МИНИСТЕРСТВО ОБРАЗОВАНИЯ
РЕСПУБЛИКИ БЕЛАРУСЬ

В.М.НОСОВ

ПРАКТИЧЕСКОЕ ИСПОЛЬЗОВАНИЕ НА ПЕРСОНАЛЬНОМ КОМПЬЮТЕРЕ ЧИСЛЕННЫХ И АНАЛИТИЧЕСКИХ МЕТОДОВ В КУРСЕ ТЕОРЕТИЧЕСКОЙ МЕХАНИКИ

Под общей редакцией доктора физико-математических наук, профессора В.И.Стражева

Допущено Министерством образования Республики Беларусь в качестве учебного пособия для студентов инженерно-технических специальностей высших учебных заведений

УП «ТЕХНОПРИНТ»
Минск 2002

Зарегистрирован в Государственном реестре информационных ресурсов Республики Беларусь.

Регистрационное свидетельство № 1180300283.

УДК 531:681.322–181.4(075.8)
ББК 32.973–01 я 73
Н 84

Рецензенты:

Председатель Научно-методического совета по ТМ Минобразования РФ,
зав. кафедрой ТМ Московского энергетического института,
академик МАН ВШ, доктор физико-математических наук,
профессор **Мартыненко Ю.Г.**

зав. кафедрой технической механики БГУИР,
доктор технических

наук, профессор **Сурин В.М.**;

зав. кафедрой "Теорет. механика и ТММ" БАТУ,
доктор физико-математических наук, профессор **Чигарев Ю.В.**

Носов В.М.

Н 84 Практическое использование на персональном компьютере численных и аналитических методов в курсе теоретической механики: Учеб. пособие.– Мн.: ТЕХНОПРИНТ, 2002. – 376 с.: ил.

ISBN 985–464–249–6

Пособие посвящено вопросам практического использования на персональном компьютере (ПК) систем компьютерной математики REDUCE (часть 1) и DERIVE (часть 4) на примерах теоретической механики и математики. Подробно описывается практическая работа на ПК в MS DOS и Windows 95/98/Me (части 2 и 3). Основной упор сделан на изложении материала, полезного в повседневной практической работе. При этом излагается минимум сведений, с помощью которых возможно решение большинства практических задач. Поэтому ***пособие может быть использовано в учебном процессе любого технического вуза, техникума, а также колледжа или школы.*** Особенно эффективно его использование в курсе теоретической механики, где оно является необходимым дополнением к широко используемому в вузах сборнику заданий для курсовых работ под ред. А.А.Яблонского (вышедшему в 4-х изданиях тиражом более 1 млн. экз.). Важную самостоятельную ценность имеют части 2 и 3, ***являющиеся самоучителем практической работы на ПК и предназначенные для массового пользователя***. Даны описания современных и широко используемых программных средств, приведены удобные приемы практической работы в Windows 95/98/Me. Пособие предназначено для студентов инженерно-технических специальностей вузов, инженерно-технических работников, учащихся техникумов, колледжей и общеобразовательных школ, а также для самообразования и массового пользователя.

УДК 531:681.322–181.4(075.8)
ББК 32.973–01 + 22.21 я 73

ISBN 985–464–249–6

МОЕЙ ДОРОГОЙ И
САМООТВЕРЖЕННОЙ МАМЕ
ЕЛЕНЕ МАТВЕЕВНЕ НОСОВОЙ
ПОСВЯЩАЕТСЯ

ПРЕДИСЛОВИЕ

Бурное развитие вычислительной математики и ее программной реализации на персональных компьютерах (ПК) привели к созданию принципиально новых систем компьютерной математики (СКМ) и аналитических вычислений (САВ).

До настоящего времени компьютеры помогали в решении математических задач лишь привилегированной научной и инженерной элите, занятой сложными и трудоемкими расчетами. Для их проведения пользователю не только нужно было освоить работу на самом компьютере, что остается необходимой задачей и в настоящее время. Он должен был математически и алгоритмически правильно поставить задачу, изучить использование довольно сложных численных методов, освоить хотя бы один язык программирования, составить программы расчетов, провести весьма трудоемкую и ответственную их отладку и тестирование.

Первым откровением стали системы REDUCE и DERIVE. Они резко уменьшили затраты времени на их освоение и программирование, а также взяли на себя груз алгебраических преобразований огромной сложности. В отличие от языков программирования высокого уровня, таких как Фортран, Си или Паскаль, системы REDUCE и DERIVE, рассматриваемые в настоящем пособии, могут решать большое количество математических задач путем введения команд, без всякого предварительного программирования. С их помощью на персональных компьютерах в настоящее время легко реализовывается интегрирование и дифференцирование символьных выражений, перестановки и перегруппировки членов, приведение подобных членов, подстановки в выражения с последующим их преобразованием.

С их помощью решение задач на ПК может быть получено как в численном виде, так и аналитически, то есть в виде формул, состоящих из математических символов. Одновременно с этим СКМ могут выполнить генерацию программы и представить результаты аналитического решения в синтаксисе, например, Фортрана, использующего обычные численные методы. Акцентированию внимания на этом и служат соответствующие эпитеты из названия пособия.

REDUCE и DERIVE являются универсальными системами, ориентированными на решение широкого круга математических задач.

Следует отметить простоту системы REDUCE и фантастические возможности работы с матрицами, что использовано в пособии как на примерах задач кинематики (см. п. 3), так и при решении систем линейных алгебраических уравнений задач статики (см. п. 2). Легкость получения нужных функциональных зависимостей в аналитической форме, никак не определяемых ранее, просто поражает. Ведь для их определения ранее нужно было выполнить численное исследование с трудоемкой подготовкой варьируемых данных в численном виде для каждого рассматриваемого положения, а затем мучиться над обработкой большого массива численных результатов.

Система DERIVE более интегрирована, обладает дружелюбным интерфейсом и большими графическими возможностями.

Системы REDUCE и DERIVE отличаются тем, что удачно сочетают возможности проведения численных и символьных вычислений с простотой и не слишком высокими требованиями к используемой технике. Это делает их незаменимыми для использования в вузах, техникумах и школах, компьютерный парк которых в основном морально устарел.

В последнее время разработана версия REDUCE 3.6 под Windows. Однако вследствие своей чрезвычайной редкости из-за строго лицензионного распространения, в учебном пособии рассмотрены широко распространенные версии 3.3 – 3.5 под MS DOS, реализуемые на *любых* IBM-совместимых ПК (см. часть 1).

Под Windows существует достаточное количество современных СКМ, распространяемых на дешевых CD–дисках и использующих аппаратные возможности современных ПК: DERIVE 4.11, MathCad 2001, MAPLE 7, MatLab 6, Mathematica 4. Из них мы в этой книге рассмотрим (из-за ограниченности ее объема) основные аспекты использования простой и удобной системы DERIVE версий 4.01–4.11.

Такое совместное рассмотрение версий REDUCE 3.3–3.5 под MS DOS и DERIVE 4.01–4.11 под Windows существенно повышает их достоинства и обеспечивает следующие преимущества:

- создание плавного перехода при практическом изучении современных СКМ;
- удобство работы и простота в освоении;

- совместное использование систем REDUCE и DERIVE вместе с текстовым процессором типа Word организует рабочее место исследователя для студента и инженера, сравнимое с самыми сложными современными интегрированными пакетами, весьма дорогостоящими и требовательными к аппаратным возможностям ПК.

Хотя и "нельзя объять необъятное", но при общении с такой информационной системой, как ПК и созданное для него огромное программное обеспечение, поневоле в какой-то форме приходится это делать. Поэтому автор старался вложить в эту книгу свой опыт и все нужное для практической работы на ПК самых разнообразных групп пользователей, реализуя развиваемый им принцип *многовекторной литературы*, чтобы каждый нашел в ней для себя нужное:

- для студентов вузов она явится учебным пособием в полном объеме своих четырех частей, в котором они получат ключ к быстрому решению практически всех вычислительных задач учебных дисциплин;
- любой желающий научиться работать на ПК (от школьника до домохозяйки) во второй и третьей частях найдет простую и понятную квинтэссенцию необходимого для удобной работы на ПК;
- учащиеся техникумов, колледжей и общеобразовательных школ после повторения примеров первой и десятой глав ("Быстрый старт"), приобретут удобных математических помощников REDUCE и DERIVE для своей учебной работы;
- преподаватель сделает свой предмет творческим, вызвав интерес у студентов или школьников;
- аспирант, инженер или ученый избавится от рутинной работы, повысив свой творческий потенциал, а также сделает более удобной и эффективной свою работу на ПК, как это помогло сделать и автору этой книги в процессе ее написания.

Разработанные современные СКМ отражают достижения происходящей в настоящее время в мире информационной революции, на несколько порядков увеличивая интеллектуальные возможности любого человека. При их постепенном освоении возникает ощущение, что ваши способности к точным наукам многократно возрастают. Поэтому использование СКМ способно совершить революцию в образовании и значительно повысить эффективность учебного процесса.

В курсе теоретической механики используются практически все основные разделы *вычислительной математики*:

- решение систем линейных алгебраических уравнений при определении реакций;
- интегрирование при нахождении центра тяжести или замены распределенных сил в статике;
- численное дифференцирование в задачах кинематики;
- интегрирование дифференциальных уравнений в задачах динамики.

Поэтому пособие можно рассматривать и как практикум по использованию СКМ, и оно также может быть использовано при выполнении расчетов в учебном процессе любой общетехнической и фундаментальной кафедры. С этой точки зрения все приводимые примеры носят обычный искусственный характер типовых задач вычислительной математики.

При изучении курса теоретической механики с применением ПК пособие непосредственно встраивается в учебный процесс, так как является необходимым дополнением к широко используемому в вузах и имеющемуся в их библиотеках сборнику заданий для курсовых работ [22], вышедшему в четырех изданиях тиражом свыше 1 млн. экземпляров. Из него взяты все типовые примеры рассматриваемых задач, аналитические решения которых методами теоретической механики рассмотрены там же и по этой причине здесь не повторяются.

Пособие является дальнейшим развитием книги В.М.Носова "Программирование на персональных ЭВМ задач теоретической механики", где весь материал рассмотрен для использования численных методов расчета на ПК для тех же типовых примеров. Это дает возможность организовать комплексное применение численно-аналитических методов на ПК, что представляется важным с практической и дидактической точек зрения.

Все сведения представлены с применением пользовательского или деятельностного подхода, что необходимо для практической работы на ПК. Такой подход одновременно с синтетичностью изложения материала, когда все необходимое для работы на ПК находится в одной книге, обеспечивает возможность неопытному пользователю без предварительной компьютерной подготовки в течение короткого времени освоить работу в нужном объеме с описываемой системой СКМ.

Эту книгу удобно использовать также для самообразования, ибо она создавалась так, что полностью заменяет собой автора и обеспечивает автономную работу практически любого пользователя на ПК.

Все это делает пособие самоучителем работы на ПК для решения основных вычислительных задач с использованием СКМ.

Книга не ставит своей целью охватить всю информацию по структуре и средствам описываемых СКМ. Она пытается ознакомить читателя с ее наиболее мощными и удобными возможностями, а также на конкретных примерах научить эффективно применять их в своей работе.

В заключение автор выражает огромную благодарность ректорату БНТУ за предоставленную возможность работы на настольно-издательской системе, что позволило ему выполнить верстку оригинал-макета этой книги.

Автор считает также приятным долгом выразить сердечную благодарность рецензентам, чьи критические замечания и пожелания помогли улучшить пособие.

Следует с особой благодарностью отметить роль заведующего лабораторией ЦИС БГУ доцента Позняка Ю.В., предоставившего в распоряжение автора необходимый справочный материал по системе REDUCE и стоявшего у самых истоков настоящей работы, которому только неблагоприятные обстоятельства, к всеобщему сожалению, не позволили принять участие в ее выполнении.

В апробации настоящей работы в учебном процессе и проверке разработанных программ и дополнений в рамках УИРС принимали участие в течение ряда лет многие студенты БНТУ. Особенно ценную техническую и информационную поддержку при подготовке пособия оказали студенты Чубанов С.С., Прилепо О.А. и Коляда В.М., которым автор выражает признательность.

Текст пособия и оригинал-макет подготовлен автором с использованием программ Word 2000, REDUCE, DERIVE и Page Maker 6.52. Все фрагменты и файлы данных помещались в оригинал-макет машинным образом и дополнительно тщательно проверялись. Автор также выражает благодарность Мальдису Я.А. за техническую поддержку и консультации при выполнении верстки.

Необходимые консультации можно получить на кафедре теоретической механики БНТУ (тел. (8-1037517) 2327425) у автора, который также открыт для сотрудничества с издательствами, использующими цивилизованные формы работы.

Автор

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ

В пособии ставится задача дать возможность студенту самому выбирать форму общения с ПК от использования готовых программ при практически нулевом уровне, до их самостоятельного составления из приведенных готовых блоков и выполнения достаточно серьезного исследования на ПК вариаций условий задачи. На каждом уровне возможен выбор различных вариантов использования ПК для решения задачи:

- получение численного решения;
- получение решения в символьном виде;
- аналитическое и графическое исследование варьирования условий задачи или различных параметров, при котором современные аппаратные и программные возможности ПК используются в качестве инструмента исследования.

Для удобства пользования в пособии приведены необходимые сведения для практической работы на ПК и по системам аналитических вычислений REDUCE и DERIVE. Для каждого раздела механики все изложение производится с использованием одного или нескольких типовых примеров, чем достигается целостность восприятия.

Все рассматриваемые примеры взяты из двух последних изданий [22] и [23] сборника заданий под ред. проф. А.А.Яблонского, широко и активно используемого в курсе теоретической механики.

Поэтому пособие может быть использовано как для организации учебного процесса с применением ПК при изучении курса теоретической механики, так и в качестве универсального практикума по информатике и САВ REDUCE и DERIVE. В последнем случае на источник получения рассматриваемых примеров обращать внимание не нужно.

Таким образом, с этой точки зрения пособие представляет собой профилирование курса информатики и вычислительной математики для нужд теоретической механики. Оно может служить основой для координации междисциплинарных связей и организации непрерывной подготовки по активному и согласованному использованию вычислительной техники при формировании будущих специалистов.

Ссылки на главы, параграфы, подпараграфы и пункты обозначаются буквой “п” (например, п. 1, п. 1.2, п. 1.2.1, п. 1.2.1.1). Если материал понимается, то на них обращать внимание не надо.

В пособии используется два вида ссылок при описании операторов и программ: на номер в круглых скобках и на номер без скобок.

Ссылка на номер в круглых скобках идентифицирует:

- весь приведенный сегмент программы, если в следующих строках данного программного модуля нет номеров со скобками;
- или только записи, находящиеся в этой же строке, если следующие операторы идентифицированы также номерами с круглыми скобками.

Нумерация с круглыми скобками сквозная в порядке возрастания по всему тексту пособия.

Ссылка на номер без скобок идентифицирует только сам оператор, находящийся в этой строке.

Номера длинных операторов, расположенных в нескольких строках, указываются в первой или в последней строке.

Номера без скобок относятся к описываемым программам и проставлены только для удобства пояснений. В программе на бланке или на экране дисплея операторы не нумеруются.

При описании вариантов и дополнений к основным программам вставляемые операторы должны располагаться в основной программе по порядку возрастания номеров (без скобок), при этом:

- если *номер* (без скобок) вставляемого оператора *совпадает* с имеющимся в основной программе, то последний *заменяется*;
- если *номер не совпадает — то оператор вставляется* между операторами с соответствующими меньшим и большим номерами в основной программе.

Для удобства пользования пособием применяются следующие ***шрифтовые выделения***:

- *все, что вводится* пользователем с клавиатуры, *выделяется* ***жирным шрифтом*** и печатается ***строчными буквами***: **(x+y*x/2+x*z/3)**3**;
- *результаты вычислений и символьных преобразований* печатаются обычными ***строчными буквами без выделения*** (хотя в большинстве версий REDUCE они будут представлены на экране ПК или в файле с использованием прописных букв) : $(x^3*(27*y^3 + 54*y^2 *z + 162*y^2 + 36*y*z^2 + 216*y*z + 324*y + 8*z^3 + 72*z^2 + 216*z + 216))/216$;

- *команды, флаги, операторы и системные функции* REDUCE печатаются прописными латинскими буквами без выделения;
- *команды MS DOS* в тексте печатаются *прописными латинскими буквами и выделяются курсивом* (с наклоном);
- *флажки и переключатели* Norton Commander и Windows печатаются соответственно латинскими или русскими буквами и **выделяются жирным шрифтом**;
- *названия клавиш* печатаются шрифтом Arial и помещаются в угловые скобки: <Enter>;
- *названия панелей, меню и подменю, кнопок и опций* Norton Commander и Windows (а также названия окон и пиктограмм) *выделяются курсивом* или печатаются шрифтом Arial.

При повторении описываемых сеансов работ в REDUCE и DERIVE следует обращать внимание на автоматическую нумерацию вводимых команд:

- если номер первой команды приводимого фрагмента *начинается с цифры 1*, то **сеанс работы начинается** и *все впервые используемые в этом фрагменте переменные* ***являются свободными***;
- *нумерация продолжается – все впервые используемые в этом фрагменте переменные* ***могут быть связанными*** и иметь значения, заданные им в течение текущего сеанса работы. Поэтому для получения приведенных результатов следует *повторить данный сеанс работы с самого начала*;
- если *номер шага не указывается* – то приводимый фрагмент является локальным: он не зависит от текущего сеанса работы и все впервые используемые в нем переменные ***являются свободными***.

Переменные и углы в программах обозначаются в форме их идентификаторов. При этом греческие буквы записываются в их латинской транскрипции, а для двухсимвольных переменных цифра или буквы I, J, K и N выполняют роль индекса (например A1, B2, P1, X0, X1, TN, TK). Этот принцип используется также при описании фрагментов программ.

Во 2-й главе индекс также может обозначаться строчной буквой за обозначением индексированной величины: реакция R в точке E будет обозначаться Re, в точке F — Rf; горизонтальные и вертикальные составляющие реакции в точке A — Xa и Ya.

Значения углов тригонометрических функций при записи уравнений равновесия указаны в градусах.

ОСНОВНЫЕ ОБОЗНАЧЕНИЯ

ДУ — дифференциальное уравнение,
ДУ2П — дифференциальное уравнение второго порядка,
ИДФ — идентификатор файла (его полное имя с указанием пути доступа к нему),
ИФ — имя файла,
ОС — операционная система,
ПК — персональный компьютер,
САВ — система аналитических вычислений,
СДУ — система дифференциальных уравнений,
СЛАУ — система линейных алгебраических уравнений,
ССМ — система символьной математики,
ССС — система сходящихся сил,
ТФ — тип файла.

ЧАСТЬ 1. СИСТЕМА АНАЛИТИЧЕСКИХ ВЫЧИСЛЕНИЙ (CAB) REDUCE

ГЛАВА 1. БЫСТРЫЙ СТАРТ

Система аналитических вычислений REDUCE предназначена для проведения точных алгебраических вычислений и преобразований любой степени сложности. Она может выполнять разложение и возведение в степень полиномов и рациональных функций в различных формах, раскрывать скобки, выносить общие множители и проводить автоматическое и контролируемое упрощение выражений, удивительно просто работать с различными матричными выражениями, а также выполнять дифференцирование и интегрирование в символьной форме.

Буквы при вводе можно писать строчные и прописные, однако, после проведения вычислений или аналитических преобразований все результаты представляются с использованием строчных букв. Операции с целыми числами осуществляются с произвольной точностью.

Предваряя описание и изложение возможностей CAB REDUCE, покажем на примере нескольких операторов и команд, что это простой и удобный программный продукт.

Наиболее легко получить много полезных результатов и научиться работать с CAB REDUCE можно в интерактивном режиме, называемом также режимом интерпретатора или режимом командной строки.

1.1. Первый сеанс работы

После загрузки CAB REDUCE, осуществляемой обычно после запуска на выполнение файла reduce.bat или reduce.exe, система печатает «шапку» типа: REDUCE 3.3, 15-Jan-88 ...

где номер версии (3.3) и дата ее создания могут быть другими для конкретной реализации REDUCE на вашем ПК. Затем система приглашает пользователя ввести информацию в командной строке, печатая в ней номер шага

1:

Теперь пользователь может ввести после приглашения (1:) правильное «редьюсовское» выражение или команду, ограниченную точкой с запятой, которая обозначает конец выражения, например, возвести число 2 в 2000-ю степень:

1: **2**2000;** (1.1)

После нажатия клавиши <Enter> система вводит выражение, вычисляет его и выдает на экране дисплея результат — строку (строки) вывода или сообщение об ошибочно введенной команде:

```
114813069527425452423283320117768198402231770208869520047764
736825766261392370313856659486316506269918445964638987462773
471189608630553314259313561666531853912998914531228000068877
914824004487142892699006348624478161546364638836394731702604
663539709049965581623988089446296056233116495361642219703326
813441689089844585056023794848079140589009347765004290027167
062583052200813223628129176126788331720659899539641812702177
985840404215985318325154088943390209192055495778358967203916
008195721663058275538042558372601552834878641943205450891527
578388262517543552880082284277081796545376218485114902937 6 (1.2)
```

Тут же выдается следующее приглашение (2:) вводить новую команду и т.д., например:

2: **(x+y*x/2+x*z/3)**3;** (1.3)

$$(x^3*(27*y^3 + 54*y^2*z + 162*y^2 + 36*y*z^2 + 216*y*z + 324*y + 8*z^3 + 72*z^2 + 216*z + 216))/216 \quad (1.4)$$

Чтобы лучше понять принципы работы системы REDUCE, давайте рассмотрим эти простые примеры.

Поскольку в алгебраических вычислениях нужны точные результаты, то арифметические операции с целыми числами выполняются практически с безграничной точностью (зависящей только от доступной памяти вашего ПК).

Это значение (1.2) числа 2**2000 состоит из 603 цифр и занимает 8 строк на экране дисплея.

Характерной чертой второго примера является то, что во введенном выражении (1.3) все знаки были напечатаны в одной строке, а в результирующем (1.4) — степени переменных подняты на одну строчку вверх.

Отметим также, что система REDUCE имеет дело с переменными и константами. Однако при вычислении выражения (1.3) переменная осталась сама собой, а вычисление было произведено в соответствии с правилами алгебры. В (1.4) приведены подобные члены, а переменные упорядочены специальным образом. Однако и способ раскрытия скобок, и упорядочивание переменных, и формат вывода, и тому подобное, все это контролируется пользователем.

Для осуществления такого контроля имеются две команды **ON** и **OFF**. Каждый режим работы представлен так называемым флагом (переключателем), который может быть либо установлен (поднят) — **ON**, либо отменен (опущен) — **OFF**. В качестве параметров команд **ON** и **OFF** служат имена флагов. Эти описания устанавливают (**ON**) или отменяют (**OFF**) соответствующий режим работы, определяемый указанным флагом (подробнее см. п. 1.5). Например, опуская флаг ALLFAC,

3: **off allfac;** (1.5)

мы отказываемся от выноса за скобки общих сомножителей в преобразуемых выражениях. После этого устанавливается новый режим работы и выдается следующее приглашение (4:).

Результаты каждого вычисленного выражения запоминаются системой в переменной *WS* (от workspace) и сохраняются до следующего вычисления, в котором этот идентификатор можно использовать. Переменная *WS* называется *рабочим полем* или *рабочей областью выражения.* Так как на 3-м шаге был изменен только режим работы, то для распечатывания предыдущего выражения (1.4) без выноса за скобки общих сомножителей, нужно просто ввести

4: **ws;**

$$(27*x^3*y^3 + 54*x^3*y^2*z + 162*x^3*y^2 + 36*x^3*y*z^2 + 216*x^3*y*z + 324*x^3*y + 8*x^3*z^3 + 72*x^3*z^2 + 216*x^3*z + 216*x^3)/216 \quad (1.6)$$

Выражение (1.4) не изменилось. Оно просто приняло другую форму. Например, для взятия от него производной по x нужно ввести

5: **df(ws,x);**

$$(27*x^2*y^3 + 54*x^2*y^2*z + 162*x^2*y^2 + 36*x^2*y*z^2 + 216*x^2*y*z + 324*x^2*y + 8*x^2*z^3 + 72*x^2*z^2 + 216*x^2*z + 216*x^2)/72 \quad (1.7)$$

Установим опять режим выноса за скобки общих сомножителей в преобразуемых выражениях и вычислим интеграл по x от выражения (1.6), чему равно теперь значение *WS*:

6: **on allfac; int(ws,x);**

$(x^3*(27*y^3 + 54*y^2*z + 162*y^2 + 36*y*z^2 + 216*y*z + 324*y + 8*z^3$
$+ 72*z^2 + 216*z + 216))/216$ (1.8)

В каждом слагаемом (1.6) присутствовало *x*. Поэтому после взятия от него производной по *x* и вычисления интеграла по *x* мы должны опять получить исходное выражение. Установка флага ALLFAC должна представить его в форме выноса за скобки общих сомножителей (1.4), что и видим на экране.

Напоминаем, что **все, что должно вводиться вами** в командной строке, **всегда выделяется нами жирным шрифтом**. Как видите, оно имеет достаточно простой вид.

Обычным шрифтом печатаются результаты преобразований и вычислений CAB REDUCE. Они могут иметь очень сложный вид. Но вам не нужно проверять и анализировать промежуточные выкладки. Желательно только продумать систему тестовых проверок (типа приведенной выше).

Другим часто используемым свойством системы является возможность выдачи результатов в форме, совместимой с Фортраном. Такие результаты могут быть использованы для создания программ на Фортране. Это особенно полезно, если рассматривать работу системы, как некоторый способ генерации алгебраических формул, которые должны быть использованы в качестве основы для обширных численных расчетов. Для этого устанавливаем флаг FORT и набираем *ws*:

8: **on fort; ws;**

```
 ans = (x**3*(27.*y**3+54.*y**2*z+162.*y**2+36.*y*z**2+
. 216.*y*z+324.*y+8.*z**3+72.*z**2+216.*z+216.))/216.      (1.9)
```

Имя ANS присваивается выражению по умолчанию и все его строки печатаются с седьмой позиции с использованием синтаксиса Фортрана. В качестве символа переноса используется точка, сопровождаемая пробелом. Вернемся к обычному режиму работы системы, опустив флаг:

10: **off fort;** (1.10)

Можно вводить за один раз несколько выражений и команд (шаги 6 и 8), но система *помнит каждую из них* и печатает следующее приглашение соответственно 8: и 10:.

Оператор DISPLAY позволяет распечатать и просмотреть столько предыдущих выражений и команд M, сколько их указано в качестве аргумента, например для M=5:

11: **display 5;** (1.11)
6**:** on allfac;
7: int(ws,x);
8**:** on fort;
9: ws;
10**:** off fort;

Если же в качестве аргумента выступает любое нечисловое выражение, то изображаются все предыдущие вводы.

Ко всем результатам можно обратиться с помощью порядкового номера команды *N*, который в этом случае печатается без следующего за ним в приглашении двоеточия. При этом возможны два способа.

- ***Для использования входного выражения в новом вычислении*** нужно ввести *INPUT(N)*. При этом будет выполняться входное предложение, соответствующее указанному номеру (происходит его повторное преобразование, вычисление или отработка соответствующей команды, оператора или описания).
- ***Для использования только результатов вычислений*** нужно ввести *WS(N)*. При этом ***не происходит*** повторная отработка соответствующей команды, оператора или описания. В этом случае, если порядковому номеру команды *N* соответствует описание изменения режимов работы **ON** или **OFF, то они не будут выполнены.** В таком случае не возникает и изменения состояния того флага, которое пользователь намеревался произвести, но *система не сообщает об этом*.

С помощью *INPUT(N)* и *WS(N)* можно использовать предыдущие входные предложения и результаты вычислений в любых выражениях, например, *INPUT(2)**2/WS(5)**3*.

Однако удобнее для возможности произвольного обращения к предыдущим выражениям и результатам использовать оператор присваивания.

Этот оператор заменяет значение переменной в левой части оператора на значение выражения, стоящего в его правой части, и имеет в простом случае вид:

```
переменная := выражение                    (1.12)
```

Между оператором присваивания и похожим на него алгебраическим равенством существует очень важное различие. Оператор X:=Y+Z означает для машины: взять содержимое ячейки памяти в символьном или числовом виде, соответствующее переменной Y, сложить с содержимым ячейки

переменной Z и передать результат также в символьном или числовом виде в ячейку памяти, соответствующую переменной X.

Знак ":=" в операторе присваивания имеет значение "необходимо заменить на ...", а в математике знак равенства означает "то же самое значение, что и ...".

С помощью операторов присваивания сеанс работы 2 – 10 повторен ниже под порядковыми номерами 12 – 20, а совпадающие результаты преобразований, отличающиеся только наличием идентификатора переменной, имеют тот же номер, но со штрихом.

12: **tt:=(x+y*x/2+x*z/3)**3;** (1.3′)

tt := $(x^3*(27*y^3 + 54*y^2*z + 162*y^2 + 36*y*z^2 + 216*y*z + 324*y + 8*z^3 + 72*z^2 + 216*z + 216))/216$ (1.4′)

13: **off allfac;** (1.5′)

14: **tt;**

$(27*x^3*y^3 + 54*x^3*y^2*z + 162*x^3*y^2 + 36*x^3*y*z^2 + 216*x^3*y*z + 324*x^3*y + 8*x^3*z^3 + 72*x^3*z^2 + 216*x^3*z + 216*x^3)/216$ (1.6′)

15: **ttd:=df(tt,x);**

ttd := $(27*x^2*y^3 + 54*x^2*y^2*z + 162*x^2*y^2 + 36*x^2*y*z^2 + 216*x^2*y*z + 324*x^2*y + 8*x^2*z^3 + 72*x^2*z^2 + 216*x^2*z + 216*x^2)/72$ (1.7′)

16: **on allfac; tti:=int(ttd,x);**

tti := $(x^3*(27*y^3 + 54*y^2*z + 162*y^2 + 36*y*z^2 + 216*y*z + 324*y + 8*z^3 + 72*z^2 + 216*z + 216))/216$ (1.8′)

18: **on fort; tti;**

```
ans = (x**3*(27.*y**3+54.*y**2*z+162.*y**2+36.*y*z**2+
. 216.*y*z+324.*y+8.*z**3+72.*z**2+216.*z+216.))/216.
```
(1.9′)

20: **off fort;** (1.10′)

Обратим Ваше внимание, что после задания переменной TT в операторе присваивания (1.3′) происходит указание имени переменной при ее распечатке в выражении (1.4′).

При изменении же режимов выполнения программы и печати результатов, например при отказе от выноса за скобки общих сомножителей из-за опускания флага ALLFAC в уже заданном выражении (1.4′), происходит только печать перестроенного выражения (1.6′) без указания его имени.

Отметим, что значение переменной *WS*, т.е. последнего вычисленного выражения, может быть присвоено любой переменной при помощи команды SAVEAS. Следует выдать эту команду с единственным параметром – именем переменной, которой надо присвоить значение переменной *WS*. Поэтому, чтобы присвоить значение выражения (1.4) из первоначального сеанса работы 2 – 10 переменной TT, нужно после получения выражения (1.4) на 3-м шаге ввести команду

3: **saveas tt;** (1.5′′)

после чего можно сразу начинать повторный сеанс работы 13 – 20, все результаты которого будут совпадать с (1.5′)– (1.10′).

Оператор присваивания связывает переменную в левой части со значением выражения в правой его части, а команда SAVEAS связывает указанную в ней переменную со значением последнего вычисленного выражения.

Эта связь в обоих случаях может быть разорвана командой CLEAR, которая стирает ненужные далее выражения указанных переменных. Например:

21: **clear tt, ttd, tti;** (1.13)

Замечание. Автор ограничен требованиями экономии места, поэтому приводятся довольно простые примеры (кроме 1:), не передающие практически безграничные возможности CAB REDUCE.

Попробуйте *сами их проверить*, выполнив операции с более сложными примерами и большими степенями. Возможно, они вызовут у вас удивление. А это очень важное чувство.

Когда-то одного маленького мальчика удивила стрелка компаса, показывающего на север. Он стал изучать, думать и размышлять. Вопрос этот однозначно не решен и до сих пор. Но впоследствии фамилия этого мальчика (Эйнштейн) стала известной всему миру.

И если при этой проверке благотворное чувство удивления посетит вас и приведет к изучению практического использования CAB на ПК, то ваши возможности также возрастут удивительным образом.

И в надежде на это мы пока завершим наш первый сеанс работы. Команды «BYE; « и «QUIT»; прекращают работу REDUCE и возвращают управление операционной системе. Поэтому

22: **bye;** (1.14)

и приступим к более систематическому изучению.

1.2. Описание элементов языка

1.2.1. Символы и предложения

В качестве основных символов системы REDUCE используются:

- прописные и строчные буквы латинского и русского (там, где он поддерживается) алфавитов;
- десятичные арабские цифры от 0 до 9;
- специальные символы: ! ( ) = - + / * , ; . : % $ “ « > <.

Система допускает при вводе использование в произвольном порядке строчных и прописных букв. Для обычного режима работы CAB REDUCE они внутренне неразличимы. Поэтому для лучшего отличия записи символов всегда будут использоваться:

- в тексте прописные буквы;
- строчными буквами жирным шрифтом с использованием абзацного отступа выделяется все, что должно вводиться в командной строке и после чего должна нажиматься клавиша <Enter> (<Ввод>);
- обычными строчными буквами без использования абзацного отступа печатаются результаты преобразований и вычислений CAB REDUCE.

Предложением в REDUCE называется последовательность символов, заканчивающихся одним из двух знаков “;” или “$”, называемых разделителями, ограничителями или терминаторами. В первом случае *результат его выполнения “распечатывается”* на экране или в файле (см., например, (1.1), (1.3), (1.4)), во втором — нет (см. (1.17)).

Предложения в программе записываются одно за другим. Их может быть как несколько в одной строке записи, так и каждое из них может занимать несколько строк.

Отметим, что при этом лишние пробелы игнорируются, а знаки переноса не используются.

Каждое предложение, имеющее численное или символьное значение, называется ***выражением*** (см., например, (1.1), (1.3), (1.4)), не имеющее — ***командой*** (см. (1.5), (1.10), (1.13)).

Для записи *выражений* в REDUCE используются имена переменных и знаки арифметических операций: + сложение, – вычитание, * умножение, / деление, ** возведение в степень. Например: B**2 – 4*A*C.

Все знаки арифметических действий должны присутствовать в выражении явно. Математическое выражение IJ (значение I умножается на значение J) в REDUCE должно быть написано обязательно так: I*J, иначе оно будет понято как символическое имя IJ и возникнет трудноуловимая ошибка, не препятствующая работе программы.

Рядом не могут находиться два знака арифметических действий, они должны отделяться скобками, например: R*(–0.01).

Последовательность выполнения арифметических операций в выражении такая же, как и при обычной записи алгебраических формул. Сначала выполняется возведение в степень, затем умножение и деление, а в конце — сложение и вычитание.

Если в выражении записано несколько одинаковых операций подряд, то они выполняются слева направо. В случае необходимости иного порядка выполнения арифметических операций, как и при написании обычных формул, используются круглые скобки.

Использование круглых скобок желательно также в сомнительных случаях для ясности записи и уверенности, что выражение будет вычисляться правильно. Нужно только следить, чтобы количество открывающих скобок совпадало с количеством закрывающих скобок.

Комментарием в REDUCE называется любая последовательность символов: от слова “COMMENT” до ближайшего разделителя (знаков “;” или “$”) или от символа “%” до конца строки. Комментарии служат для записи пояснений в программах и системой не обрабатываются.

1.2.2. Числа

В системе REDUCE числа могут быть *целые, рациональные* или *вещественные*.

Целые числа представляют собой последовательности десятичных цифр со знаком или без него, написанных без использования десятичной точки: 88, +5, –7. Они могут входить в любые выражения. Система REDUCE ориентирована на работу с целыми числами и не накладывает ограничений на количество цифр для их представления с произвольной точностью (см., например, целое число, выражающее результат возведения 2**2000 и представленное под номером (1.2)).

Рациональные числа представляются в виде отношения двух целых чисел с сокращенными общими множителями. Так обычно представляются

дробные числа (в виде отношения двух целых несократимых чисел), *знак* которых хранится в числителе. Поскольку длина целого числа в REDUCE ограничивается лишь размерами используемой оперативной памяти, то это позволяет производить вычисление рациональных значений без округления:

aa:=(1/(-2)+1/7+1/11)15;** (1.15)

aa := – 155509831499153791088860 1 /
64984654238835018083651806471782 4

Вещественные числа записываются в двух формах:

- ***в основной форме*** число записывается в виде целой и дробной частей, разделенных точкой, со знаком +, – или без знака: +151.2, –0.0001512, 1.512. В большинстве версий REDUCE разрешается использовать не более 8 десятичных цифр;
- ***в показательной форме*** запись числа состоит из основной формы с указанием десятичного порядка, который обозначается буквой E, а следующая за ней целая константа со знаком или без него определяет величину порядка. Например, +0.1512E3, –0.1512E–3, 151.2E–2.

Предупреждение. Число не может начинаться с точки. Поэтому при записи дробного числа, меньшего единицы, ноль обязательно указывается. Существует мнение, что реализация режима арифметики с плавающей запятой в большинстве версий REDUCE не всегда удачна [7, с. 19].

1.2.3. Идентификаторы

Идентификаторы используются в качестве переменных, меток, а также имен массивов, операций и процедур.

Идентификаторы представляют собой набор букв, цифр и допустимых символов, начинающийся с буквы: N, A1, B2, WORK, KNNEKO. В качестве букв используются буквы латинского алфавита от A до Z, цифры: 0,1,...,9.

Длина идентификатора зависит от конкретной реализации системы (обычно от 1 до 24 символов). Идентификатор нельзя писать в двух строках текста.

Если в идентификатор включается недопустимый символ, то перед таким символом следует употреблять восклицательный знак или закрывающую скобку, что зависит от конкретной реализации системы. Также следует поступать при желании начать идентификатор с цифры.

Однако использование такой возможности нежелательно, так как многие служебные идентификаторы системы REDUCE содержат специальные

символы. Их совпадение с идентификатором пользователя может привести к непредсказуемым последствиям. Особенно избегайте использования знака "*", входящего во многие внутренние имена REDUCE.

Запрещено использовать в этом качестве также и *ключевые слова*, означающие имена встроенных в систему команд и процедур. Поэтому они **не должны использоваться в качестве идентификаторов.** Их более восьмидесяти (см. Руководство пользователя по системе REDUCE [32]). Некоторые из них, имеющие 6 и менее символов, приведены ниже:

ALL, ARRAY, BEGIN, BYE, CLEAR, CMD, CONT, CORE, CREATE, DCL, DEFINE, DEPEND, DOWN, EDIT, EMB, END, EXEC, EXPR, FACTOR, FEXPR, FLAGOP, FLOAD, FOR, FORALL, FSLOUT, GO, HELP, IF, IN, INDEX, INPUT, INTEGER, KORDER, LAMBDA, LET, LINEAR, LISP, MACRO, MASS, MATCH, MATRIX, MSHELL, OFF, ON, OPERATOR, ORDER, OUT, PAUSE, PROCEDURE, QUIT, RATIONAL, REAL, REMFAC, REPEAT, RETRY, RETURN, RLISP, SAVEAS, SCALAR, SHARE, SHUT, SLISP, SMACRO, TP, TRST, UP, VECTOR, WEIGHT, WHILE, WRITE, WS.

1.2.4. Переменные

Переменные — это величины, которым присвоены наименования. Для каждой переменной в памяти машины отводится место, в котором хранится ее численное или символьное значение. Для обозначения переменных используются символические имена (идентификаторы).

Каждая переменная обозначается именем, которое является ее первоначальным значением. Необходимо различать имя переменной и ее значение, даже если они совпадают.

Имя переменной — обозначение "емкости", которую с помощью операторов присваивания (1.12) наполняют содержимым (численным или символьным значением).

Свободной называется переменная, которой ничего не присвоено и значение которой совпадает с ее именем.

Связанной — которой ранее было присвоено некоторое значение. Присвоенное переменной значение сохраняется неизменным при дальнейшем выполнении программы или сеанса работы в REDUCE до тех пор, пока не будет изменено оператором присваивания (1.12) или командами CLEAR (1.13), SAVEAS (1.5′′) (или LET). Команда CLEAR делает связанные переменные свободными, какими они считаются в момент начала выполнения программы.

Если в операторе присваивания справа в качестве выражения стоит свободная переменная, то ее *имя присваивается* в качестве вычисленного значения переменной слева:

a:=b; (1.16)

a := b

Если в операторе присваивания справа в качестве выражения стоит связанная переменная, то переменной слева *присваивается вычисленное ранее значение переменной* справа:

b:=(x+y*x/2+x*z/3)3$** (1.17)

a:=b;

a := $(x^3*(27*y^3 + 54*y^2*z + 162*y^2 + 36*y*z^2 + 216*y*z + 324*y + 8*z^3 + 72*z^2 + 216*z + 216))/216$

Всегда, когда происходит вычисление выражения, все входящие в него имена исследуются на предмет их совпадения с именами связанных переменных. Если такое совпадение обнаружится, то данное имя заменяется на значение связанной переменной. Имена свободных переменных, которыми являются в нашем примере X, Y и Z, участвуют в дальнейших преобразованиях.

Используются переменные следующих типов: RATIONAL — рациональные, INTEGER, — целые, REAL — вещественные для числовых переменных и SCALAR — для символьных переменных. Рядовой пользователь может не принимать во внимание тип переменных. Если он заранее не объявлен, то по умолчанию переменные имеют тип SCALAR. В качестве значений они принимают представление любого алгебраического выражения. Если оно отсутствует, то они обозначают сами себя.

Некоторые переменные имеют в REDUCE определенные значения:

- **E** — представляет основание натуральных логарифмов. Если в выражении встречается LOG(E), то оно автоматически заменяется на единицу. В режиме “ON NUMVAL;” символ E будет заменен его значением с плавающей точкой, округленным с текущей степенью точности арифметики с плавающей запятой;
- **I** — представляет собой квадратный корень из -1, I**2 заменяется -1. Аналогично для более высоких степеней I. Используется для представления комплексных чисел;
- **PI** — число “π”. В режиме “ON NUMVAL;” заменяется значением числа “π” с плавающей точкой и текущей точностью;

- T — имеет значение «истина» и используется в символьном режиме;
- NIL — в алгебраическом режиме работы NIL является синонимом нуля, в символьном режиме имеет значение “ложь”.

1.3. Имена со скобками

Помимо переменных, описанных выше, в REDUCE используются также так называемые *имена со скобками*, в которых указываются списки параметров, разделенных запятыми: Q(5), F(X), SIN(ALFA), COS(BETA), FY(A, F(X), SIN(ALFA)). В качестве параметров могут выступать не только простые имена, но и любые выражения, в том числе включающие в себя имена со скобками.

Если список параметров состоит из одного элемента, то скобки, его ограничивающие, можно заменить пробелами. Однако этим нарушается ясность записи. Поэтому мы сами так никогда поступать не станем, и вам не советуем.

Первая операция, которую выполняет REDUCE, встречая в программе имя со скобками, — вычисление значений выражений, записанных в списке параметров. Иначе говоря, *список параметров всегда преобразуется к списку их значений*, и только после этого начинается анализ самого имени со скобками.

В REDUCE имена со скобками применяются для обозначения операторов (см. п. 1.3.1), системных функций (см. п. 1.3.2), элементов массивов (см. п. 1.3.3) и матриц (см. п. 1.3.4), процедур (см. п. 4.6).

1.3.1. Операторы

Функции действующие в системе, называются операторами. Основным свойством оператора является то, что само его имя вместе со списком значений параметров, который у оператора принято называть списком аргументов, может быть его значением.

Иными словами, *имя со скобкой, при использовании его в качестве оператора может условно рассматриваться в качестве свободной переменной.*

Это свойство свободных имен со скобками в REDUCE при использовании их в качестве системных функций, являющихся также разновидностью операторов, позволит нам в дальнейшем с легкостью

проводить исследование вариаций различных геометрических факторов и действующих нагрузок в различных задачах (см. пп. 2.3.3, 2.3.4 и т.п.).

Оператор следует описать, т.е. указать системе REDUCE, что данное имя со скобками должно ниже обрабатываться как оператор. Для описания операторов используется команда OPERATOR со списком имен, которые далее будут обозначать операторы. Аргументы и скобки у этих имен при описании указывать не надо. Описание каждого оператора проводится один раз в любом месте программы, но до его использования. Например, чтобы считать операторами приведенные в п. 1.3 имена со скобками, нужно написать:

```
OPERATOR Q, F, FY;
```

Если вышеприведенный оператор отсутствует и в программе встречается никак ранее не описанное имя со скобками, например F(X), то в пакетном режиме при работе с файлами (см. п. 1.4) такое имя автоматически описывается как оператор. При этом выдается сообщение:

***F declared operator

В интерактивном режиме в подобном случае выдается вопрос:

Declare F operator? (Y or N)

на который необходимо ответить, набрав либо Y (да), либо N (нет) и нажать клавишу <Enter> – <Ввод>.

При этом ***никакого терминатора после букв Y или N не ставится***, поскольку они являются ответами на вопрос, а не предложениями программы.

В случае утвердительного ответа имя F описывается, как имя оператора, в случае отрицательного — выражение с F(X) игнорируется и выдается сообщение об ошибке в указанном предложении.

1.3.2. Системные функции

В систему REDUCE встроены следующие функции: SIN, COS, TAN, SINH, COSH, TANH, соответствующие тригонометрическим и гиперболическим функциям, и обратные им функции ASIN, ACOS, ATAN, ASINH, ACOSH, ATANH, а также EXP, LOG, SQRT, и некоторые другие (COT, DILOG, ERF, EXPINT).

В качестве своих аргументов эти функции могут использовать только скалярные выражения. Системе известны только элементарные соотношения и свойства этих функций:

COS(–X) = COS(X)
COS(N*PI) = (–1)**N
LOG(E) = 1
LOG(1) = 0
LOG(E**X) = X
SQRT(–X) = I*SQRT(X)
SQRT(200) = 10*SQRT(2)

SIN(–X)= –SIN(X)
SIN(N*PI) = 0
E**(I*PI/2) = I
E**(I*PI) = –1
E**(3*I*PI/2) = –I
SQRT(A**2)) = A

и другие.

Обычно в системе REDUCE происходит замена выражения SQRT(A**2) на A, которая вполне оправдана, но может повлечь за собой трудности, если выражению A будет случайно присвоено отрицательное значение. Функция извлечения квадратного корня имеет имя SQRT или может быть задана при помощи возведения в степень **(1/2). При выводе печатаются неупрощенные функции извлечения квадратного корня SQRT.

Если установлен режим ON NUMVAL, то функции ACOS, ASIN, COS, EXP, LOG, SIN, SQRT, TAN с числовыми аргументами принимают численные значения в форме с плавающей точкой и текущей степенью точности.

1.3.3. Переменные с индексами. Массивы и оператор ARRAY

Массив — это упорядоченная последовательность величин, называемых элементами массива, обозначаемая одним символическим именем. Каждый элемент массива определяется именем массива и его положением в массиве, т.е. значениями индексов. Индексы заключаются в круглые скобки и разделяются запятыми, причем для двумерного массива первый индекс определяет номер строки, второй — номер столбца. При обозначении массивов пользуются теми же правилами задания типа, что и при обозначении переменных.

Следует отличать размерность массива от его размера:

- ***размерность*** — число измерений (т.е. число индексов, определяющих положение элемента в массиве);
- ***размер*** — число его элементов.

Массивы, используемые в программе, должны быть обязательно описаны. Описание массивов дается в начале программы и содержит информацию об именах, размерностях и максимальных размерах

используемых массивов. Для описания массивов обычно используется оператор ARRAY, общий вид которого:

```
ARRAY имя_массива
(верхние границы изменения индексов через запятую), ...
```

Индексы в массиве изменяются от 0 до заданной величины. В момент объявления ARRAY все элементы массива обнуляются.

Таким образом, *переменные, определяющие элементы массива, всегда являются* ***связанными***, а значение элемента массива не может совпадать с его именем. В качестве границ массива можно использовать целое положительное выражение.

Не накладывается никаких ограничений на число переменных в одном предложении ARRAY. Он относится к неисполняемым операторам в том смысле, что не порождает команд в программе: по его описанию выделяется место в памяти для размещения всех указанных массивов. Так, после выполнения предложения

```
ARRAY B(6), A(6,6);                                    (1.18)
```

в программе могут быть использованы переменные (элементы массива) B(0), B(1), ..., B(6) и A(0,0), A(0,1), ..., A(0,6), A(1,0), A(1,1), ..., A(1,6), A(2,0), ..., A(5,6), A(6,0), ..., A(6,6).

Оператор ARRAY может появиться в любом месте программы. Если символ объявлен именем массива, он не может быть именем оператора или процедуры. Однако при желании этот символ может быть переобъявлен массивом иной размерности. Отменить описание массива можно командой CLEAR, в качестве параметров которой указываются только имена массивов (без скобок с числами):

```
CLEAR A, B;                                            (1.19)
```

Теперь все значения элементов массивов A и B стираются и перестают занимать память ПК. Сами имена A и B можно использовать для любых целей, в том числе и для обозначения новых массивов другой длины (см. (1.20)).

1.3.4. Работа с матрицами. Операторы MATRIX и MAT

Вместо термина “двумерный массив” часто употребляется термин “матрица”. Для работы с матрицами используется оператор MATRIX. С его помощью любой идентификатор может быть объявлен матричной переменной, которая обозначает всю матрицу целиком. Размерность матрицы

может быть явно задана в описании MATRIX после имени матричной переменной.

Если размерность неизвестна заранее, ее можно задать неявно, присваивая матричной переменной некоторое матричное выражение. Например, предложение

```
N:=9; MATRIX A(N,N), B(N,1), X(N,1), BB;          (1.20)
```

присваивает переменной N значение и вводит:

- двумерную матрицу A с числом строк N и столбцов N=9, для которой обеспечивается резервирование памяти для хранения 9×9=81 значения элементов A,
- матрицы-столбцы B и X, содержащих по N=9 элементов каждый,
- матрицу BB, размерность которой определится при присвоении ее элементам каких-либо значений.

В начальном состоянии все элементы некоторой матрицы, размерность которой была описана в операторе MATRIX, равны 0. В результате ни один из матричных элементов не может означать сам себя.

Обратим ваше внимание, что индексы у матрицы при использовании операторы MATRIX изменяются от 1 до заданной величины (а не от 0, как у оператора ARRAY), поэтому его удобнее применять при работе с матрицами.

В операторе MATRIX первое число дает количество строк, а второе количество столбцов.В задачах статики число строк (число уравнений равновесия N), всегда равняется числу столбцов (количеству неизвестных в задаче), поэтому матрица всегда квадратная A (N,N):

```
A(1,1), A(1,2), A(1,3), ..., A(1,N)
A(2,1), A(2,2), A(2,3), ..., A(2,N)
A(3,1), A(3,2), A(3,3), ..., A(3,N)                (1.21)
.................................
A(N,1), A(N,2), A(N,3), ..., A(N,N)
```

Первый индекс обозначает номер строки элемента, а второй — номер столбца. Например, элемент A(2,3) находится во второй строке в третьем столбце.

Замечание. Если матрица введена посредством оператора MATRIX, то ссылаться на отдельный элемент этой матрицы или на матрицу можно только после представления размерности матрицы и представления значений ее элементов выполнением матричного присваивания.

Присвоить элементам матрицы некоторые значения можно двумя способами. Рассмотрим их на примере матрицы A(3,3), у которой только

элементы главной диагонали отличны от нуля: A(1,1)=1; A(2,2)=2; A(3,3)=3. Для экономии места распечатку представим построчно, а не каждый элемент в отдельной строке, как на экране дисплея.

1. По первому из них ***присвоение производится только ненулевым элементам*** с помощью операторов присваивания. При этом нулевые значения элементов можно не указывать, так как *все элементы матрицы, размерность которой была описана, равны 0.* Поэтому достаточно только задать ненулевые элементы:

   matrix a(3,3); a(1,1):=1; a(2,2):=2; a(3,3):=3; a; (1.22)

```
mat(1,1) := 1   mat(1,2) := 0   mat(1,3) := 0
mat(1,1) := 0   mat(1,2) := 2   mat(1,3) := 0          (1.23)
mat(1,1) := 0   mat(1,2) := 0   mat(1,3) := 3
```

2. Другой способ присвоения — это ***использование оператора MAT***, в котором ***элементы матрицы задаются в полном виде по строкам***, каждая из которых заключена во внутренние скобки**.**

   matrix a(3,3); a:=mat((1,0,0), (0,2,0), (0,0,3)); a; (1.24)

   Результаты выполнения предложения (1.24) полностью совпадают с (1.23), потому повторно не приводятся.

Отметим, что ***аргументы оператора MAT обязательно должны заключаться в скобки***, даже если представляется матрица, содержащая 1 столбец или 1 элемент.

Поэтому матрица-столбец задается в виде MAT ((X), (У)). Внутренние скобки нужны здесь, чтобы отличать этот случай от случая матрицы-строки, которая задается в виде MAT ((X,У)).

Матричные выражения. Весьма впечатляющей особенностью системы REDUCE является та легкость, с которой могут быть выполнены матричные вычисления.

Они *следуют обычным правилам матричной алгебры.* Поэтому для выполнения нужных действий следует просто написать соответствующее алгебраическое выражение с использованием матричных переменных.

Это позволяет легко решать системы линейных алгебраических уравнений в численном и символьном виде (см. п. 2), а также проводить любые преобразования с использованием матричных переменных [24, с. 140-144].

При этом следует помнить, что в суммах и произведениях матричных выражений размерности должны быть приведены в соответствие.

1.4. Команды работы с файлами. Создание, редактирование и использование файлов

Система REDUCE создавалась как интерактивная система. Однако она может работать и в пакетном режиме с предварительно созданными файлами редьюсовских программ.

Это удобнее по многим причинам:

- программы можно предварительно внимательно проверить;
- результат их выполнения также можно записать в файл на диск;
- после первоначального выполнения можно неоднократно доводить программу до правильного решения, внося необходимые изменения;
- создание и редактирование файлов можно выполнять с помощью практически любого текстового редактора, которые гораздо удобнее встроенного "редьюсовского";
- после доведения решения до правильного можно исследовать влияние вариаций различных факторов на базе легко получаемых копий этого файла.

Для работы с файлами в системе REDUCE имеется несколько команд: OUT, SHUT и IN.

1.4.1. Команды OUT и SHUT

В качестве параметра команды OUT может быть идентификатор единственного файла, заключенный в кавычки в большинстве версий REDUCE:

```
OUT "C:\REDUCE\PROG\PR3.LIS";                    (1.25)
```

Эта команда дает указание системе, что с места ее расположения в программе ***все результаты следует направлять в указанный файл*** *C:\REDUCE\PROG\PR3.LIS до тех пор, пока другая команда OUT не изменит файл вывода или команда SHUT не закроет его.*

Напомним, что идентификатор файла полностью описывает расположение файла на диске и имеет следующий формат:

```
[диск:][путь\]имя_файла[.расширение]             (1.26)
```

Здесь элементы, заключенные в квадратные скобки, являются необязательными. В случае, если в описании не указывается дисковод или путь, то подразумевается текущий дисковод или текущий каталог.

Идентификатором находящегося в нем файла ИДФ будет только имя файла с его расширением (если оно имеется).

Последняя возможность очень удобна. Поэтому для файла, находящегося в текущем каталоге, предложение (1.25) примет вид:

```
OUT "PR3.LIS";                                    (1.27)
```

Одновременно для вывода могут быть открыты несколько файлов. Однако вывод будет производиться только в файл, открытый последним:

```
OUT "FR1.L"; {команды REDUCE};
OUT "FR2.L"; {команды REDUCE};
OUT T; {команды REDUCE};
OUT "FR1.L";
```

- после команды OUT "FR1.L"; *вывод будет происходить* в файл "FR1.L";
- после OUT "FR2.L"; — в файл "FR2.L";
- по OUT T; —направляться на терминал (дисплей);
- после второй команды OUT "FR1.L" вывод будет происходить в конец файла "FR1.L".

Результаты, посылаемые в файл, будут иметь такой же вид, какой они имели бы на экране дисплея. По умолчанию в стандартных математических обозначениях (установлен флаг ON NAT;). Например, X**2 будет записано как X^2.

Если целью вывода является *сохранение результатов для их дальнейшего использования* в системе REDUCE, то такая форма является неприемлемой.

Поэтому мы должны перед созданием файла установить переключатель в состояние OFF NAT.

После этого в конце любого выражения в качестве ограничителя появится знак "$", а сама информация будет записана в виде, позволяющем использовать ее в системе REDUCE для последующего ввода.

Далее в конце файла следует ввести слово END командой

```
WRITE ";END"$
```

что является стандартным способом заканчивать файлы, предназначенные для считывания.

При любой форме вывода результатов все открытые файлы следует закрыть командой SHUT, после которой в кавычках пишется список идентификаторов файлов, которые были открыты командой OUT. По команде SHUT эти файлы будут закрыты:

```
SHUT "PR3.LIS";                                    (1.28)
```

При выходе из системы REDUCE с файлом, незакрытым командой SHUT, может произойти потеря информации.

Если же файл закрыт командой SHUT и выдается команда OUT для этого файла, то он стирается, а затем записывается новый выходной файл. Попытки закрыть файлы, которые не были открыты командой OUT, приводят к возникновению ошибки.

При работе в интерактивном режиме при обнаружении неверно записанного выражения (синтаксической ошибки) выдается сообщение об ошибке, поэтому пользователь имеет возможность ввести вместо неправильно записанного предложения правильное.

В пакетном режиме при обнаружении ошибки дальнейшее выполнение программы сводится лишь к проверке на правильность записи (синтаксическому анализу), но никаких вычислений после этого не производится.

Пример. Последовательность команд

off nat; out "vivf.l"; k:=(c+d)4; write ";end";**

shut "vivf.l"; on nat;

позволит сформулировать выводной файл VIVF.L, который будет содержать запись информации в виде, позволяющем использовать ее для последующего ввода:

```
k := c**4 + 4*c**3*d + 6*c**2*d**2 + 4*c*d**3 + d**4$
;end$
```

Отметим, что сохранение результатов для их дальнейшего использования может быть также выполнено в синтаксисе Фортрана с помощью флага FORT (см. дополнения 3.2 и 3.3).

1.4.2. Команда IN

В качестве аргументов этой команды могут быть допустимые имена файлов.

```
IN "PR3","C:\REDUCE\PROG\PR4";                     (1.29)
```

По этой команде в систему будет загружен файл PR3, а затем C:\REDUCE\PROG\PR4.

Если команда IN ограничена символом ";", то содержимое файла отображается на экране или в выходном файле. Чтобы предотвратить

указанное отображение, необходимо использовать либо ограничитель "$" в IN, либо во входном файле команду

```
OFF ECHO;
```

Файлы, считываемые по команде IN, обычно завершаются конструкцией

```
;END;                                                    (1.30)
```

Знак "точка с запятой" перед словом END используется для большей надежности, хотя его можно и не указывать. Он запускает специальный учет контроля за файлами, улучшающий эффективность работы системы, а также защищает от некоторых ошибок при использовании сложных составных операторов. Если же слово END пропущено, то выдается сообщение об ошибке.

Замечание. Работать в текущем каталоге очень удобно: все команды открытия (1.27), закрытия (1.28) или работы с файлами (1.29) предстают в своей самой простой форме.

Однако текущим каталогом при работе с системой REDUCE будет сам подкаталог REDUCE, ибо из него в начале всегда запускается сама система. Работать в подкаталоге в окружении системных файлов очень неудобно, ибо всегда есть опасность удалить или запортить один из них с непредсказуемыми последствиями.

Поэтому в дальнейшем *будем организовывать для работы подкаталог в корневом каталоге текущего диска*, например \P–R, в котором будут располагаться сами программы и результаты их работы. В этом случае предложения (1.27), (1.28) и (1.29) соответственно примут вид (1.31) – (1.33):

```
OUT "\P-R\PR3.LIS";                                      (1.31)
SHUT "\P-R\PR3.LIS";                                     (1.32)
IN "\P-R\PR3","\P-R\PR4";                                (1.33)
```

Пример. Чтобы команды (1.3′) – (1.10′) из п. 1.1 выполнить из внешнего файла, например SR1, расположенного в подкаталоге \P–R корневого каталога текущего диска, а результат их выполнения записать на диск в файл SR1.LIS в том же подкаталоге, нужно:

1. Создать в корневом каталоге текущего диска (где находится и подкаталог \REDUCE с расположенными в нем системными файлами reduce.exe или reduce.bat) подкаталог с нужным именем, например, P–R. Для этого в корневом каталоге текущего диска следует нажать функциональную клавишу <F7> и в выводимой на экран строке следует указать имя нового подкаталога (например, P–R). После нажатия клавиши <Enter> в оглавлении текущего каталога

появится новый пустой подкаталог с указанным именем P–R. После этого нужно совместить курсор с его именем и нажать клавишу <Enter> (<ВВОД>). Norton Commander (Volkov Commander или другая аналогичная оболочка) "войдет" в этот подкаталог.

2. В подкаталоге \P–R корневого каталога текущего диска создать в любом текстовом редакторе новый файл с нужным именем, например SR1. Для этого необходимо при любом положении курсора нажать клавиши <Shift>+<F4> (<Верх>+<Ф4>). Затем в появившемся на экране приглашении набрать имя файла (например, SR1) и нажать клавишу <Enter>. Если файл с таким именем существует, то он вызывается на редактирование, если нет — то откроется пустой файл;
3. Начать набор исходных данных:
   - в первой команде OUT в качестве параметра указать идентификатор открываемого для вывода файла результатов \P-R\SR1.LIS (например, OUT "\P-R\SR1.LIS";);
   - далее в удобной форме (в одной или нескольких строках) набрать последовательность команд (1.3') — (1.10') из п. 1.1 (выделенных жирным шрифтом);
   - в предпоследней команде SHUT в качестве параметра указать идентификатор закрываемого файла результатов \P-R\SR1.LIS (например, SHUT "\P-R\SR1.LIS";);
   - в конце следует поместить команду END.
4. После окончания ввода команд, проверки набранной информации и исправления ошибок, ее необходимо сохранить. Для этого нужно нажать клавишу <F2> (<Ф2>) для записи на диск с именем редактируемого файла \P-R\SR1.
5. Для выхода из режима редактирования нужно нажать клавишу <F10> (<Ф10>). На экране опять появляются панели Norton Commander.

   В результате файл SR1 примет, например, следующий вид:

```
OUT "\P-R\SR1.LIS"; TT:=(X+Y*X/2+X*Z/3)**3; OFF ALLFAC;
TT; TTD:=DF(TT,X); ON ALLFAC; TTI:=INT(TTD,X);      (1.34)
ON FORT; TTI; OFF FORT; SHUT "\P-R\SR1.LIS"; ;END;
```

После этого нужно выйти из подкаталога \P-R и перейти в подкаталог \REDUCE с расположенными в нем системными файлами. Затем следует загрузить систему, для чего нужно запустить на выполнение файл reduce.exe или reduce.bat, совместив с их именами подсветку курсора и нажав клавишу

<Enter> (<Ввод>). После появления приглашения в командной строке следует ввести:

1: **in "\P-R\SR1"$** (1.35)

По этой команде в систему будет загружен файл \P-R\SR1, выполнены все находящиеся в нем команды (1.34), а результаты их выполнения будут записаны на диске в файле SR1.LIS в том же подкаталоге \P-R текущего диска.

Если в качестве ограничителя оператора IN (1.35) используется:

- ***точка с запятой,*** то операторы, содержащиеся в файле (для нашего примера \P-R\SR1), вместе с результатами их работы записываются в текущий файл вывода (\P-R\SR1.LIS);
- ***знак $,*** то сами операторы файла ввода (\P-R\SR1) не распечатываются, а в файле вывода (\P-R\SR1.LIS) записываются только результаты их работы.

В обоих случаях для просмотра результатов работы программы из файла \P-R\SR1 (1.34):

- следует выйти из сеанса работы с REDUCE по команде **bye;**
- выйти из подкаталога \REDUCE и перейти в подкаталог \P–R корневого каталога текущего диска;
- совместить подсветку курсора с именем файла результатов SR1.LIS и нажать клавишу <F3> (<Ф3>).

Результаты работы программы будут иметь для нашего примера следующий вид, соответственно совпадающий с выражениями из п. 1.1 (1.4′), (1.6′) — (1.9′).

tt := $(x^3*(27*y^3 + 54*y^2*z + 162*y^2 + 36*y*z^2 + 216*y*z + 324*y + 8*z^3 + 72*z^2 + 216*z + 216))/216$

$(27*x^3*y^3 + 54*x^3*y^2*z + 162*x^3*y^2 + 36*x^3*y*z^2 + 216*x^3*y*z + 324*x^3*y + 8*x^3*z^3 + 72*x^3*z^2 + 216*x^3*z + 216*x^3)/216$

ttd := $(27*x^2*y^3 + 54*x^2*y^2*z + 162*x^2*y^2 + 36*x^2*y*z^2 + 216*x^2*y*z + 324*x^2*y + 8*x^2*z^3 + 72*x^2*z^2 + 216*x^2*z + 216*x^2)/72$

tti := $(x^3*(27*y^3 + 54*y^2*z + 162*y^2 + 36*y*z^2 + 216*y*z + 324*y + 8*z^3 + 72*z^2 + 216*z + 216))/216$

```
ans = (x**3*(27.*y**3+54.*y**2*z+162.*y**2+36.*y*z**2+
. 216.*y*z+324.*y+8.*z**3+72.*z**2+216.*z+216.))/216.
```

Замечание. Иногда при дальнейшем использовании файла в интерактивном режиме нам требуется ввести дополнительные данные (например, присвоить значения некоторой переменной или ввести некоторую функцию) или провести некоторые вычисления до завершения программы.

Тогда в соответствующем месте файла (программы) нужно вставить команду PAUSE. При последующем использовании этого файла система останавливает выполнение программы и печатает на терминале сообщение CONT? .

Если мы хотим *продолжить* выполнение программы, то следует ввести

y

(соответствует английскому слову yes).

Для временной *остановки* выполнения программы следует напечатать

n

(соответствует английскому слову no).

Во время этой остановки пользователь может вводить с терминала нужные команды и выражения, а также выполнять требуемые вычисления, после чего следует ввести команду

cont;

и выполнение программы продолжится.

Если вы не испытываете затруднений при общении с компьютером, то можно сразу перейти к изучению п. 2. В противном случае быстрый старт несколько затянется и вам следует подробнее изучить нужные места второй части.

ГЛАВА 2. РЕШЕНИЕ СИСТЕМ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ (СЛАУ) В СИСТЕМЕ REDUCE

Любую задачу статики по определению реакций опор после составления уравнений равновесия можно записать в матричной форме в виде системы линейных алгебраических уравнений (СЛАУ):

$$(A)(X) = B, \qquad (2.1)$$

где A или A(I,J) — заданная квадратная матрица коэффициентов перед неизвестными, в которой выписаны только отличные от нуля значения элементов матрицы A(I,J);

X или X(I,1) — неизвестная матрица-столбец с N компонентами (усилия в стержнях, опорные реакции, неизвестные силы и т.п.);

B или B(I,1) — заданная матрица-столбец с N компонентами — правые части уравнений равновесия, в которые должны быть перенесены все свободные члены уравнений, не содержащие неизвестных.

Число строк матрицы A, т.е. число уравнений равновесия N, равняется числу столбцов, т.е. количеству неизвестных в задаче. Поэтому в задачах статики матрица A всегда квадратная и ее размерность A(N,N).

Важной особенностью СЛАУ задач статики является их сильно разреженный характер, т.е. наличие в их составе большого количества нулевых элементов. Поэтому, хотя в полном виде уравнение (2.1) имеет вид

$$\begin{aligned} &A_{11}X_1 + A_{12}X_2 + \ldots + A_{1N}X_N = B_1, \\ &A_{21}X_1 + A_{22}X_2 + \ldots + A_{2N}X_N = B_2, \qquad (2.2) \\ &\ldots\ldots\ldots\ldots\ldots\ldots \\ &A_{N1}X_N + A_{N2}X_2 + \ldots + A_{NN}X_N = B_N. \end{aligned}$$

но в уравнениях задач статики выписаны только ненулевые элементы.

Возможность вычислять в REDUCE обратную матрицу позволяет в общем виде решать системы линейных уравнений. Тогда решение системы (2.1) будет выглядеть так:

$$X = (A)^{-1}(B).$$

Заметим, что отдельное вычисление обратной матрицы 1/A и потом умножение ее справа на матрицу-столбец B производится системой гораздо медленнее, чем непосредственное вычисление выражения (1/A)*B, так как в этом случае система в целом обрабатывает гораздо меньшее количество элементов.

2.1. Этапы решения СЛАУ и структура программ

Для решения на ПК любой СЛАУ, возникающей в задачах статики, ее сначала нужно формализовать, приведя к виду (2.2).

Это достигается постановкой в соответствие неизвестным в задаче (усилиям в стержнях, опорным реакциям, неизвестным силам и т.п.) определенного элемента одномерного массива X(I) (см. соответствие идентификаторов в п.п. 2.2, 2.3, 2.4).

Формализация помогает как правильному представлению полного вида матрицы A (2.2) для ее задания с помощью оператора MAT, так и заданию только ненулевых ее элементов с помощью операторов присваивания.

Вторым этапом следует численное или символьное определение всех коэффициентов при неизвестных в уравнениях равновесия и их свободных членов (не содержащих неизвестных). Последние переносятся в правую часть этих уравнений, образуя матрицу-столбец свободных членов B(I).

После выполнения 2-го этапа система уравнений любой задачи статики по определению реакций опор приобретает явный вид СЛАУ (2.2), в которой выписаны только отличные от нуля элементы.

Для дальнейшего решения с помощью ПК в системе имеются две возможности составления программ:

- ввод всех элементов матрицы A(I,J) и матрицы-столбца B(I,1) для использования оператора MAT, что требует представления полного вида матриц с помощью добавления соответствующего количества нулевых элементов;
- ввод только ненулевых элементов с указанием их расположения, что требует использования операторов присваивания.

Соответственно этому ***структура простейшей программы*** для решения задач статики в системе REDUCE состоит из следующих основных блоков:

1. Открытие командой OUT выводного файла с идентификатором ИДФ (1.26), куда будут записываться результаты работы программы (`OUT «ИДФ»;`).
2. Задание значений с помощью операторов присваивания используемым вспомогательным переменным и тригонометрическим функциям (`N:=6;`).
3. Включение или выключение соответствующих переключателей, устанавливающих нужные режимы печати и представления результатов:
   - включение режима FLOAT при работе с вещественной арифметикой (`ON FLOAT;`) или выключение его при использовании рациональных чисел (`OFF FLOAT;`);
   - запрещение печати нулевых значений при контрольной распечатке матриц A и B, что достигается включением флага NERO (`ON NERO;`).
4. Описание матричных переменных оператором MATRIX с явным заданием ее размерностей:

```
MATRIX A(N,N), B(N,1), X(N,1);
```

5. Ввод исходных данных одним из двух способов:
   - с помощью оператора MAT, в котором элементы матрицы задаются *в полном виде по строкам*, каждая из которых заключена во внутренние скобки:

```
A:=MAT(( ,..., ), (по строкам во внутренних скобках все
элементы через запятую), ( ,..., ));
B:=MAT(( ),...(по элементу во внутренних скобках),( ));
```

   - использование операторов присваивания для ввода *только ненулевых значений соответствующих элементов матрицы A(I,J) и матрицы-столбца B(I,1) с указанием их расположения* (см. дополнение 2.1).
6. Контрольная печать введенных исходных данных автоматически получается применением терминатора ";" при любом способе ввода и может быть в двух формах в зависимости от используемой версии REDUCE (по этой причине она в рассматриваемых примерах не приводится, но является очень важной для проверки правильности введенных в ПК исходных данных):
   - представление матриц с указанием номера соответствующего элемента и его значения по одному элементу в каждой строке (версии до REDUCE 3.3 включительно);

- распечатка значений элементов матриц в их естественной форме по строкам без указания номера соответствующего элемента (версия REDUCE 3.5), при этом не работает флаг NERO;
- разрешение печати нулевых значений матрицы-столбца результатов решения X, что достигается выключением флага NERO (`OFF NERO;`).

7. Непосредственное решение СЛАУ задачи статики и печать выходной информации (значений элементов матрицы-столбца X(I,1)), которая автоматически получается применением терминатора ";":
   `X:=A**(-1)*B;`
8. Закрытие командой SHUT выводного файла с ИДФ (1.26), куда записывались результаты работы программы (`SHUT «ИДФ»;`).
9. В конце следует ввести оператор END, что является стандартным способом заканчивать файлы, предназначенные для считывания (`END;` или для большей надежности `;END;`).

Для удобства изложения можно разбить все рассматриваемые типовые примеры по формальному признаку количества уравнений равновесия на три группы: с малым (до 6-ти — см. п. 2.2), средним (до 12-ти — см. п. 2.3) и большим (свыше 12-ти — см. п. 2.4) числом уравнений. Отметим, что достоинства использования операторов присваивания для ввода только ненулевых элементов резко возрастают с увеличением числа уравнений равновесия.

Рассмотрим решение задач статики с использованием CAB REDUCE на типовых примерах, приведенных в сборнике [22], которым присвоен 31-й вариант.

2.2. Решение задач статики на ПК в CAB REDUCE с "малым» количеством уравнений

2.2.1. Формализация уравнений равновесия

В качестве примера задачи статики с "малым" количеством уравнений равновесия воспользуемся аналитическим решением типового задания С-8 по определению усилий в шести стержнях пространственной шарнирно-стержневой конструкции для ССС [22, с. 52-54]. Его расчетная схема приведена на рис. 2.1.

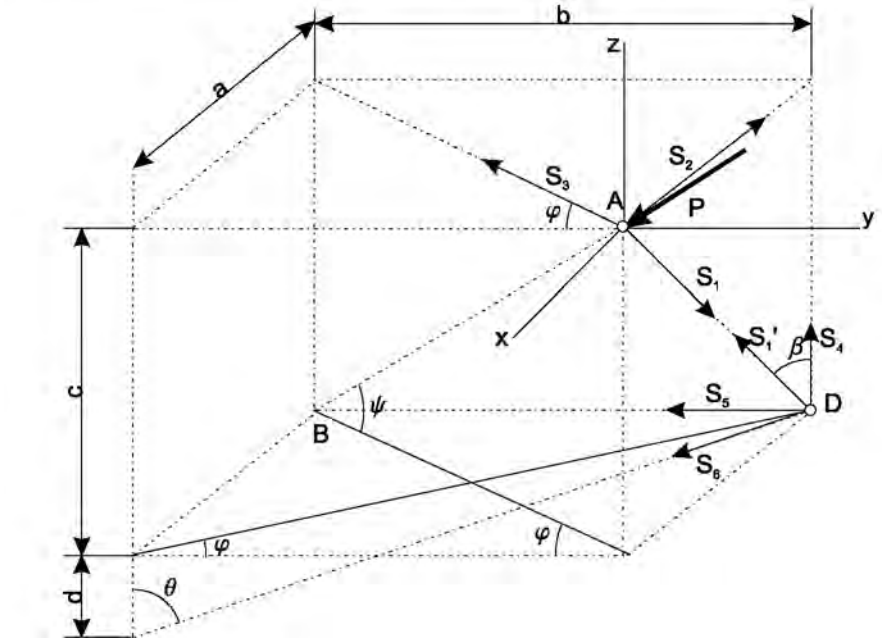

Рис. 2.1. Расчетная схема пространственной шарнирно-стержневой конструкции

Составим по три уравнения равновесия для сил, сходящихся соответственно в узлах A и D (они приведены под номерами (1)–(6) в пособии [22, с. 52-53]):

$$\begin{aligned}
&\sum X_i = 0; && -P*\cos\psi*\sin\varphi - S_1*\sin\beta - S_2 - S_3*\sin\varphi = 0,\\
&\sum Y_i = 0; && -P*\cos\psi*\cos\varphi - S_3*\cos\varphi = 0,\\
&\sum Z_i = 0; && -P*\sin\psi - S_1*\cos\beta = 0,\\
&\sum X_i = 0; && S_1'*\sin\beta + S_6*\sin\theta*\sin\varphi = 0,\\
&\sum Y_i = 0; && -S_5 - S_6*\sin\theta*\cos\varphi = 0,\\
&\sum Z_i = 0; && S_1'*\cos\beta + S_4 - S_6*\cos\theta = 0.
\end{aligned} \qquad (2.3)$$

Система уравнений равновесия (2.3) является системой линейных алгебраических уравнений, в которой выписаны только отличные от нуля значения элементов матрицы A(I,J) в неформализованном виде.

Рассмотрим подготовку исходных данных для решения этого примера на ПК в CAB REDUCE.

Отметим, что внутренние силы в первом стержне $\mathbf{S}_1$ и $\mathbf{S}_1'$ при мысленном его разбиении для рассмотрения равновесия узлов A и D становятся внешними для каждого из них, выражая действие отброшенной части. Они являются силами действия и противодействия, их модули равны друг другу и они противоположны по направлению. Этот факт выражают векторные равенства:

$$\mathbf{S}_1 = -\mathbf{S}_1'. \qquad (2.4)$$

Обычно векторные уравнения (2.4), выражающие соотношения равенства по модулю сил действия и противодействия и их противоположность по направлению, учитывают на рисунке. Для этого соответствующие вектора для штрихованных и не штрихованных величин направляют в противоположные стороны, как это показано для $\mathbf{S}_1$ и $\mathbf{S}_1'$ на рис. 2.1 и на рис. 53 [22, с.53].

Так как разность направлений штрихованных и не штрихованных величин уже учтена на рисунках (т.е. учтен знак минус в уравнении (2.4)), то этому векторному выражению (2.4) соответствует алгебраическое уравнение (2.5), учитывающее равенство значений этих величин вплоть до знака, так что им можно присвоить один идентификатор:

$$S_1 = S_1' = X_1 \qquad (2.5)$$

Эти равенства (2.5) используют при решении задачи. Такой подход мы рекомендуем применять всегда не только при определении усилий в стержнях ферм в п. 5.2, но и при рассмотрении равновесия систем тел при направлении внутренних сил в односторонних связях и промежуточных шарнирах (когда в узле сходятся не более двух стержней).

Соответствие идентификаторов для рассматриваемого типового примера с учетом вышеизложенного примет вид:

						(2.6)
X_1	X_2	X_3	X_4	X_5	X_6	
$S_1=S_1'$	S_2	S_3	S_4	S_5	S_6	

Для удобства составления программ заменим также используемые греческие буквы в обозначениях углов их идентификаторами, записанными в их латинской транскрипции:

				(2.7)
BETA	FI	TETA	PSI	
β	φ	θ	ψ	

Перенесем свободные члены, не содержащие неизвестных, в правые части уравнений (2.3), которые с учетом соответствия идентификаторов (2.6)–(2.7) теперь примут следующий формализованный вид (2.8):

```
-X1*sin(BETA) - X2 - X3*sin(FI) = P*cos(PSI)*sin(FI),
-X3*cos(FI) = P*cos(PSI)*cos(FI),
-X1*cos(BETA) = P*sin(PSI),                                    (2.8)
X1*sin(BETA) + X6*sin(TETA)*sin(FI) = 0,
-X5 - X6*sin(TETA)*cos(FI) = 0,
X1*cos(BETA) + X4 - X6*cos(TETA) = 0.
```

Теперь система уравнений (2.8) приобрела явный вид СЛАУ (2.2), в которой выписаны только отличные от нуля элементы. Для дальнейшего её решения или исследования с помощью ПК в системе REDUCE имеются различные возможности.

2.2.2. Решение в виде вещественных и рациональных чисел

Перед составлением программы для численного решения системы уравнений (2.8) в виде вещественных чисел по заданным размерам (a = 4 м, b = 5, c = 4, d = 1 м) вычисляем тригонометрические функции используемых углов [см. 22, с. 52-54]:

$$\sin\varphi = \sin(FI) = \frac{a}{\sqrt{a^2+b^2}} = \frac{4}{\sqrt{4^2+5^2}} = \frac{4}{\sqrt{41}} = 0.6247;$$

$$\cos\varphi = \cos(FI) = \frac{b}{\sqrt{a^2+b^2}} = \frac{5}{\sqrt{4^2+5^2}} = \frac{5}{\sqrt{41}} = 0.7809;$$

$$\sin\psi = \sin(PSI) = \frac{c}{\sqrt{a^2+b^2+c^2}} = \frac{4}{\sqrt{4^2+5^2+4^2}} = \frac{4}{\sqrt{57}} = 0.5298;$$

$$\cos\psi = \cos(PSI) = \frac{\sqrt{a^2+b^2}}{\sqrt{a^2+b^2+c^2}} = \frac{\sqrt{4^2+5^2}}{\sqrt{4^2+5^2+4^2}} = \frac{\sqrt{41}}{\sqrt{57}} = 0.8481;$$

$$\sin\beta = \sin(BETA) = \frac{a}{\sqrt{a^2+c^2}} = \frac{4}{\sqrt{4^2+4^2}} = \frac{1}{\sqrt{2}} = 0.7071; \qquad (2.9)$$

$$\cos\beta = \cos(BETA) = \frac{c}{\sqrt{a^2+c^2}} = \frac{4}{\sqrt{4^2+4^2}} = \frac{1}{\sqrt{2}} = 0.7071;$$

$$\sin\theta = \sin(TETA) = \frac{\sqrt{a^2+b^2}}{\sqrt{a^2+b^2+d^2}} = \frac{\sqrt{41}}{\sqrt{4^2+5^2+1^2}} = \frac{\sqrt{41}}{\sqrt{42}} = 0.988;$$

$$\cos\theta = \cos(TETA) = \frac{d}{\sqrt{a^2+b^2+d^2}} = \frac{1}{\sqrt{4^2+5^2+1^2}} = \frac{1}{\sqrt{42}} = 0.1543.$$

После численного определения всех коэффициентов при неизвестных и вычисления при значении P = 4 свободных членов (не содержащих неизвестных), уравнения (2.8) примут формализованный вид (2.10):

```
1. -0.7071*X1 - X2 - 0.6247*X3 = 2.1192,
2. -0.7809*  X3 = 2.6491,
3. -0.7071*X1 = 2.1192,                                    (2.10)
4. 0.7071*X1 + 0.6172*X6 = 0,
5. -X5 - 0.7715*X6 = 0,
6. 0.7071*X1 + X4 - 0.1543*X6 = 0.
```

Для удобства пользования уравнения пронумерованы сверху вниз, начиная с 1, где номером без скобок обозначается порядковый номер строки в системе уравнений (2.10).

Теперь система уравнений (2.10) приобрела явный вид СЛАУ (2.2), в которой выписаны в вещественной форме только отличные от нуля элементы. Поэтому квадратную матрицу A для ее представления в операторе MAT следует дополнить нулями до ее полного вида N×N:

```
0.7071,-1,-0.6247,0,0,0
0,0,-0.7809,0,0,0
-0.7071,0,0,0,0,0                                          (2.11)
0.7071,0,0,0,0,0.6172
0,0,0,0,-1,-0.7715
0.7071,0,0,1,0,-0.1543
```

Структура простейшей программы в системе REDUCE из п. 2.1 для решения системы уравнений (2.10) может быть реализована, например, сле-

дующим образом (по вопросам записи программы 2.1, например, в файле PR2-1 в подкаталоге \C8 корневого каталога текущего диска и запуске ее на выполнение см. п. 7.3 и пример в п. 1.4.2):

```
COMMENT Программа 2.1: решение типового примера С-8 на REDUCE
с заданием с помощью оператора MAT всех элементов матрицы A и
матрицы-столбца B с представлением данных ввода и результа-
тов в численном виде;
OUT "\C8\PR2-1.LIS";                                          10
ON FLOAT, NERO;                                               15
N:=6;                                                         20
MATRIX A(N,N), B(N,1), X(N,1);                                30
A:=MAT((-0.7071,-1,-0.6247,0,0,0),                            35
 (0,0,-0.7809,0,0,0),
 (-0.7071,0,0,0,0,0),
 (0.7071,0,0,0,0,0.6172),                                  (2.12)
 (0,0,0,0,-1,-0.7715),
 (0.7071,0,0,1,0,-0.1543));
B:=MAT((2.1192),(2.6491),(2.1192),(0),(0),(0));               55
OFF NERO;                                                     80
X:=A**(-1)*B;                                                 85
OFF FLOAT;                                                    90
SHUT "\C8\PR2-1.LIS";                                         95
;END;                                                         99
```

Номера справа операторов и команд проставлены только для удобства дальнейших пояснений. В программе на экране дисплея ***операторы не нумеруются***. Этот прием будет использоваться и в дальнейшем при описании программ.

Также соображения удобства пояснений определили расположение почти каждого оператора или команды программы 2.1 в отдельной строке.

Напомним, что их может быть:

- *несколько в одной строке записи* (см., например, оператор 15, что будет использоваться нами в дальнейшем при описании однотипных операторов);
- *каждый из них может занимать несколько строк*, при этом лишние пробелы игнорируются и никаких знаков переноса не используется (см., например, оператор 35).

Комментарий, помещенный между зарезервированным словом COMMENT и разделителем в виде знака “точка с запятой”, описывает назначение программы 2.1 и при обработке игнорируется. В комментариях также обычно указываются номера группы, задания и варианта, а также фамилия и инициалы автора данной программы.

Команда 10 открывает выводной файл PR2-1.LIS, находящийся в подкаталоге \C8 корневого каталога текущего диска, куда будут записываться результаты работы программы 2.1. Отметим, что имя файла и подкаталога для ваших программ может быть любым разрешенным идентификатором и должно отражать индивидуальность вашу (для файла) или группы (для подкаталога).

Команда 15 включает режим FLOAT для работы с вещественной арифметикой и запрещает печать нулевых значений при контрольной распечатке матриц A и B, что достигается включением флага NERO.

Оператор присваивания 20 задает значение используемой вспомогательной переменной N, равное количеству шести уравнений равновесия в данной задаче.

Оператор 30 описывает матричные переменные с явным заданием их размерностей:

- матрицу A с числом строк N и столбцов N=6, для которой обеспечивается резервирование памяти для хранения 6×6=36 значений элементов A;
- матрицы-столбцы B и X, содержащих по N=6 элементов каждый.

Ввод исходных данных матрицы A и матрицы-столбца B осуществляется с помощью операторов 35 и 55 соответственно. Элементы матриц в форме вещественных чисел задаются в полном виде по строкам, каждая из которых заключена во внутренние скобки (даже если она состоит из одного элемента), как это требуется для оператора MAT. Контрольная печать введенных исходных данных будет получена автоматически применением терминатора “;”.

Команда 80 разрешает печать нулевых значений перед решением СЛАУ, что достигается выключением флага NERO. Это сделано для возможности распечатки всех значений матрицы-столбца X.

Оператор 85 выполняет непосредственное решение СЛАУ задачи статики и печать выходной информации (значений элементов матрицы-столбца X(I,1)).

Команда 90 выключает режим работы с вещественной арифметикой, что достигается выключением флага FLOAT.

Команда 95 закрывает выводной файл PR2-1.LIS, находящийся в подкаталоге \C8 корневого каталога текущего диска, куда записывались результаты работы программы 2.1.

Оператор 99 запускает специальный учет контроля за файлами, улучшающий эффективность работы системы, что является стандартным способом заканчивать файлы, предназначенные для считывания. Знак "точка с запятой" перед словом END используется для большой надежности, хотя его можно и не указывать. В дальнейшем мы будем применять его только при использовании сложных составных операторов.

Результаты решения после окончания работы программы 2.1 (вместе со всей предусмотренной в ней печатью) окажутся в файле PR2-1.LIS, находящемся в подкаталоге \C8 корневого каталога текущего диска, и будут иметь для рассматриваемого примера следующие значения:

X(1,1) := – 2.99701	X(2,1) := 2.119203	
X(3,1) := – 3.39236	X(4,1) := 2.648988	(2.13)
X(5,1) := – 2.64898	X(6,1) := 3.43355	

Расположение элементов матриц на экране дисплея, как отмечалось в п. 2.1, будет зависеть от используемой версии REDUCE.

Дополнение 2.1. ***Положение элемента A(I,J) в матрице A характеризуется двойным индексом:***

- ***первый индекс I означает номер строки,*** *в которой стоит элемент A(I,J) – это порядковый номер уравнения равновесия;*
- ***второй индекс J означает номер столбца*** *– номер идентификатора Xj, при котором стоит элемент A(I,J).* Отметим, что нумерация строк производится сверху вниз, а столбцов — слева направо.

Для матрицы A ненулевыми элементами A(I,J) системы уравнений (2.10) будут:

- *для 1-й строки* (уравнение 1.) первый индекс I=1, второй индекс J совпадает с номером идентификатора, при котором стоит значение элемента A(1,J): A(1,1)=–0.7071 (J=1, так как значение этого элемента, равное –0.7071, стоит при идентификаторе X_1), A(1,2)=–1 (J=2, так как значение элемента A(1,2) стоит при идентификаторе X_2), A(1,3)=–0.6247 (J=3, так как стоит при X_3). Напомним, что все остальные элементы первой строки будут равны нулю: A(1,4)=A(1,5)=A(1,6)=0;

- *для 2-й строки* (уравнение 2.) первый индекс I=2, второй индекс совпадает с номером идентификатора, при котором стоит значение элемента A(2,J): A(2,3)=–0.7809 (значение стоит при X_2, значит второй индекс J=2);
- *для 3-й строки* (уравнение 3.): A(3,1)=–0.7071;
- *для оставшихся строк 4-6* аналогично: A(4,1)=0.7071, A(4,6)=0.6172, A(5,5)=–1, A(5,6)=–0.7715, A(6,1)=0.7071, A(6,4)=1, A(6,6)=–0.1543.

Напомним, что в каждой строке по N=6 элементов, но значения всех остальных невыписанных элементов равны нулю.

Положение элемента вектора-столбца B(I) характеризуется только номером I-й строки, *в которой стоит элемент B(I).* При представлении его в REDUCE в виде матрицы-столбца это примет вид B(I,1), где 1 – номер единственного первого столбца.

Для матрицы-столбца B ненулевыми элементами единственного первого столбца B(I,1) системы уравнений (2.10) будут:

- *для 1-й строки* I=1 и B(1,1)=2.1192;
- *для 2-й* I=2 и B(2,1)=2.6491;
- *для 3-й строки* I=3 и B(3,1)=2.1192.

В столбце также N=6 элементов, но остальные равны нулю.

Определив положение и значения всех ненулевых элементов матриц A и B, ввод исходных данных для работы программы 2.1 можно организовать без использования операторов MAT 35 и 55, в которых элементы матрицы A(I,J) и матрицы-столбца B(I,1) задаются в *полном виде* по строкам.

Применение операторов присваивания позволяет вводить только ненулевые значения элементов соответствующих матриц с указанием их расположения. При этом нулевые значения элементов можно не указывать, так как все элементы матрицы A(I,J) и матрицы-столбца B(I,1), размерность которых была описана оператором 30, равны 0.

Поэтому достаточно задать в программе 2.1 только ненулевые элементы:

```
A(1,1):=-0.7071; A(1,2):=-1; A(1,3):=-0.6247;          35
A(2,3):=-0.7809; A(3,1):=-0.7071; A(4,1):=0.7071;      40
A(4,6):=0.6172; A(5,5):=-1; A(5,6):=-0.7715;           45
A(6,1):=0.7071; A(6,4):=1; A(6,6):=-0.1543;            50
B(1,1):=2.1192; B(2,1):=2.6491; B(3,1):=2.1192;        55
```

Напомним, что при описании вариантов и дополнений к основным программам вставляемые операторы, команды или предложения должны рас-

полагаться в основной программе по порядку возрастания вспомогательных номеров, при этом:

- *если номер вставляемого оператора* ***совпадает*** с имеющимся в основной программе, то последний ***заменяется***;
- *если номер* **не совпадает** — то предложение **вставляется** между операторами с соответствующими меньшим и большим номерами в основной программе.

Дополнение 2.2. Если команду 80 представить в виде

```
OFF FLOAT, NERO;                                                    80
```

удалив из программы используемую здесь команду 90, то отмена установленного режима FLOAT для работы с вещественной арифметикой произойдет *до распечатки результатов решения*. Это приведет к тому, что введенные операторами 35 и 45 вещественные числа будут представлены в виде рациональных, о чем для каждого используемого значения числа будет выдано соответствующее сообщение. Результаты решения (2.13) будут также представлены в виде рациональных чисел, отчего их значение, конечно, не изменится:

$$X(1,1) := -\frac{447681}{149375} \qquad X(2,1) := \frac{7001719}{3303891}$$

$$X(3,1) := -\frac{26878}{7923} \qquad X(4,1) := \frac{3029791101}{1143748750} \qquad (2.14)$$

$$X(5,1) := -\frac{2649}{1000} \qquad X(6,1) := \frac{5298}{1543}$$

Однако проще не вычислять значения действительных чисел при подготовке исходных данных, а оставить их после определения тригонометрических функций в форме рациональной дроби и корней целых чисел, в которой они естественным образом определяются соотношениями (2.9). Программа 2.1 в этом случае может принять, например, следующий вид:

```
% Программа 2.2
OUT "\C8\PR2-2.LIS";                                                10
ON NERO;                                                            15
N:=6;                                                               20
SIN(FI):=4/SQRT(41); COS(FI):=5/SQRT(41);                           22
SIN(PSI):=4/SQRT(57); COS(PSI):=SQRT(41)/SQRT(57);                  24
```

```
SIN(BETA):=1/SQRT(2); COS(BETA):=1/SQRT(2);                    26
SIN(TETA):=SQRT(41)/SQRT(42); COS(TETA):=1/SQRT(42);           28
P:=4;                                                          29
MATRIX A(N,N),B(N,1),X(N,1);                                   30
A:=MAT((-SIN(BETA),-1,-SIN(FI),0,0,0),                         35
 (0,0,-COS(FI),0,0,0),
 (-COS(BETA),0,0,0,0,0),
 (SIN(BETA),0,0,0,0,SIN(TETA)*SIN(FI)),                  (2.15)
 (0,0,0,0,-1,-SIN(TETA)*COS(FI)),
 (COS(BETA),0,0,1,0,-COS(TETA)));
B:=MAT((P*COS(PSI)*SIN(FI)),(P*COS(PSI)*COS(FI)),              55
 (P*SIN(PSI)),(0),(0),(0));
OFF NERO;                                                      80
X:=A**(-1)*B;                                                  85
SHUT "\C8\PR2-2.LIS";                                          95
END;                                                           99
```

Команды и операторы 15, 20, 30, 80, 85 и 99 эквивалентны соответствующим командам и операторам программы 2.1 и повторно не описываются. Отметим отличия программ 2.2 и 2.1.

Команды 10 и 95 отличаются только именем внешнего файла PR2-2.LIS, соответственно открываемого и закрываемого этими командами, куда будут записываться результаты работы программы 2.2.

Команда 15 включает только флаг NERO, чем запрещается печать нулевых значений при контрольной распечатке матриц A и B.

Включение режима FLOAT для работы с вещественной арифметикой из команды 15 удалено, что делает ненужным и его выключение командой 90, которая также убрана из программы 2.2 (подробнее см. дополнение 2.4).

Новые операторы присваивания 22–28 выражают в программе 2.2 значения тригонометрических функций в форме рациональной дроби и корней целых чисел, в которой они естественным образом определяются соотношениями (2.9).

Оператор присваивания 29 задает значение силе P, которая будет использоваться в качестве связанной переменной в общей записи элементов матрицы-столбца B (правых частей уравнений равновесия (2.8)).

Операторы 35 и 55 вводят исходные данные соответственно матрицы A и матрицы-столбца B в символьной форме, элементы которых также задаются в полном виде по строкам сразу из формализованных уравнений (2.8).

Дополнение 2.3. Операторы присваивания 22–28, выражающие тригонометрические функции в программе 2.2, можно также представить в нижеследующей степенной форме:

```
SIN(FI):=4/41**(1/2); COS(FI):=5/41**(1/2);                22
SIN(PSI):=4/57**(1/2); COS(PSI):=41**(1/2)/57**(1/2);      24
SIN(BETA):=1/2**(1/2); COS(BETA):=1/2**(1/2);              26
SIN(TETA):=41**(1/2)/42**(1/2); COS(TETA):=1/42**(1/2);    28
```

Распечатка результатов решения после окончания работы программы 2.2 при обоих способах задания тригонометрических функций (в программе 2.2 и по дополнению 2.3) будет иметь следующий вид:

$$X(1,1) := -\frac{16 * \mathrm{sqrt}(2)}{\mathrm{sqrt}(57)} \qquad X(2,1) := \frac{16}{\mathrm{sqrt}(57)}$$

$$X(3,1) := -\frac{4 * \mathrm{sqrt}(41)}{\mathrm{sqrt}(57)} \qquad X(4,1) := \frac{20}{\mathrm{sqrt}(57)} \qquad (2.16)$$

$$X(5,1) := -\frac{20}{\mathrm{sqrt}(57)} \qquad X(6,1) := \frac{4 * \mathrm{sqrt}(42)}{\mathrm{sqrt}(57)}$$

Конечно, это только точная форма представлений приближенных вещественных значений результатов (2.13) в виде дробей из целых чисел и их корней.

Дополнение 2.4. Если в процессе вычисления встречается действительное (с плавающей точкой) число, то система обычно преобразует его в отношение двух целых чисел и выдает сообщение о проделанном преобразовании.

Если пользователь желает использовать действительную арифметику, он может воспрепятствовать этому преобразованию командой ON FLOAT, которая только запрещает преобразовывать число с плавающей точкой в рациональную дробь во время вычислений.

Поэтому если в команде 15 включить режим FLOAT для работы с вещественной арифметикой (как в программе 2.1)

```
ON FLOAT, NERO;                                    15
```

то никаких изменений в форме представлений результатов решения (2.16) программы 2.2 не будет, так как среди вводимых данных нет действительных или рациональных чисел.

Кроме этого, *CAB REDUCE предназначена для выполнения точных алгебраических преобразований.*

Поэтому *использование приближенных вещественных чисел при сложных вычислениях*, к которым относится и определение обратной матрицы при решении СЛАУ, *может привести к ошибкам.*

Однако использование действительной арифметики удобнее для представления результатов решения в численной форме.

Поэтому все расчеты следует проводить с использованием рациональных чисел, а результаты решения представить в приближенном виде с использованием действительных чисел.

Для этого после окончания расчета в программе 2.2 следует:

- установить флаги BIGFLOAT и NUMVAL командой 87;
- распечатать результаты решения выражением 88, при этом они предстанут в виде (2.13) с использованием действительных чисел (только вместо идентификатора X будет использоваться MAT);
- отменить установленные на время режимы командой 90:

```
ON BIGFLOAT, NUMVAL;                               87
X;                                                 88
OFF BIGFLOAT, NUMVAL;                              90
```

Вышеописанный результат достигается только при совместном действии флагов BIGFLOAT и NUMVAL, причем вместо флага BIGFLOAT *не может использоваться* флаг FLOAT.

Флаг BIGFLOAT обеспечивает использование в многочленах вещественных коэффициентов повышенной точности (по умолчанию в этом режиме точность вещественных чисел определяется десятью десятичными числами).

Флаг NUMVAL устанавливает режим вычисления значений элементарных функций. Функции SIN, COS, TAN, ASIN, ACOS, SQRT, EXP, LOG с числовым аргументом и зарезервированные переменные E и PI принимают численные значения в форме с плавающей запятой с текущей степенью точности.

Дополнение 2.5. Ввод исходных данных в программе 2.2 также можно организовать без операторов MAT 35 и 55 с использованием операторов присваивания для *ненулевых значений* элементов соответствующих матриц. В отличие от дополнения 2.1, ненулевые значения элементов здесь указываются с применением тригонометрических функций:

```
A(1,1):=-SIN(BETA); A(1,2):=-1; A(1,3):=-SIN(FI);          35
A(2,3):=-COS(FI); A(3,1):=-COS(BETA);                      40
A(4,1):=SIN(BETA); A(4,6):=SIN(TETA)*SIN(FI);              42
A(5,5):=-1; A(5,6):=-SIN(TETA)*COS(FI);                    45
A(6,1):=COS(BETA); A(6,4):=1; A(6,6):=-COS(TETA);          50
B(1,1):=P*COS(PSI)*SIN(FI);                                55
B(2,1):=P*COS(PSI)*COS(FI); B(3,1):=P*SIN(PSI);            60
```

Так как тригонометрические функции предварительно определены операторами 22–28 (в программе 2.2 и по дополнению 2.3), то при обоих способах их задания распечатка результатов решения (2.16) не изменится.

Программа 2.2 с учетом дополнения 2.5 представляет только другую форму численного решения СЛАУ (2.8) и этим не слишком сильно отличается от программы 2.1. Однако она уже гораздо лучше приспособлена для возможности исследования влияния на результаты решения вариации различных факторов: величины и направления действующих сил, а также геометрических параметров конструкции.

Дополнение 2.6. В CAB REDUCE можно записать и многократный оператор присваивания, формат которого в общем случае можно представить в следующем виде:

```
выражение := выражение := … := выражение                  (2.17)
```

В нем каждое выражение принимает значение, вычисленное для самого правого выражения.

Отметим, что в любой части оператора (2.17) вместо выражения может использоваться также переменная или элемент массива (см. его простую форму представления (1.12)).

Реализуя такую возможность, операторы 22–28 программы 2.2, задающие значения тригонометрических функций применяемых углов, можно представить в виде, облегчающем запись операторов MAT 35 и 55. Теперь вместо тригонометрических функций просто используются переменные, идентификаторы которых для простоты состоят из первых букв обозначений самой функции и используемого угла:

```
SF:=SIN(FI):=4/SQRT(41); CF:=COS(FI):=5/SQRT(41);        22
SP:=SIN(PSI):=4/SQRT(57);                                24
CP:=COS(PSI):=SQRT(41)/SQRT(57);                         25
SB:=SIN(BETA):=1/SQRT(2); CB:=COS(BETA):=1/SQRT(2);      26
ST:=SIN(TETA):=SQRT(41)/SQRT(42);                        27
CT:=COS(TETA):=1/SQRT(42);                               28
A:=MAT((-SB,-1,-SF,0,0,0),                               35
 (0,0,-CF,0,0,0),                                    (2.18)
 (-CB,0,0,0,0,0),
 (SB,0,0,0,0,ST*SF),
 (0,0,0,0,-1,-ST*CF),
 (CB,0,0,1,0,-CT));
B:=MAT((P*CP*SF),(P*CP*CF), (P*SP),(0),(0),(0));         55
```

Дополнение 2.7. Фрагмент программы на REDUCE дополнения 2.5, осуществляющий ввод ненулевых значений исходных данных в программе 2.2, также удобнее представить с использованием вспомогательных переменных из многократных операторов присваивания 22–28 дополнения 2.6, что может иметь следующий вид:

```
A(1,1):=-SB; A(1,2):=-1; A(1,3):=-SF;                    35
A(2,3):=-CF; A(3,1):=-CB;                                40
A(4,1):=SB; A(4,6):=ST*SF;                               42
A(5,5):=-1; A(5,6):=-ST*CF;                              45
A(6,1):=CB; A(6,4):=1; A(6,6):=-CT;                      50
B(1,1):=P*CP*SF;                                         55
B(2,1):=P*CP*CF; B(3,1):=P*SP;                           60
```

Дополнение 2.8. В системе REDUCE в простых случаях нет необходимости в команде вывода информации на терминал, поскольку значение любого выражения печатается автоматически при использовании ограничителя “;”. Однако в REDUCE для этой же цели введена специальная команда WRITE, которая снимает некоторые ограничения, связанные с использовании ограничителя “;” в некоторых сложных случаях. Формат команды следующий:

```
WRITE список параметров, разделенных запятыми;
```

В качестве параметров могут использоваться:

- ***выражения*** (включая переменные и константы), которые при выводе оцениваются и печатаются;

- ***операторы присваивания***, в которых выражение, стоящее в правой части, при выводе оценивается, операция присваивания выполняется и распечатывается вместе с левой частью;
- ***произвольный текст, заключённый в кавычки***, выводится в том же виде без кавычек на печать.

Все значения указанных параметров вместе с сопровождающим текстом печатаются одна за другой на данной строке. Для их отделения друг от друга в кавычках указываются пробелы.

Если количество выводимой информации превышает длину строки, то печать продолжается на другой строке. После оценивания команды WRITE печатаемая строка закрывается. Поэтому список параметров, состоящий только из одного пробела в кавычках (`WRITE " ";`), приведёт к пропуску пустой строки.

Программу 2.2 желательно дополнить поясняющими сообщениями, что можно сделать, например, следующим образом:

```
WRITE "ВХОДНАЯ МАТРИЦА ЛЕВОЙ ЧАСТИ А";                33
WRITE "ВЕКТОР ПРАВЫХ ЧАСТЕЙ В";                       53
WRITE "РЕЗУЛЬТАТЫ РЕШЕНИЯ СИСТЕМЫ УРАВНЕНИЙ";         68
```

В данном случае поясняющие сообщения команды `WRITE` можно заменить использованием комментария, например, для команды 33:

```
COMMENT ВХОДНАЯ МАТРИЦА ЛЕВОЙ ЧАСТИ А;                33
```

или

```
% ВХОДНАЯ МАТРИЦА ЛЕВОЙ ЧАСТИ А                       33
```

Отметим, что для выполнения WRITE, как и для любой *команды*, безразлично, какой после нее использован ограничитель «;» или «$»: в любом случае она *выполняется* и указанный список распечатывается.

2.2.3. Решение в символьном виде. Исследование влияния вариации нагрузки

Представляется исключительно интересным исследовать влияние параметров СЛАУ на результаты решения. Применительно к задачам статики это означает исследование влияния вариации нагрузки и геометрических факторов на значения реакций опор или усилий в стержнях конструкций. При численном решении на ПК с использованием алгоритмических языков для этого нужно проделать большую работу по подготовке исходных данных [17.]

CAB REDUCE поразительно легко справляется с подобной задачей. Это достигается использованием в процессе символьных вычислений *свободных переменных и свободных имен со скобками*.

Основанием для этого служит тот факт, что в REDUCE различают имя переменной и ее значение, даже если они совпадают.

Каждая переменная обозначается именем, которое является ее первоначальным значением.

Имя переменной — обозначение "емкости", которую с помощью операторов присваивания наполняют содержимым (численным или символьным значением).

***Свободной** называется переменная, которой ничего не присвоено и значение которой совпадает с ее именем.*

Связанной — которой ранее было присвоено некоторое значение.

Для свободных переменных алгебраические преобразования выполняются с их именами, которые и входят в окончательный ответ, определяя его зависимость от значения последних.

Всем использованным переменным ранее всегда предварительно задавались значения, то есть они являлись связанными.

Поэтому если из программы 2.2 удалить оператор 29, сделав свободной переменной величину нагрузки силу P, очистив для надежности ее значение командой

```
CLEAR P;                                                    29
```

то мы получим значения усилий в стержнях рассматриваемой пространственной конструкции в зависимости от значения силы P:

$$X(1,1):=-\frac{4*\mathrm{sqrt}(2)*p}{\mathrm{sqrt}(57)} \qquad X(2,1):=\frac{4*p}{\mathrm{sqrt}(57)}$$

$$X(3,1):=-\frac{\mathrm{sqrt}(41)*p}{\mathrm{sqrt}(57)} \qquad X(4,1):=\frac{5*p}{\mathrm{sqrt}(57)} \qquad (2.19)$$

$$X(5,1):=-\frac{5*p}{\mathrm{sqrt}(57)} \qquad X(6,1):=\frac{\mathrm{sqrt}(42)*p}{\mathrm{sqrt}(57)}$$

Их графики представляют собой прямые линии, выходящие из начала координат, с тангенсами углов наклона, равными коэффициентам перед силой P в соотношениях (2.19).

Еще более впечатляющий результат достигается использованием свободных имен со скобками. В REDUCE основным свойством оператора является то, что само его имя вместе со списком параметров, может быть его значением. Иными словами, *имя со скобкой, при использовании его в качестве оператора может условно рассматриваться в качестве свободной переменной.*

Это свойство свободных имен со скобками при использовании их в качестве системных функций, являющихся разновидностью операторов, позволяет нам с легкостью проводить исследование вариаций различных геометрических факторов и направлений действующих нагрузок в различных задачах.

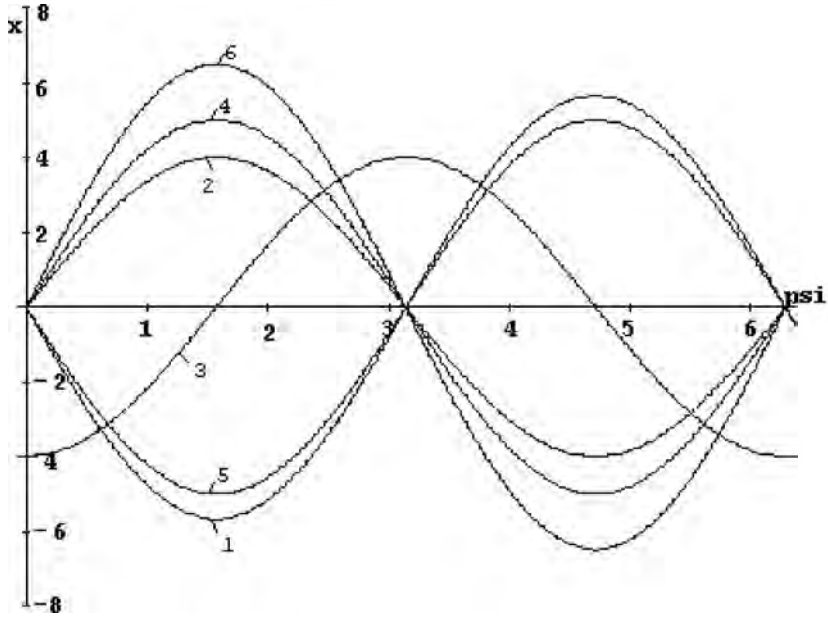

Рис. 2.2. Зависимости усилий (X, кН) в стержнях 1-6 пространственной конструкции (рис. 2.1) от изменения угла PSI (рад) при вращении силы P в диагональной плоскости: 1 – S_1, 2 – S_2, 3 – S_3, 4 – S_4, 5 – S_5, 6 – S_6

В программе 2.2 операторы 24 задают значения SIN(PSI) и COS(PSI), которые определяют направление действия силы P в диагональной плоскости. Если восстановить в программе 2.2 оператор 29, а удалить операторы 24 (очистив для надежности их значения командой CLEAR), то имена со скобками SIN(PSI) и COS(PSI), идентифицирующие тригонометрические функции, можно условно рассматривать в качестве свободных переменных. В результате работы такой измененной программы мы получим значения

усилий в зависимости от вариации направления постоянной по модулю силы P в диагональной плоскости:

```
X(1,1):= – 4*sqrt(2)*sin(psi)   X(2,1):= 4*sin(psi)
X(3,1):= – 4*cos(psi)           X(4,1):= 5*sin(psi)          (2.20)
X(5,1):= – 5*sin(psi)           X(6,1):= sqrt(42)*sin(psi)
```

Графики на рис. 2.2 наглядно показывают зависимости усилий (X, кН) в стержнях 1-6 пространственной конструкции (рис. 2.1) от изменения угла PSI (рад) при вращении силы P в диагональной плоскости. Как видно из их сравнения, значения усилий в стержнях пространственной конструкции при вращении постоянной по модулю силы P в диагональной плоскости изменяются по синусоидальному закону:

В программе 2.2 операторы 22 задают значения SIN(FI) и COS(FI), которые определяют направление действия силы P при ее вращении вокруг вертикальной оси с постоянным углом PSI.

Если восстановить в программе 2.2 операторы 24, а удалить операторы 22, то в результате ее работы мы получим зависимости результатов решения от изменения направления постоянной по модулю силы P при ее вращении вокруг вертикальной оси с постоянным углом наклона к горизонтальной плоскости:

$$X(1,1):=-\frac{16*\mathrm{sqrt}(2)}{\mathrm{sqrt}(57)} \qquad X(2,1):=\frac{16}{\mathrm{sqrt}(57)}$$

$$X(3,1):=-\frac{4*\mathrm{sqrt}(41)}{\mathrm{sqrt}(57)} \qquad X(4,1):=\frac{16*(\mathrm{sqrt}(41)*\sin(fi)+1)}{\mathrm{sqrt}(2337)*\sin(fi)} \quad (2.21)$$

$$X(5,1):=-\frac{16*\cos(fi)}{\mathrm{sqrt}(57)*\sin(fi)} \qquad X(6,1):=\frac{16*\mathrm{sqrt}(42)}{\mathrm{sqrt}(2337)*\sin(fi)}$$

Простота выполнения вариаций различных факторов открывает такие широкие возможности, которые могут просто погубить легкомысленного исследователя, о чем в аналогичной ситуации предупреждал еще Бармалея всем известный добрый доктор в фильме «Айболит-66». Применительно к нам это означает, что начав исследовать влияние вращения силы P путем вариации угла φ (идентификатором которого в уравнениях 2.21 является FI), мы не обратили внимание, что он также определяет положение третьего стержня конструкции (см. рис. 53, [22, с. 53]), являясь углом между диагоналями верхней и нижней плоскости и горизонтальной прямой.

Поэтому при вращении силы P мы также *несогласованным образом* стали изменять размеры конструкции. При $\varphi = 0$ или $\varphi = \pi$ *пространствен-*

ная шарнирно-стержневая конструкция вырождается в *плоскую*, которая не может существовать при заданных размерах.

В результате геометрической несогласованности происходит потеря равновесия и устойчивости конструкции, что математически находит свое отражение в стремлении к бесконечности реакций трех последних стержней 4–6 (2.21), где в выражениях для X(4,1) – X(6,1) при $\varphi = 0$ или $\varphi = \pi$ происходит деление на 0.

Это очень хорошо видно на графиках зависимости значений реакций стержней 1–6 от величины угла φ, построенных по соотношениям (2.21):

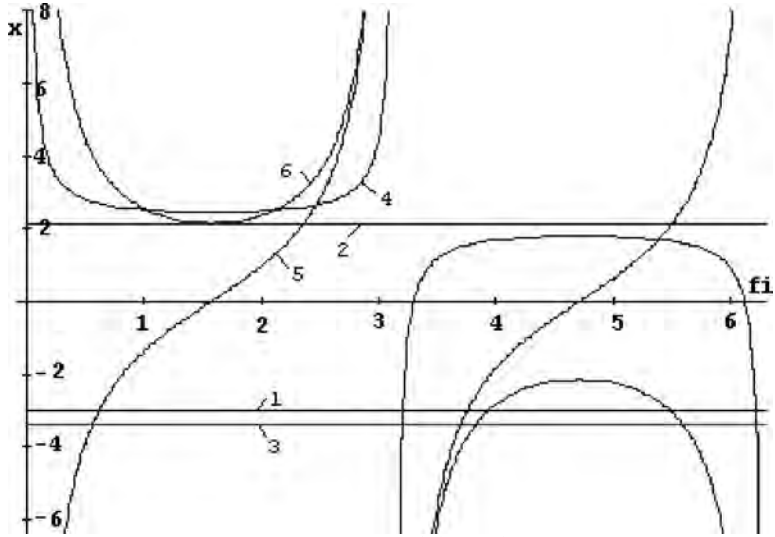

Рис. 2.3. Зависимости усилий (X, кН) в стержнях 1-6 пространственной фермы при формальной вариации угла FI (рад) для всей конструкции (рис. 2.1): 1 – S_1, 2 – S_2, 3 – S_3, 4 – S_4, 5 – S_5, 6 – S_6

Чтобы этого не происходило, нужно угол между вертикальной плоскостью действия силы P и горизонтальной осью обозначить другой буквой, например α (с идентификатором ALFA).

Его значение для начального положения равно φ, но при вращении силы P вокруг вертикальной оси угол α изменяется соответствующим образом независимо от постоянного значения угла φ. В программе 2.2 оператор 55 с учетом этого примет следующий вид (где жирным шрифтом выделены измененные обозначения):

```
B:=MAT((P*COS(PSI)*SIN(ALFA)),(P*COS(PSI)*COS(ALFA)),   55
 (P*SIN(PSI)),(0),(0),(0));                              (2.22)
```

Значение угла ALFA в программе 2.2 не задано, поэтому он является свободной переменной.

Теперь в результате работы такой измененной программы 2.2 с новым оператором 55 мы получим настоящие зависимости результатов решения (2.23) от вариации угла α при вращении силы P вокруг вертикальной оси с постоянным углом наклона к горизонтальной плоскости. Они весьма сильно отличаются от легкомысленно полученных соотношений (2.21), что видно из их сравнения:

$$X(1,1):=-\frac{16*\mathrm{sqrt}(2)}{\mathrm{sqrt}(57)}$$

$$X(2,1):=\frac{4*(4*\mathrm{sqrt}(41)*\cos(\mathrm{alfa})-5*\mathrm{sqrt}(41)*\sin(\mathrm{alfa})+20)}{5*\mathrm{sqrt}(57)}$$

$$X(3,1):=-\frac{164*\cos(\mathrm{alfa})}{5*\mathrm{sqrt}(57)} \qquad X(4,1):=\frac{20}{\mathrm{sqrt}(57)} \qquad (2.23)$$

$$X(5,1):=-\frac{20}{\mathrm{sqrt}(57)} \qquad X(6,1):=\frac{4*\mathrm{sqrt}(42)}{\mathrm{sqrt}(57)}$$

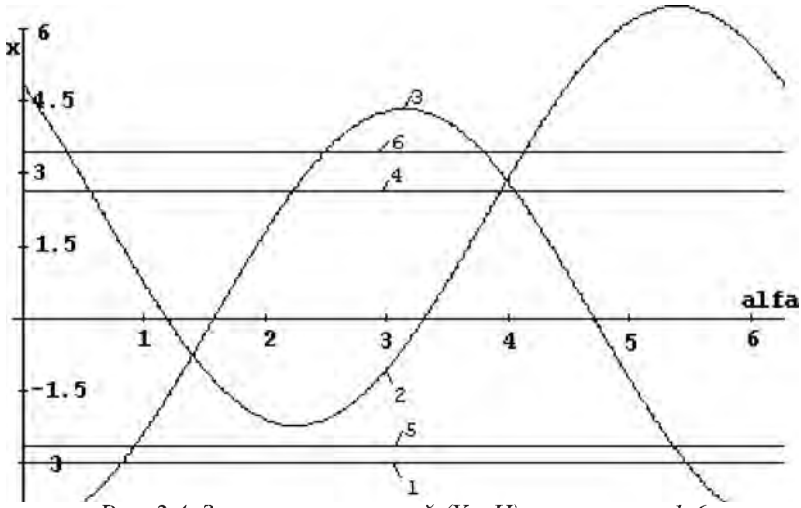

Рис. 2.4. Зависимости усилий (X, кН) в стержнях 1-6 пространственной конструкции (рис. 2.1) от изменения угла ALFA (рад) при вращении силы P вокруг вертикальной оси: 1 – S_1, 2 – S_2, 3 – S_3, 4 – S_4, 5 – S_5, 6 – S_6

Графики зависимостей значений реакций стержней 1–6 пространственной конструкции (рис. 2.1) от изменения угла ALFA (рад) при вращении

силы P вокруг вертикальной оси, построенных по соотношениям (2.23) на рис. 2.4, также наглядно подтверждают это.

Оказывается, значение реакций при вращении силы P вокруг вертикальной оси с постоянным углом наклона к горизонтальной плоскости ψ:

- в стержнях 1, 4–6 является постоянным;
- в третьем — изменяется по косинусоидальному закону;
- для второго стержня — определяется разностью функций $cos\alpha$ и $sin\alpha$, а сама зависимость смещена по вертикальной оси от нулевого значения.

Одновременное удаление из программы 2.2 операторов 24, 29 при изменении оператора 55 по фрагменту (2.22) даст возможность получить *зависимости значений реакций* пространственной конструкции (рис. 2.1) *при всевозможном изменении величины и направления силы P:*

$$X(1,1):= -\,sqrt(2)*sin(psi)*p$$

$$X(2,1):=(p*(4*cos(alfa)*cos(psi)-5*cos(psi)*sin(alfa)+5*sin(psi)))/5$$

$$X(3,1):= -\frac{sqrt(41)*cos(alfa)*cos(psi)*p}{5} \qquad (2.24)$$

$$X(4,1):=\frac{5*sin(psi)*p}{4} \qquad X(5,1):= -\frac{5*sin(psi)*p}{4}$$

$$X(6,1):=\frac{sqrt(42)*sin(psi)*p}{4}$$

Теперь слишком широкие возможности при выполнении вариаций нагрузки просто придавливают нас тяжестью последующего анализа зависимостей реакций в стержнях от двух-трех переменных параметров (P, ψ и α) в соотношениях (2.24). Поэтому здесь, как и во многих других случаях, можно лишний раз вспомнить, как там у Епифания Премудрого сказано: "*Простота без пестроты*" и "*Не мудрствуя лукаво*". Затем, будучи ободренными этими древними методическими принципами, в дальнейшем всегда будем проводить исследования:

- варьируя только один переменный параметр по образцам (2.19), (2.20) и (2.22);
- дополняя полученные аналитические соотношения их графическим двумерным представлением типа рис. 2.2 – 2.4;
- внимательно осмысливая и анализируя полученные аналитические и графические зависимости, определяя зону оптимальной действующей нагрузки и других варьируемых параметров. Ведь этого за вас не сможет сделать ПК! И теперь, как и ранее, остается верным принцип: *"Если не умеешь думать, нечего садиться за компьютер!"*.

2.2.4. Исследование влияния вариации геометрических факторов

При вариации геометрических факторов нас еще больше поражают открывающиеся перед нами возможности, однако, и опасности возрастают. Ведь недаром расчеты на ПК сравнивают с прогулкой в горах: чем выше забираешься, тем большие красоты открываются твоему взору, но и возможности свалиться в пропасть увеличиваются. Поэтому, утвердившись в мысли, что *"лучше гор могут быть только" ... символьные аналитические преобразования на компьютере,* вернемся к теме нашего исследования.

Основной опасностью при вариации геометрических факторов являются *несогласованные изменения размеров конструкции.*

Ранее, начав исследовать влияние вращения силы P путем вариации угла φ при получении уравнений 2.21, мы обратили на это внимание. Рассмотрим этот вопрос подробнее.

При вариации угла φ происходит изменение его тригонометрических функций (первые два уравнения соотношений (2.9)), которые определяются размерами **a** и **b**. Это приводит к соответствующему ***изменению этих величин.***

Но они входят в свою очередь в определение остальных тригонометрических функций значений углов (2.9): **a** для трех (ψ, β, θ) и **b** для двух (ψ, θ), ***которые остались постоянными***. Значит, в этих соотношениях **a** и **b не изменились**. Это противоречие и называется *несогласованными изменениями размеров конструкции,* при которых она в данном виде не может существовать.

Значит, потеря равновесия и устойчивости данной конструкции происходит не только в особых точках при φ = 0 или φ = π, в которых в уравнениях (2.21) происходит деление на 0, а этим соотношениям нельзя доверять при любых значениях угла φ, так как они не имеют никакого отношения к рассматриваемой конструкции.

Поэтому в программе 2.2 нельзя удалить операторы 22–29 и, получив соотношения (2.25) считать, что они описывают значения реакций стержней при всевозможных согласованных изменениях направлений стержней в плоскостях их действия и силы P. Это будет просто результат неосознанного вождения самого себя за нос:

$$X(1,1):=-\frac{\sin(psi)*p}{\cos(beta)} \qquad X(2,1):=\frac{\sin(beta)*\sin(psi)*p}{\cos(beta)}$$

$$X(3,1):=-\cos(psi)*p \qquad (2.25)$$

$$X(4,1):=\frac{(\sin(psi)*p*(\cos(beta)*\sin(fi)*\sin(teta)+\cos(teta)*\sin(beta)))}{(\cos(beta)*\sin(fi)*\sin(teta))}$$

$$X(5,1):=-\frac{\cos(fi)*\sin(beta)*\sin(psi)*p}{\cos(beta)*\sin(fi)}$$

$$X(6,1):=\frac{\sin(beta)*\sin(psi)*p}{\cos(beta)*\sin(fi)*\sin(teta)}$$

Для выполнения исследования ***влияния вариации геометрических факторов на значения реакций стержней*** на самом деле нужно сделать следующее:

1. В программе 2.2 представить выражения для тригонометрических функций в общем виде через размеры данной конструкции **a**, **b**, **c** и **d**, например, с использованием идентификаторов AR, BR (для отличия от соответствующих матриц A, B), C и D:

```
SIN(FI):=AR/SQRT(AR**2+BR**2);                                    21
COS(FI):=BR/SQRT(AR**2+BR**2);                                    22
SIN(PSI):=C/SQRT(AR**2+BR**2+C**2);                               23
COS(PSI):=SQRT(AR**2+BR**2)/SQRT(AR**2+BR**2+C**2);               24
SIN(BETA):=AR/SQRT(AR**2+C**2);                                   25
COS(BETA):=C/SQRT(AR**2+C**2);                                    26
SIN(TETA):=SQRT(AR**2+BR**2)/SQRT(AR**2+BR**2+D**2);              27
COS(TETA):=D/SQRT(AR**2+BR**2+D**2);                              28
```

2. Предварительно операторами присваивания следует задать исходные размеры данной конструкции:

```
AR:=4; BR:=5; C:=4; D:=1;                                         18
```

3. Представить оператор 55 в одной из двух форм с использованием:

   - *численных значений,* задающих имеющиеся в нем тригонометрические функции, для исследования влияния вариации геометрических факторов на значения реакций стержней:

```
B:=MAT((P*SQRT(41/57)*4/SQRT(41)),                                55
 (P*SQRT(41/57)*5/SQRT(41)),                                  (2.26)
 (P*4/SQRT(57)),(0),(0),(0));
```

   - *символьных обозначений* для возможности независимого варьирования направления действия силы P, применяя для задания имеющихся в нем углов *свободные переменные* GAMMA и ALFA, не

имеющие никаких значений в программе 2.2 и не используемые там более нигде, например:

```
B:=MAT((P*COS(GAMMA)*SIN(ALFA)),                                    55
 (P*COS(GAMMA)*COS(ALFA)),                                    (2.27)
 (P*SIN(GAMMA)),(0),(0),(0));
```

Теперь достаточно осознанно скомпоновать универсальную программу 2.3, приспособленную для вариации геометрических размеров конструкции. Это можно сделать, например, следующим образом:

```
% Программа 2.3
OUT "\C8\PR2-3.LIS";                                                10
ON NERO;                                                            15
AR:=4; BR:=5; C:=4; D:=1;                                           18
N:=6;                                                               20
SIN(FI):=AR/SQRT(AR**2+BR**2);                                      21
COS(FI):=BR/SQRT(AR**2+BR**2);                                      22
SIN(PSI):=C/SQRT(AR**2+BR**2+C**2);                                 23
COS(PSI):=SQRT(AR**2+BR**2)/SQRT(AR**2+BR**2+C**2);                 24
SIN(BETA):=AR/SQRT(AR**2+C**2);                                     25
COS(BETA):=C/SQRT(AR**2+C**2);                                      26
SIN(TETA):=SQRT(AR**2+BR**2)/SQRT(AR**2+BR**2+D**2);                27
COS(TETA):=D/SQRT(AR**2+BR**2+D**2);                                28
P:=4;                                                               29
MATRIX A(N,N),B(N,1),X(N,1);                                        30
A:=MAT((-SIN(BETA),-1,-SIN(FI),0,0,0),                              35
 (0,0,-COS(FI),0,0,0),
 (-COS(BETA),0,0,0,0,0),                                      (2.28)
 (SIN(BETA),0,0,0,0,SIN(TETA)*SIN(FI)),
 (0,0,0,0,-1,-SIN(TETA)*COS(FI)),
 (COS(BETA),0,0,1,0,-COS(TETA)));
B:=MAT((P*SQRT(41/57)*4/SQRT(41)),                                  55
 (P*SQRT(41/57)*5/SQRT(41)),
 (P*4/SQRT(57)),(0),(0),(0));
OFF NERO;                                                           80
X:=A**(-1)*B;                                                       85
SHUT "\C8\PR2-3.LIS";                                               95
END;                                                                99
```

Все команды и операторы программы 2.3 рассматривались ранее и повторно не описываются. Отметим, что это тестовый вариант программы 2.3, в результате работы которой должны получиться результаты решения в форме (2.16), в чем следует убедиться, запустив ее на выполнение. *Так следует всегда поступать:* ***перед началом вариации удостовериться на известном численном примере в правильности результата базовой программы,*** подготовленной для проведения исследований.

Для изучения раздельного влияния вариации геометрических факторов на значения реакций стержней нужно из предложения 18 исключить соответствующий оператор присваивания, задающий данный размер **a**, **b**, **c** или **d**. ***Предварительно нужно очистить исключаемую переменную, указав ее в качестве параметра команды CLEAR*** и сделав свободной. Если этого не осуществить, то после запуска программы 2.3 ***в памяти системы в течение сеанса работы останутся все заданные там переменные.***

Поэтому нельзя будет запускать без выхода из CAB REDUCE предварительно подготовленные файлы с модификациями предложения 18 программы 2.3 или даже с его полным отсутствием. В этом случае никакие исключения из соответствующих операторов присваивания, задающих данный размер конструкции **a**, **b**, **c** или **d**, не будут давать никаких изменений в форме результатов решения (2.16). Система будет помнить в течение сеанса работы все заданные ей переменные и брать недостающие данные из своей памяти.

Поэтому перед выполнением каких-либо вариаций всегда следует очищать исключаемую переменную. Будем считать это признаком хорошего тона, и всегда использовать для этой цели команду ***CLEAR***.

В результате предложение 18 программы 2.3 для изучения раздельного влияния вариации геометрических факторов на значения реакций стержней должно принимать следующие формы (2.29), (2.31), (2.33), (2.35). Теперь соответствующий идентификатор AR, BR, C и D становится свободной переменной и входит в символьном виде в результаты решения, выражая их зависимость от данной величины. При этом *все изменения размеров конструкции происходят согласованно,* а результаты решения легко поддаются аналитическому и графическому анализу по аналогии с вышерассмотренной вариацией силы P, что мы предлагаем выполнить самостоятельно для ваших вариантов заданий.

Приведем для рассматриваемого типового примера пространственной шарнирно-стержневой конструкции [22, с. 52-54], схема которого представлена также на рис. 2.1, формы предложения 18 программы 2.3 с полученными соответствующими аналитическими зависимостями,

выражающими влияние раздельного изменения размеров конструкции **a**, **b**, **c** и **d** на значения реакций стержней. Покажем также геометрическую интерпретацию полученных соотношений.

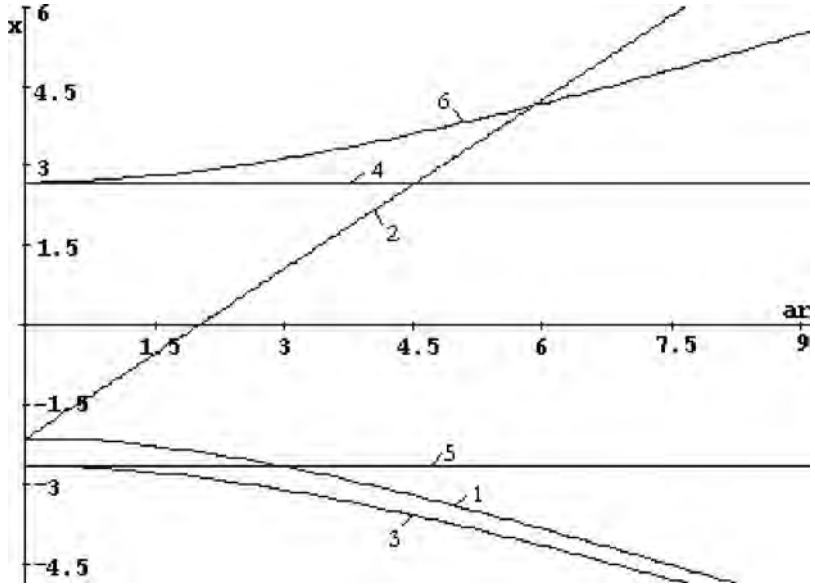

Рис. 2.5. Зависимости усилий (X, кН) в стержнях 1-6 пространственной конструкции (рис. 2.1) от изменения ее размера ***a*** *(м): 1 – S_1, 2 – S_2, 3 – S_3, 4 – S_4, 5 – S_5, 6 – S_6*

В результате все это примет соответственно следующий вид:

- *вариация размера* ***a***, обозначаемого во фрагменте (2.29) программы 2.3 идентификатором AR (и ar в распечатке результатов (2.30)):

```
CLEAR AR; BR:=5; C:=4; D:=1;                                    (2.29)
```

$$X(1,1):=-\frac{4*\mathrm{sqrt}(ar^2+16)}{\mathrm{sqrt}(57)} \qquad X(2,1):=\frac{8*(ar-2)}{\mathrm{sqrt}(57)}$$

$$X(3,1):=-\frac{4*\mathrm{sqrt}(ar^2+25)}{\mathrm{sqrt}(57)} \qquad X(4,1):=\frac{20}{\mathrm{sqrt}(57)} \qquad (2.30)$$

$$X(5,1):=-\frac{20}{\mathrm{sqrt}(57)} \qquad X(6,1):=\frac{4*\mathrm{sqrt}(ar^2+26)}{\mathrm{sqrt}(57)}$$

На рис. 2.5 показаны графики, выражающие зависимости усилий (2.30) в стержнях 1- 6 пространственной конструкции (рис. 2.1) от изменения ее размера **a**.

- *вариация размера* ***b***, обозначаемого во фрагменте (2.31) программы 2.3 идентификатором BR (и br в распечатке результатов (2.32)):

```
CLEAR BR; AR:=4; C:=4; D:=1;                                    (2.31)
```

$$X(1,1):=-\frac{16*\mathrm{sqrt}(2)}{\mathrm{sqrt}(57)} \qquad X(2,1):=\frac{80}{\mathrm{sqrt}(57)*br}$$

$$X(3,1):=-\frac{20*\mathrm{sqrt}(br^2+16)}{\mathrm{sqrt}(57)*br} \qquad X(4,1):=\frac{20}{\mathrm{sqrt}(57)} \qquad (2.32)$$

$$X(5,1):=-\frac{4*br}{\mathrm{sqrt}(57)} \qquad X(6,1):=\frac{4*\mathrm{sqrt}(br^2+17)}{\mathrm{sqrt}(57)}$$

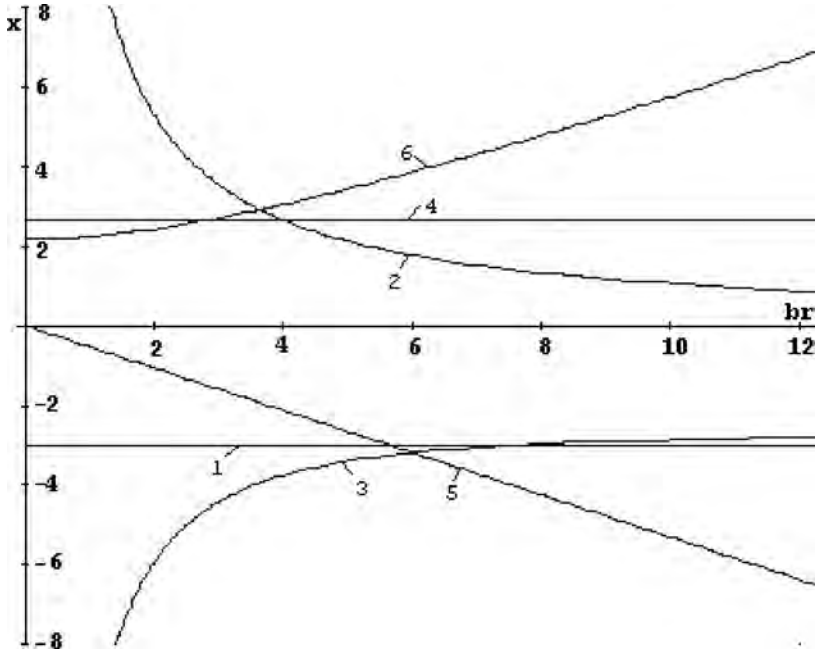

Рис. 2.6. Зависимости усилий (X, кН) в стержнях 1-6 пространственной конструкции (рис. 2.1) от изменения ее размера ***b*** *(м): 1 – S_1, 2 – S_2, 3 – S_3, 4 – S_4, 5 – S_5, 6 – S_6*

- *вариация размера* ***c***, обозначаемого во фрагменте (2.33) программы 2.3 идентификатором C (и c в распечатке результатов (2.34)):

```
CLEAR C; AR:=4; BR:=5; D:=1;                                    (2.33)
```

$$X(1,1):=-\frac{16*\mathrm{sqrt}(c^2+16)}{\mathrm{sqrt}(57)*c} \qquad X(2,1):=\frac{64}{\mathrm{sqrt}(57)*c}$$

$$X(3,1):=-\frac{4*\mathrm{sqrt}(41)}{\mathrm{sqrt}(57)} \qquad X(4,1):=\frac{16*(c+1)}{\mathrm{sqrt}(57)*c} \qquad (2.34)$$

$$X(5,1):=-\frac{80}{\mathrm{sqrt}(57)*c} \qquad X(6,1):=\frac{16*\mathrm{sqrt}(42)}{\mathrm{sqrt}(57)*c}$$

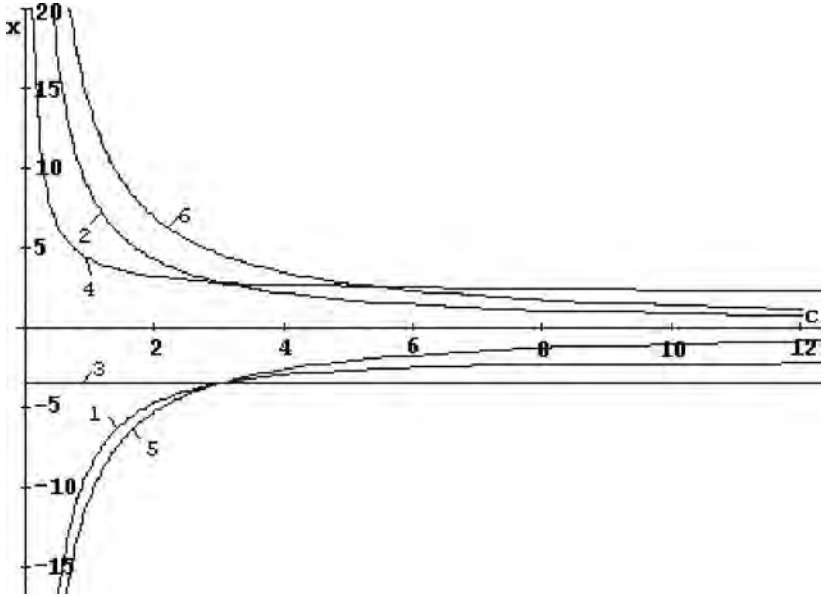

Рис. 2.7. Зависимости усилий (X, кН) в стержнях 1-6 пространственной конструкции (рис. 2.1) от изменения ее размера ***c*** *(м): 1 – S_1, 2 – S_2, 3 – S_3, 4 – S_4, 5 – S_5, 6 – S_6*

- *вариация размера* ***d***, обозначаемого во фрагменте (2.35) программы 2.3 идентификатором `D` (и `d` в распечатке результатов (2.36)):

```
CLEAR D; AR:=4; BR:=5; C:=4;                                    (2.35)
```

$$X(1,1):=-\frac{16*\mathrm{sqrt}(2)}{\mathrm{sqrt}(57)} \qquad X(2,1):=\frac{16}{\mathrm{sqrt}(57)}$$

$$X(3,1):=-\frac{4*\mathrm{sqrt}(41)}{\mathrm{sqrt}(57)} \qquad X(4,1):=\frac{4*(d+4)}{\mathrm{sqrt}(57)} \qquad (2.36)$$

$$X(5,1):= -\frac{20}{\text{sqrt}(57)} \qquad X(6,1):=\frac{4*\text{sqrt}(d^2+41)}{\text{sqrt}(57)}$$

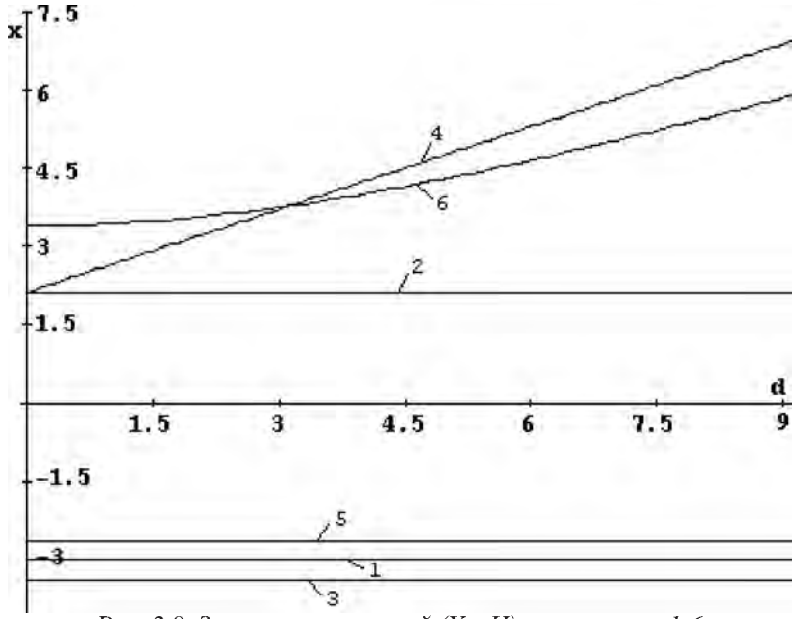

Рис. 2.8. Зависимости усилий (X, кН) в стержнях 1-6 пространственной конструкции (рис. 2.1) от изменения ее размера ***d*** *(м): 1 – S_1, 2 – S_2, 3 – S_3, 4 – S_4, 5 – S_5, 6 – S_6*

Теперь выполним следующие действия:

- полностью очистим все используемые переменные и имена со скобками командой (2.37):

```
CLEAR AR, BR, C, D, P, SIN(ALFA), COS(ALFA),
SIN(GAMMA), COS(GAMMA);                                    (2.37)
```

- удалим из программы 2.3 предложение 18 и оператор 29;
- представим оператор 55 по фрагменту (2.27).

После работы такой модифицированной программы 2.3 получим соотношения (2.38), описывающие значения реакций при всевозможных изменениях величины и направления силы P и направлений наклонных стержней в плоскостях их действия:

$X(1,1):= -\text{sqrt}(ar^2 + c^2)*\sin(\text{gamma})*p / c$

$X(2,1):= (p*(\cos(\text{alfa})*\cos(\text{gamma})*ar*c - \cos(\text{gamma})*\sin(\text{alfa})*br*c +$
$\sin(\text{gamma})*ar*br)) / (br*c)$

$$X(3,1):= -\sqrt{ar^2 + br^2}\cdot\cos(alfa)\cdot\cos(gamma)\cdot p / br \qquad (2.38)$$

$$X(4,1):= \sin(gamma)\cdot p\cdot(c + d) / c$$

$$X(5,1):= -\sin(gamma)\cdot br\cdot p / c$$

$$X(6,1):= \sqrt{ar^2 + br^2 + d^2}\cdot\sin(gamma)\cdot p / c$$

Выражения такого общего вида позволяют легко получать значения реакций при любом наборе данных для всей конструкции и действующей нагрузки, проверяя наши предположения, сделанные на основании анализа вариаций каких-либо отдельных факторов.

Как видим, они сильно отличаются от механически полученных ранее уравнений (2.25), наглядно показывающих, как сильно можно увести себя не в ту сторону при бездумном выполнении подобных исследований.

Замечание. *Сложные аналитические преобразования* до недавнего времени были доступны только посвященным. Всех остальных они приводили в священный трепет, как что-то им совсем недоступное.

Хочется решительно предостеречь от подобного чувства испуга при взгляде на все приведенные аналитические выражения и их графическую интерпретацию. Все они ***получены ПК в результате ваших простых, но аккуратных и внимательных действий,*** *полностью описанных в настоящем пособии.*

Настало время, когда рутинную аналитическую работу практически любой степени сложности с легкостью берет на себя персональный компьютер.

Для человека она сопряжена с большими техническими трудностями, в результате которых возрастает вероятность ошибок при действиях "вручную". В некоторых случаях результат практически невозможно получить без использования ПК.

Однако это не означает, что теперь не нужно будет учить математику, мучиться над анализом результатов, осмысливать их, делая выводы, а все за вас сделает компьютер.

Весьма вероятно, что вами будет выполнена примерно такая же ***умственная работа***. При использовании ПК она ***просто перераспределяется: ее рутинная часть резко уменьшается, но зато также должна возрасти ее творческая составляющая*** по осмыслению и анализу целого спектра всевозможных аналитических решений практически любой степени сложности. Вследствие чего и результаты также резко возрастут.

Если этого не произойдет, вы будете просто с использованием современных информационных технологий элегантно водить себя за нос.

Но это удовольствие прекратится и идиллия сразу нарушится при первом же практическом использовании результатов вашего анализа, что закончится весьма печально для вас (и зачастую, к большому сожалению, для окружающих).

Это замечание в особой степени касается проведения исследований влияния вариации различных факторов для выбора оптимальных размеров конструкции или условий нагружения.

2.3. Решение СЛАУ с “большим” количеством уравнений

2.3.1. Формализация задачи

Алгоритм решения различных задач статики с использованием CAB REDUCE одинаков, поэтому их программные реализации аналогичны. Это обуславливается тем фактором, что любая задача статики по определению реакций опор (тела или системы тел) или усилий в стержнях шарнирно-стержневой конструкции представляет из себя систему линейных алгебраических уравнений.

Таким образом, любые конкретные реализации пространственных или плоских объектов равновесия или их систем ***различаются только количеством систем уравнений, размерами получающихся матриц и значениями их элементов***.

Поэтому после достаточно детального рассмотрения п. 2.2 можно было переходить к дальнейшему изучению CAB REDUCE.

Однако очень долго для различных задач статики (по определению реакций опор или усилий в стержнях) писались разные программы вплоть до недавнего времени [21], что лучше доказать эти общие рассуждения на конкретных примерах.

В качестве примера задачи статики с “большим” количеством уравнений равновесия воспользуемся аналитическим решением типового задания по определению усилий в стержнях плоской шарнирно-стержневой конструкции для ССС [22, с. 5-12].

Его расчетная схема приведена на рис. 2.9,а.

Отличие рис. 2.9,б от рис.5 [22, с. 10] состоит только в том, что реакция Rb показана в общем виде в форме проекций Xb и Yb.

Это обусловлено тем фактом, что при решении СЛАУ задач статики с "большим" количеством уравнений на ПК ***не требуется предварительного определения реакций опор***, с которого обычно начинается аналитическое решение. Его обычно делают только для облегчения последующих "ручных" преобразований.

Поэтому на рис.4 и 5 [22, с. 9-10] показано *истинное* направление реакции Rb шарнирно-неподвижной опоры в точке B. Оно определено в результате проведения предварительного дополнительного расчета рассматриваемого типового примера [22, с. 9] и совпадает с рис. 2.9,а.

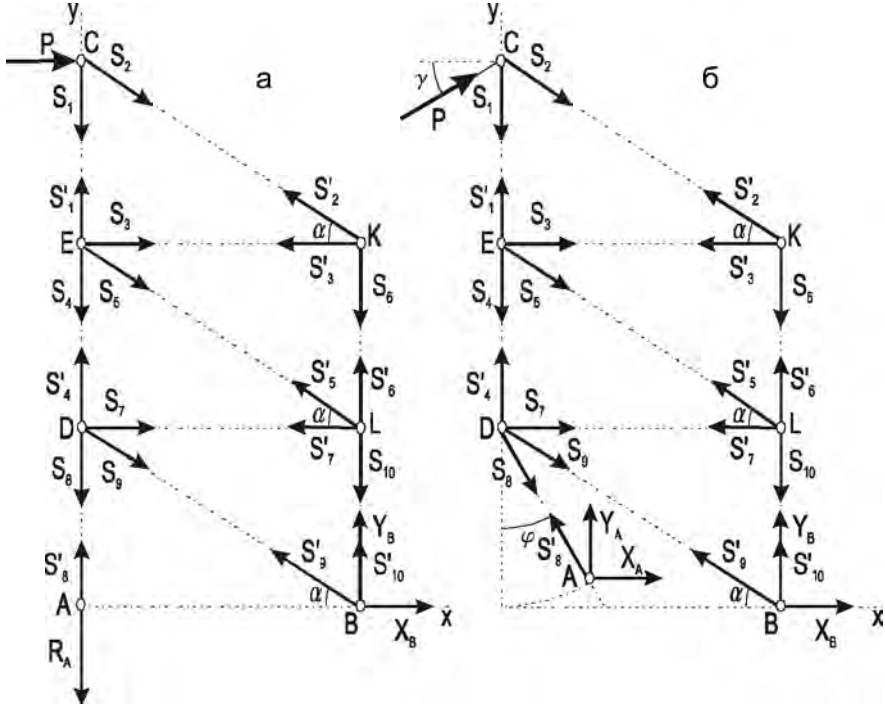

Рис. 2.9. Расчетная схема плоской фермы: а — исходное положение; б — вариации направления силы P и положения шарнирно-неподвижной опоры в точке A

При использовании ПК такая дополнительная работа является не только ненужной, но даже вредной, ибо она затрудняет проведение дальнейших исследований. Поэтому перед составлением уравнений равновесия для их решения на ПК:

- введенные проекции Xb и Yb неизвестной реакции Rb просто направим в сторону положительного направления осей координат (рис. 2.9,б), тогда знак ответа покажет их истинное направление;
- никакого предварительного нахождения реакций опор делать не будем, ибо упрощение аналитических выкладок для ПК не имеет никакого значения. Искомые усилия в стержнях, как и реакции опор, получатся из численного или символьного решения на ПК системы уравнений равновесия любой степени сложности для всех узлов простой плоской фермы;
- для изучения вариации нагрузки (см. п. 2.3.3) или геометрических факторов (см. п. 2.3.4) варьируемые параметры покажем в самой общей постановке в произвольном текущем положении (направление силы P и положение шарнирно-неподвижной опоры в точке A на рис. 2.9,б).

Для исходного положения плоской фермы (рис. 2.9,а) составим по два уравнения равновесия для сил, сходящихся соответственно в узлах C, K, E, L, D, A и B (они приведены также последовательно в пособии [22, с. 9-11]), что в результате будет иметь следующий вид:

Узел C:	$\sum X_i = 0;$	1. $P + S_2 * \cos\alpha = 0,$	
	$\sum Y_i = 0;$	2. $-S_1 - S_2 * \sin\alpha = 0,$	
Узел K:	$\sum X_i = 0;$	3. $-S_2' * \cos\alpha - S_3' = 0,$	
	$\sum Y_i = 0;$	4. $S_2' * \sin\alpha - S_6 = 0,$	
Узел E:	$\sum X_i = 0;$	5. $S_3 + S_5 * \cos\alpha = 0,$	
	$\sum Y_i = 0;$	6. $S_1' - S_4 - S_5 * \sin\alpha = 0,$	
Узел L:	$\sum X_i = 0;$	7. $-S_5' * \cos\alpha - S_7' = 0,$	(2.39)
	$\sum Y_i = 0;$	8. $S_5' * \sin\alpha + S_6' - S_{10} = 0,$	
Узел D:	$\sum X_i = 0;$	9. $S_7 + S_9 * \cos\alpha = 0,$	
	$\sum Y_i = 0;$	10. $S_4' - S_8 - S_9 * \sin\alpha = 0,$	
Узел A:	$\sum X_i = 0;$	11. $S_8' - Ra = 0,$	
Узел B:	$\sum X_i = 0;$	12. $-S_9' * \cos\alpha + Xb = 0,$	
	$\sum Y_i = 0;$	13. $S_9' * \sin\alpha + S_{10}' + Yb = 0.$	

Согласно условию значение P=11. На рис. 2.9 и рис. 5 [22, с. 10] уже учтена разность направлений штрихованных и не штрихованных реакций связей для каждого стержня. Поэтому вместо векторных уравнений типа

(2.4) используются соответствующие скалярные равенства типа (2.5), учитывающие равенство значений этих величин вплоть до знака, так что им можно присвоить один идентификатор: Si = Si′ = Xi.

Рассмотрим подготовку исходных данных для решения этого примера в CAB REDUCE. Соответствие идентификаторов для формализации системы уравнений (2.39) можно представить в следующей форме:

X_1	X_2	X_3	X_4	X_5	X_6
$S_1=S_1'$	$S_2=S_2'$	$S_3=S_3'$	$S_4=S_4'$	$S_5=S_5'$	$S_6=S_6'$

(2.40)

X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}
$S_7=S_7'$	$S_8=S_8'$	$S_9=S_9'$	$S_{10}=S_{10}'$	Ra	Xb	Yb

Для удобства составления программ заменим также используемую греческую букву для обозначения угла α ее идентификатором, записанным в латинской транскрипции: α = ALFA.

Перенесем свободные члены, не содержащие неизвестных, в правые части уравнений (2.39), которые с учетом соответствия идентификаторов (2.40) теперь примут следующий формализованный вид (2.41):

1. X_2*cos(ALFA) = -P,
2. -X_1 - X_2*sin(ALFA) = 0,
3. -X_2*cos(ALFA) - X_3 = 0,
4. X_2*sin(ALFA) - X_6 = 0,
5. X_3 + X_5*cos(ALFA) = 0,
6. X_1 - X_4 - X_5*sin(ALFA) = 0,
7. -X_5*cos(ALFA) - X_7 = 0, (2.41)
8. X_5*sin(ALFA) + X_6 - X_{10} = 0,
9. X_7 + X_9*cos(ALFA) = 0,
10. X_4 - X_8 - X_9*sin(ALFA) = 0,
11. X_8 - X_{11} = 0,
12. -X_9*cos(ALFA) + X_{12} = 0,
13. X_9*sin(ALFA) + X_{10} + X_{13}=0.

Для удобства пользования уравнения пронумерованы сверху вниз, начиная с 1, где номером без скобок обозначается порядковый номер строки. Теперь система уравнений (2.41) приобрела явный вид СЛАУ (2.2), в которой выписаны в символьной форме только отличные от нуля элементы. Для дальнейшего её решения или исследования с помощью ПК в системе REDUCE имеются различные возможности.

2.3.2. Решение в численном виде

С увеличением количества уравнений равновесия резко *возрастают достоинства работы с использованием операторов присваивания для ненулевых значений* элементов соответствующих матриц. Они состоят в *сокращении вводимых элементов* (только ненулевых) и в *точном их помещении с указанием места.* Все это значительно облегчает ввод данных и поиск допущенных ошибок.

Мы рассмотрим также для создания вам возможности осознанного выбора и использование операторов MAT. В них элементы матрицы A(I,J) и матрицы-столбца B(I,1) задаются в *полном виде* по строкам.

Сначала представим с использованием многократного оператора присваивания (2.17) значения используемых тригонометрических функций, что является общим для обеих возможностей:

```
SA:=SIN(ALFA):=1/2;                                    22
CA:=COS(ALFA):= SQRT(3)/2;                             24
```

Для задания элементов матрицы A(I,J) и матрицы-столбца B(I,1) в *полном виде* по строкам в операторе MAT их следует дополнить нулями до полных форм N×N и N×1, что будет с использованием идентификаторов многократных операторов присваивания 22–24 иметь следующий вид:

```
A:=MAT((0,CA,0,0,0,0,0,0,0,0,0,0,0),          35
 (-1,-SA,0,0,0,0,0,0,0,0,0,0,0),
 (0,-CA,-1,0,0,0,0,0,0,0,0,0,0),
 (0,SA,0,0,0,-1,0,0,0,0,0,0,0),
 (0,0,1,0,CA,0,0,0,0,0,0,0,0),
 (1,0,0,-1,-SA,0,0,0,0,0,0,0,0),
 (0,0,0,0,-CA,0,-1,0,0,0,0,0,0),                                (2.42)
 (0,0,0,0,SA,1,0,0,0,-1,0,0,0),
 (0,0,0,0,0,0,1,0,CA,0,0,0,0),
 (0,0,0,1,0,0,0,-1,-SA,0,0,0,0),
```

```
 (0,0,0,0,0,0,0,1,0,0,-1,0,0),
 (0,0,0,0,0,0,0,0,-CA,0,0,1,0),
 (0,0,0,0,0,0,0,0,SA,1,0,0,1));
B:=MAT((-P),(0),(0),(0),(0),(0),(0),(0),      55
 (0),(0),(0),(0),(0));                              (2.43)
```

Как видите, достаточно трудно не перепутать местами какой-нибудь нолик с единичкой, а потом долго и упорно чисто механически пытаться найти ошибку.

Конечно, никакая добросовестная работа не бывает бесполезной, и эта после достижения правильного результата значительно усовершенствует ваши внимательность и терпение.

Однако лучше развитие этих замечательных качеств соединить с небольшим теоретическим багажом и прочитать дополнение 2.1. Затем организовать ввод исходных данных с использованием многократных операторов присваивания 22–24 для *ненулевых значений* элементов соответствующих матриц, что может иметь следующий вид:

```
A(1,2):=CA; A(2,1):=-1; A(2,2):=-SA; A(3,2):=-CA;       35
A(3,3):=-1; A(4,2):=SA; A(4,6):=-1;  A(5,3):=1;         37
A(5,5):=CA; A(6,1):=1;  A(6,4):=-1;  A(6,5):=-SA;       40
A(7,5):=-CA; A(7,7):=-1; A(8,5):=SA; A(8,6):=1;         42
A(8,10):=-1; A(9,7):=1; A(9,9):=CA;  A(10,4):=1;        45
A(10,8):=-1; A(10,9):=-SA; A(11,8):=1;                  47
A(11,11):=-1; A(12,9):=-CA; A(12,12):=1;                48
A(13,9):=SA; A(13,10):= 1; A(13,13):=1;                 50
B(1,1):=-P;                                             55
```

Теперь достаточно осознанно скомпоновать универсальную программу 2.4, приспособленную для решения системы уравнений (2.41) рассматриваемого типового примера и последующего выполнения вариаций различных факторов.

Это можно сделать, например, с использованием многократных операторов присваивания 35–55 для *ненулевых значений* элементов, следующим образом (по вопросам записи программы 2.4 в файле PR2-4 в подкаталоге \C1 корневого каталога текущего диска и запуске ее на выполнение см. п. 7.3 и пример в п. 1.4.2):

```
COMMENT Программа 2.4: решение типового примера С-1 на REDUCE
с использованием многократных операторов присваивания 35-55
```

```
для ненулевых значений элементов матрицы A и матрицы-столбца
B с представлением данных ввода и результатов в численном
виде;
OUT "\C1\PR2-4.LIS";                                          10
ON NERO;                                                      15
N:=13;                                                        20
SA:=SIN(ALFA):=1/2;                                           22
CA:=COS(ALFA):= SQRT(3)/2;                                    24
P:=11;                                                        29
MATRIX A(N,N), B(N,1), X(N,1);                                30
A(1,2):=CA; A(2,1):=-1; A(2,2):=-SA; A(3,2):=-CA;             35
A(3,3):=-1; A(4,2):=SA; A(4,6):=-1;  A(5,3):=1;               37
A(5,5):=CA; A(6,1):=1;  A(6,4):=-1;  A(6,5):=-SA;             40
A(7,5):=-CA; A(7,7):=-1; A(8,5):=SA; A(8,6):=1;               42
A(8,10):=-1; A(9,7):=1; A(9,9):=CA;  A(10,4):=1;              45
A(10,8):=-1; A(10,9):=-SA; A(11,8):=1;                        47
A(11,11):=-1; A(12,9):=-CA; A(12,12):=1;                      48
A(13,9):=SA; A(13,10):= 1; A(13,13):=1;                       50
B(1,1):=-P;                                                   55
OFF NERO;                                                     80
X:=A**(-1)*B;                                                 85
SHUT "\C1\PR2-4.LIS";                                         95
;END;                                                         99
```

Команда 10 открывает выводной файл PR2-4.LIS, находящийся в подкаталоге \C1 корневого каталога текущего диска, куда будут записываться результаты работы программы 2.4.

Команда 15 запрещает печать нулевых значений при контрольной распечатке матриц A и B, что достигается включением флага NERO.

Оператор присваивания 20 задает значение используемой вспомогательной переменной N, равное количеству тринадцати уравнений равновесия в данной задаче.

Многократные операторы присваивания 22 и 24 задают значения тригонометрических функций угла α. В них каждое выражение принимает значение, вычисленное для самой правой части.

Используя такую возможность, операторы присваивания 35–55 ввода данных программы 2.4 можно представить в виде, облегчающем их запись.

Теперь вместо тригонометрических функций просто используются переменные, идентификаторы которых для простоты состоят из первых букв обозначений самой функции и используемого угла.

Оператор присваивания 29 задает значение силе P из условия задачи, делая ее связанной переменной при записи элементов матрицы-столбца B (правых частей уравнений равновесия (2.41)).

Оператор 30 описывает матричные переменные с явным заданием ее размерностей:

- матрицу A с числом строк N и столбцов N=13, для которой обеспечивается резервирование памяти для хранения 13×13=169 значений элементов A;
- матрицы-столбцы B и X, содержащих по N=13 элементов каждый.

Операторы присваивания 35 – 50 и 55 вводят ненулевые элементы соответственно матрицы A и матрицы-столбца B в символьной форме, которые задаются с использованием обозначений для переменных из операторов 22 и 24.

Контрольная печать введенных исходных данных будет получена автоматически применением терминатора ";".

Команда 80 разрешает печать нулевых значений перед решением СЛАУ, что достигается выключением флага NERO. Это сделано для возможности распечатки всех значений матрицы-столбца X.

Оператор 85 выполняет непосредственное решение СЛАУ задачи статики и печать выходной информации (значений элементов матрицы-столбца X(I,1)).

Команда 95 закрывает выводной файл PR2-4.LIS, находящийся в подкаталоге \C1 корневого каталога текущего диска, куда записывались результаты работы программы 2.4.

Команда 99 стандартным способом заканчивает файл PR2-4, предназначенный для считывания.

Результаты решения после окончания работы программы 2.4 (вместе со всей предусмотренной в ней печатью) окажутся в файле PR2-4.LIS, находящемся в подкаталоге \C1 корневого каталога текущего диска, и будут иметь для рассматриваемого примера следующие значения:

$$X(1,1):=\frac{11}{sqrt(3)} \qquad X(2,1):=-\frac{22}{sqrt(3)}$$

$$X(3,1):=11 \qquad X(4,1):=\frac{22}{sqrt(3)}$$

$$X(5,1):=-\frac{22}{sqrt(3)} \qquad X(6,1):=-\frac{11}{sqrt(3)}$$

$$X(7,1):=11 \qquad X(8,1):=\frac{33}{sqrt(3)} \qquad (2.44)$$

$$X(9,1):=-\frac{22}{sqrt(3)} \qquad X(10,1):=-\frac{22}{sqrt(3)}$$

$$X(11,1):=\frac{33}{sqrt(3)} \qquad X(12,1):=-11$$

$$X(13,1):=\frac{33}{sqrt(3)}$$

Расположение элементов матриц на экране дисплея, как отмечалось в п. 2.1, будет зависеть от используемой версии REDUCE.

Дополнение 2.9. Ввод исходных данных можно также организовать с использованием операторов MAT 35 и 55, в которых элементы матрицы A(I,J) и матрицы-столбца B(I,1) задаются в *полном виде* по строкам.

Это придется сделать, если вам:

- не хочется отягощать свою память кажущейся вам лишней информацией дополнения 2.1;
- применение операторов присваивания для ввода *только ненулевых значений* элементов соответствующих матриц с указанием их расположения вы считаете, в отличие от нас, неудобным.

В этом случае вместо операторов присваивания 35 – 50 и 55 достаточно только представить в программе 2.4 операторы 35 и 55 соответственно по фрагментам (2.42) и (2.43).

Это пример как бы обратного применения дополнения 2.1. Надеемся, что вы теперь в состоянии сами осознанно применить к программе 2.4 и все остальные дополнения 2.2 – 2.8 в прямом или обратном виде, все ближе на практике знакомясь с возможностями CAB REDUCE.

Например, для представления результатов решения в приближенном виде с использованием действительных чисел, что важно для сравнения полученных результатов с вашим аналитическим решением, нужно использовать дополнение 2.4.

Для этого в программе 2.4 следует:

- установить флаги BIGFLOAT и NUMVAL оператором 87:

```
ON BIGFLOAT, NUMVAL;                                    87
```

- распечатать результаты решения оператором 88:

```
X;                                                      88
```

- отменить установленные на время режимы оператором 89:

```
OFF BIGFLOAT, NUMVAL;                                   89
```

После работы такой дополненной программы 2.4 результаты решения (2.44) предстанут в виде действительных чисел (только вместо идентификатора X будет использоваться MAT):

MAT(1,1):= 6.351039261 MAT(2,1):= – 12.70207852
MAT(3,1):= 11 MAT(4,1):= 12.70207852
MAT(5,1):= – 12.70207852 MAT(6,1):= – 6.351039261
MAT(7,1):= 11 MAT(8,1):= 19.05311778 (2.45)
MAT(9,1):= – 12.70207852 MAT(10,1):= – 12.70207852
MAT(11,1):= 19.05311778 MAT(12,1):= –11
MAT(13,1):= 19.05311778

2.3.3. Исследование влияния вариации нагрузки

Исследуем влияние вариации нагрузки на значения реакций опор и усилий в стержнях плоской шарнирно-стержневой конструкции, изображенной на рис. 2.9,а.

Сначала ***исследуем их зависимость от значения силы P***. Для этого сделаем ее свободной переменной, удалив из программы 2.4 оператор 29, задающий силе P конкретное значение, а также очистим для надежности ее значение командой

```
CLEAR P;                                                29
```

Теперь после работы измененной программы 2.4 мы получим значения усилий в стержнях рассматриваемой плоской конструкции в зависимости от значения силы P:

$$X(1,1):=\frac{P}{sqrt(3)} \qquad X(2,1):=-\frac{2*P}{sqrt(3)}$$

$$X(3,1):=P \qquad X(4,1):=\frac{2*P}{sqrt(3)}$$

$$X(5,1):=-\frac{2*P}{sqrt(3)} \qquad X(6,1):=-\frac{P}{sqrt(3)}$$

$$X(7,1):=P \qquad X(8,1):=\frac{3*P}{sqrt(3)} \qquad (2.46)$$

$$X(9,1):=-\frac{2*P}{sqrt(3)} \qquad X(10,1):=-\frac{2*P}{sqrt(3)}$$

$$X(11,1):=\frac{3*P}{sqrt(3)} \qquad X(12,1):=-P$$

$$X(13,1):=\frac{3*P}{sqrt(3)}$$

Их графики представляют собой прямые линии, выходящие из начала координат, с тангенсами углов наклона, равным коэффициентам перед силой P в соотношениях (2.46).

Чтобы исследовать влияние вариации угла поворота силы на значения реакций опор и усилий в стержнях рассматриваемой плоской шарнирно-стержневой конструкции, силу P нужно представить в произвольном положении.

Для этого ее следует от горизонтального положения на рис. 2.9а, которое будем считать начальным, повернуть против часовой стрелки на небольшой угол γ (рис. 2.9б).

В результате уравнения равновесия 1 и 2 системы (2.39) примут следующий вид:

1. $P*\cos\gamma + S_2*\cos\alpha = 0;$

2. $P*\sin\gamma - S_1 - S_2*\sin\alpha = 0;$

Выполним вышеописанные действия по их формализации:

- заменим используемые греческие буквы для обозначения углов α и γ их идентификаторами, записанными в латинской транскрипции (α = ALFA, γ = GAMMA),

- перенесем свободные члены, не содержащие неизвестных, в правые части уравнений 1 и 2, которые с учетом соответствия идентификаторов (2.40) теперь примут следующий измененный вид в системе (2.41):

```
1. X2*cos(ALFA) = -P*COS(GAMMA)
2. -X1 - X2*sin(ALFA) = -P*SIN(GAMMA)
```

Теперь оператор 55 программы 2.4 должен быть представлен в форме:

```
B(1,1):=-P*COS(GAMMA); B(2,1):=-P*SIN(GAMMA);          55
```

Имена со скобками SIN(GAMMA) и COS(GAMMA) можно условно рассматривать в качестве свободных переменных, так как в программе 2.4 им не присваивалось никаких значений.

Для большей надежности работы программы предварительно очистим их командой CLEAR:

```
CLEAR SIN(GAMMA),COS(GAMMA);                           27
```

В результате работы такой измененной программы 2.4 мы получим значения реакций опор и усилий в стержнях плоской конструкции в зависимости от вариации направления постоянной по модулю силы P, определяемого изменением угла γ (рис. 2.9,б):

$$X(1,1):=\frac{11*(\cos(gamma)+\mathrm{sqrt}(3)*\sin(gamma))}{\mathrm{sqrt}(3)}$$

$$X(2,1):=-\frac{22*\cos(gamma)}{\mathrm{sqrt}(3)}$$

$$X(3,1):=11*\cos(gamma)$$

$$X(4,1):=\frac{11*(2*\cos(gamma)+\mathrm{sqrt}(3)*\sin(gamma))}{\mathrm{sqrt}(3)}$$

$$X(5,1):=-\frac{22*\cos(gamma)}{\mathrm{sqrt}(3)}$$

$$X(6,1):=-\frac{11*\cos(gamma)}{\mathrm{sqrt}(3)}$$

$$X(7,1):=11*\cos(gamma) \qquad (2.47)$$

$$X(8,1):=\frac{11*(3*\cos(gamma)+\mathrm{sqrt}(3)*\sin(gamma))}{\mathrm{sqrt}(3)}$$

$$X(9,1) := -\frac{22 * \cos(gamma)}{sqrt(3)}$$

$$X(10,1) := -\frac{22 * \cos(gamma)}{sqrt(3)}$$

$$X(11,1) := \frac{11 * (3 * \cos(gamma) + sqrt(3) * \sin(gamma))}{sqrt(3)}$$

$$X(12,1) := -11 * \cos(gamma)$$

$$X(13,1) := \frac{33 * \cos(gamma)}{sqrt(3)}$$

Нижеследующие графики наглядно показывают эти зависимости. Как видно из их сравнения, значения реакций опор и усилий в стержнях плоской конструкции, изображенной на рис. 2.9б, при вращении постоянной по модулю силы P в той же плоскости также изменяются по синусоидальному закону:

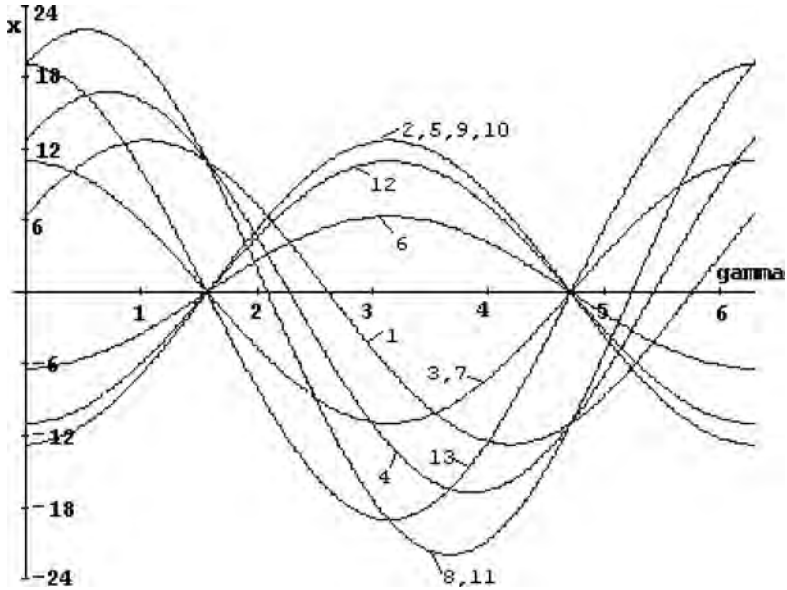

Рис. 2.10. Зависимости усилий в стержнях и реакций опор (X, кН) плоской конструкции (рис. 2.9б) от изменения угла GAMMA *(рад) при вращении силы P: 1 – S_1, 2 – S_2, 3 – S_3, 4 – S_4, 5 – S_5, 6 – S_6, 7 – S_7, 8 – S_8, 9 – S_9, 10 – S_{10}, 11 – Ra, 12 – Xb, 13 – Yb*

2.3.4. Исследование влияния вариации геометрических факторов

Изучим влияние вариации геометрических факторов рассматриваемой плоской фермы на примере изменения положения ее левой шарнирно-неподвижной опоры в точке A, с которой связан промежуточный элемент в виде 8-го стержня. Его положение при вариации будет определяться углом φ (с идентификатором FI), отсчитываемом от вертикальной оси (начального положения) против часовой стрелки (рис. 2.9,б). Точка A при этом вместе с 8-м стержнем будет вращаться вокруг точки D, а угол φ изменяться от 0 до 2π радиан.

Реакция Ra будет также направлена вдоль 8-го стержня при любом его положении. Однако теперь для определения ее модуля и направления удобнее ввести проекции Xa и Ya, направленные в сторону положительного направления осей координат (рис. 2.9,б).

Теперь для составления уравнений равновесия, позволяющих исследовать влияние вариации положения левой шарнирно-неподвижной опоры, рассматриваемую плоскую ферму нужно представить в произвольном положении. Для этого 8-й стержень следует от вертикального положения (рис. 2.9,а) повернуть против часовой стрелки на небольшой угол φ (рис. 2.9,б). В результате изменятся уравнения равновесия 9-11 системы (2.39) и к ней добавится 14-е уравнение:

$$
\begin{aligned}
&9.\ S_7 + S_9\cos\alpha + S_8\sin\varphi = 0,\\
&10.\ S_4' - S_8\cos\varphi - S_9\sin\alpha = 0,\\
&11.\ Xa - S_8'\sin\varphi = 0,\\
&14.\ Ya + S_8'\cos\varphi = 0.
\end{aligned}
\qquad (2.48)
$$

Внесем также изменения в соответствие идентификаторов (2.40), положив в них X_{11} = Xa и X_{14} = Ya. Введем обозначения α = ALFA, φ = FI. С учетом этого формализация системы (2.48) будет иметь следующий вид:

$$
\begin{aligned}
&9.\ X_7 + X_9\cos(ALFA) + X_8\sin(FI) = 0,\\
&10.\ X_4 - X_8\cos(FI) - X_9\sin(ALFA) = 0,\\
&11.\ X_{11} - X_8\sin(FI) = 0,\\
&14.\ X_{14} + X_8\cos(FI) = 0.
\end{aligned}
\qquad (2.49)
$$

Теперь в программе 2.4 следует соответствующим образом изменить предложения 20 и 45–52. Они задают с помощью операторов присваивания значение N и ненулевые элементы фрагмента (2.49). В них для удобства жирным шрифтом выделены отличающиеся от программы 2.4 элементы:

```
N:=14;                                                    20
A(8,10):=-1; A(9,7):=1; A(9,9):=CA;  A(10,4):=1;          45
A(10,8):=-COS(FI); A(10,9):=-SA; A(11,8):=-SIN(FI);       47
A(11,11):=1; A(12,9):=-CA; A(12,12):=1;                   48
A(13,9):=SA; A(13,10):= 1; A(13,13):=1;                   50
A(9,8):=SIN(FI); A(14,8):=COS(FI); A(14,14):=1;           52
```

Очистим для большей надежности работы программы имена со скобками SIN(FI) и COS(FI) командой CLEAR:

```
CLEAR SIN(FI),COS(FI);                                    27
```

В результате работы такой измененной программы 2.4 мы получим значения усилий в стержнях плоской конструкции (рис. 2.9,б) в зависимости от вариации положения ее левой шарнирно-неподвижной опоры:

$$X(1,1):=\frac{11*(\mathrm{sqrt}(3)*\cos(fi)-\sin(fi))}{3*\cos(fi)-\mathrm{sqrt}(3)*\sin(fi)}$$

$$X(2,1):=-\frac{22*(\mathrm{sqrt}(3)*\cos(fi)-\sin(fi))}{3*\cos(fi)-\mathrm{sqrt}(3)*\sin(fi)}$$

$$X(3,1):=11 \qquad X(4,1):=\frac{22*(\mathrm{sqrt}(3)*\cos(fi)-\sin(fi))}{3*\cos(fi)-\mathrm{sqrt}(3)*\sin(fi)}$$

$$X(5,1):=-\frac{22*(\mathrm{sqrt}(3)*\cos(fi)-\sin(fi))}{3*\cos(fi)-\mathrm{sqrt}(3)*\sin(fi)}$$

$$X(6,1):=-\frac{11*(\mathrm{sqrt}(3)*\cos(fi)-\sin(fi))}{3*\cos(fi)-\mathrm{sqrt}(3)*\sin(fi)} \qquad (2.50)$$

$$X(7,1):=11 \qquad X(8,1):=\frac{33}{\mathrm{sqrt}(3)*\cos(fi)-\sin(fi)}$$

$$X(9,1):=-\frac{22*(\mathrm{sqrt}(3)*\cos(fi)+2*\sin(fi))}{3*\cos(fi)-\mathrm{sqrt}(3)*\sin(fi)}$$

$$X(10,1):=-\frac{22*(\mathrm{sqrt}(3)*\cos(fi)-\sin(fi))}{3*\cos(fi)-\mathrm{sqrt}(3)*\sin(fi)}$$

$$X(11,1):=\frac{33*\sin(fi)}{\mathrm{sqrt}(3)*\cos(fi)-\sin(fi)}$$

$$X(12,1) := -\frac{11*(\mathrm{sqrt}(3)*\cos(fi)+2*\sin(fi))}{\mathrm{sqrt}(3)*\cos(fi)-\sin(fi)}$$

$$X(13,1) := \frac{33*\cos(fi)}{\mathrm{sqrt}(3)*\cos(fi)-\sin(fi)}$$

$$X(14,1) := -\frac{33*\cos(fi)}{\mathrm{sqrt}(3)*\cos(fi)-\sin(fi)}$$

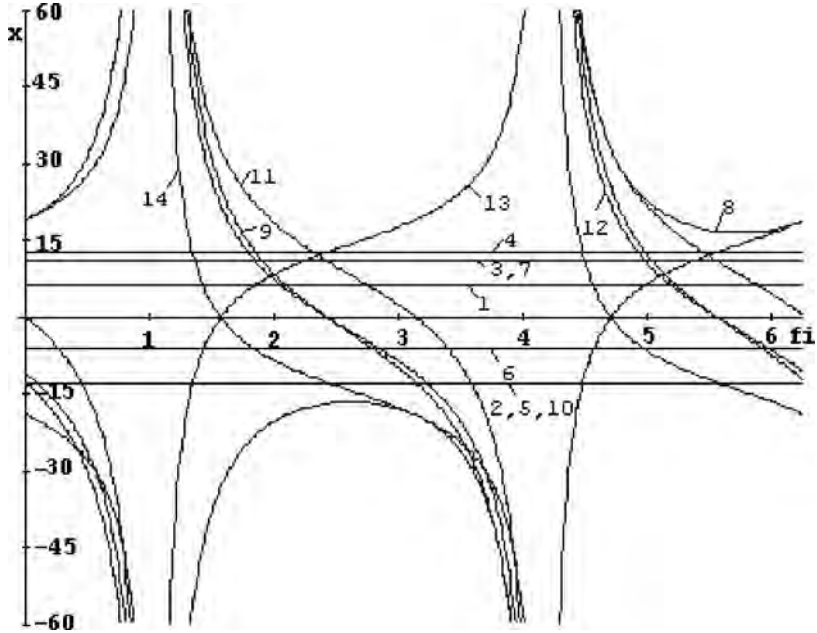

Рис. 2.11. Зависимости усилий в стержнях и реакций опор (X, кН) плоской конструкции (рис. 2.9б) от вариации положения шарнирно-неподвижной опоры в точке A, определяемом углом FI (рад): 1 – S_1, 2 – S_2, 3 – S_3, 4 – S_4, 5 – S_5, 6 – S_6, 7 – S_7, 8 – S_8, 9 – S_9, 10 – S_{10}, 11 – Xa, 12 – Xb, 13 – Yb, 14 – Ya

Графики на рис. 2.11 наглядно показывают эти зависимости. Как видно из их сравнения, значения усилий в стержнях плоской конструкции при изменении положения опоры в той же плоскости изменяются немонотонным образом. При значениях $\varphi = \pi/3$ и $\varphi = 4\pi/3$ конструкция находится в неустойчивом положении. При этих значениях угла φ происходит потеря равновесия и устойчивости конструкции, что математически находит свое

отражение в стремлении к бесконечности соответствующих реакций, в выражениях для которых при $\varphi = \pi/3$ или $\varphi = 4\pi/3$ происходит деление на 0.

Это очень хорошо видно на графиках (рис. 2.11) зависимости значений реакций стержней 8, 9 и опор 11–14 от величины угла φ, построенных по уравнениям (2.50).

Теперь ***получим соотношения, описывающие значения реакций при вариации опоры одновременно с изменением положения и модуля действующей силы P.***

Для этого следующим образом ***изменим программу 2.4***:

- полностью очистим все используемые переменные и имена со скобками командой 27:

```
CLEAR P, SIN(FI), COS(FI), SIN(GAMMA), COS(GAMMA);      27
```

- удалим оператор 29;
- представим оператор 55 в следующей форме:

```
B(1,1):=-P*COS(GAMMA); B(2,1):=-P*SIN(GAMMA);           55
```

- изменим соответствующим образом предложения 20 и 45–52. Они задают с помощью операторов присваивания значение N и ненулевые элементы фрагмента (2.49). В них для удобства жирным шрифтом выделены отличающиеся от программы 2.4 элементы:

```
N:=14;                                                  20
A(8,10):=-1; A(9,7):=1; A(9,9):=CA;  A(10,4):=1;        45
A(10,8):=-COS(FI); A(10,9):=-SA; A(11,8):=-SIN(FI);     47
A(11,11):=-1; A(12,9):=-CA; A(12,12):=1;                48
A(13,9):=SA; A(13,10):= 1; A(13,13):=1;                 50
A(9,8):=SIN(FI); A(14,8):=COS(FI); A(14,14):=1;         52
```

Теперь после работы такой модифицированной программы 2.4 получим соотношения (2.51), описывающие значения реакций при *вариации опоры одновременно с изменением положения и модуля действующей силы P*.

$$X(1,1):= (p*(sqrt(3)*cos(fi)*cos(gamma) + 3*cos(fi)*sin(gamma) - cos(gamma)*sin(fi) - sqrt(3)*sin(fi)*sin(gamma))) / (3*cos(fi) - sqrt(3)*sin(fi))$$

$$X(2,1):= -\frac{2*\cos(gamma)*p*(sqrt(3)*\cos(fi) - \sin(fi))}{3*\cos(fi) - sqrt(3)*\sin(fi)}$$

X(3,1):= cos(gamma)*p,

X(4,1):= (p*(2*sqrt(3)*cos(fi)*cos(gamma) + 3*cos(fi)*sin(gamma) –
2*cos(gamma)*sin(fi) – sqrt(3)*sin(fi)*sin(gamma))) /
(3*cos(fi) – sqrt(3)*sin(fi))

$$X(5,1):= -\frac{2 * \cos(gamma) * p * (sqrt(3) * \cos(fi) - \sin(fi))}{3 * \cos(fi) - sqrt(3) * \sin(fi)}$$

$$X(6,1):= -\frac{\cos(gamma) * p * (sqrt(3) * \cos(fi) - \sin(fi))}{3 * \cos(fi) - sqrt(3) * \sin(fi)}$$

X(7,1):= cos(gamma)*p (2.51)

$$X(8,1):= \frac{p * (3 * \cos(gamma) + sqrt(3) * \sin(gamma))}{sqrt(3) * \cos(fi) - \sin(fi)}$$

X(9,1):= – (2*p*(sqrt(3)*cos(fi)*cos(gamma) + 2*cos(gamma)*sin(fi) +
sqrt(3)*sin(fi)*sin(gamma))) / (3*cos(fi) - sqrt(3)* sin(fi))

$$X(10,1):= -\frac{2 * \cos(gamma) * p * (sqrt(3) * \cos(fi) - \sin(fi))}{3 * \cos(fi) - sqrt(3) * \sin(fi)}$$

$$X(11,1):= \frac{\sin(fi) * p * (3 * \cos(gamma) + sqrt(3) * \sin(gamma))}{sqrt(3) * \cos(fi) - \sin(fi)}$$

X(12,1):= – (p*(sqrt(3)*cos(fi)*cos(gamma) + 2*cos(gamma)*sin(fi) +
sqrt(3)*sin(fi)*sin(gamma))) / (sqrt(3)*cos(fi) – sin(fi))

$$X(13,1):= \frac{p * (3 * \cos(fi) * \cos(gamma) + \sin(fi) * \sin(gamma))}{sqrt(3) * \cos(fi) - \sin(fi)}$$

$$X(14,1):= -\frac{\cos(fi) * p * (3 * \cos(gamma) + sqrt(3) * \sin(gamma))}{sqrt(3) * \cos(fi) - \sin(fi)}$$

Как уже отмечалось, выражения такого общего вида позволяют легко получать значения реакций при любом наборе данных для всей конструкции и действующей нагрузки, проверяя наши предположения, сделанные на основании анализа вариаций каких-либо отдельных факторов.

2.4. Многовариантные задачи

2.4.1. Формализация уравнений

В качестве примера многовариантной задачи статики со "средним" количеством уравнений равновесия воспользуемся аналитическим решением типового задания по определению реакций опор составных конструкций с внутренними односторонними связями для системы трех тел *(задание С-9 [23, с. 50-59])*.

Его расчетная схема приведена на рис. 2.12,а.

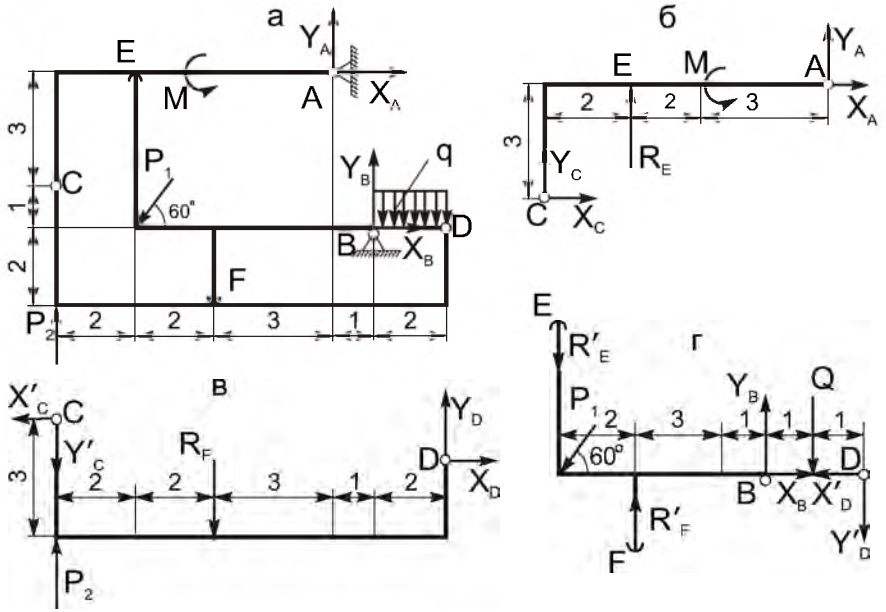

Рис. 2.12. Расчетная схема плоской составной конструкции с внутренними односторонними связями (а) и ее составных частей (б–г)

В задаче рассматривается два случая, когда "работает" какая-либо одна из односторонних связей: Re или Rf, а вторая равна нулю (рис. 2.12,б–г). Соответственно этому необходимо решить на ПК две системы уравнений равновесия (14) и (24) [23, с. 56 и 57], которые имеют следующий вид:

- **1 случай (Rf = 0):**

```
1. Re*2 + M – Xa*3 + Ya*7 = 0,
2. Xa + Xc = 0,
3. Re + Ya + Yc = 0,
4. Xc'*1 + Yc'*10 – P2*10 = 0,
5. -Xc' + Xd = 0,                                    (2.52)
6. -Yc' + P2 + Yd =0,
7. Re*6 + 6*P1*sin60° – Q*1 – Yd'*2 = 0,
8. Xв – Xd' – P1*cos60° = 0,
9. -Re – P1*sin60° + Yв – Q – Yd' = 0;
```

- **2 случай (Re = 0):**

```
1. M – Xa*3 + Ya*7 = 0,
2. Xa + Xc = 0,
3. Ya + Yc = 0,
4. Rf*6 + Xc'*1 + Yc'*10 – P2*10 = 0,
5. -Xc' + Xd = 0,                                    (2.53)
6. -Rf – Yc' + P2 + Yd =0,
7. -Rf*4 + 6*P1*sin60° – Q*1 – Yd'*2 = 0,
8. Xв – Xd' – P1*cos60° = 0,
9. Rf – P1*sin60° + Yв – Q – Yd' = 0.
```

Здесь уравнения пронумерованы сверху вниз, для каждой СЛАУ начиная с 1, где номером без скобок обозначен порядковый номер строки в системе уравнений.

Рассмотрим подготовку исходных данных для численного решения СЛАУ (2.52) и (2.53), отличающихся друг от друга только использованием одной неизвестной величины: (Re или Rf). Составим соответствие идентификаторов для СЛАУ (2.52), присвоив значение реакции Re=Re'=X_1. Для СЛАУ (2.53) оно будет таким же, только X_1 вместо Re приравнивается Rf: Rf=Rf'=X_1.

Соответствие идентификаторов для СЛАУ (2.52):

X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	
Re=Re'	Xa	Ya	Xв	Yв	Xd=Xd'	Yd=Yd'	Xc=Xc'	Yc=Yc'	(2.54)

Для удобства составления программ обозначим также идентификатором ALFA используемый угол 60^0: ALFA = 60^0. Перепишем СЛАУ (2.52)–(2.53) с учетом таблицы идентификаторов, перенеся все члены, не содержащие неизвестных, в правую часть и подсчитаем их с учетом условия задачи (P_1=10 кН, P_2=3 кН, M=20 кН.м, Q=2 кН):

$$
\begin{aligned}
&1.\ 2*X_1 - 3*X_2 + 7*X_3 = -M = -20,\\
&2.\ X_2 + X_8 = 0,\\
&3.\ X_1 + X_3 + X_9 = 0,\\
&4.\ X_8 + 10*X_9 = 10*P_2 = 30,\\
&5.\ X_6 - X_8 = 0,\\
&6.\ X_7 - X_9 = -P_2 = -3,\\
&7.\ 6*X_1 - 2*X_7 = Q - 6*P_1*\sin(ALFA) = 2 - 30*\sqrt{3},\\
&8.\ X_4 - X_6 = P_1*\cos(ALFA) = 5,\\
&9.\ -X_1 + X_5 - X_7 = Q+P_1*\sin(ALFA) = 2 + 5*\sqrt{3};
\end{aligned}
\qquad (2.55)
$$

$$
\begin{aligned}
&1.\ -3*X_2 + 7*X_3 = -20,\\
&2.\ X_2 + X_8 = 0,\\
&3.\ X_3 + X_9 = 0,\\
&4.\ 6*X_1 + X_8 + 10*X_9 = 30,\\
&5.\ X_6 - X_8 = 0,\\
&6.\ -X_1 + X_7 - X_9 = -3,\\
&7.\ -4*X_1 - 2*X_7 = Q - 6*P_1*\sin(ALFA) = 2 - 30*\sqrt{3},\\
&8.\ X_4 - X_6 = 5,\\
&9.\ X_1 + X_5 - X_7 = Q+P_1*\sin(ALFA) = 2 + 5*\sqrt{3}.
\end{aligned}
\qquad (2.56)
$$

2.4.2. Решение в виде рациональных и вещественных чисел

Представим формализованные СЛАУ (2.55) и (2.56) в матричной форме соответственно в виде:

$$(A_1)\ \ (X_1)=B_1, \qquad (2.57)$$

$$(A_2)\ \ (X_2)=B_2. \qquad (2.58)$$

Теперь системы уравнений (2.55) и (2.56) соответственно приобрели явный вид СЛАУ (2.57) и (2.58), в которой выписаны только отличные от нуля элементы. Из их сравнения видно, что матрицы A_1 и A_2 отличаются только первым столбцом (“работает” реакция Re или Rf), а матрицы-столбцы B_1 и B_2 совпадают, так как нагрузка не меняется.

Для дальнейшего их решения или исследования с помощью ПК в системе REDUCE имеются различные возможности. Рассмотрим сначала их самое простое решение, когда матрицы A_1 и A_2 задаются независимо

друг от друга, причем покажем как использование операторов присваивания только для ненулевых значений элементов, так и применение для этой цели операторов MAT.

Для задания только ненулевых значений элементов соответствующих матриц $A_1(I,J)$, $A_2(I,J)$ и матрицы-столбца $B_1(I,1)$ следует организовать ввод исходных данных с использованием операторов присваивания. Предварительно следует определить согласно дополнению 2.1 положение и значения всех ненулевых элементов матриц A_1, A_2 и B_1.

Применение операторов присваивания позволяет вводить только их ненулевые значения с указанием их расположения. При этом нулевые значения можно не указывать, так как все элементы матриц $A_1(I,J)$, $A_2(I,J)$ и матрицы-столбца $B_1(I,1)$, размерность которых была описана оператором 30, равны 0. Поэтому достаточно только задать в программе 2.5 их ненулевые элементы.

Теперь структура простейшей программы п. 2.1 для решения двухвариантных систем уравнений (2.55) и (2.56) в системе REDUCE может быть реализована, например, следующим образом (по вопросам записи программы 2.5 в файле PR2-5 в подкаталоге \C9 корневого каталога текущего диска и запуске ее на выполнение см. п. 7.3 и пример в п. 1.4.2):

```
COMMENT Программа 2.5: решение типового примера С-9 на REDUCE
с описанием только ненулевых элементов матриц A1, A2 и матри-
цы-столбца B1, копированием B2=B1 с представлением данных
ввода и результатов в численном виде;
OUT "\C9\PR2-5.LIS";                                          10
ON NERO;                                                      15
N:=9;                                                         20
SIN(ALFA):= SQRT(3)/2;                                        22
COS(ALFA):= 1/2;                                              24
P1:=10; P2:=3; Q:=2;                                          29
MATRIX A1(N,N),B1(N,1),X1(N,1),                               30
 A2(N,N),B2(N,1),X2(N,1);
A1(1,1):=2; A1(1,2):=-3; A1(1,3):=7; A1(2,2):=1;              35
A1(2,8):=1; A1(3,1):= 1; A1(3,3):=1; A1(3,9):=1;              37
A1(4,8):=1; A1(4,9):=10; A1(5,6):=1; A1(5,8):=-1;             40
A1(6,7):=1; A1(6,9):=-1; A1(7,1):=6; A1(7,7):=-2;             42
A1(8,4):=1; A1(8,6):=-1; A1(9,1):=-1;                         45
A1(9,5):=1; A1(9,7):=-1;                                      47
```

```
B1(1,1):=-20; B1(4,1):=30; B1(6,1):=-P2;                          55
B1(7,1):=Q-6*P1*SIN(ALFA); B1(8,1):= P1*COS(ALFA);                57
B1(9,1):=Q+P1*SIN(ALFA);                                          58
A2(1,2):=-3; A2(1,3):= 7; A2(2,2):= 1; A2(2,8):=1;                60
A2(3,3):= 1; A2(3,9):= 1; A2(4,1):= 6; A2(4,8):=1;                62
A2(4,9):=10; A2(5,6):= 1; A2(5,8):=-1; A2(6,1):=-1;               65
A2(6,7):= 1; A2(6,9):=-1; A2(7,1):=-4; A2(7,7):=-2;               67
A2(8,4):= 1; A2(8,6):=-1; A2(9,1):=1;                             70
A2(9,5):= 1; A2(9,7):=-1;                                         72
B2:=B1;                                                           75
OFF NERO;                                                         80
X1:=A1**(-1)*B1;                                                  85
X2:=A2**(-1)*B2;                                                  86
SHUT "\C9\PR2-5.LIS";                                             95
END;                                                              99
```

Команда 10 открывает выводной файл PR2-5.LIS, находящийся в подкаталоге \C9 корневого каталога текущего диска, куда будут записываться результаты работы программы 2.5.

Команда 15 запрещает печать нулевых значений при контрольной распечатке матриц A_1, A_2 и матриц-столбцов B_1 и B_2, что достигается включением флага NERO.

Оператор присваивания 20 задает значение используемой вспомогательной переменной N, равное количеству девяти уравнений равновесия в данной задаче.

Операторы присваивания 22 и 24 задают значения тригонометрических функций угла α, обозначенного посредством идентификатора ALFA.

Операторы присваивания 29 задают значения силам P_1, P_2 и Q из условия задачи, делая их связанными переменными при записи элементов матрицы-столбца B_1 (правых частей уравнений равновесия (2.55)).

Оператор 30 описывает матричные переменные с явным заданием их размерностей:

- матрицы A_1 и A_2 с числом строк N и столбцов N=9, для которых обеспечивается резервирование памяти для хранения 9×9=81 значений элементов для каждой из них;
- матрицы-столбцы B_1, B_2 и X_1, X_2 содержащие по N=9 элементов каждый.

Операторы присваивания 35 – 47 и 55 – 58 вводят ненулевые элементы соответственно матрицы A_1 и матрицы-столбца B_1 в символьной форме, которые задаются с использованием обозначений для переменных из операторов 22, 24 и 29. Контрольная печать введенных исходных данных будет получена автоматически применением терминатора ";".

Операторы 60 – 72 аналогично присваивают ненулевым элементам матрицы A_2 соответствующие значения.

Оператор присваивания 75 приравнивает между собой матрицы-столбцы B_2 и B_1, так как нагрузка не меняется и они одинаковы. В результате этого каждый элемент B_2 получает значение B_1. Такой элегантный и простой способ копирования матриц лишний раз подчеркивает достоинства системы REDUCE.

В любом из языков программирования высокого уровня, например Фортране, такая операция может быть выполнена только с использованием соответствующего цикла. Он с помощью арифметического оператора присваивания для всех N=9 элементов матриц-столбцов присвоит значение каждого элемента $B_1(I,1)$ соответствующему элементу $B_2(I,1)$: $B_2(1,1) = B_1(1,1)$, $B_2(2,1) = B_1(2,1)$,..., $B_2(9,1) = B_1(9,1)$.

Команда 80 разрешает печать нулевых значений перед решением СЛАУ, что достигается выключением флага NERO. Это сделано для возможности распечатки всех значений матриц-столбцов X_1 и X_2.

Операторы 85 и 86 выполняют непосредственное решение СЛАУ (2.55) и (2.56) соответственно и печать выходной информации (значений элементов матриц-столбцов $X_1(I,1)$ и $X_2(I,1)$).

Команда 95 закрывает выводной файл PR2-5.LIS, находящийся в подкаталоге \C9 корневого каталога текущего диска, куда записывались результаты работы программы 2.5, а команда 99 стандартным способом заканчивает файл PR2-5, предназначенный для считывания.

Для задания элементов матриц $A_1(I,J)$, $A_2(I,J)$ и матрицы-столбца $B_1(I,1)$ ***в полном виде по строкам:***

- уравнения (2.55) и (2.56) следует дополнить нулями до полных форм 9×9 и 9×1;
- из программы 2.5 нужно удалить операторы 35 – 47, 55 – 58 и 60 – 72;
- после этого все значения элементов используемых матриц следует задать по строкам в соответствующих операторах MAT, что будет иметь для программы 2.5 следующий вид:

```
A1:=MAT((2,-3,7,0,0,0,0,0,0),                                    35
 (0,1,0,0,0,0,0,1,0),
 (1,0,1,0,0,0,0,0,1),
 (0,0,0,0,0,0,0,1,10),
 (0,0,0,0,0,1,0,-1,0),
 (0,0,0,0,0,0,1,0,-1),
 (6,0,0,0,0,0,-2,0,0),
 (0,0,0,1,0,-1,0,0,0),
 (-1,0,0,0,1,0,-1,0,0));                                        (2.59)
B1:=MAT((-20),(0),(0),(30),(0),(-P2),                            55
 (Q-6*P1*SIN(ALFA)),(P1*COS(ALFA)),(Q+P1*SIN(ALFA)));
A2:=MAT((0,-3,7,0,0,0,0,0,0),                                    60
 (0,1,0,0,0,0,0,1,0),
 (0,0,1,0,0,0,0,0,1),
 (6,0,0,0,0,0,0,1,10),
 (0,0,0,0,0,1,0,-1,0),
 (-1,0,0,0,0,0,1,0,-1),
 (-4,0,0,0,0,0,-2,0,0),
 (0,0,0,1,0,-1,0,0,0),
 (1,0,0,0,1,0,-1,0,0));
```

В обоих случаях результаты решения после окончания работы программы 2.5 (вместе со всей предусмотренной в ней печатью) окажутся в файле PR2-5.LIS, находящемся в подкаталоге \C9 корневого каталога текущего диска, и будут иметь для рассматриваемого примера следующие значения:

$$X1(1,1):=-\frac{3*(185*\mathrm{sqrt}(3)-12)}{116} \qquad X1(2,1):=\frac{5*(75*\mathrm{sqrt}(3)-8)}{58}$$

$$X1(3,1):=\frac{2*(60*\mathrm{sqrt}(3)-47)}{29} \qquad X1(4,1):=-\frac{15*(25*\mathrm{sqrt}(3)-22)}{58}$$

$$X1(5,1):=\frac{5*(5*\mathrm{sqrt}(3)+13)}{29} \qquad X1(6,1):=-\frac{5*(75*\mathrm{sqrt}(3)-8)}{58}$$

$$X1(7,1):=\frac{75*\mathrm{sqrt}(3)-8}{116} \qquad X1(8,1):=-\frac{5*(75*\mathrm{sqrt}(3)-8)}{58}$$

$$X1(9,1):=\frac{5*(15*\mathrm{sqrt}(3)+68)}{116} \qquad (2.60)$$

$$X2(1,1):=\frac{185*\mathrm{sqrt}(3)-12}{31} \qquad X2(2,1):=\frac{2*(105*\mathrm{sqrt}(3)-11)}{31}$$

$$X2(3,1):=\frac{2*(45*\mathrm{sqrt}(3)-49)}{31} \qquad X2(4,1):=-\frac{3*(70*\mathrm{sqrt}(3)-59)}{31}$$

$$X2(5,1):=\frac{65*\mathrm{sqrt}(3)+67}{31} \qquad X2(6,1):=-\frac{2*(105*\mathrm{sqrt}(3)-11)}{31}$$

$$X2(7,1):=\frac{95*\mathrm{sqrt}(3)-7}{31} \qquad X2(8,1):=-\frac{2*(105*\mathrm{sqrt}(3)-11)}{31}$$

$$X2(9,1):=-\frac{2*(45*\mathrm{sqrt}(3)-49)}{31}$$

Используем дополнение 2.4 и представим результаты решения (2.60) в приближенном виде с использованием действительных чисел, что нужно для сравнения полученных результатов с вашим аналитическим или численным решением на ПК. Для этого в программу 2.5 следует внести следующие дополнения:

- установить флаги BIGFLOAT и NUMVAL оператором 87:

```
ON BIGFLOAT, NUMVAL;                                    87
```

- распечатать результаты решения операторами 88–89:

```
X1;                                                     88
X2;                                                     89
```

- отменить установленные на время режимы оператором 90:

```
OFF BIGFLOAT, NUMVAL;                                   90
```

После работы такой дополненной программы 2.5 результаты решения (2.60) предстанут в виде действительных чисел (только вместо идентификаторов X1 и X2 в обоих случаях будет использоваться MAT), что мы и покажем в нижеследующей распечатке, соответствующей работе операторов 88–89:

X1;

MAT(1,1):= – 7.976379296 MAT(2,1):= 10.50862067

MAT(3,1):= 3.925517234 MAT(4,1):= – 5.50862068

MAT(5,1):= 3.734482752 MAT(6,1):= – 10.50862067

MAT(7,1):= 1.050862067 MAT(8,1):= – 10.50862067

MAT(9,1):= 4.050862062 (2.61)

X2;

MAT(1,1):= 9.949031743 MAT(2,1):= 11.02322523

MAT(3,1):= 1.867096643 MAT(4,1):= – 6.023225674

MAT(5,1):= 5.792903577 MAT(6,1):= – 11.02322567

MAT(7,1):= 5.081935718 MAT(8,1):= – 11.02322567

MAT(9,1):= – 1.867096717

2.4.3. Оператор цикла FOR ... DO. Учет особенностей задания С-9

В CAB REDUCE **оператор цикла FOR ... DO** позволяет указанное число раз повторить нужную последовательность выражений и команд при различных значениях некоторой переменной, называемой счетчиком цикла.

Он может быть записан в следующей общей форме:

```
FOR I:= N1 STEP N2 UNTIL N3 DO оператор          (2.62)
```

где: I — неиндексированная целая переменная, соответствующая последовательно изменяющейся в цикле величине и называемая параметром или счетчиком цикла;

N1, N2, N3 — константы, неиндексированные переменные или арифметические выражения, имеющие целочисленные значения, причем N1 и N3 — соответственно начальное и конечное значения параметра I, а N2 — шаг его изменения при каждом прохождении цикла.

Оператор, стоящий после DO, выполняется сначала при значении целой переменной I=N1, затем при I=N1+N2, затем при I=N1+2*N2, I=N1+3*N2 и т.д., пока значение I меньше или равно N3.

Как только станет значение параметра цикла I=N2+N3, то выполнение цикла прекращается и программа переходит к выполнению оператора, следующего непосредственно за оператором цикла.

Если конечное значение N3 меньше (а для отрицательных значений шага N2 больше), чем начальное значение N1, то оператор цикла вообще не выполняется.

Оператор FOR ... DO также не предназначен для использования в качестве выражения: его значение в этом случае равно нулю.

После DO должен стоять один оператор. Если же необходимо использовать несколько операторов, применяют групповой оператор.

Конструкцию I:= N1 STEP N2 UNTIL N3 обычно называют заголовком цикла.

Если N2=1, то в этом случае STEP 1 UNTIL можно заменить двоеточием и общая форма оператора цикла FOR ... DO (2.62) примет упрощенный вид:

```
FOR I:= N1:N3 DO оператор                                    (2.63)
```

Например, в результате выполнения цикла:

```
FOR I:= 1:N DO A2(I,1):=0;                                   (2.64)
```

будут заданы нулевые значения всем N элементам первого столбца матрицы A . Конечно, значение переменной N, присутствующей в заголовке цикла, обязательно должно быть предварительно задано в программе (например, оператором присваивания 20 (N:=9) в программе 2.5).

Замечание. Значение переменной, используемой в качестве параметра цикла, не связано с ее значением вне цикла.

Поэтому в качестве имени параметра цикла можно использовать I, хотя вне цикла I есть мнимая единица.

Таким образом, выполнение оператора FOR I:= … не меняет заложенное в систему REDUCE соотношение I**2= –1.

Теперь можно учесть еще одну особенность задания С-9, рекомендованного в сборнике [23] для применения ЭВМ к решению задач статики.

Она заключается в том, что матрицы A_1 и A_2 отличаются только первым столбцом (см. уравнения (2.52)-(2.53)). Это позволяет почти наполовину сократить необходимые вводимые данные. Для этого в базовой программе 2.5 нужно:

1. Удалить операторы присваивания 60–72.
2. Предусмотреть после ввода матрицы A_1 ее копирование в рабочий массив A_2:

```
A2:=A1;                                                      60
```

3. Затем следует обнулить первый столбец матрицы A_2, например, с использованием цикла (2.64):

```
FOR I:= 1:N DO A2(I,1):=0;                                   62
```

4. Теперь нужно ввести только отличающиеся ненулевые элементы 1-го столбца матрицы A_2 (2.53):

```
A2(4,1):=6; A2(6,1):=-1; A2(7,1):=-4; A2(9,1):=1;            65
```

После работы программы 2.5 с вышеописанными изменениями результаты решения СЛАУ (2.52)-(2.53) предстанут, конечно, в том же виде (2.60).

Дополнение 2.10. Результат выполнения оператора цикла 62 не контролируется пользователем, так как он не распечатывается при выполнении модифицированной программы 2.5.

Чтобы получить печать выполнения оператора из цикла, нужно использовать команду WRITE. В ней в качестве параметра должен стоять нужный оператор присваивания:

```
FOR I:= 1:N DO WRITE A2(I,1):=0;                    62
```

Однако мы и теперь не получим желаемого результата: нулевые значения первого столбца матрицы A_2 так и не будут распечатаны. После несколько недоуменного изучения программы 2.5 мы вскоре поймем причину: оператор 62 стоит до выключения режима NERO, запрещающего печать нулевых значений.

Поэтому команду 80, которая разрешает печать нулевых значений перед решением СЛАУ, выключая флаг NERO, следует расположить впереди оператора 62:

```
OFF NERO;                                           61
```

Вот теперь нужная контрольная печать будет получена: все девять нулевых значений первого столбца матрицы A_2 будут распечатаны *по одному в каждой строке*: A2(1,1):=0, …, A2(9,1):=0.

Заметим, что циклы с операцией DO WRITE используются, как правило, для распечатки массивов переменных.

2.4.4. Исследование вариации нагрузки

В рассматриваемом типовом примере сила P_1 представлена на рис. 2.18 в произвольном положении, определяемом углом α = ALFA = 60°. Он отсчитывается против часовой стрелки от горизонтальной прямой, идущей вправо.

Поэтому ***для исследования влияния вариации угла поворота силы P_1*** на значения реакций опор рассматриваемой составной конструкции тригонометрические функции используемого угла ALFA следует сделать в программе 2.5 свободными переменными, очистив их командой CLEAR:

```
CLEAR SIN(ALFA),COS(ALFA);                          27
```

В результате работы такой измененной программы 2.5 мы получим значения реакций опор рассматриваемой системы тел в зависимости от вариации направления постоянной по модулю силы P_1 для двух вариантов работы конструкции:

- **1 случай (Rf = 0):**

$$X1(1,1):=-\frac{3*(185*\sin(alfa)-6)}{58} \qquad X1(2,1):=\frac{5*(75*\sin(alfa)-4)}{29}$$

$$X1(3,1):=\frac{2*(120*\sin(alfa)-47)}{29}$$

$$X1(4,1):=\frac{5*(58*\cos(alfa)-75*\sin(alfa)+4)}{29} \qquad (2.65)$$

$$X1(5,1):=\frac{5*(10*\sin(alfa)+13)}{29} \qquad X1(6,1):=-\frac{5*(75*\sin(alfa)-4)}{29}$$

$$X1(7,1):=\frac{75*\sin(alfa)-4}{58} \qquad X1(8,1):=-\frac{5*(75*\sin(alfa)-4)}{29}$$

$$X1(9,1):=\frac{5*(15*\sin(alfa)+34)}{58}$$

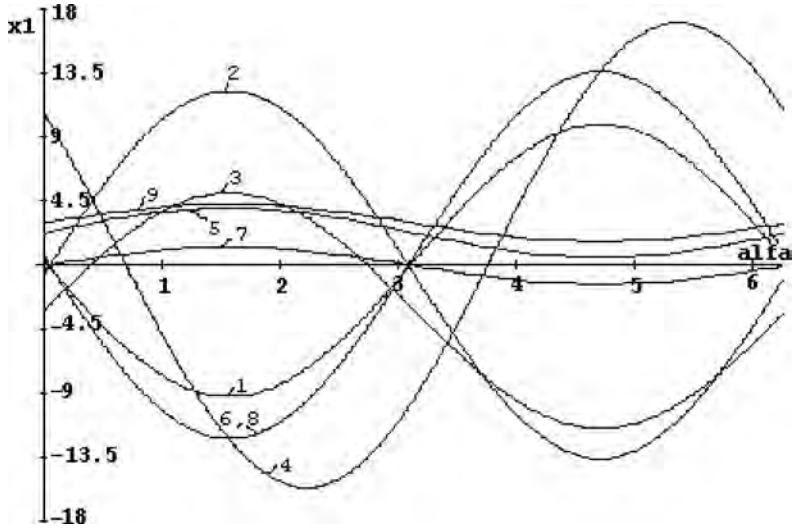

*Рис. 2.13. Зависимости реакций опор (X, кН) плоской составной конструкции (рис. 2.18) от изменения угла ALFA (рад) при вращении силы P_1: **1 – Re=Re'**, 2 – Xa, 3 – Ya, 4 – Xв, 5 – Yв, 6 – Xd=Xd', 7 – Yd=Yd', 8 – Xc=Xc', 9 – Yc=Yc'*

- **2 случай (Re = 0):**

$$X2(1,1):=\frac{2*(185*\sin(alfa)-6)}{31} \qquad X2(2,1):=\frac{2*(210*\sin(alfa)-11)}{31}$$

$$X2(3,1):=\frac{2*(90*\sin(alfa)-49)}{31}$$

$$X2(4,1):=\frac{2*(155*\cos(alfa)-210*\sin(alfa)+11)}{31} \qquad (2.66)$$

$$X2(5,1):=\frac{130*\sin(alfa)+67}{31} \qquad X2(6,1):=-\frac{2*(210*\sin(alfa)-11)}{31}$$

$$X2(7,1):=\frac{190*\sin(alfa)-7}{31} \qquad X2(8,1):=-\frac{2*(210*\sin(alfa)-11)}{31}$$

$$X2(9,1):=-\frac{2*(90*\sin(alfa)-49)}{31}$$

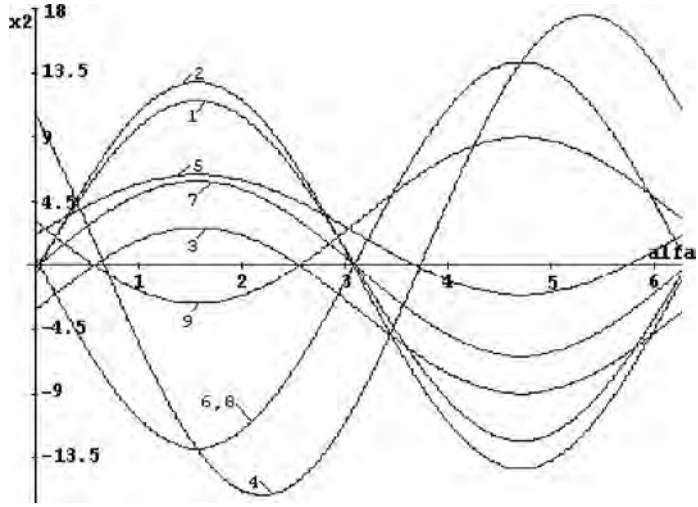

*Рис. 2.14. Зависимости реакций опор (X, кН) плоской составной конструкции (рис. 2.18) от изменения угла ALFA (рад) при вращении силы P_1: **1 – Rf=Rf'**, 2 – Xa, 3 – Ya, 4 – Xв, 5 – Yв, 6 – Xd=Xd', 7 – Yd=Yd', 8 – Xc=Xc', 9 – Yc=Yc'*

На рис. 2.13 и 2.14 представлены зависимости значений реакций опор рассматриваемой плоской составной конструкции (рис. 2.18) *при вращении*

постоянной по модулю силы P_1 в той же плоскости, построенные соответственно по соотношениям (2.65) и (2.66). Их сравнительный анализ показывает:

- зависимости подчиняются синусоидальному закону;
- синусоидальный характер зависимостей не изменяется в обоих случаях (при Rf = 0 или Re = 0).

Для удобства дальнейшего анализа выпишем отдельно из (2.65) и (2.66) выражения, определяющие значения реакций Re = $X_1(1,1)$ и Rf = $X_2(1,1)$, при различных положениях силы P_1:

$$X1(1,1):=-\frac{3*(185*\sin(alfa)-6)}{58} \qquad X2(1,1):=\frac{2*(185*\sin(alfa)-6)}{31}$$

Построим соответствующие графики с использованием системы DERIVE (см. [5, 6]), передав в нее значения реакций Re = $X_1(1,1)$ и Rf = $X_2(1,1)$ из (2.65) и (2.66) с помощью буфера обмена (<Ctrl>+<C> — <Ctrl>+<V>):

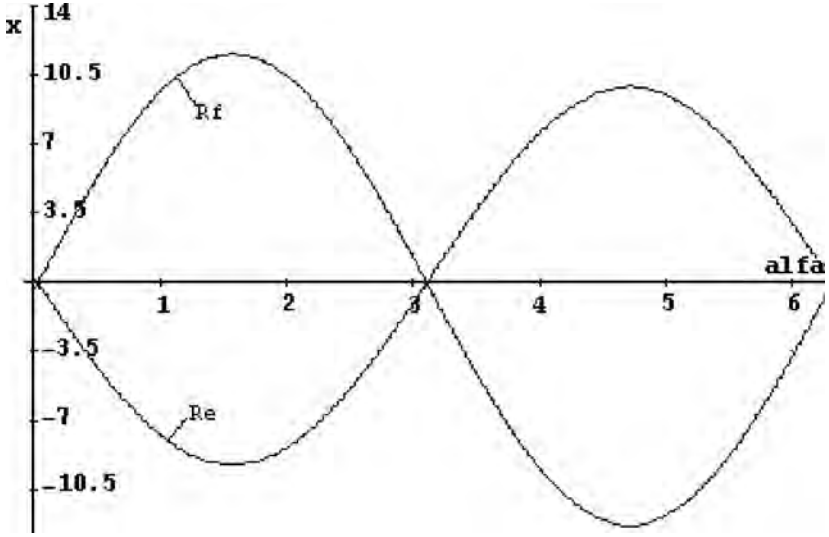

Рис. 2.15. Зависимости реакций опор Re и Rf (X, кН) плоской составной конструкции с внутренними односторонними связями (рис. 2.18) от изменения угла ALFA (рад) при вращении силы P_1

Из сравнения представленных зависимостей видно, что при значениях угла ALFA от 0 до π "работает" реакция Rf (ее значения положительны, а Re – отрицательны).

При значениях угла ALFA от π до 2π "работает" реакция Re.

Для исследования влияния вариации угла поворота силы P_2 на значения реакций опор рассматриваемой составной конструкции силу P_2 нужно представить в произвольном положении.

Будем отсчитывать изменяемый при вращении силы P_2 угол β по часовой стрелке от горизонтальной прямой, проходящей через начало силы вправо (рис. 2.18).

Тогда изображенному на рис. 57–58 и 60 [23, с. 55] вертикальному положению силы P_2 будет соответствовать угол $\beta = 90^o$, а произвольному положению на рис. 2.18 — угол $\beta < 90^o$.

После этого уравнения равновесия 4-6 систем (2.52) и (2.53) примут для произвольного положения силы P_2 соответственно следующий вид (2.67) и (2.68):

```
4. Xc*1 + Yc*10 –P2*cosβ*2 - P2*sinβ*10 = 0,
5. -Xc + Xd - P2*cosβ = 0,                          (2.67)
6. -Yc + P2*sinβ  + Yd =0;
```

и

```
4. Rf*6 + Xc*1 + Yc*10 –P2*cosβ*2 - P2*sinβ*10 = 0,
5. -Xc + Xd - P2*cosβ = 0,                          (2.68)
6. -Rf – Yc + P2*sinβ  + Yd =0.
```

Выполним вышеописанные действия по их формализации:

- заменим используемую греческую букву для обозначения угла β ее идентификатором, записанным в латинской транскрипции (β = BETA),
- перенесем свободные члены, не содержащие неизвестных, в правые части уравнений 4–6 систем (2.67) и (2.68), которые с учетом соответствия идентификаторов (2.54) теперь примут следующий измененный вид (2.69) и (2.70):

```
4. X8 + 10*X9 = P2*cos(BETA)*2 + P2*sin(BETA)*10,
5. X6 – X8 = P2*cos(BETA),                          (2.69)
6. X7 – X9 = -P2*sin(BETA),
```

и

```
4. 6*X1 + X8 + 10*X9 = P2*cos(BETA)*2 + P2*sin(BETA)*10,
5. X6 – X8 = P2*cos(BETA),                          (2.70)
6. -X1 + X7 – X9 = -P2*sin(BETA).
```

Напомним, что остальные уравнения равновесия 1–3 и 7–9 в системах (2.55) и (2.56) не изменились, а уравнения 4–6 там имеют соответственно вид (2.69) и (2.70).

Теперь оператор 55 программы 2.5 предстанет в измененной форме, а она сама дополнится оператором 56:

```
B1(1,1):=-20; B1(4,1):=P2*COS(BETA)*2+P2*SIN(BETA)*10;  55
B1(5,1):= P2*COS(BETA); B1(6,1):= - P2*SIN(BETA);       56
```

Теперь наша измененная программа 2.5 учитывает направление силы P_2 в общем виде, определяемом углом b.

Проверим, что для изображенного на рис. 57–58 и 60 [23, с. 55] вертикального положения силы P_2 она дает те же результаты решения (2.60) (или (2.61) при использовании дополнения 2.4).

Для этого дополним программу 2.5 операторами, определяющими значение тригонометрических функций при $\beta = 90^o$:

```
SIN(BETA):= 1; COS(BETA):= 0;                           26
```

Убедившись после запуска измененной программы 2.5 в совпадении результатов решения, исследуем влияние вариации направления постоянной по модулю силы P_2 на значения реакций опор рассматриваемой составной конструкции.

Для этого тригонометрические функции используемого угла BETA сделаем в программе 2.5 свободными переменными, очистив их командой CLEAR:

```
CLEAR SIN(BETA),COS(BETA);                              27
```

В результате работы такой измененной программы 2.5 мы получим значения реакций опор рассматриваемой составной конструкции в зависимости от вариации направления постоянной по модулю силы P_2:

- **1 случай (Rf = 0):**

$$X1(1,1):=\frac{3*(6*\cos(beta)-185*sqrt(3)-7*\sin(beta)+19)}{116}$$

$$X1(2,1):=-\frac{78*\cos(beta)-375*sqrt(3)+315*\sin(beta)-275}{58}$$

$$X1(3,1):=-\frac{2*(9*\cos(beta)-60*sqrt(3)+33*\sin(beta)+14)}{29}$$

$$X1(4,1):=\frac{3*(84*\cos(beta)-125*sqrt(3)+105*\sin(beta)+5)}{58}$$

$$X1(5,1):=\frac{18*\cos(beta)+25*\sqrt{3}-21*\sin(beta)+86}{29} \qquad (2.71)$$

$$X1(6,1):=\frac{252*\cos(beta)-375*\sqrt{3}+315*\sin(beta)-275}{58}$$

$$X1(7,1):=\frac{54*\cos(beta)+75*\sqrt{3}-63*\sin(beta)+55}{116}$$

$$X1(8,1):=\frac{78*\cos(beta)-375*\sqrt{3}+315*\sin(beta)-275}{58}$$

$$X1(9,1):=\frac{54*\cos(beta)+75*\sqrt{3}+285*\sin(beta)+55}{116}$$

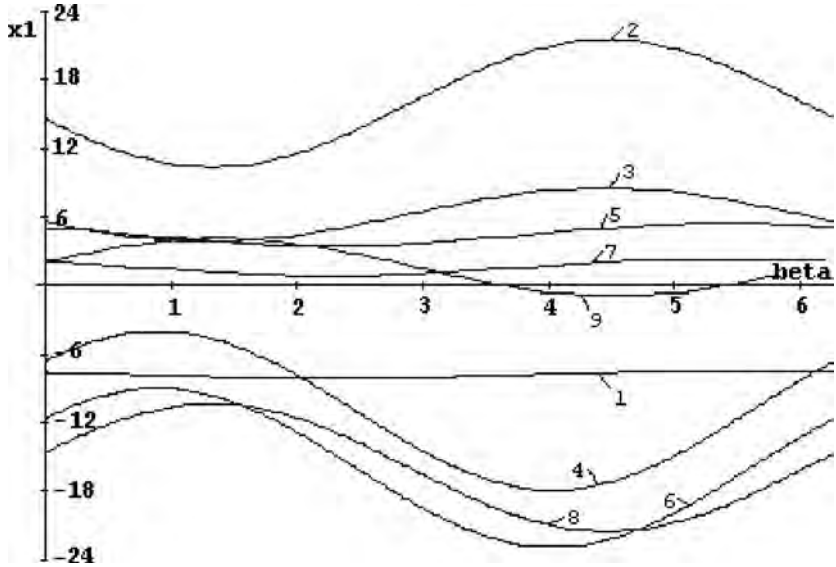

*Рис. 2.16. Зависимости реакций опор (X, кН) плоской составной конструкции с внутренними односторонними связями (рис. 2.18) от изменения угла BETA (рад) при вращении силы P_2: **1 – Re=Re'**, 2 – Xa, 3 – Ya, 4 – Xв, 5 – Yв, 6 – Xd=Xd', 7 – Yd=Yd', 8 – Xc=Xc', 9 – Yc=Yc'*

Как видно из сравнения графиков, построенных на рис. 2.16 по соотношениям (2.71), при **Rf = 0** значения реакций опор рассматриваемой составной конструкции (рис. 2.18) при вращении постоянной по модулю силы $\mathbf{P_2}$ в плоскости рисунка также изменяются по синусоидальному закону.

- **2 случай (Re = 0):**

$$X2(1,1):=-\frac{6*\cos(beta)-185*\mathrm{sqrt}(3)-7*\sin(beta)+19}{31}$$

$$X2(2,1):=-\frac{2*(21*\cos(beta)-105*\mathrm{sqrt}(3)+84*\sin(beta)-73)}{31}$$

$$X2(3,1):=-\frac{2*(9*\cos(beta)-45*\mathrm{sqrt}(3)+36*\sin(beta)+13)}{31}$$

$$X2(4,1):=\frac{3*(45*\cos(beta)-70*\mathrm{sqrt}(3)+56*\sin(beta)+3)}{31}$$

$$X2(5,1):=\frac{18*\cos(beta)+65*\mathrm{sqrt}(3)-21*\sin(beta)+88}{31} \qquad (2.72)$$

$$X2(6,1):=\frac{135*\cos(beta)-210*\mathrm{sqrt}(3)+168*\sin(beta)-146}{31}$$

$$X2(7,1):=\frac{12*\cos(beta)+95*\mathrm{sqrt}(3)-14*\sin(beta)+7}{31}$$

$$X2(8,1):=\frac{2*(21*\cos(beta)-105*\mathrm{sqrt}(3)+84*\sin(beta)-73)}{31}$$

$$X2(9,1):=\frac{2*(9*\cos(beta)-45*\mathrm{sqrt}(3)+36*\sin(beta)+13)}{31}$$

Напомним, что эти зависимости значений реакций опор рассматриваемой составной конструкции в зависимости от вариации направления постоянной по модулю силы P_2 мы получили в результате работы измененной программы 2.5.

Для этого тригонометрические функции используемого угла BETA в программе 2.5 делались свободными переменными и очищались от своих значений командой CLEAR:

```
CLEAR SIN(BETA),COS(BETA);                                    27
```

Графики зависимостей значений реакций опор составной конструкции (рис. 2.18) для **Re = 0** при вращении силы **P_2,** представлены на рис. 2.17. Они также изменяются по синусоидальному закону.

Сравнительный анализ графиков, построенных на рис. 2.16 и 2.17 соответственно по соотношениям (2.71) и (2.72) показывает:

- зависимости значений реакций опор рассматриваемой плоской составной конструкции (рис. 2.18) *при вращении постоянной по модулю силы P_2* также подчиняются синусоидальному закону;

- изменение значений реакций опор в обоих случаях (Rf = 0 или Re = 0) при вращении силы P_2 происходит почти во всех случаях *без изменения знаков* этих величин (кроме зависимости для Yc – кривая 9 на рис. 2.16):
  - в этом проявляется основное отличие влияния вариации силы P_2 от соответствующего воздействия вариации силы P_1 на значения реакций опор рассматриваемой конструкции (см. графики на рис. 2.13, 2.14 и на рис. 2.16, 2.17);
- графики для Re и Rf (кривые 1 на рис. 2.16 и 2.17 соответственно) весьма характерны: они представляют синусоидальные кривые очень небольшой амплитуды, расположенные в положительной (Rf) или отрицательной (Re) областях. Поэтому их можно не рассматривать отдельно и сразу сделать вывод:
  - *вариация направления силы P_2 не может изменить знаки односторонних связей:* при любых значениях угла β "работает" реакция Rf (ее значения положительны, а Re – отрицательны).

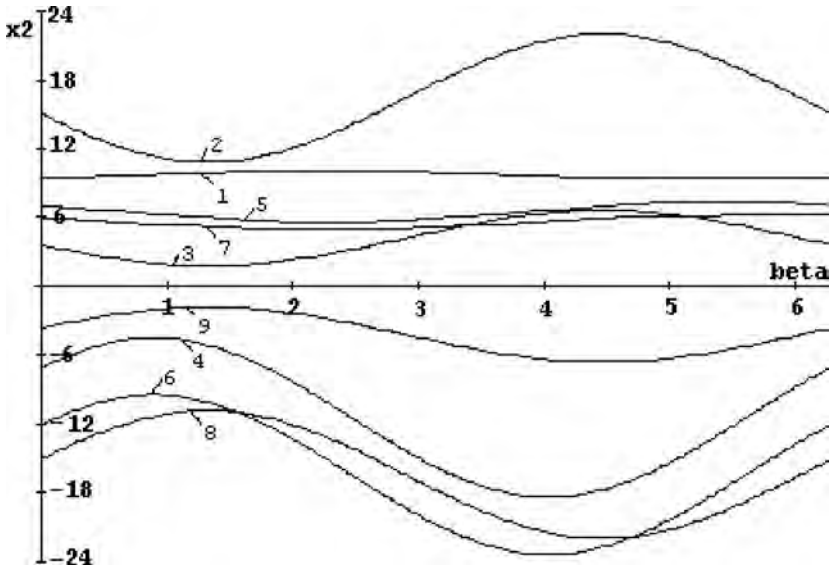

Рис. 2.17. Зависимости реакций опор (X, кН) плоской составной конструкции с внутренними односторонними связями (рис. 2.18) от изменения угла BETA (рад) при вращении силы P_2: ***1 – Rf=Rf'****, 2 – Xa, 3 – Ya, 4 – Xв, 5 – Yв, 6 – Xd=Xd', 7 – Yd=Yd', 8 – Xc=Xc', 9 – Yc=Yc'*

2.4.5. Исследование вариации геометрических факторов

Для вариации геометрических факторов размеры конструкции, изображенной на рис. 57 [23, с. 55], следует представить в общем виде. Для указанных там величин 1, 2 и 3м положим: **a**=1м, **b**=2, **c**=3м (рис. 2.18).

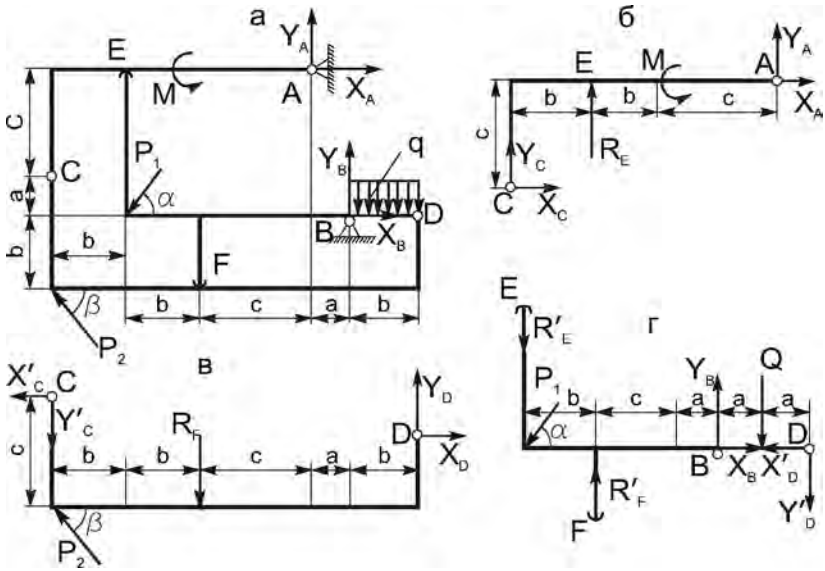

Рис. 2.18. Расчетная схема составной конструкции с внутренними односторонними связями (а) и ее составных частей (б–г), представленная для возможности исследования вариации нагрузки и геометрических факторов

С учетом используемых обозначений размеров **a**, **b** и **c** перепишем уравнения (2.52)–(2.53) в общем виде:

- **1 случай (Rf = 0):**

```
1. Re*b + M - Xa*c + Ya*(2b + c) = 0,
2. Xa + Xc = 0,
3. Re + Ya + Yc = 0,
4. Xc*a + Yc*(3b + a + c) - P2*(3b + a + c) = 0,
5. -Xc + Xd =0,                                          (2.73)
```

```
6. -Yc + P2 + Yd = 0,
7. Re*(a+b+c)+ (a+b+c)*P1*sin60° - q*b*b/2 - Yd*b = 0,
8. Xb - Xd - P1*cos60° = 0,
9. -Re - P1*sin60° + Yb - q*b - Yd = 0;
```

- **2 случай (Re = 0):**

```
1. M -Xa*c + Ya*(2b + c) = 0,
2. Xa + Xc = 0,
3. Ya + Yc = 0,
4. Rf*(a+b+c) + Xc*a + Yc*(3b+a+c) - P2*(3b+a+c) = 0,
5. -Xc + Xd =0,                                          (2.74)
6. -Rf - Yc + P2 + Yd = 0,
7. -Rf*(a+c) + (a+b+c)*P1*sin60° - q*b*b/2 - Yd*b = 0,
8. Xb - Xd - P1*cos60° = 0,
9. Rf - P1*sin60° + Yb - q*b - Yd = 0.
```

В уравнениях 7 и 9 систем (2.73)–(2.74) значение сосредоточенной силы Q представлено в виде, явно зависящем от соответствующего размера конструкции b: **Q = q*b.**

Обратим ваше внимание, что рис. 2.18 использовался нами ранее для исследования влияния вариации нагрузки (см. п. 2.4.4), поэтому на нем направления сил P_1 и P_2 показаны в произвольном положении с использованием углов α и β.

Теперь при вариации геометрических факторов действующие силы P_1 и P_2 *постоянны по направлению.*

Поэтому, хотя конструкция, изображенная на рис. 2.18, представлена в общем виде с использованием обозначений размеров **a**, **b**, **c** и углов α и β, *направления сил* P_1 и P_2 для исследования вариации геометрических размеров *совпадают со своим исходным положением на рис. 2.12:*

- угол α, определяющий направление силы P_1, постоянен и равен 60°;
- направление силы P_2 на рис. 2.12 вертикально, что соответствует углу $\beta = 90^\circ$ на рис. 2.18.

Для более полного совпадения с формализованными уравнениями равновесия (2.55)–(2.56):

- используемый угол 60° обозначается далее идентификатором ALFA;
- угол β в уравнениях не применяется.

С использованием соответствия идентификаторов (2.54), системы уравнений (2.73)–(2.74) примут формализованную форму (2.75)–(2.76). Она просто представляет собой запись соотношений (2.55)–(2.56) с представлением размеров конструкции, изображенной на рис. 2.12, в алгебраической форме: **a**=1м, **b**=2, **c**=3м:

- **1 случай (Rf = 0):**

1. $X_1*b - X_2*c + X_3*(2b + c) = - M,$
2. $X_2 + X_8 = 0,$
3. $X_1 + X_3 + X_9 = 0,$
4. $X_8*a + X_9*(3b + a + c) = P_2(3b + a + c),$
5. $X_6 - X_8 = 0,$ (2.75)
6. $X_7 - X_9 = - P_2,$
7. $X_1*(a+b+c) - X_7*b = q*b*b/2 - P_1*\sin(ALFA)*(a+b+c),$
8. $X_4 - X_6 = P_1*\cos(ALFA),$
9. $-X_1 + X_5 - X_7 = P_1*\sin(ALFA) + q*b;$

- **2 случай (Re = 0):**

1. $-X_2*c + X_3*(2b + c) = - M,$
2. $X_2 + X_8 = 0,$
3. $X_3 + X_9 = 0,$
4. $X_1*(a+b+c) + X_8*a + X_9*(3b+a+c) = P_2(3b+a+c),$
5. $X_6 - X_8 = 0,$ (2.76)
6. $-X_1 + X_7 - X_9 = - P_2,$
7. $- X_1*(a+c) - X_7*b = q*b*b/2 - P_1*\sin(ALFA)*(a+b+c),$
8. $X_4 - X_6 = P_1*\cos(ALFA),$
9. $X_1 + X_5 - X_7 = q*b + P_1*\sin(ALFA).$

Для вариации размеров исследуемой конструкции, изображенной на рис. 57 [23, с. 55] и в общем виде на рис. 2.18, выполним следующие действия:

1. Определим согласно дополнению 2.1 положение и значение всех ненулевых элементов матриц A_1 и B_1:

```
A1(1,1):=B; A1(1,2):=-C; A1(1,3):=2*B+C; A1(2,2):=1;     35
A1(2,8):=1; A1(3,1):=1; A1(3,3):=1; A1(3,9):=1;          37
```

```
A1(4,8):=A; A1(4,9):=3*B+A+C; A1(5,6):=1; A1(5,8):=-1;  40
A1(6,7):=1; A1(6,9):=-1; A1(7,1):=A+B+C;                42
A1(7,7):=-B; A1(8,4):=1; A1(8,6):=-1;                   45
A1(9,1):=-1; A1(9,5):=1; A1(9,7):=-1;                   47
B1(1,1):=-M; B1(4,1):=P2*(3*B+A+C); B1(6,1):=-P2;        55
B1(7,1):=-P1*SIN(ALFA)*(A+B+C)+q*B*B/2;                  57
B1(8,1):=P1*COS(ALFA); B1(9,1):=P1*SIN(ALFA)+q*B;        58
```

2. Учтем особенность выполнения двухвариантной задачи рассматриваемого типового примера С–9 [23, с. 51, 55–59] согласно п. 2.4.3:

   - удалим из программы 2.5 операторы присваивания 60–72;
   - выполним после ввода матрицы A_1 ее копирование в рабочий массив A_2:

```
A2:=A1;                                                  60
```

   - обнулим первый столбец матрицы A_2, например, с использованием цикла (2.64):

```
FOR I:= 1:N DO A2(I,1):=0;                               62
```

   - введем отличающиеся ненулевые элементы 1-го столбца матрицы A_2 (2.76).

```
A2(4,1):=A+B+C; A2(6,1):=-1; A2(7,1):=-C-A; A2(9,1):=1;  65
```

3. Зададим геометрические размеры конструкции предложением 18:

```
A:=1; B:=2; C:=3;                                        18
```

Осознанно скомпоновав приведенные фрагменты, мы получим из программы 2.5 базовую программу 2.6, приспособленную для вариации геометрических размеров конструкции:

```
COMMENT Программа 2.6: решение типового примера С-9 на REDUCE
с описанием только ненулевых элементов матриц A1 и B1, копи-
рованием B2:=B1 и A2:=A1, обнулением 1-го столбца A2  и
заданием ее 4-х отличающихся элементов;
OUT "\C9\PR2-6.LIS";                                     10
ON NERO;                                                 15
A:=1; B:=2; C:=3;                                        18
N:=9;                                                    20
SIN(ALFA):= SQRT(3)/2;                                   22
COS(ALFA):= 1/2;                                         24
```

```
P1:=10; P2:=3;                                                    29
MATRIX A1(N,N),B1(N,1),X1(N,1),                                   30
 A2(N,N),B2(N,1),X2(N,1);
A1(1,1):=B; A1(1,2):=-C; A1(1,3):=2*B+C; A1(2,2):=1;              35
A1(2,8):=1; A1(3,1):=1; A1(3,3):=1; A1(3,9):=1;                   37
A1(4,8):=A; A1(4,9):=3*B+A+C; A1(5,6):=1; A1(5,8):=-1;            40
A1(6,7):=1; A1(6,9):=-1; A1(7,1):=A+B+C;                          42
A1(7,7):=-B; A1(8,4):=1; A1(8,6):=-1;                             45
A1(9,1):=-1; A1(9,5):=1; A1(9,7):=-1;                             47
B1(1,1):=-M; B1(4,1):=P2*(3*B+A+C); B1(6,1):=-P2;                 55
B1(7,1):=-P1*SIN(ALFA)*(A+B+C)+q*B*B/2;                           57
B1(8,1):=P1*COS(ALFA); B1(9,1):=P1*SIN(ALFA)+q*B;                 58
A2:=A1;                                                           60
FOR I:= 1:N DO A2(I,1):=0;                                        62
A2(4,1):=A+B+C; A2(6,1):=-1; A2(7,1):=-C-A; A2(9,1):=1;           65
B2:=B1;                                                           75
OFF NERO;                                                         80
X1:=A1**(-1)*B1;                                                  85
X2:=A2**(-1)*B2;                                                  86
SHUT "\C9\PR2-6.LIS";                                             95
END;                                                              99
```

Все операторы, команды и предложения программы 2.6 были предварительно описаны. После ее работы результаты решения СЛАУ (2.73)-(2.74) будут записаны в файле PR2-6.LIS, находящемся в подкаталоге \C9 корневого каталога текущего диска. Их значения, конечно, должны совпасть с результатами (2.60).

Убедившись в этом и проверив таким образом работу базовой программы, можно приступать к вариации геометрических размеров конструкции **a**, **b** и **c**.

Для изучения раздельного влияния вариации геометрических факторов на значения реакций опор нужно из предложения 18 исключать соответствующий оператор присваивания, задающий данный размер конструкции **a**, **b** или **c**.

Предварительно нужно очистить исключаемую переменную и сделать свободной, указав ее в качестве параметра команды CLEAR.

При желании выполнить исследование с минимальными переделками программы, можно просто командой CLEAR каждый раз только очищать варьируемую переменную, делая ее свободной, что и мы также будем делать.

Тогда соответствующий идентификатор A, B или C становится свободной переменной и входит в символьном виде в результаты решения, выражая их зависимость от данной величины.

Приведем для рассматриваемого типового примера плоской двухвариантной системы трех тел [23, с. 51, 55-59] формы команды CLEAR, которой нужно дополнить программу 2.6.

Представим также выборку результатов ее работы в виде соответствующих аналитических зависимостей, выражающих влияние раздельного изменения размеров конструкции **a**, **b** и **c** на значения реакций Re (X1(1,1)) и Rf (X2(1,1)):

- вариация размера **a**:

```
CLEAR A;                                                    27
```

X1(1,1):= – (50*sqrt(3)*a**2 + 385*sqrt(3)*a + 675*sqrt(3) – 18*a – 54) / (10*a**2 + 87*a + 135)$ (2.77)

X2(1,1):= (50*sqrt(3)*a**2 + 385*sqrt(3)*a + 675*sqrt(3) – 18*a – 54) / (10*a**2 + 71*a + 105)$ (2.78)

- вариация размера **b**:

```
CLEAR B;                                                    27
```

X1(1,1):= – (110*sqrt(3)*b**2 + 590*sqrt(3)*b + 600*sqrt(3) – 11*b**3 – 3*b**2 – 22*b) / (4*(6*b**2 + 31*b + 30))$ (2.79)

X2(1,1):= (110*sqrt(3)*b**2 +590*sqrt(3)*b + 600*sqrt(3) – 11*b**3 – 3*b**2 – 22*b) /(2*(8*b**2 +47*b + 60))$ (2.80)

- вариация размера **c**:

```
CLEAR C;                                                    27
```

X1(1,1) := – (5*sqrt(3)*c**3 + 55*sqrt(3)*c**2 + 140*sqrt(3)*c + 60* sqrt(3) – 2*c**2 – 10*c – 24) / (c**3 + 11*c**2 + 30*c + 16)$ (2.81)

X2(1,1):= (5*sqrt(3)*c**3 + 55*sqrt(3)*c**2 + 140*sqrt(3)*c + 60* sqrt(3) – 2*c**2 – 10*c – 24) / (c**3 + 9*c**2 + 22*c + 12)$ (2.82)

На рис. 2.19 покажем также геометрическую интерпретацию полученных соотношений (2.77) – (2.82).

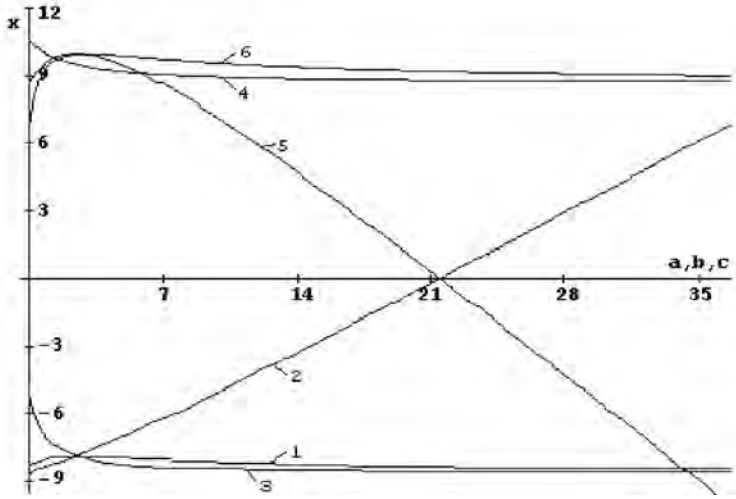

*Рис. 2.19. Зависимости величин реакций Re (кривые 1–3) и Rf (кривые 4–6) от изменения геометрических размеров конструкции **a**, **b** и **c**, изображенной на рис. 57 [23, с. 55], при вариации **a** (кривые 1 и 4), **b** (кривые 2 и 5) и **c** (кривые 3 и 6). Значения Re и Rf даны в кН, геометрических размеров **a, b** и **c** - в метрах*

Все графики, как видно из приведенного рисунка, кроме небольшого начального участка носят монотонный характер.

Изменение в широких пределах значений **a** и **c** не приводит к изменению знаков реакций Re и Rf и практически не влияет на их значения (кривые 1, 3 для Re и 4, 6 для Rf).

Изменение же размера **b** приводит к почти постоянному увеличению значений реакции Re и уменьшению Rf. Это приводит в итоге к изменению условий работы односторонних связей рассматриваемой плоской двухвариантной системы трех тел: вместо Rf начинает “работать”Re. Однако это происходит при значениях **b**>21 метра, что редко будет иметь место при практическом использовании.

В этой связи следует заметить, что в настоящем пособии в учебных целях приведены методы решения задач при вариации самых различных факторов: величины и направления действия сил, геометрических размеров и относительного расположения элементов конструкции.

Реальные требования конкретной задачи из инженерной практики определят необходимость варьирования тех или иных параметров.

Мы пока только изучаем методику проведения аналитического исследования на ПК и программные возможности для этого.

Поэтому в заключение не откажем себе в удовольствии и получим выражения для реакций опор при одновременном изменении всех геометрических размеров конструкции. Для этого очистим в программе 2.6 и сделаем свободными все геометрические размеры **a**, **b** и **c**, указав их в качестве параметров команды CLEAR:

```
CLEAR A,B,C;                                                    27
```

Представим выборку результатов работы программы 2.6 в виде соответствующих аналитических зависимостей, выражающих влияние одновременного изменения размеров конструкции **a**, **b** и **c** только на значения реакций Re (X1(1,1)) и Rf (X2(1,1)):

$$X1(1,1):= -(20*\mathrm{sqrt}(3)*a^2*b + 20*\mathrm{sqrt}(3)*a^2*c + 20*\mathrm{sqrt}(3)*a*b^2 + 70*\mathrm{sqrt}(3)*a*b*c + 30*\mathrm{sqrt}(3)*a*c^2 + 30*\mathrm{sqrt}(3)*b^2*c + 40*\mathrm{sqrt}(3)*b*c^2 + 10*\mathrm{sqrt}(3)*c^3 - 2*a*b^3 - 2*a*b^2*c + 12*a*b^2 + 6*a*b*c - 40*a*b - 3*b^3*c - b^2*c^2) / (2*(2*a^2*b + 2*a^2*c + 3*a*b^2 + 8*a*b*c + 3*a*c^2 + 3*b^2*c + 4*b*c^2 + c^3)) \quad (2.83)$$

$$X1(2,1):= (10*\mathrm{sqrt}(3)*a^2*b + 10*\mathrm{sqrt}(3)*a^2*c + 40*\mathrm{sqrt}(3)*a*b^2 + 60*\mathrm{sqrt}(3)*a*b*c + 20*\mathrm{sqrt}(3)*a*c^2 + 30*\mathrm{sqrt}(3)*b^3 + 70*\mathrm{sqrt}(3)*b^2*c + 50*\mathrm{sqrt}(3)*b*c^2 + 10*\mathrm{sqrt}(3)*c^3 - 12*a^2*b - 6*a^2*c + 40*a^2 - a*b^3 - a*b^2*c - 48*a*b^2 - 48*a*b*c + 160*a*b - 12*a*c^2 + 80*a*c - 3*b^4 - 4*b^3*c - 36*b^3 - b^2*c^2 - 66*b^2*c + 120*b^2 - 36*b*c^2 + 160*b*c - 6*c^3 + 40*c^2) / (2*(2*a^2*b + 2*a^2*c + 3*a*b^2 + 8*a*b*c + 3*a*c^2 + 3*b^2*c + 4*b*c^2 + c^3)) \quad (2.84)$$

Как уже отмечалось, выражения такого общего вида позволяют получать значения реакций при любом наборе данных для всей конструкции, проверяя наши предположения, сделанные на основании анализа вариаций каких-либо отдельных факторов.

ГЛАВА 3. СИМВОЛЬНОЕ ДИФФЕРЕНЦИРОВАНИЕ В CAB REDUCE И ЕГО ИСПОЛЬЗОВАНИЕ В ЗАДАЧАХ КИНЕМАТИКИ

Решение задач кинематики с применением ПК ранее сводилось к численному дифференцированию уравнений движения, функций положения и т.п. Поэтому рассмотрим различные аспекты применения символьного дифференцирования на ПК в CAB REDUCE.

3.1. Оператор дифференцирования DF

Оператор DF используется для выполнения дифференцирования в частных производных по отношению к одной или нескольким переменным.

Его синтаксис можно представить в следующей общей форме:

`DF(функция,аргумент1,порядок1,…,аргументN,порядокN)` (3.1)

В (3.1) первым параметром в скобках (функция) является скалярное выражение, которое надо дифференцировать. Остальные параметры определяют независимые переменные (аргументI), по которым производится дифференцирование, и порядок каждой частной производной (порядокI). В качестве аргумента могут использоваться простые имена или операторы, но не алгебраические выражения в виде их произведений или сумм.

Таким образом, лишь первый параметр оператора дифференцирования может иметь значением алгебраическое выражение. Значениями последующих параметров должны быть либо простого вида переменные, либо целые неотрицательные числа, задающие соответствующий порядок производной, которую требуется взять по стоящей перед цифрой переменной. Если число, выражающее порядок любой частной производной (порядокI) равно единице, то оно может быть опущено. Например:

- для обозначения первой производной нижеследующие два выражения эквивалентны: df(y, x,1) и df(y, x) означают $\frac{dy}{dx}$;

- для второй и более высоких производных порядок следует указывать обязательно:

$$\text{df(y, x,2)} = \frac{d^2y}{dx^2} \text{ и df(y, x,2,z,3,t)} = \frac{d^6y}{dx^2dz^3dt}.$$

Выполнение оператора DF(Y,X) происходит следующим образом: сначала определяются значения Y и X. Предположим, что аргумент X является свободной переменной, так что его значением является сам символ X. Каждое слагаемое (или другая часть) функции Y, которое зависит от X, будут продифференцированы по обычным правилам. Если Y (или его часть) является выражением, не зависящим от X, то производная Y по X (или ее часть) принимается равной 0.

Если требуется указать на зависимость переменных, то она вводится посредством команды DEPEND, после которой пишутся через запятую параметры. Тем самым устанавливается зависимость первого параметра от всех последующих. Так, запись

depend y, t; df(y5,t);** (3.2)

указывает на то, что Y является функцией T. Поэтому результат дифференцирования будет иметь следующий вид:

$5*df(y,t)*y^4$

Зависимость Y от T в явном виде установлена не была, поэтому в результате используется символ DF(Y,T).

Введенные зависимости могут быть отменены командой NODEPEND. После

nodepend y, t; df(y5,t);** (3.3)

переменная Y не будет далее зависеть от T, поэтому производная Y по T принимается равной 0.

Рассмотрим применение оператора дифференцирования DF на примерах задач кинематики. Произведем с использованием CAB REDUCE определение кинематических характеристик (скорости, ускорения) и радиуса кривизны траектории материальной точки по заданным уравнениям ее движения.

В качестве типовых выберем примеры решения заданий на соответствующие темы: “Кинематика точки” (К-1 [23, с. 60-62] или [22, с. 76-79], К-2 [22, с. 82-87]) и “Сложное движение точки” (К-7 [23, с. 99-106] или К-10 [22, с. 137-143]). Ссылки на задания будем производить по их номеру, причем рассматриваемым примерам присвоим 31-й вариант.

3.2. Постановка задач и структура программ по кинематике точки

Задачи на определение кинематических характеристик являются прекрасной иллюстрацией возможностей символьного дифференцирования системы REDUCE. После составления уравнений движения материальной точки и определения зависимостей ее координат X и Y (а для пространственных задач и Z) от времени T, их нужно дважды продифференцировать по времени для определения проекций скорости и ускорения на соответствующие координатные оси. Еще одно дифференцирование выражения для модуля скорости может быть использовано для определения тангенциального ускорения.

Соответственно этому ***структура простейшей программы*** для определения радиуса кривизны траектории материальной точки с использованием символьного дифференцирования в системе REDUCE состоит из следующих основных блоков.

1. Открытие командой OUT выводного файла с идентификатором ИДФ (1.26), куда будут записываться результаты работы программы `(OUT «ИДФ»;)`. Включение или выключение соответствующих переключателей, устанавливающих нужные режимы представления результатов:
   - выключение режима NAT при получении достаточно длинных результатов для представления их по строкам `(OFF NAT;)` или включение его при желании представить их в "естественной" форме `(ON NAT;)`. При опущенном флаге NAT все степени печатаются через "**", для отделения числителя от знаменателя используется символ "/" и в конце каждого предложения ставится терминатор "$";
   - объявление вывода в фортран-синтаксисе для удобства проведения дальнейших сложных численных расчетов, что достигается включением флага FORT `(ON FORT;)`. Этот режим включает в себя все достоинства выключения флага NAT при получении длинных результатов для представления их по строкам. Кроме этого строго соблюдается синтаксис фортран-программ:
     - выражения будут начинаться с 7-й колонки;
     - если запись не помещается на одной строке, то в последующих строках появится признак продолжения – знак "." (точка) в 6-й колонке (максимальное число строк продолжения 19).

2. Задание значений с помощью операторов присваивания используемым вспомогательным переменным и тригонометрическим функциям. Описание требуемых уравнений движения материальной точки, определяющих зависимость ее координат X и Y (а для пространственных задач и Z) от времени.
3. Определение проекций скорости точки путем дифференцирования выражений для соответствующих координат по времени. Напомним, что аргументом, по которому производится дифференцирование, должна быть независимая переменная. Переменная T, которой обычно обозначают время, в системе REDUCE является зарезервированной: имеет значение «истина» и используется в символьном режиме. Поэтому применять для обозначения аргумента переменную T возможно, так как он в операторе дифференцирования DF является формальным параметром. Однако ***переменной T нельзя присвоить никакого значения.*** Это неудобно, ибо делает невозможной проверку получающихся сложных аналитических выражений путем сравнения их численных значений при заданном времени T1 с решением без ПК. Поэтому будем:
   - ***применять для обозначения аргумента переменную T1;***
   - ***очищать ее командой CLEAR*** перед первым использованием в программе в операторе дифференцирования DF и после присваивания ей численных значений, ***делая переменную T1 свободной.***
4. Нахождение модуля скорости точки путем извлечения квадратного корня из сумм квадратов проекций скорости.
5. Определение проекций ускорения точки на координатные оси одним из двух способов:
   - путем взятия *второй производной* от выражений для соответствующих координат по времени T1;
   - получением *первой производной* от выражений для соответствующих проекций скорости точки.
6. Нахождение модуля ускорения точки путем извлечения квадратного корня из сумм квадратов проекций ускорения точки.
7. Определение тангенциального ускорения точки одним из двух способов:
   - получением *первой производной* от соответствующего выражения для скорости точки;

- использованием обычной формулы для определения тангенциального ускорения по известным значениям проекций скорости и ускорения точки.

8. Определение нормального ускорения точки и радиуса кривизны траектории.
9. Закрытие командой SHUT выводного файла с идентификатором ИДФ (1.26), куда записывались результаты работы программы (`SHUT «ИДФ»;`). В конце следует ввести оператор END, что является стандартным способом заканчивать файлы, предназначенные для считывания (`END;` или для большей надежности `;END;`).

3.3. Определение скорости и ускорения точки по заданным уравнениям ее движения

Требуемые для начала решения уравнения движения материальной точки задаются в К-1 ([23, с. 60-62] или [22, с. 76-79]), что позволяет акцентировать внимание на различных аспектах его численного и символьного решения в CAB REDUCE.

Поэтому мы начнем с него показ возможностей символьного дифференцирования.

Уравнения движения материальной точки для типового примера задания К-1 имеют вид ([23, с. 60] или [22, с. 76]):

$$x = 4*t;\ y = 16*t^2 - 1. \qquad (3.4)$$

Для составления простейшей программы по определению кинематических характеристик в системе REDUCE следует реализовать блочную структуру, описанную в п. 3.2.

Это может быть сделано, например, следующим образом (по вопросам записи программы 3.1 в файле PR3-1 в подкаталоге \KIN корневого каталога текущего диска и запуске ее на выполнение см. п 7.3 и пример в п. 1.4.2):

```
COMMENT Программа 3.1: Определение скорости, ускорения точки
и радиуса кривизны траектории типового примера К-1 ([23,
с. 60] или [22, с. 76])на REDUCE с представлением результатов
в символьном виде;
OUT "\KIN\PR3-1.LIS";                                        10
CLEAR T1;                                                    20
X:=4*T1;                                                     25
```

```
Y:=16*T1*T1-1;                                    27
VX:=DF(X,T1);                                     30
VY:=DF(Y,T1);                                     32
V:=SQRT(VX*VX+VY*VY);                             40
AX:=DF(VX,T1);                                    50
AY:=DF(VY,T1);                                    52
A:=SQRT(AX*AX+AY*AY);                             60
AT:=(VX*AX+VY*AY)/V;                              70
AN:=SQRT(A*A-AT*AT);                              80
RO:=V*V/AN;                                       85
SHUT "\KIN\PR3-1.LIS";                            90
END;                                              99
```

Номера операторов справа проставлены только для удобства дальнейших пояснений. Этот прием будет использоваться и в дальнейшем.

Напомним, что в программе 3.1 и последующих ее модификациях для удобства первоначальной ориентировки 1-я цифра номера указывает на соответствующий блок п. 3.2: например, операторы 50 и 52 относятся к блоку 5 (определение проекций ускорения точки на координатные оси).

Комментарий, помещенный между зарезервированным словом COMMENT и разделителем в виде знака “точка с запятой”, описывает назначение программы 3.1 и при обработке игнорируется.

Команда 10 открывает выводной файл PR3-1.LIS, находящийся в подкаталоге \KIN корневого каталога текущего диска, куда будут записываться результаты работы программы 3.1. Напомним, что имя файла и подкаталога для ваших программ может быть любым разрешенным идентификатором и должно отражать индивидуальность вашу (для файла) или группы (для подкаталога).

Команда CLEAR 20 для большей надежности очищает переменную T1 перед первым использованием в программе, делая ее свободной, так что ее значением является сам символ T1.

Операторы 25 и 27 присваивают алгебраические выражения, описывающие уравнения движения материальной точки в виде зависимости ее проекций от времени T1, переменным X и Y.

Операторы 30 и 32 определяют проекции скорости точки путем дифференцирования алгебраических выражений для соответствующих координат X и Y по времени T1. Здесь порядок производной равен единице, поэтому он опущен и не указывается в качестве третьего параметра.

Оператор 40 находит значение модуля скорости точки V путем извлечения квадратного корня из сумм квадратов проекций скорости VX и VY.

Операторы 50 и 52 определяют проекции ускорения точки путем взятия первой производной от алгебраических выражений для соответствующих проекций скорости точки VX и VY по времени T1. Здесь порядок производной, равный единице, также не указывается.

Операторы 50 и 52 можно представить также в форме второй производной от соответствующих координат X и Y, где порядок следует указывать обязательно. Это может иметь, например, следующий вид:

```
AX:=DF(X,T1,2);                                    50
AY:=DF(Y,T1,2);                                    52
```

Оператор 60 находит значение ускорения точки A путем извлечения квадратного корня из сумм квадратов проекций ускорений AX и AY.

Оператор 70 определяет тангенциальное ускорение точки AT с использованием обычной формулы по известным значениям проекций скорости и ускорения точки.

Оператор 80 находит значение нормального ускорения точки AN путем извлечения квадратного корня из разности квадратов полного A и тангенциального AT ускорений.

Оператор 85 определяет радиус кривизны траектории RO из формулы для нормального ускорения, деля квадрат скорости точки V*V на ее нормальное ускорение AN.

Команда 90 закрывает выводной файл PR3-1.LIS, находящийся в подкаталоге \KIN корневого каталога текущего диска, куда записывались результаты работы программы 3.1.

Оператор 99 запускает специальный учет контроля за файлами, улучшающий эффективность работы системы, что является стандартным способом заканчивать файлы, предназначенные для считывания.

Допустим, что программа 3.1 находится в файле PR3-1 подкаталога \KIN корневого каталога текущего диска. Тогда **для запуска ее на выполнение нужно:**

- выйти из подкаталога \ KIN и перейти в подкаталог CAB REDUCE с расположенными в нем системными файлами;
- затем следует загрузить систему, для чего нужно запустить на выполнение файл reduce.exe или reduce.bat., совместив с их именами подсветку курсора и нажав клавишу <Enter> (<Ввод>);
- после появления приглашения в командной строке следует ввести:

1: **in «\KIN\PR3-1»$** (3.5)

По этой команде в систему будет загружен файл \KIN\PR3-1, выполнены все находящиеся в нем операторы и команды программы 3.1, а результаты их выполнения будут записаны на диске в файле PR3-1.LIS в том же подкаталоге \KIN корневого каталога текущего диска. Они будут иметь для рассматриваемого примера (3.4) следующие значения:

x:=4*t1 y:=16*t1^2 – 1

vx:=4 vy:=32*t1

v:=4*sqrt(64*t1^2 + 1) ax:=0 (3.6)

ay:=32 a:=32

$$at:=\frac{256*t1}{sqrt(64*t1^2+1)} \qquad an:=\frac{32}{sqrt(64*t1^2+1)}$$

$$ro:=\frac{sqrt(64*t1^2+1)*(64*t1^2+1)}{2}$$

Обратим ваше внимание, что команда (3.5) ограничена символом $, поэтому содержимое выполняемого файла (то есть сама программа PR3-1) в выходном файле PR3-1.LIS не отображается. Это сделано для экономии места. С этой же целью результаты представлены по двое в одной строке, а не в каждой по отдельности, как это реально имеет место в выходном файле PR3-1.LIS.

Чтобы получить отображение самой программы PR3-1 в выходном файле, достаточно в команде (3.5) использовать ограничитель «;»:

1: **in «\KIN\PR3-1»;** (3.7)

В этом случае после каждого оператора программы PR3-1 будет указан соответствующий результат ее выполнения (3.6), в чем вы легко можете убедиться сами.

Отметим, что даже при использовании команды IN в форме (3.7), предотвратить указанное отображение можно использованием во входном файле (в программе PR3-1) команды

```
OFF ECHO;                                                    15
```

В этом случае полученные результаты в выходном файле PR3-1.LIS будут также иметь форму (3.6). Построенные по ним графики зависимостей найденных кинематических характеристик от времени T1 будут иметь следующий вид.

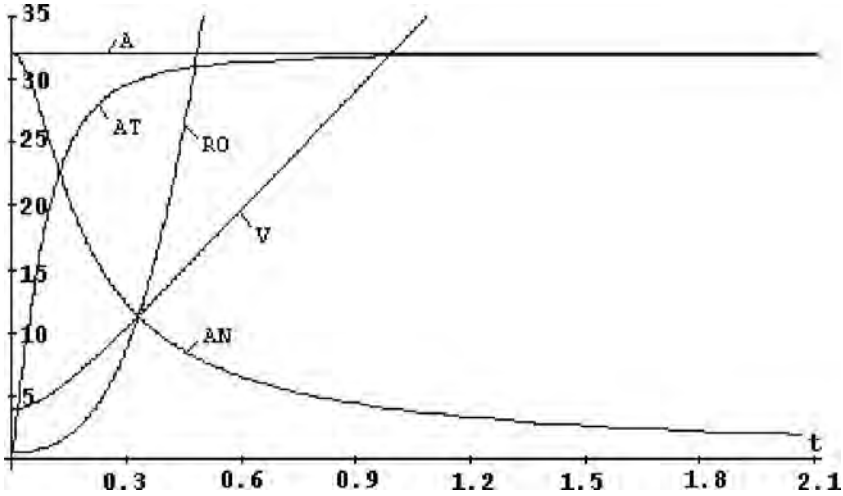

Рис. 3.1. Зависимости кинематических характеристик материальной точки от времени t (с): скорости V (м/с), тангенциального AT, нормального AN и полного ускорения A (м/с^2), а также радиуса кривизны траектории RO (м) для типового примера задания К–1 [23, с. 60–62] или [22, с. 76–79].

Дополнение 3.1. Теперь следует дополнить полученное символьное решение (3.6) соответствующим численным при конкретном значении времени T1, заданным в условии задачи.

Затем следует убедиться в совпадении результатов работы численного решения дополненной программы 3.1 и вашего аналитического (безмашинного) решения при одинаковом значении времени T1.

Представим результаты решения в численной форме с использованием действительной арифметики согласно дополнения 2.4.

Для получения численных значений V, A и RO программу 3.1 следует дополнить нижеследующим фрагментом.

```
T1:=1/2;                                                    41
ON BIGFLOAT, NUMVAL;                                        42
V;                                                          43
OFF BIGFLOAT, NUMVAL; CLEAR T1;                             44
T1:=1/2;                                                    61
ON BIGFLOAT, NUMVAL;                                        62
A;                                                          63
```

```
OFF BIGFLOAT, NUMVAL; CLEAR T1;                                    64
T1:=1/2;                                                           86
ON BIGFLOAT, NUMVAL;                                               87
RO;                                                                88
OFF BIGFLOAT, NUMVAL; CLEAR T1;                                    89
```

Оператор 41 присваивает заданное в условии типового примера задания К-1 ([23, с. 60] или [22, с. 76]) значение переменной T1.

Команда 42 устанавливает флаги BIGFLOAT и NUMVAL. Флаг BIGFLOAT обеспечивает использование в многочленах вещественных коэффициентов повышенной точности. Флаг NUMVAL устанавливает режим вычисления значений элементарных функций в форме с плавающей запятой с текущей степенью точности.

Выражение 43 распечатывает результаты численного решения для скорости точки V.

Команда 44 отменяет установленные на время флаги BIGFLOAT и NUMVAL и очищает значение переменной T1, делая ее свободной и пригодной для продолжения символьного решения задачи.

Вышеописанная последовательность операторов 41-44 встречается еще во фрагменте дополнения 3.1 дважды под номерами 61-64 и 86-89, распечатывая результаты численного решения соответственно для ускорения точки A и радиуса кривизны RO.

Оператор 41 под номерами 61 и 86 присваивает заданное в условии типового примера задания К-1 ([23, с. 60] или [22, с. 76]) значение переменной T1.

Команда 42 под номерами 62 и 87 устанавливает флаги BIGFLOAT и NUMVAL.

Выражения 63 и 88 распечатывают результаты численного решения для ускорения точки A и радиуса кривизны траектории RO соответственно.

Команда 44 под номерами 64 и 89 отменяет установленные на время флаги BIGFLOAT и NUMVAL и очищает значение переменной T1, делая ее свободной и пригодной для продолжения символьного решения задачи.

Результаты численного решения для скорости точки V, ускорения точки A и радиуса кривизны траектории RO для рассматриваемого типового примера будут иметь соответственно следующие значения: 16.4924225, 32 и 35.04639782.

Сравнив их с результатами вашего аналитического (безмашинного) расчета при одинаковом значении времени T1 убеждаемся в правильности решения.

Напомним, что вышеописанный результат достигается только при совместном действии флагов BIGFLOAT и NUMVAL, причем вместо флага BIGFLOAT *не может использоваться* флаг FLOAT.

3.4. Составление уравнений движения точки и определение ее скорости и ускорения

Составление уравнения движения материальной точки является необходимым условием для аналитического решения задания К-2 [22, с. 81-87], также как и для его символьного решения на ПК.

Поэтому рассмотрим их получение для типового примера этого задания ([22, с. 82, 86-87]), схема которого представлена на рис. 3.2.

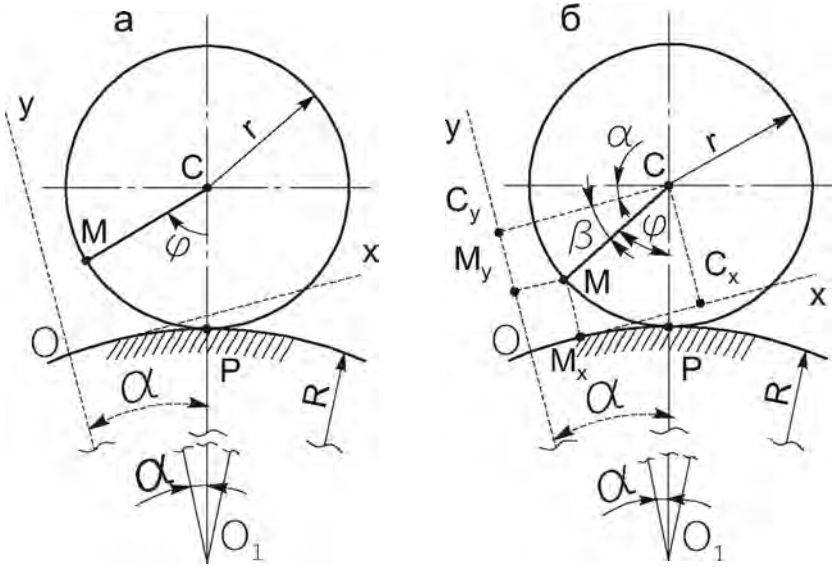

Рис. 3.2. Схема механизма типового примера задания К-2 [22, с. 86]: а — исходное положение; б — дополнения для составления уравнений движения материальной точки М

Исходные данные к представленной на рис. 3.2 схеме типового примера задания К-2 имеют вид [22, с. 82]:

$\varphi = p*t$ (рад), $r = 20$ см, $R = 100$ см, $t_1 = 1/3$ с.

Из условия равенства дуг ОР = МР также имеем: $R*\alpha = r*\varphi$ и

$$\alpha = r*\varphi/R = 0.2*\varphi = 0.2*\pi*t. \qquad (3.8)$$

Чтобы составить уравнения движения материальной точки, нужно для произвольного момента времени Т выразить ее координаты через данные задачи и параметр Т.

Время Т обычно входит в исходные данные (выражения углов или в уравнения движения других точек рассматриваемого механизма).

Полученные таким образом выражения координат для одного произвольного положения материальной точки включают переменную Т и остаются справедливыми для любого момента времени, являясь *уравнениями движения материальной точки в координатном виде в параметрической форме*.

Условимся, что проекции точек на координатные оси будем обозначать той же буквой с индексом данной оси: M_X, M_Y, C_X, C_Y (см. рис. 3.2б).

Тогда для произвольного положения точки М ее координаты X и Y через данные задачи и параметр Т можно выразить следующим образом:

$$X = OM_X = OC_X - C_XM_X = O_1C*\sin\alpha - CM*\cos\beta =$$
$$= (R+r)*\sin\alpha - r*\cos(90^\circ - (\varphi + \alpha)) = (R+r)*\sin\alpha -$$
$$- r*\sin(\varphi + \alpha), \qquad (3.9)$$
$$Y = OM_Y = O_1C_Y - O_1O - C_YM_Y = O_1C*\cos\alpha - R - CM*\sin\beta =$$
$$= (R+r)*\cos\alpha - R - r*\sin(90^\circ - (\varphi + \alpha)) = (R+r)*\cos\alpha - R -$$
$$- r*\cos(\varphi + \alpha).$$

Из рис. 3.2б видно, что угол $\beta = 90^\circ - (\varphi + \alpha)$. Поэтому при выводе уравнений движения (3.9) использовано, что

$$\cos\beta = \cos(90^\circ - (\varphi + \alpha)) = \sin(\varphi + \alpha),$$
$$\sin\beta = \sin(90^\circ - (\varphi + \alpha)) = \cos(\varphi + \alpha).$$

Для удобства составления программ заменим также используемые греческие буквы в обозначениях углов их идентификаторами, записанными в их латинской транскрипции:

AL	FI	PI
α	φ	π

(3.10)

С учетом соотношений (3.10), значений радиусов (r = 20 см, R = 100 см), углов φ и α (3.8) уравнения движения (3.9) примут следующий вид

```
X=20*(6*SIN(AL)- SIN(FI+AL)),
Y=20*(6*COS(AL)- COS(FI+AL)- 5),                    (3.11)
```

где PI=3.14159265, FI=PI*T, AL=0.2*FI.

Теперь для решения типового примера задания К-2 в символьном виде в программе 3.1 нужно исправить уравнения движения по фрагменту (3.11). Модифицированную программу 3.1 желательно поместить в другом файле, например, \KIN\PR3-1M. Чтобы не затирались предыдущие результаты расчета, следует также использовать другой файл для записи выходной информации. Для этого в программу 3.1 нужно внести следующие изменения и дополнения, отразив их в комментарии:

```
COMMENT Программа 3.1М: Составление уравнений движения точки
и определение ее скорости и ускорения для типового примера К-
2 ([22, с. 81-87] на REDUCE с представлением результатов в
символьном виде;
OUT "\KIN\PR3-1M.LIS";                                  10
FI:=PI*T1;                                              23
AL:=0.2*FI;                                             24
X:=20*(6*SIN(AL)-SIN(FI+AL));                           25
Y:=20*(6*COS(AL)-COS(FI+AL)-5);                         27
SHUT "\KIN\PR3-1M.LIS";                                 90
```

Команда 10 открывает, а команда 90 закрывает выводной файл PR3-1.LIS, находящийся в подкаталоге \KIN корневого каталога текущего диска, куда записываются результаты работы программы 3.1.

Операторы 23 и 24 присваивают алгебраические выражения, определяющие значения используемых углов FI и AL.

Операторы 25 и 27 присваивают алгебраические выражения, описывающие уравнения движения материальной точки в виде зависимости ее проекций от времени T1, переменным X и Y.

После исправления программы 3.1 с учетом вышеприведенного фрагмента, нужно:

- выйти из рабочего подкаталога \KIN в корневой каталог;
- перейти в подкаталог CAB REDUCE с расположенными в нем системными файлами и загрузить систему;

- после появления приглашения в командной строке следует ввести команду IN в форме (3.5) или (3.7) с указанием имени запускаемого на выполнение файла, например:

  1: **in "\KIN\PR3-1M"$** (3.12)

  По этой команде:

- в систему будет загружен файл \KIN\PR3-1M;
- выполнены все находящиеся в нем операторы и команды программы 3.1M;
- результаты их выполнения будут записаны на диске в файле PR3-1M.LIS в том же подкаталоге \KIN корневого каталога текущего диска.

Они будут иметь для рассматриваемого примера (3.11) следующие значения.

В целях экономии места приведем только получающиеся аналитические зависимости для:

- *скорости V*

$$v := 24*\mathrm{sqrt}\left(\cos\left(\frac{pi*t1}{5}\right)^2 - 2*\cos\left(\frac{pi*t1}{5}\right)*\cos\left(\frac{6*pi*t1}{5}\right) + \cos\left(\frac{6*pi*t1}{5}\right)^2 + \sin\left(\frac{pi*t1}{5}\right)^2 - 2*\sin\left(\frac{pi*t1}{5}\right)*\sin\left(\frac{6*pi*t1}{5}\right) + \sin\left(\frac{6*pi*t1}{5}\right)^2\right)*pi \quad (3.13)$$

- *ускорения A*

$$a := \left(24*\mathrm{sqrt}\left(\cos\left(\frac{pi*t1}{5}\right)^2 - 12*\cos\left(\frac{pi*t1}{5}\right)^2*\cos\left(\frac{6*pi*t1}{5}\right)^2 + 36*\cos\left(\frac{6*pi*t1}{5}\right)^2 + \sin\left(\frac{pi*t1}{5}\right)^2 - 12*\sin\left(\frac{pi*t1}{5}\right)*\sin\left(\frac{6*pi*t1}{5}\right) + 36*\sin\left(\frac{6*pi*t1}{5}\right)^2\right)*pi\right)/5 \quad (3.14)$$

- *радиуса кривизны RO*

$$\text{ro} := \left(120*\text{sqrt}\left(\cos\left(\frac{\text{pi}*\text{t1}}{5}\right)^2 - 2*\cos\left(\frac{\text{pi}*\text{t1}}{5}\right)*\cos\left(\frac{6*\text{pi}*\text{t1}}{5}\right) + \cos\left(\frac{6*\text{pi}*\text{t1}}{5}\right)^2 + \sin\left(\frac{\text{pi}*\text{t1}}{5}\right)^2 - 2*\sin\left(\frac{\text{pi}*\text{t1}}{5}\right)*\sin\left(\frac{6*\text{pi}*\text{t1}}{5}\right) + \sin\left(\frac{6*\text{pi}*\text{t1}}{5}\right)^2 \right)*\left(\cos\left(\frac{\text{pi}*\text{t1}}{5}\right)^2 - 2*\cos\left(\frac{\text{pi}*\text{t1}}{5}\right)*\cos\left(\frac{6*\text{pi}*\text{t1}}{5}\right) + \cos\left(\frac{6*\text{pi}*\text{t1}}{5}\right)^2 + \sin\left(\frac{\text{pi}*\text{t1}}{5}\right)^2 - 2*\sin\left(\frac{\text{pi}*\text{t1}}{5}\right)*\sin\left(\frac{6*\text{pi}*\text{t1}}{5}\right) + \sin\left(\frac{6*\text{pi}*\text{t1}}{5}\right)^2 \right) \right) / \left(\cos\left(\frac{\text{pi}*\text{t1}}{5}\right)^2 - 7*\cos\left(\frac{\text{pi}*\text{t1}}{5}\right)*\cos\left(\frac{6*\text{pi}*\text{t1}}{5}\right) + 6*\cos\left(\frac{6*\text{pi}*\text{t1}}{5}\right)^2 + \sin\left(\frac{\text{pi}*\text{t1}}{5}\right)^2 - 7*\sin\left(\frac{\text{pi}*\text{t1}}{5}\right)*\sin\left(\frac{6*\text{pi}*\text{t1}}{5}\right) + 6*\sin\left(\frac{6*\text{pi}*\text{t1}}{5}\right)^2 \right) \qquad (3.15)$$

Теперь дополним полученное символьное решение (3.13)–(3.15) соответствующим численным при заданном времени T1. Для этого в программе 3.1M, дополненной по фрагменту дополнения 3.1, в операторах 41, 61 и 86 укажем значение времени, заданное в условии задачи T1=1/3 с.

После выполнения такой дополненной программы 3.1M по команде (3.12), результаты численного решения для скорости точки V, ускорения точки A и радиуса кривизны траектории RO для рассматриваемого типового примера будут иметь соответственно следующие значения:

V=75.39822369, A=263.767832 и RO=34.28571429. (3.16)

Сравнив их с результатами вашего аналитического расчета при одинаковом значении времени T1, убеждаемся в правильности решения.

После этого с легким сердцем строим по соотношениям (3.13)–(3.15) на ПК с использованием системы DERIVE графики (см. [5, 6]), представленные на рис. 3.3.

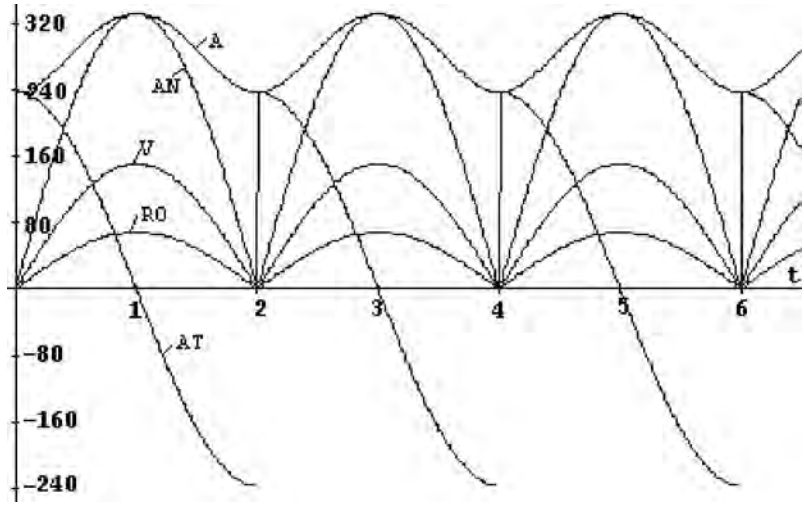

Рис. 3.3. Зависимости кинематических характеристик материальной точки от времени t (с): скорости V (м/с), тангенциального AT, нормального AN и полного ускорения A (м/с^2), а также радиуса кривизны траектории RO (м) для типового примера задания К–2 [22, с. 82, 86–87].

Качественный анализ приведенных кривых подтверждает правильность их построения, ибо выполняются все необходимые соотношения:

- при максимальном значении скорости V тангенциальное ускорение AT = dV/dt = 0;
- при любом значении времени t выполняется, что квадрат полного ускорения A равняется сумме квадратов тангенциального и нормального ускорений, что подтверждает визуальная проверка;
- при $t_1 = 1/3$ из графиков рис. 3.3 примерно получаются определенные ранее численные значения (3.16) для V, A и RO;
- уравнения движения точки (3.11) представляют собой циклоиду, у которой при четных значениях t_1 (0, 2, 4, 6с.) скорость V резко меняет свое направление. В таких острых точках графика понятие производной не имеет смысла, поэтому тангенциальное ускорение AT=dV/dt имеет в них разрыв, что и видно на соответствующей кривой на рис. 3.3.

Отметим, что пользователем выполняются только следующие действия:

- записываются нужные уравнения движения материальной точки;
- составляется программа 3.1M на REDUCE;

- программа 3.1М записывается в файле с идентификатором, например, \KIN\PR3-1M с использованием любого текстового редактора;
- файл \KIN\PR3-1M запускается на выполнение в REDUCE, в результате которого получается символьное решение (3.13)–(3.15);
- программа 3.1М дополняется для получения численного решения: в операторах 41, 61 и 86 указывается значение времени, заданное в условии задачи T_1=1/3 с (см. дополнение 3.1);
- файл с дополненной программой запускается на выполнение в REDUCE, в результате которого получается численное решение (3.16).

Одновременно с этим обычными методами механики студентом выполняется параллельное аналитическое (безмашинное) решение указанных заданий при одном заданном времени T_1.

Далее производится сравнение аналитического решения с численным (3.16) при времени T_1, после совпадения которых убеждаемся в правильности решения.

3.5. Определение кинематических характеристик при сложном движении точки

Для символьного решения на ПК К-7 [23, с. 99-106] (или К-10 [22, с. 137-143]) составление уравнений движения является дополнительной работой, так как без ПК решение заданий на сложное движение точки проводится другим способом с использованием кинематической теоремы Кориолиса. Поэтому рассмотрим их получение для типового примера этого задания (К-7 [23, с. 99, 104-106] или К-10 [22, с. 137, 141-143]), схема которого представлена на рис. 3.4,а.

Исходные данные к представленной на рис. 3.4 схеме типового примера имеют одинаковый вид ([23, с. 99] или [22, с. 137]):

$$s_r = 16 - 8\cos(3\pi t) \text{ см};$$

$$\varphi_e = 0.9t^2 - 9t^3 \text{ рад}; \qquad (3.17)$$

$$t_1 = 2/9 \text{ с}.$$

Чтобы составить уравнения движения материальной точки, нужно для произвольного момента времени T выразить ее координаты через данные задачи и параметр T. Время T обычно входит в исходные данные (в уравнения относительного s_r и переносного φ_e движений).

Полученные таким образом выражения координат для одного произвольного положения материальной точки включают переменную T и остаются справедливыми для любого момента времени, являясь уравнениями движения материальной точки в координатном виде в параметрической форме.

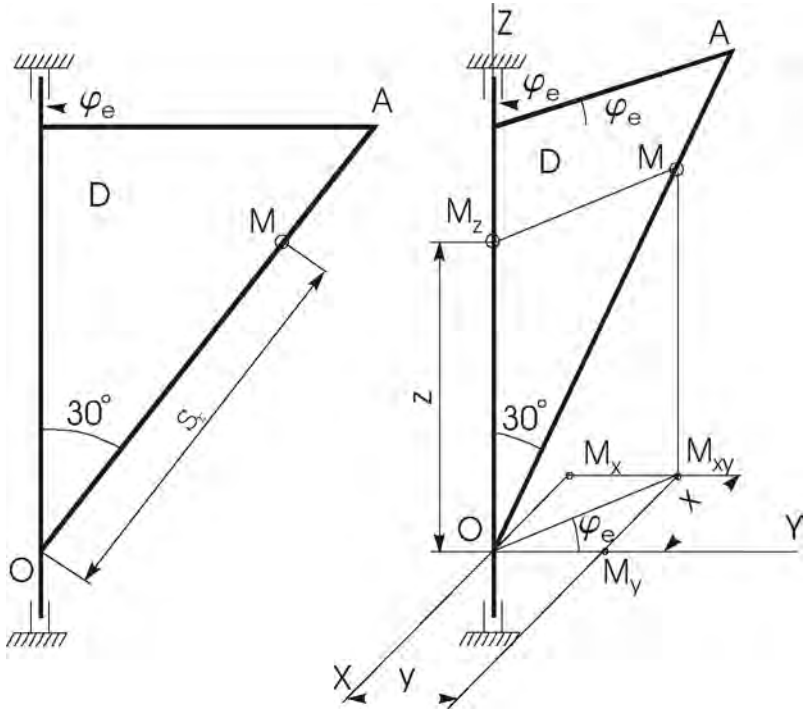

Рис. 3.4. Схема механизма типового примера задания К-7 ([23, с. 99, 104-106] или К-10 [22, с. 137, 141-143]): а — схема механизма; б — дополнения для составления уравнений движения материальной точки М

Заметим, что в большинстве вариантов задания К-7 [23, с. 101-103] (и К-10 [22, с. 138-140]) исходные схемы следует считать представленными в начальном положении переносного движения при $\varphi_e = 0$.

Поэтому для составления уравнений движения нужно дать произвольное перемещение тела, повернув его *на небольшой угол* φ_e *по направлению переносного движения* из заданного положения (рис. 3.4а) в произвольное (рис. 3.4б).

В этих заданиях также не определены положения координатных осей и, в зависимости от их выбора, уравнения могут принимать разнообразную форму, отличаясь друг от друга в конечном счете на постоянную.

Это приводит к *разным значениям координат точек*, определяемых по этим разным уравнениям, но к *одинаковым значениям скоростей и ускорений*.

Выберем координатные оси и переместим тело D из исходного положения на рис. 3.4а в произвольное положение по направлению переносного движения (в сторону положительного отсчета φ_e), как показано на рис. 3.4б.

Также условимся, что проекции точек на координатные оси или плоскость будем обозначать той же буквой с индексом данной оси или плоскости: M_X, M_Y, M_Z, M_{XY} (см. рис. 3.4б).

Тогда для произвольного положения точки M ее координаты X, Y и Z через данные задачи и параметр T можно выразить следующим образом:

$$X = -OM_X = -OM_{XY}*\sin\varphi_e = -OM*\sin 30°*\sin\varphi_e = -(s_r/2)*\sin\varphi_e;$$

$$Y = OM_Y = OM_{XY}*\cos\varphi_e = OM*\sin 30°*\cos\varphi_e = (s_r/2)*\cos\varphi_e; \quad (3.18)$$

$$Z = OM_Z = OM*\cos 30° = s_r*\cos 30°.$$

Для удобства составления программ заменим также используемые греческие и латинские буквы в обозначениях углов и индексах переменных их идентификаторами, записанными в латинской транскрипции:

SR	FIE	PI	(3.19)
s_r	φ_e	π	

С учетом соотношений (3.19), значений s_r и φ_e (3.17) уравнения движения (3.18) примут следующий вид:

```
X=-(SR/2)*SIN(FIE),

Y=(SR/2)*COS(FIE),                                    (3.20)

Z=SR*COS(PI/6),

где SR=16-8*COS(3*PI*T), FIE=0.9*T**2-9*T**3,
PI=3.1415926.
```

Отметим, что типовые примеры заданий К-7 [23, с. 99, 104-106] и К-10 [22, с. 137, 141-143] одинаковы. Поэтому их программная реализация совпадает и рассматривается на примере (3.20).

Теперь для решения типового примера задания К-7 в символьном виде в программе 3.1 нужно:

- исправить уравнения движения (операторы 25 и 27) по фрагменту (3.20);
- учесть пространственный характер задачи:
- дать соответствующие выражения для VZ и AZ;
- исправить определения для скорости V и ускорения A, введя в операторы 40 и 60 дополнительные слагаемые VZ*VZ и AZ*AZ соответственно;
- удалить операторы 70-85 из-за отсутствия необходимости при сложном движении точки определять радиус кривизны траектории.

Хотя изложенных рекомендаций достаточно для самостоятельного составления программы, для большей ясности изложения мы приведем получающийся результат полностью в нижеследующей программе 3.2.

```
COMMENT Программа 3.2: Составление уравнений движения при
сложном движении точки и определение ее скорости и ускорения
для типового примера К-7 [23, с. 99, 104-106] (или К-10 [22,
с. 137, 141-143]) на REDUCE с представлением результатов в
символьном виде;
OUT "\KIN\PR3-2.LIS";                                          10
ON FORT;                                                       15
CLEAR T1;                                                      20
SR:=16-8*COS(3*PI*T1);                                         23
FIE:=0.9*T1**2-9*T1**3;                                        24
X:=-(SR/2)*SIN(FIE);                                           25
Y:=(SR/2)*COS(FIE);                                            27
Z:=SR*COS(PI/6);                                               29
VX:=DF(X,T1);                                                  30
VY:=DF(Y,T1);                                                  32
VZ:=DF(Z,T1);                                                  34
V:=SQRT(VX*VX+VY*VY+VZ*VZ);                                    40
AX:=DF(VX,T1);                                                 50
AY:=DF(VY,T1);                                                 52
AZ:=DF(VZ,T1);                                                 54
A:=SQRT(AX*AX+AY*AY+AZ*AZ);                                    60
SHUT "\KIN\PR3-2.LIS";                                         90
OFF FORT;                                                      95
END;                                                           99
```

Команда 10 открывает, а команда 90 закрывает выводной файл PR3-2.LIS, находящийся в подкаталоге \KIN корневого каталога текущего диска, куда записываются результаты работы программы 3.2.

Напомним, что здесь, как и ранее предполагается, что программа 3.2 расположена в том же текущем подкаталоге \KIN в файле, например, с именем PR3-2.

Команда 15 поднимает, а команда 95 опускает флаг FORT, устанавливающий выдачу результатов в форме, совместимой с Фортраном. Это сделано как для удобства представления предполагаемых сложных и длинных результатов расчета, так и для показа возможности проведения дальнейших сложных численных расчетов с использованием фортран-программ.

Команда CLEAR 20 для большей надежности очищает переменную T1 перед первым использованием в программе, делая ее свободной, так что ее значением является сам символ T1.

Операторы 23 и 24 задают значения вспомогательных переменных SR и FIE с учетом данных условия (3.17).

Операторы 25, 27 и 29 присваивают алгебраические выражения, описывающие уравнения движения материальной точки в виде зависимости ее проекций от времени T1, переменным X, Y и Z.

Операторы 30, 32 и 34 определяют проекции скорости точки путем дифференцирования алгебраических выражений для соответствующих координат X, Y и Z по времени T1. Здесь число, выражающее порядок производной, равно единице, поэтому оно опущено и не указывается в качестве третьего параметра.

Оператор 40 находит значение модуля скорости точки V путем извлечения квадратного корня из сумм квадратов проекций скорости VX, VY и VZ.

Операторы 50, 52 и 54 определяют проекции ускорения точки путем взятия первой производной от алгебраических выражений для соответствующих проекций скорости точки VX, VY и VZ по времени T1. Здесь порядок производной, равный единице, также не указывается.

Операторы 50, 52 и 54 можно представить также в форме второй производной от соответствующих координат X, Y и Z, где порядок следует указывать обязательно. Это может иметь, например, следующий вид:

```
AX:=DF(X,T1,2);                                    50
AY:=DF(Y,T1,2);                                    52
AZ:=DF(Z,T1,2);                                    54
```

Оператор 60 находит значение ускорения точки A путем извлечения квадратного корня из сумм квадратов проекций ускорений AX, AY и AZ.

Оператор 99, как обычно, запускает специальный учет контроля за файлами, улучшающий эффективность работы системы, что является стандартным способом заканчивать файлы, предназначенные для считывания.

Теперь следует записать на диске набранную программу 3.2 в файл, например, с именем \KIN\PR3-2. После чего для запуска ее на выполнение нужно выйти из рабочего подкаталога \KIN корневого каталога текущего диска, перейти в подкаталог CAB REDUCE с расположенными в нем системными файлами и загрузить систему. Затем после появления приглашения в командной строке следует ввести команду IN в форме (3.5) или (3.7) с указанием имени запускаемого на выполнение файла, например:

1: **in"\KIN\PR3-2"$** (3.21)

По этой команде в систему будет загружен файл \KIN\PR3-2, выполнены все находящиеся в нем операторы и команды программы 3.2, а результаты их выполнения будут записаны на диске в файле PR3-2.LIS в том же подкаталоге \KIN корневого каталога текущего диска. Они будут иметь для рассматриваемого примера (3.20) следующие значения. В целях экономии места приведем только получающиеся аналитические зависимости для:

- *скорости V*

```
 ANS4=-270.*COS(3.*PI*T1)**2*SIN((90.*T1**3-9.*T1**2)/
. 10.)**2*T1**3+9.*COS(3.*PI*T1)**2*SIN((90.*T1**3-9.*
. T1**2)/10.)**2*T1**2-8100.*COS(3.*PI*T1)*SIN((90.*T1
. **3-9.*T1**2)/10.)**2*T1**4+1080.*COS(3.*PI*T1)*SIN(
. (90.*T1**3-9.*T1**2)/10.)**2*T1**3-36.*COS(3.*PI*T1)
. *SIN((90.*T1**3-9.*T1**2)/10.)**2*T1**2+25.*SIN((90.
. *T1**3-9.*T1**2)/10.)**2*SIN(3.*PI*T1)**2*PI**2+
. 8100*SIN((90.*T1**3-9.*T1**2)/10.)**2*T1**4-1080.*
. SIN((90.*T1**3-9.*T1**2)/10.)**2*T1**3+36.*SIN((90.*
. T1**3-9.*T1**2)/10.)**2*T1**2+75.*SIN(3.*PI*T1)**2*
. PI**2                                              (3.22)
 ANS3=2025.*COS((90.*T1**3-9.*T1**2)/10.)**2*COS(3.*PI
. *T1)**2*T1**4-270.*COS((90.*T1**3-9.*T1**2)/10.)**2*
```

```
. COS(3.*PI*T1)**2*T1**3+9.*COS((90.*T1**3-9.*T1**2)/
. 10.)**2*COS(3.*PI*T1)**2*T1**2-8100.*COS((90.*T1**3-
. 9.*T1**2)/10.)**2*COS(3.*PI*T1)*T1**4+1080.*COS((90.*
. T1**3-9.*T1**2)/10.)**2*COS(3.*PI*T1)*T1**3-36.*COS(
. (90.*T1**3-9.*T1**2)/10.)**2*COS(3.*PI*T1)*T1**2+25.
. *COS((90.*T1**3-9.*T1**2)/10.)**2*SIN(3.*PI*T1)**2*
. PI**2+8100.*COS((90.*T1**3-9.*T1**2)/10.)**2*T1**4-
. 1080.*COS((90.*T1**3-9.*T1**2)/10.)**2*T1**3+36.*COS(
. (90.*T1**3-9.*T1**2)/10.)**2*T1**2+2025.*COS(3.*PI*
. T1)**2*SIN((90.*T1**3-9.*T1**2)/10.)**2*T1**4+ANS4
 ANS2=SQRT(ANS3)
 ANS1=12.*ANS2
 V=ANS1/5.
```

- *ускорения A*

```
 ANS10=437400.*SIN((90.*T1**3-9.*T1**2)/10.)**2*T1**6-
. 19440.*SIN((90.*T1**3-9.*T1**2)/10.)**2*T1**5+324.*
. SIN((90.*T1**3-9.*T1**2)/10.)**2*T1**4+90000.*SIN((
. 90.*T1**3-9.*T1**2)/10.)**2*T1**2-6000.*SIN((90.*T1**
. 3-9.*T1**2)/10.)**2*T1+100.*SIN((90.*T1**3-9.*T1**2)
. /10.)**2
 ANS9=-90000.*COS(3.*PI*T1)*SIN((90.*T1**3-9.*T1**2)/
. 10.)**2*T1**2+6000.*COS(3.*PI*T1)*SIN((90.*T1**3-9.*
. T1**2)/10.)**2*T1-100.*COS(3.*PI*T1)*SIN((90.*T1**3-
. 9.*T1**2)/10.)**2+202500.*SIN((90.*T1**3-9.*T1**2)/
. 10.)**2*SIN(3.*PI*T1)**2*PI**2*T1**4-27000.*SIN((90.*
. T1**3-9.*T1**2)/10.)**2*SIN(3.*PI*T1)**2*PI**2*T1**3
. +900.*SIN((90.*T1**3-9.*T1**2)/10.)**2*SIN(3.*PI*T1)
. **2*PI**2*T1**2+270000.*SIN((90.*T1**3-9.*T1**2)/10.
. )**2*SIN(3.*PI*T1)*PI*T1**3-27000.*SIN((90.*T1**3-9.
```

```
     . *T1**2)/10.)**2*SIN(3.*PI*T1)*PI*T1**2+600.*SIN((90.
     . *T1**3-9.*T1**2)/10.)**2*SIN(3.*PI*T1)*PI*T1+
     . 16402500.*SIN((90.*T1**3-9.*T1**2)/10.)**2*T1**8-
     . 4374000.*SIN((90.*T1**3-9.*T1**2)/10.)**2*T1**7+ANS10
      ANS8=13500.*COS(3.*PI*T1)*SIN((90.*T1**3-9.*T1**2)/
     . 10.)**2*SIN(3.*PI*T1)*PI*T1**2-300.*COS(3.*PI*T1)*SIN
     . ((90.*T1**3-9.*T1**2)/10.)**2*SIN(3.*PI*T1)*PI*T1-
     . 202500.*COS(3.*PI*T1)*SIN((90.*T1**3-9.*T1**2)/10.)**
     . 2*PI**2*T1**4+27000.*COS(3.*PI*T1)*SIN((90.*T1**3-9.
     . *T1**2)/10.)**2*PI**2*T1**3-900.*COS(3.*PI*T1)*SIN((
     . 90.*T1**3-9.*T1**2)/10.)**2*PI**2*T1**2-16402500.*COS
     . (3.*PI*T1)*SIN((90.*T1**3-9.*T1**2)/10.)**2*T1**8+
     . 4374000.*COS(3.*PI*T1)*SIN((90.*T1**3-9.*T1**2)/10.)
     . **2*T1**7-437400.*COS(3.*PI*T1)*SIN((90.*T1**3-9.*T1
     . **2)/10.)**2*T1**6+19440.*COS(3.*PI*T1)*SIN((90.*T1
     . **3-9.*T1**2)/10.)**2*T1**5-324.*COS(3.*PI*T1)*SIN((
     . 90.*T1**3-9.*T1**2)/10.)**2*T1**4+ANS9
      ANS7=450.*COS(3.*PI*T1)**2*SIN((90.*T1**3-9.*T1**2)/
     . 10.)**2*PI**2*T1**2+4100625.*COS(3.*PI*T1)**2*SIN((
     . 90.*T1**3-9.*T1**2)/10.)**2*T1**8-1093500.*COS(3.*PI*
     . T1)**2*SIN((90.*T1**3-9.*T1**2)/10.)**2*T1**7+
     . 109350.*COS(3.*PI*T1)**2*SIN((90.*T1**3-9.*T1**2)/10.
     . )**2*T1**6-4860.*COS(3.*PI*T1)**2*SIN((90.*T1**3-9.*
     . T1**2)/10.)**2*T1**5+81.*COS(3.*PI*T1)**2*SIN((90.*
     . T1**3-9.*T1**2)/10.)**2*T1**4+22500.*COS(3.*PI*T1)**
     . 2*SIN((90.*T1**3-9.*T1**2)/10.)**2*T1**2-1500.*COS(
     . 3.*PI*T1)**2*SIN((90.*T1**3-9.*T1**2)/10.)**2*T1+25.*
     . COS(3.*PI*T1)**2*SIN((90.*T1**3-9.*T1**2)/10.)**2+
     . 1875.*COS(3.*PI*T1)**2*PI**4-135000.*COS(3.*PI*T1)*
     . SIN((90.*T1**3-9.*T1**2)/10.)**2*SIN(3.*PI*T1)*PI*T1
     . **3+ANS8
```



```
      ANS6=600.*COS((90.*T1**3-9.*T1**2)/10.)**2*SIN(3.*PI*
     . T1)*PI*T1+16402500.*COS((90.*T1**3-9.*T1**2)/10.)**2
     . *T1**8-4374000.*COS((90.*T1**3-9.*T1**2)/10.)**2*T1
     . **7+437400.*COS((90.*T1**3-9.*T1**2)/10.)**2*T1**6-
     . 19440.*COS((90.*T1**3-9.*T1**2)/10.)**2*T1**5+324.*
     . COS((90.*T1**3-9.*T1**2)/10.)**2*T1**4+90000.*COS((
     . 90.*T1**3-9.*T1**2)/10.)**2*T1**2-6000.*COS((90.*T1**
     . 3-9.*T1**2)/10.)**2*T1+100.*COS((90.*T1**3-9.*T1**2)
     . /10.)**2+625.*COS(3.*PI*T1)**2*SIN((90.*T1**3-9.*T1
     . **2)/10.)**2*PI**4+101250.*COS(3.*PI*T1)**2*SIN((90.
     . *T1**3-9.*T1**2)/10.)**2*PI**2*T1**4-13500.*COS(3.*
     . PI*T1)**2*SIN((90.*T1**3-9.*T1**2)/10.)**2*PI**2*T1
     . **3+ANS7                                          (3.23)
      ANS5=-437400.*COS((90.*T1**3-9.*T1**2)/10.)**2*COS(3.
     . *PI*T1)*T1**6+19440.*COS((90.*T1**3-9.*T1**2)/10.)**
     . 2*COS(3.*PI*T1)*T1**5-324.*COS((90.*T1**3-9.*T1**2)/
     . 10.)**2*COS(3.*PI*T1)*T1**4-90000.*COS((90.*T1**3-9.*
     . T1**2)/10.)**2*COS(3.*PI*T1)*T1**2+6000.*COS((90.*T1
     . **3-9.*T1**2)/10.)**2*COS(3.*PI*T1)*T1-100.*COS((90.
     . *T1**3-9.*T1**2)/10.)**2*COS(3.*PI*T1)+202500.*COS((
     . 90.*T1**3-9.*T1**2)/10.)**2*SIN(3.*PI*T1)**2*PI**2*T1
     . **4-27000.*COS((90.*T1**3-9.*T1**2)/10.)**2*SIN(3.*
     . PI*T1)**2*PI**2*T1**3+900.*COS((90.*T1**3-9.*T1**2)/
     . 10.)**2*SIN(3.*PI*T1)**2*PI**2*T1**2+270000.*COS((90.
     . *T1**3-9.*T1**2)/10.)**2*SIN(3.*PI*T1)*PI*T1**3-
     . 27000.*COS((90.*T1**3-9.*T1**2)/10.)**2*SIN(3.*PI*T1)
     . *PI*T1**2+ANS6
      ANS4=-1500.*COS((90.*T1**3-9.*T1**2)/10.)**2*COS(3.*
     . PI*T1)**2*T1+25.*COS((90.*T1**3-9.*T1**2)/10.)**2*
```

```
     . COS(3.*PI*T1)**2-135000.*COS((90.*T1**3-9.*T1**2)/
     . 10.)**2*COS(3.*PI*T1)*SIN(3.*PI*T1)*PI*T1**3+13500.*
     . COS((90.*T1**3-9.*T1**2)/10.)**2*COS(3.*PI*T1)*SIN(
     . 3.*PI*T1)*PI*T1**2-300.*COS((90.*T1**3-9.*T1**2)/10.)
     . **2*COS(3.*PI*T1)*SIN(3.*PI*T1)*PI*T1-202500.*COS((
     . 90.*T1**3-9.*T1**2)/10.)**2*COS(3.*PI*T1)*PI**2*T1**4
     . +27000.*COS((90.*T1**3-9.*T1**2)/10.)**2*COS(3.*PI*
     . T1)*PI**2*T1**3-900.*COS((90.*T1**3-9.*T1**2)/10.)**
     . 2*COS(3.*PI*T1)*PI**2*T1**2-16402500.*COS((90.*T1**3
     . -9.*T1**2)/10.)**2*COS(3.*PI*T1)*T1**8+4374000.*COS(
     . (90.*T1**3-9.*T1**2)/10.)**2*COS(3.*PI*T1)*T1**7+
     . ANS5
      ANS3=625.*COS((90.*T1**3-9.*T1**2)/10.)**2*COS(3.*PI*
     . T1)**2*PI**4+101250.*COS((90.*T1**3-9.*T1**2)/10.)**
     . 2*COS(3.*PI*T1)**2*PI**2*T1**4-13500.*COS((90.*T1**3
     . -9.*T1**2)/10.)**2*COS(3.*PI*T1)**2*PI**2*T1**3+450.
     . *COS((90.*T1**3-9.*T1**2)/10.)**2*COS(3.*PI*T1)**2*
     . PI**2*T1**2+4100625.*COS((90.*T1**3-9.*T1**2)/10.)**
     . 2*COS(3.*PI*T1)**2*T1**8-1093500.*COS((90.*T1**3-9.*
     . T1**2)/10.)**2*COS(3.*PI*T1)**2*T1**7+109350.*COS((
     . 90.*T1**3-9.*T1**2)/10.)**2*COS(3.*PI*T1)**2*T1**6-
     . 4860.*COS((90.*T1**3-9.*T1**2)/10.)**2*COS(3.*PI*T1)
     . **2*T1**5+81.*COS((90.*T1**3-9.*T1**2)/10.)**2*COS(
     . 3.*PI*T1)**2*T1**4+22500.*COS((90.*T1**3-9.*T1**2)/
     . 10.)**2*COS(3.*PI*T1)**2*T1**2+ANS4
      ANS2=SQRT(ANS3)
      ANS1=36.*ANS2
      A=ANS1/25.
```

Отметим, что именно *для выработки бесстрашия перед такими огромными алгебраическими выражениями, ужасающими для ручной обработки, они здесь и приведены.*

Получены они компьютером, им же будут и дальше обрабатываться. Ваше дело — лишь вовремя давать ему указания путём набора весьма простых команд и, после получения аналитического решения и его графической интерпретации, сосредоточиться на их осмыслении и анализе.

Режим FORT включает все достоинства выключения флага NAT при получении длинных результатов для представления их по строкам. Кроме этого строго соблюдается синтаксис фортран-программ:

- все выражения начинаются с 7-й колонки;
- если запись не помещается на одной строке, то в последующих строках появится признак продолжения – знак "." (точка) в 6-й колонке, сопровождаемая пробелом (максимальное количество строк продолжения 19);
- при записи длинных выражений используются вспомогательные переменные (ANS4–ANS1 для скорости V в (3.22) и ANS10– ANS1 для ускорения A в (3.23)). Они разбивают громоздкое выражение на ряд более простых. Их число, как и остальные режимы вывода, могут быть по желанию изменены (см. дополнение 3.2). Имя ANS с соответствующим номером присваивается каждому выражению по умолчанию и все его строки также печатаются с седьмой позиции с использованием синтаксиса Фортрана.

Теперь дополним полученное символьное решение (3.22)–(3.23) соответствующим численным при конкретном значении времени T_1. Для этого следует в программе 3.2, дополненной по фрагменту дополнения 3.1 без команд 86-89, в операторах 41 и 61 указать значение времени, заданное в условии задачи: $T_1=2/9$ с.

После выполнения такой дополненной программы 3.2 по команде (3.21), результаты численного решения для скорости точки V и ускорения точки A для рассматриваемого типового примера будут иметь соответственно следующие значения:

$$V = 65.9604443,\ A = 394.9225358. \qquad (3.24)$$

Сравнив их с результатами вашего аналитического (безмашинного) расчета при одинаковом значении времени T1 убеждаемся в правильности решения. После чего строим по соотношениям (3.22)–(3.23) на ПК с использованием системы DERIVE (см. [5, 6]) графики, представленные на рис. 3.5.

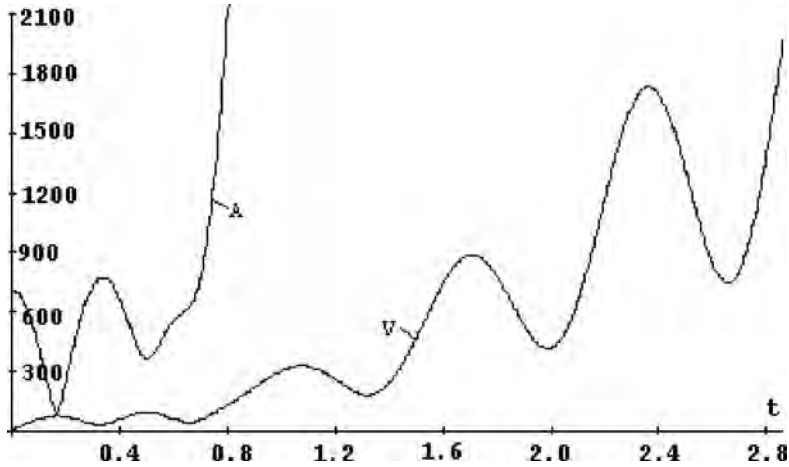

Рис. 3.5. Зависимости скорости V (м/с) и полного ускорения A (м/с²) от времени t (с) при сложном движении материальной точки для типового примера заданий К–7 [23, с. 99, 104–106] или К–10 [22, с. 137, 141–143]

Отметим, что пользователем также составляются уравнения движения материальной точки при ее сложном движении и программа на REDUCE для символьного определения кинематических характеристик при различных временах T (что называется символьным решением). Затем следует сеанс работы на ПК и получение символьного (3.22)–(3.23) и численного (3.24) решения. Одновременно с этим обычными методами механики студентом производится параллельное аналитическое (безмашинное) решение указанных заданий при одном заданном времени t_1 (которое рассмотрено в сборнике [23, с. 99, 104–106] или К–10 [22, с. 137, 141–143] и в пособии не повторяется). После его совпадения с численным решением (3.24), убедившись в правильности решения, на ПК с использованием системы DERIVE (см. [5, 6]) строятся графики.

При этом не следует пугаться внешнего вида формул типа (3.22)–(3.23). Вам не придется их проверять и набирать вручную для ввода в CAB DERIVE. Они будут считаны в буфер обмена, быстро и безошибочно вставлены в текстовый редактор Word, где машинным образом некоторые специальные символы будут представлены в понимаемой системой DERIVE форме, опять скопированы в буфер и вставлены в CAB DERIVE, после чего эта система быстро и точно построит нужные графики. Все будет сделано действительно с легким сердцем и удовольствием, которые доставляют современные CAB.

Дополнение 3.2. Имеется несколько способов, позволяющих изменить автоматический формат вывода в фортрановском виде.

1. Задание с помощью оператора присваивания соответствующего значения внутренней редьюсовской переменной CARDNO*

```
CARDNO!* := число;                                    (3.25)
```

определит количество строк продолжения для записи одного выражения. От заданного значения зависит число вспомогательных переменных, которые используются для записи сложных выражений. В начальном состоянии параметр CARDNO* устанавливается равным 20. При этом значении для скорости V в (3.22) используются вспомогательные переменные ANS4–ANS1 и ANS10–ANS1 для ускорения A в (3.23).

Их использование, как обычно считается, удобно для записи программы на Фортране, ибо они разбивают громоздкое выражение на ряд более простых.

Но при машинной генерации выражений в аналитическом виде, записи их во внешний файл и последующей также компьютерной обработке, это достоинство не несет практически никаких выгод, ибо простота этих выражений весьма относительна (см. (3.22) и (3.23)).

Кроме этого, использование вспомогательных переменных приводит к тому, что мы не можем передать все сложное выражение за один раз через буфер обмена (<Ctrl>+C – <Ctrl>+V), а вынуждены считывать его порциями по значениям каждой вспомогательной переменной. Емкость буфера обмена не накладывает практически никаких ограничений на величину копируемого объекта. Поэтому сложность и громоздкость передаваемого выражения не имеют никакого значения. Важно, чтобы оно могло быть считано в буфер обмена за один раз.

Для достижения этой цели достаточно только соответствующим образом изменить значение переменной CARDNO*. Например. для получения выражений (3.22) и (3.23) без использования вспомогательных переменных следует задать значение параметра CARDNO* соответственно равным 34 для скорости V (см. оператор 41 фрагмента программы дополнения 3.3) и 145 для ускорения A.

Отметим, что для включения в состав имен переменных (кроме латинских букв и цифр) разрешенных символов (например, “*” у переменной CARDNO*), ***перед ним*** всегда указывается специальный символ, которым может быть в зависимости от версии REDUCE восклицательный знак (“!”) или закрывающая скобка (“]”).

2. Задание с помощью оператора присваивания соответствующего значения внутренней редьюсовской переменной FORTWIDTH*

```
FORTWIDTH!* := число;                                    (3.26)
```

определит количество столбцов при выводе сложного выражения. В исходном состоянии системы это число равно 70.

3. Система REDUCE после каждого изолированного целочисленного коэффициента в фортрановском выражении автоматически вставляет десятичную точку (таким образом, что 4 становится 4.). Для предотвращения этого следует опустить флаг PERIOD

```
OFF PERIOD;                                              (3.27)
```

4. Имя ANS, автоматически присвоенное непоименованному переменному выражению и его частям, может быть изменено с помощью операции VARNAME.

Эта операция имеет в качестве параметра один идентификатор, который и заменяет идентификатор ANS в качестве имени выражения. Значением операции VARNAME является её параметр. Например:

```
VARNAME PEREM;                                           17
```

Теперь, после включения предложения 17 в программу 3.2, в результате ее работы аналитические ее выражения (3.22) и (3.23) будут представлены с использованием вспомогательных переменных PEREM4 – PEREM1 для скорости V и PEREM10 – PEREM1 для ускорения A (вместо соответственно ANS4 – ANS1 в (3.22) и ANS10 – ANS1 в (3.23)).

Дополнение 3.3. Приведем пример записи программ в синтаксисе Фортрана во внешний файл.

Эта возможность организована при помощи флага FORT с использованием оператора WRITE. Возьмем в качестве основы программу 3.2. Ограничимся с целью экономии места дополнением для представления в синтаксисе Фортрана только скорости точки V, которое получим без использования вспомогательных переменных, задав соответствующим образом значение CARDNO*. Для этого в программу 3.2 нужно добавить следующий фрагмент:

```
CARDNO!*:=34$                                            41
OUT "\PR\D3-2.FOR";                                      42
WRITE "      PI=3.141592654"$                            43
WRITE "      T1=2./9."$                                  44
V;                                                       45
WRITE "      PRINT 20, ANS"$                             46
```

```
WRITE "20    FORMAT(5X,'СКОРОСТЬ V = ',G12.5)"$          47
WRITE "      STOP"$                                      48
WRITE "      END"$                                       49
SHUT "\PR\D3-2.FOR"$                                    49а
OUT "\KIN\PR3-2.LIS";                                   49б
```

Оператор 41 задает значение внутренней редьюсовской переменной CARDNO*, которая определяет количество строк продолжения для записи одного выражения. Оно предварительно подобрано таким образом, чтобы вызвать печать значения для скорости V без использования вспомогательных переменных.

Команда 42 открывает второй выводной файл D3-2.FOR, находящийся в подкаталоге \PR корневого каталога текущего диска, куда будут записываться результаты работы операторов 43–49 программы 3.2 с дополнением 3.3. Напомним, что сама программа 3.2 расположена в подкаталоге \KIN в файле с именем PR3-2. Там же расположен и первый выводной файл PR3-2.LIS, открытый командой 10. В него записывались результаты работы операторов 22-40 программы 3.2.

В командах WRITE 43–44 и 46–49, в качестве списка параметров используется текст вспомогательных операторов Фортрана, заключённый в кавычки. Он выводится в том же виде без кавычек на печать. Для записи операторов Фортрана с 7-й позиции и для их отделения друг от друга в кавычках указываются пробелы.

Команды 43–44 печатают арифметические операторы присваивания Фортрана, задающие значения переменным PI и T1. Отметим, что в REDUCE переменная PI является зарезервированной и попытка присваивания ей значения приведет к ошибке, в то время как для Фортрана задание ее значения обязательно.

Команда 45 производит печать выражения для скорости точки V, определяемого оператором 40 программы 3.2. Вывод производится в синтаксисе Фортрана с учетом значения внутренней редьюсовской переменной CARDNO*. Так как выражение только распечатывается, а не определяется, как в операторе 40, то оно присваивается по умолчанию переменной ANS.

Команды 46 и 47 печатают операторы PRINT и FORMAT, выполняющих печать значения переменной ANS, вычисленной после работы программы на Фортране, впереди которого будет напечатан заголовок “СКОРОСТЬ V = ”.

Команды 48 и 47 печатают операторы Фортрана STOP и END, указывающие соответственно на конец вычислений и конец основной программы.

Команда 49а закрывает второй выводной файл D3-2.FOR, находящийся в подкаталоге \PR корневого каталога текущего диска, куда записывались результаты работы операторов 43–49 программы 3.2 с дополнением 3.3.

Команда 49б повторно открывает первый выводной файл PR3-2.LIS, расположенный в подкаталоге \KIN. Если этого не сделать, то после закрытия командой 49а выводного файла D3-2.FOR, вывод будет направляться на терминал.

После второй команды 10 под номером 49б вывод будет происходить в конец файла PR3-2.LIS. Теперь результаты работы операторов 50-60 также будут записаны в этом файле. Поэтому основной вывод программы 3.2 не изменится после добавления операторов 41-49б дополнения 3.3 (кроме числа вспомогательных переменных для печати ускорения A вследствие изменения оператором 41 значения внутренней редьюсовской переменной CARDNO*).

После запуска на выполнение файла PR3-2 с дополнением операторов 41-49б по команде (3.21), в файле D3-2.FOR, находящемся в подкаталоге PR корневого каталога диска C, будет находиться исходный текст программы на Фортране для численного определения скорости V по полученным в REDUCE выражениям, приведенный ниже:

```
      PI=3.141592654
      T1=2./9.
     ANS=(12.*SQRT(2025.*COS((90.*T1**3-9.*T1**2)/10.)**2*
     . COS(3.*PI*T1)**2*T1**4-270.*COS((90.*T1**3-9.*T1**2)
     . /10.)**2*COS(3.*PI*T1)**2*T1**3+9.*COS((90.*T1**3-9.
     . *T1**2)/10.)**2*COS(3.*PI*T1)**2*T1**2-8100.*COS((
     . 90.*T1**3-9.*T1**2)/10.)**2*COS(3.*PI*T1)*T1**4+1080.
     . *COS((90.*T1**3-9.*T1**2)/10.)**2*COS(3.*PI*T1)*T1**
     . 3-36.*COS((90.*T1**3-9.*T1**2)/10.)**2*COS(3.*PI*T1)
     . *T1**2+25.*COS((90.*T1**3-9.*T1**2)/10.)**2*SIN(3.*
     . PI*T1)**2*PI**2+8100.*COS((90.*T1**3-9.*T1**2)/10.)
     . **2*T1**4-1080.*COS((90.*T1**3-9.*T1**2)/10.)**2*T1
     . **3+36.*COS((90.*T1**3-9.*T1**2)/10.)**2*T1**2+2025.
     . *COS(3.*PI*T1)**2*SIN((90.*T1**3-9.*T1**2)/10.)**2*
     . T1**4-270.*COS(3.*PI*T1)**2*SIN((90.*T1**3-9.*T1**2)
     . /10.)**2*T1**3+9.*COS(3.*PI*T1)**2*SIN((90.*T1**3-9.
     . *T1**2)/10.)**2*T1**2-8100.*COS(3.*PI*T1)*SIN((90.*
     . T1**3-9.*T1**2)/10.)**2*T1**4+1080.*COS(3.*PI*T1)*
```

```
     . SIN((90.*T1**3-9.*T1**2)/10.)**2*T1**3-36.*COS(3.*PI
     . *T1)*SIN((90.*T1**3-9.*T1**2)/10.)**2*T1**2+25.*SIN(
     . (90.*T1**3-9.*T1**2)/10.)**2*SIN(3.*PI*T1)**2*PI**2+
     . 8100.*SIN((90.*T1**3-9.*T1**2)/10.)**2*T1**4-1080.*
     . SIN((90.*T1**3-9.*T1**2)/10.)**2*T1**3+36.*SIN((90.*
     . T1**3-9.*T1**2)/10.)**2*T1**2+75.*SIN(3.*PI*T1)**2*
     . PI**2))/5.                                          (3.28)
20    FORMAT(5X,'СКОРОСТЬ V = ",G12.5)
      PRINT 20, ANS
      STOP
      END
```

Дополнение 3.4. Рассмотрим для удобства пользования пособием дальнейшую обработку полученной программы (3.28) на Фортране: последующую ее трансляцию и выполнение на ПК.

Ввод ее исходного текста (в файл D3-2.FOR) был осуществлен в результате работы модифицированной по дополнению 3.3 программы 3.2 на REDUCE. Для получения выполняемой программы (с расширением .exe) нужно произвести ее компиляцию и редактирование. Вызов компилятора (транслятора) производится по команде FL, в которой указываются режимы компиляции, начинающиеся с символа "/" (наклонная черта), и имя обрабатываемого файла. После ее окончания создается объектный файл (тип .obj). Сообщения об ошибках в программе выводятся на экран и помещаются также в отдельный файл (тип .lst).

Если в программе отсутствуют серьезные ошибки, то с помощью команды LINK вызывается Редактор связей для создания выполняемого файла (тип .exe). В процессе редактирования к основному объектному файлу присоединяются необходимые программы из библиотеки Фортрана, а также объектные файлы, дополнительно указанные в команде.

Команда FL имеет следующий формат:

```
FL [режим] [имя-файла] [режим] [имя-файла]...
  [/link [библиотеки] [режимы-редактирования]]   (3.29)
```

где `режим` – один из режимов компилятора (см. п. 8.4. 1 [17, с. 141-143]);

`имя-файла` – идентификатор исходного файла;

`библиотеки` – список имен библиотечных файлов, явно заданных для редактирования;

`режимы-редактирования` - режимы Редактора связей (см. п. 8.4.2 [17, с. 143-145]).

Если режим `/link` используется в команде FL, то компилятор автоматически вызывает Редактор связей для создания выполняемого файла, если нет – то потребуется отдельный шаг редактирования с помощью команды LINK (см. п. 8.4 [17, с. 140-145]). Команды FL и LINK вследствие своей сложности помещаются обычно в командные файлы (с типом .bat).

Отметим, что режимы могут быть записаны в любом месте команды FL. Они относятся ко всем файлам, имена которых следуют за ними в команде.

Не смотря на то, что имена файлов и команд могут задаваться как строчными, так и прописными буквами, *режимы в команде FL следует задавать так, как они указаны в пособии, например, /FPi.*

При указании режима /HELP (или /help) выдается справочная информация о команде FL. Например, по команде FL /HELP она выводится на экран, а по FL /HELP >PRN перенаправляется на печатающее устройство.

Пусть исполняемые файлы компилятора Фортрана помещены в подкаталоге C:\F5\BIN, библиотечные - в C:\F5\LIB, графические - в C:\F5\INCLUDE, а временно создаваемые файлы в подкаталоге C:\F5\TMP. Создание этих подкаталогов и помещение в них файлов выполняется автоматически после указания их имен в ответ на соответствующие вопросы при инсталляции Фортрана.

Будем также считать, что к ним указан путь в команде PATH файла AUTOEXEC.BAT:

```
PATH C:\F5\BIN;C:\F5\LIB;C:\F5\INCLUDE;C:\F5\TMP    (3.30)
```

Отметим, что в списке каталогов, задаваемых командой PATH, обычно перечисляются через точку с запятой те каталоги данного ПК, в которых находятся исполняемые программы общего назначения. Поэтому не следует обращать внимание на их список.

Важно только, чтобы в их числе находились пути команды (3.30). Рекомендуется также имена каталогов в команде PATH указывать полностью от корневого каталога соответствующего диска (как указано в (3.30)), что позволит командному процессору DOS правильно их находить из любого текущего каталога и диска.

В подкаталоге C:\F5\BIN создадим командный файл FFLL.BAT для запуска компилятора Фортрана (что позволит вызывать его из любого текущего подкаталога):

```
@ECHO OFF                                          10
PATH C:\F5\BIN;                                    20
```

```
SET LIB=C:\F5\LIB                                30
SET INCLUDE=C:\F5\INCLUDE                        40     (3.31)
SET TMP=C:\F5\TMP                                50
FL /Fs /Fpi %1.FOR                               60
```

Команда 10 предназначена для того, чтобы последующие не выводились на экран.

Команда 20 указывает путь в подкаталог, в котором находятся исполняемые файлы компилятора.

Команды 30-50 задают переменные окружения, указывающие расположение подкаталогов, в которых находятся библиотечные, графические и временно создаваемые файлы.

Команда 60 производит вызов программы компилятора Фортрана. Режим `/Fs` используется для вывода распечатки исходного текста компилируемых модулей программ (с типом .LST), которая полезна при их отладке.

По режиму `/c` выполняется только компиляция исходных файлов, заданных в команде. По умолчанию объектному файлу (с типом .OBJ) присваивается такое же основное имя, как у соответствующего исходного файла.

Если режим `/c` в команде FL отсутствует, то используется по умолчанию режим `/link`, полученный объектный файл редактируется и создается выполняемый файл (с типом .EXE). Это удобно при работе с программами, все модули которых располагаются в одном файле, как в нашем случае. При компиляции подпрограмм, находящихся в отдельных файлах, следует использовать режим `/c` (см. п. 8.4 [17, с. 140-145]).

Согласование режима `/FP`, осуществляющего управление операциями с плавающей точкой, и используемой библиотеки Фортрана является особенно важным (см. [17, с. 140-145]). Режим `/FPi` полезен, когда нет уверенности в наличии математического сопроцессора. Если он присутствует, то программа его использует. При отсутствии сопроцессора режим `/FPi` обеспечивает получение наиболее высокой точности выполнения операций с плавающей точкой.

Имя компилируемой программы, указываемое при запуске после имени командного файла (FFLL в нашем случае), является заменяемым параметром. Оно будет подставлено при выполнении команд вместо символа %1. Тип файла (.FOR) уже указан в процедуре (3.31), поэтому повторно его указывать не нужно.

Например, для компиляции программы (3.28), находящейся в файле D3-2.FOR, следует из соответствующего текущего подкаталога ввести команду:

```
FFLL D3-2                                        (3.32)
```

При запуске выполняемого файла (с расширением .EXE) во всех случаях следует учитывать рекомендации п. 7.6.3 [17, с. 89-90]:

- предварительно использовать в программах описанный там же оператор OPEN;
- применять перенаправление потоков ввода-вывода (см. п. 2.2.3 [17, с. 26])).

В последнем случае нужно указать посредством использования в команде знака “меньше” (<) соответствующее имя файла, из которого следует брать исходные данные. Для указания того, куда помещать результаты расчета, используется знак “больше” (>).

Исходные данные в нашем случае задаются в самой программе, поэтому для выполняемого файла D3-2.EXE это может иметь вид:

```
D3-2.EXE > D3-2.LIS                                    (3.33)
```

После запуска команды (3.33) на выполнение в том же текущем подкаталоге PR корневого каталога диска C будет находиться файл D3-2.LIS, в котором будут находиться результаты работы фортран-программы (3.28), приведенные ниже:

```
 СКОРОСТЬ V =  65.960
```

Сравнив их с полученными ранее на REDUCE результатами (3.24) для скорости точки, убеждаемся в правильности решения.

В заключение отметим, что в работе [9] содержится большое количество процедур, а в пособии [24] – программ, которые удобно использовать в различных задачах механики и математики. Если при этом вы встретите трудности, почувствовав недостаток теоретического багажа, то в монографии [7] наиболее подробно описаны возможности операторов и команд базовой версии REDUCE.

ГЛАВА 4. ОСНОВНЫЕ ВОЗМОЖНОСТИ CAB REDUCE

После быстрого старта в первой главе и набора минимально необходимых сведений для практической работы с системой REDUCE, во 2-й и 3-й главе было подробно изучено все самое простое и лучшее, что она позволяет достичь (для технического вуза на примерах теоретической механики).

Рассмотрим теперь в более систематическом изложении основные возможности CAB REDUCE. Они содержат массу полезных справочных сведений как для тех, кто впервые знакомится с системой REDUCE, так и для продолжения изучения работы с ней. Они будут полезны и нужны всем, кто столкнулся с проблемой автоматизации вычислений в любой области знания. Если у вас возникнет желание узнать больше о возможностях системы REDUCE, то кроме собственного опыта, полезно будет обратиться также к дополнительной литературе, список которой приведен в конце книги.

4.1. Управление формой представления результатов расчета

4.1.1. Команды ON, OFF и флаги

Для управления формой представления результатов расчета используются команды–флаги (переключатели), которые идентифицируются именами. Их можно **поднимать (включать) командой ON** и ***опускать (выключать) командой OFF:*** ALLFAC, MCD, EXP, PRI, NAT, … (в исходном состоянии в положении ON), DIV, GCD, LIST, FLOAT, BIGFLOAT, NUMVAL, …. (в исходном состоянии в положении OFF).

Состояние флагов изменяется командами ON или OFF, формат которых имеет вид

```
ON F1, …, FN;                                    (4.1)
OFF F1, …, FN;                                   (4.2)
```

где F1, …, FN – список имен поднимаемых (4.1) или опускаемых (4.2) флагов. Для поднятия флага необходимо, чтобы его имя фигурировало в списке команды ON, а для опускания — команды OFF.

В начале работы системы REDUCE по умолчанию состояние флагов таково, что выполняются следующие преобразования:

- из выражения выносится за скобки общий числовой множитель, что выполняется всегда при любом состоянии флагов;
- также выносится за скобки общий "простой" символьный множитель, т.е. множитель, состоящий из произведений свободных переменных и операторов (ON ALLFAC$);
- производится возведение в целочисленную степень, все скобки раскрываются и приводятся подобные члены (ON EXP$);
- все слагаемые приводятся к общему знаменателю, который записывается после числителя через знак деления / (ON MCD$).

Рассмотрим управление режимами работы REDUCE 3.3 на примере сеанса работы в этой системе (см. п. 1.1).

В других ее версиях исходное состояние некоторых флагов может незначительно отличаться, поэтому форма приводимых ниже выражений может быть там иной.

Флаг ALLFAC *в исходном состоянии поднят*. При этом из всего выражения или из любых выражений, заключенных в скобки, выносится общий символьный множитель (числовой выносится всегда).

Если флаг ALLFAC опустить командой (4.3) то присваиваемое переменной **q** выражение (4.4) будет распечатано в том же виде (4.5) без всяких преобразований.

1: **off allfac;** (4.3)

2: **q:= x**2*y**2 – x*y;** (4.4)

$q := x^2*y^2 - x*y$ (4.5)

После возвращения системы в исходное состояние по команде (4.6), оно будет распечатано по команде (4.7) в виде (4.8) с вынесенным за скобки общим символьным множителем.

3: **on allfac;** (4.6)

4: **q;** (4.7)

x*y*(x*y – 1) (4.8)

Флаг MCD *в исходном состоянии поднят*. В таком состоянии системы алгебраические выражения приводятся к общему знаменателю.

Если флаг MCD опустить командой (4.9), то присваиваемое переменной **qq** выражение (4.10) будет распечатано в том же виде (4.11) без всяких

преобразований (только вместо наклонной черты для записи выражения в знаменателе будет использована степенная форма).

5: **off mcd;** (4.9)

6: **qq:= 1/x + 1/y;** (4.10)

$qq:= x^{-1} + y^{-1}$ (4.11)

После возвращения системы в исходное состояние по команде (4.12), выражение (4.11) будет распечатано по команде (4.13) в виде (4.14), приведенным к общему знаменателю.

7: **on mcd;** (4.12)

8: **qq;** (4.13)

$$\frac{x + y}{x * y} \quad (4.14)$$

Замечание. Опуская флаг MCD, мы запрещаем приведение выражения к общему знаменателю. Однако запрет распространяется лишь на проведение соответствующих преобразований. Если в некотором выражении слагаемые уже приведены к общему знаменателю, то ***обратного разбиения на отдельные слагаемые с отдельными символьными знаменателями обычно не производится.*** Исключение составляют лишь некоторые простейшие случаи (например, случай численного знаменателя).

Дополнение 4.1. Необходимо также отметить, что действие ряда флагов (ALLFAC, MCD, EXP) не всегда приводит к ожидаемому результату преобразования выражений. Например, существуют такие состояния системы, когда опускание флага EXP не блокирует раскрытия скобок, поднятие флага ALLFAC не ведет к выносу общего знаменателя и т. п. К ошибкам расчета это не приводит, т.к. ***значение выражения в памяти ПК не зависит от формы его представления при печати результатов.***

Часто *разница в формах представления переменных при печати результатов вызвана различием форм, в которых они хранятся в памяти ПК.* Это происходит в зависимости от того состояния флагов EXP и MCD, при которых выполнялось присваивание переменной соответствующего значения. Поясним сказанное на примере флага MCD.

Присвоим при опущенном флаге MCD (4.15) значение переменной **xy** (4.16) и распечатаем его (4.17):

9: **off mcd;** (4.15)

10: **xy:= a/(b+c) + d/(e+f);** (4.16)

$$xy := (b + c)^{-1} * a + (e + f)^{-1} * d \quad (4.17)$$

После этого поднимем флаг MCD (4.18), вернув систему в исходное состояние, и присвоим переменной **xx** (4.19) значение **xy** (4.16), также распечатав его (4.20):

11: **on mcd;** (4.18)

12: **xx:= xy;** (4.19)

$$xx := \frac{a*e + a*f + b*d + c*d}{b*e + b*f + c*e + c*f} \quad (4.20)$$

Теперь опять опустим флаг MCD (4.21), распечатаем значения равных друг другу переменных **xy** (4.22) и **xx** (4.24) при одном и том же состоянии флага MCD и вернем систему в исходное состояние (4.26):

13: **off mcd;** (4.21)

14: **xy;** (4.22)

$$(b + c)^{-1} * a + (e + f)^{-1} * d \quad (4.23)$$

15: **xx;** (4.24)

$$(a*e + a*f + b*d + c*d)/(b*e + b*f + c*e + c*f) \quad (4.25)$$

16: **on mcd;** (4.26)

Теперь постараемся разобраться в полученном неожиданном результате. Ведь мы получили ***разные формы представления (4.23) и (4.25) равных друг другу переменных xy и xx (4.19) при одном и том же состоянии флага MCD.***

Значение переменной **xy** *(4.16)* присваивается при опущенном флаге MCD. Поэтому оно *сохраняется в памяти ПК не приведенным к общему знаменателю.*

Переменной **xx** *(4.19)* значение **xy** (4.16) присваивается при поднятом флаге MCD. Поэтому оно *сохраняется в памяти ПК в форме с общим знаменателем.*

Печать разных форм представления (4.23) и (4.25) *равных друг другу переменных* **xy** *и* **xx** (4.19) происходит при опущенном состоянии флага MCD. Она *иллюстрирует различия в обработке значений, еще не приведенных к общему знаменателю (4.17) и заранее приведенных (4.20).*

Выражение (4.16), сохраняемое в памяти ПК не приведенным к общему знаменателю (4.17), распечатывается при опущенном состоянии флага MCD в той же форме (4.23) (также *не приведенным к общему знаменателю*, причем выражения (4.17) и (4.23) полностью одинаковы по форме).

Опускание флага MCD также не влияет на форму представления того же выражения (4.19), хранящегося в памяти ПК, приведенным к общему знаменателю (4.20). Опускание флага MCD не приводит к обратному преобразованию и оно распечатывается также *приведенным к общему знаменателю* (4.25), только в одну строку и с использованием наклонной черты в качестве знаменателя.

Флаг DIV сокращает при выводе общие простые множители в числителе и знаменателе сложных выражений. При этом могут появиться отрицательные степени и рациональные дроби, но само выражение упрощается.

Флаг DIV *в исходном состоянии опущен.* Поэтому присваиваемое переменной **tt** выражение (4.27) будет распечатано в форме (4.28) без сокращения простых сомножителей.

```
17: tt:=2*y+((x**2+2*x*y+y**2)*(y**2+z**2))/
    (z*x**2+z*2*x*y+ z*y**2);                    (4.27)
```

$$tt:=(x^2*y^2+2*x^2*y*z+x^2*z^2+2*x*y^3+4*x*y^2*z +2*x*y*z^2+y^4+2*y^3*z+y^2*z^2)/(z*(x^2+2*x*y+y^2)) \quad (4.28)$$

Обратим ваше внимание, что в исходном состоянии системы REDUCE подняты флаги MCD и ALLFAC. Поэтому в распечатанном виде (4.28) есть следующие отличия от формы заданного выражения в (4.27):

- оно приведено к общему знаменателю (действие флага MCD);
- из него вынесен общий символьный множитель (**z** в знаменателе – действие флага ALLFAC);
- все скобки в алгебраических выражениях в числителе раскрыты, что обусловлено действием флага EXP, который также поднят в исходном состоянии системы (см. ниже).

Как видно из сравнения выражений (4.27) и (4.28), на представление результатов оказывают действие все флаги, находящиеся в исходном состоянии системы в состоянии ON или поднятые в процессе работы этой же командой.

Действие каждого флага будет распространяться до тех пор, пока команда OFF не изменит его состояние.

Если поднять флаг DIV командой (4.29) и снова распечатать выражение **tt**, то при выводе будут сокращены простые множители в числителе и знаменателе и оно примет вид (4.30).

```
18: on div;                                      (4.29)
```

19: **tt;**

$$y^2 * z^{-1} + 2 * y + z \qquad (4.30)$$

Флаг LIST *в исходном положении опущен*. Если его поднять командой (4.29) и снова распечатать выражение **tt**, то каждое слагаемое выражения (4.30) будет в (4.32) печататься на отдельной строке.

20: **on list;** (4.31)

21: **tt;**

$$y^2 * z^{-1}$$
$$+2 * y \qquad (4.32)$$
$$+z$$

Поэтому на практике флаг LIST поднимают лишь перед печатью очень громоздких выражений. Опустим его и приведем систему в исходное состояние, опустив также поднятый ранее флаг DIV.

22: **off list, div;** (4.33)

Для проверки можете распечатать выражение для переменной **tt**, которая опять в результате будет выведена в форме (4.28).

Обратите также внимание, что одна команда (4.33) (ON или OFF) может устанавливать или отменять сразу несколько флагов.

Флаг RAT *в исходном положении опущен*. Он используется вместе с командой FACTOR, которая группирует коэффициенты при степенях тех свободных переменных, которые указаны в списке параметров этой команды (команды FACTOR и отменяющая ее REMFAC описаны в п. 4.1.2).

Если выполняется команда FACTOR (4.34) и поднят флаг RAT (4.35), то алгебраическое выражение при печати раскладывается на сумму алгебраических дробей (4.36). При этом могут появляться отрицательные степени:

23: **factor z;** (4.34)

24: **on rat;** (4.35)

25: **tt;**

$$z + 2 * y + z^{-1} * y^2 \qquad (4.36)$$

Опять приведем систему в исходное состояние.

26: **off rat;**

27: **remfac z;**

Флаг EXP *в исходном состоянии поднят.* В таком состоянии системы все скобки в алгебраических выражениях раскрываются (см. (4.28)). Можно убрать операцию раскрытия степеней от полиномиальных выражений, опустив флаг EXP командой (4.37). В результате значение переменной **tt** (4.28) предстанет в форме (4.38) без раскрытия скобок, после чего опять приведем систему в исходное состояние командой (4.39).

28: **off exp;** (4.37)

29: **tt;**

$$\frac{(x+2*y)*(y^2+2*y*z+z^2)*x+y^4+2*y^3*z+y^2*z^2}{(x^2+2*x*y+y^2)*z} \qquad (4.38)$$

30: **on exp;** (4.39)

Флаг GCD *в исходном положении опущен.* Если он поднят, то система выделяет общий наибольший множитель в числителе и знаменателе и производит сокращение на него. Но и при опущенном флаге GCD система сокращает числитель и знаменатель на множители сравнительно простого вида. *Поднятие флага GCD приводит к сокращению числителя и знаменателя на сложный общий множитель.* На выполнение алгоритма сокращения при поднятом флаге GCD (4.40) требуется сравнительно много времени, поэтому им следует пользоваться в отдельных частях программы для упрощения алгебраических выражений (4.41), после чего его нужно опускать (4.42).

31: **on gcd;** (4.40)

32: **tt;**

$$\frac{y^2+2*y*z+z^2}{z} \qquad (4.41)$$

33: **off gcd;** (4.42)

Как видим, произошло сокращение числителя и знаменателя на общий множитель *(x+y)**2*. В результате выражение для **tt** (4.28) предстало после его распечатывания в значительно более простой форме (4.41).

Обратим также ваше внимание, что флаг GCD ни в коем случае нельзя поднимать при опущенном флаге MCD. Такой режим нарушает нормальную работу системы и может привести к ошибкам расчета.

Флаг NAT *в исходном положении поднят.* Он обеспечивает представление выражений в "естественном виде" с индексной формой для показателей степеней (типа (4.38), (4.41)). Однако такое представление выражений,

наряду с большой его наглядностью, обладает рядом существенных недостатков, особенно резко нарастающих с увеличением громоздкости выражений. Самым крупным из них является тот, что выражение в такой форме несовместимо с возможностью его дальнейшего использования:

- путем передачи через буфер обмена в другие windows-приложения (Word, Derive);
- для записи во внешний файл с целью его дальнейшего использования в качестве вводного файла.

В обоих случаях "естественный" способ печати несовместим с синтаксисом вводимых выражений в системе REDUCE или других windows-приложениях, что делает невозможным его дальнейшую машинную обработку. Это очень крупный недостаток, т.к. выражения очень быстро становятся совсем непригодными для обычной ручной обработки из-за своего огромного размера (см., например, (3.22) и (3.23)).

Для отмены "естественного" способа печати необходимо опустить флаг NAT:

34: **off nat;** (4.43)

После этого выражения не становятся существенно менее громоздкими, однако они теперь становятся пригодными для дальнейшей машинной обработки, важность чего невозможно переоценить. При этом у них в конце выражения в качестве ограничителя появится знак "$", а все выражение примет однострочную форму записи (4.44):

35: **tt;**

```
(x**2*y**2 + 2*x**2*y*z + x**2*z**2 + 2*x*y**3 +
4*x*y**2*z + 2*x*y*z**2 + y**4 + 2*y**3*z +
y**2*z**2)/(z*(x**2 + 2*x*y + y**2))$
```
(4.44)

Снова приведем систему в исходное состояние, подняв флаг NAT (4.45):

36: **on nat;** (4.45)

Флаг FORT *в исходном положении опущен.* Он используется при своем поднятии (4.46) для представления выводимых результатов (4.47) в формате языка Фортран (4.48):

37: **on fort;** (4.46)

38: **tt;** (4.47)

```
 ans=(x**2*y**2+2.*x**2*y*z+x**2*z**2+2.*x*y**3+4.*x*y
. **2*z+2.*x*y*z**2+y**4+2.*y**3*z+y**2*z**2)/(z*(x**2
. +2.*x*y+y**2))
```
(4.48)

При этом выводимое выражение :

- начинается в седьмом столбце;
- если оно превышает одну строку, то на следующей появится признак продолжения – точка в шестом столбце, сопровождаемая пробелом;
- если распечатываемое выражение есть результат присвоения некоторой переменной определенного значения, то эта переменная печатается как имя выражения (в его левой части);
- если распечатывается выражение, которому уже ранее было присвоено некоторое значение, то оно автоматически приобретает имя ANS (см. (4.48));
- если идентификатор или число выходят за границы, допускаемые синтаксисом языка Фортран, то возникает ошибка.

Такое представление результатов расчета оказывается очень удобным, особенно для сложных выражений (см., например, (3.22) и (3.23)):

- для записи во внешний файл с целью дальнейшего численного расчета сложного алгебраического выражения с использованием языка Фортран (см. дополнение 3.3);
- для передачи через буфер обмена в другие windows-приложения (Word, Derive) (см. дополнение 3.2 об изменениях режимов вывода в синтаксисе Фортрана).

Для отмены такой форматированной печати необходимо опустить флаг FORT:

39: **off fort;** (4.49)

Флаг PRI *в исходном положении поднят.* Он определяет удобную для чтения печать. В этом его положении рассмотрено действие всех вышеописанных флагов.

Следует, однако, отметить, что преобразование больших алгебраических выражений с целью получения измененных форматов вывода требуют определенного времени и объема памяти. Если требуется ускорить печать, то необходимо опустить флаг PRI, используя команду OFF (4.50). После этого вывод (4.51) будет производиться в одном фиксированном формате (4.52), который в основном отражает внутреннее представления выражения:

40: **off pri;** (4.50)

41: **tt;** (4.51)

$((y^2 + 2*z*y + z^2)*x^2 + (2*y^3 + 4*z*y^2 + 2*z^2*y)*x + y^4 + 2*z*y^3 + z^2*y^2) / (z*x^2 + 2*z*y*x + z*y^2)$ (4.52)

При опущенном флаге PRI будут работать флаги, изменяющие только *форму представления* этого фиксированного формата: LIST, GCD, NAT, FORT, что мы рекомендуем проверить самостоятельно.

Опять приведем систему в исходное состояние, подняв флаг PRI (4.53):

42: **on pri;** (4.53)

Флаг FLOAT *в исходном положении опущен.* В таком состоянии системы, которое является обычным для проведения алгебраических преобразований, если в процессе вычисления встречается действительное (с плавающей точкой) число, то система обычно преобразует его в отношение двух целых чисел и выдает сообщение о проделанном преобразовании.

При желании использовать действительную арифметику, пользователь может воспрепятствовать этому преобразованию поднятием флага FLOAT (ON FLOAT;). Он запрещает преобразовывать число с плавающей точкой в рациональную дробь во время вычислений (см., например, команды 15 и 90 программы 2.1 и их описания).

Флаг NERO *в исходном положении опущен.* Его поднимают тогда, когда чтение распечатки затруднено большим количеством нулевых значений элементов. Это часто бывает при решении систем линейных алгебраических уравнений задач статики, которые являются достаточно разреженными. Печать таких выражений может быть отменена, если установить флаг NERO.

Так, например, команды 15 программ 2.1–2.6 запрещают печать нулевых значений при контрольной распечатке матриц A и B, что достигается включением флага NERO. Команда 80 программ 2.1–2.6 разрешает печать нулевых значений перед решением СЛАУ, что достигается выключением флага NERO. Это сделано для возможности распечатки всех значений матрицы-столбца X.

Флаг BIGFLOAT *в исходном положении опущен.* Однако в системе имеется возможность проводить вычисления с вещественными числами, точность представления которых больше, чем при установленном флаге FLOAT. Поднятие флага BIGFLOAT обеспечивает использование в многочленах вещественных коэффициентов повышенной точности (по умолчанию в этом режиме точность вещественных чисел определяется десятью десятичными числами). Для задания нужной точности представления чисел можно воспользоваться командой PRECISION, в которой в качестве параметра указывается нужное число десятичных чисел. Например, для определения точности вещественных чисел двадцатью десятичными числами следует задать команду PRECISION 20$.

Особенно часто флаг BIGFLOAT используется совместно с флагом NUMVAL. Этот флаг устанавливает режим вычисления значений элементарных функций. Функции SIN, COS, TAN, ASIN, ACOS, SQRT, EXP, LOG с числовым аргументом и зарезервированные переменные E и PI принимают численные значения в форме с плавающей запятой с текущей степенью точности.

Система REDUCE предназначена для выполнения точных алгебраических преобразований. Поэтому использование приближенных вещественных чисел при сложных вычислениях может привести к ошибкам. Однако использование действительной арифметики удобнее для представления результатов решения в численной форме.

Поэтому все расчеты обычно проводят с использованием рациональных чисел, а только результаты решения представляют в приближенном виде с использованием действительных чисел. Для этого перед печатью результатов устанавливают флаги BIGFLOAT и NUMVAL, после чего систему опять возвращают в исходное состояние (см. дополнение 2.4).

Напомним, что вышеописанный результат достигается только при совместном действии флагов BIGFLOAT и NUMVAL, причем вместо флага BIGFLOAT *не может использоваться* флаг FLOAT.

В заключение отметим, что ***все изменения формы представления результатов расчета путем использования любых флагов не меняют его значения***.

Оно может быть изменено только присваиванием нового значения этой переменной или командами SAVEAS, CLEAR и LET.

4.1.2. Управление порядком имен в выражениях. Команды ORDER, FACTOR и REMFAC

В системе REDUCE вычисления и преобразования производятся в соответствии с правилами алгебры, а выражения представляются в определенной форме. Например, в примере (4.41) приведены подобные члены, числовые коэффициенты вынесены на первое место, а переменные упорядочены специальным образом:

- имена, состоящие из одной латинской буквы, расставлены в алфавитном порядке;
- более сложные — в зависимости от их длины и последовательности, в которой они впервые были использованы в программе.

Часто возникает потребность установить другое расположение переменных. ***Управление порядком имен в выражениях*** осуществляется командами ORDER, FACTOR и REMFAC.

Команда ORDER предназначена для получения нужного порядка переменных при выводе на печать различных выражений.

Формат использования команды:

```
ORDER P1, …, PN;                                    (4.54)
```

Списком ее параметров служит расположенный в требуемом порядке список переменных P1, …, PN, которые следует печатать среди прочих.

Если в произведении или сумме встречаются несколько имен из этого списка, то они расставляются в той же последовательности, что и в списке. Такой порядок соблюдается во всех распечатываемых далее значениях.

Поэтому для перестройки выражения для **tt** (4.28) не по алфавитному порядку **x**, **y**, **z**, а по **z**, **y**, **x** нужно ввести команду (4.55). Теперь распечатав выражение для **tt** командой (4.56) мы получим нужную форму его представления (4.57).

43: **order z,y;** (4.55)

44: **tt;** (4.56)

$$(z^2*y^2+2*z^2*y*x+z^2*x^2+2*z*y^3+4*z*y^2*x+2*z*y*x^2+y^4+ 2*y^3*x+y^2*x^2)\ /\ (z*(y^2+2*y*x+x^2)) \quad (4.57)$$

Команда (4.55) устанавливает такой порядок вывода в (4.57), что **z** стоит впереди **y**, а обе они стоят впереди всех остальных переменных, не указанных в команде ORDER.

Для возвращения к исходному порядку переменных по умолчанию используется команда ORDER NIL.

45: **order nil;** (4.58)

После ее ввода, опять распечатав выражение для **tt** командой (4.56), мы получим исходную форму его представления (4.28).

Дополнение 4.2. Отметим, что порядок переменных может быть в дальнейшем изменен с помощью дополнительных объявлений ORDER. Однако тогда все "переупорядоченные" переменные будут иметь младший порядок по сравнению с тем, что был объявлен в ранних объявлениях ORDER для этих переменных.

Поэтому вам придется в результате быть очень внимательным, чтобы путем использования дополнительных объявлений ORDER действитель-

но иметь тот порядок расположения переменных, который бы вы хотели получить.

Например, команды

46: **aa:= a + b + c + d;** (4.59)

aa:= a + b + c + d (4.60)

47: **order d, c, b, a;** (4.61)

48**: aa;** (4.62)

d + c + b + a (4.63)

49: **order d, c;** (4.64)

приведут к тому, что **b** и **a** будут стоять впереди **d** и **c**:

50: **aa;** (4.65)

b + a + d + c (4.66)

Последующие же использования этой команды запутают вас окончательно.

Потому проще и надежнее после первоначального объявления команды ORDER вернуть систему в исходное состояние (порядка переменных, принятого по умолчанию) командой (4.58). В зависимости от реализации системы REDUCE порядок, принятый по умолчанию, может быть разным, но, как правило, алфавитным.

Теперь, после ввода команды (4.58), опять первоначальным объявлением команды ORDER, действующей на систему в исходном состоянии, следует установить нужный вам порядок переменных. Таким образом следует поступать каждый раз перед использованием команды ORDER.

Команда FACTOR объявляет выражения в качестве коэффициентов-сомножителей при выдаче на печать.

Аргументом этой команды является список идентификаторов или выражений.

Все слагаемые, содержащие постоянные степени объявленных выражений, печатаются в виде произведения целочисленной степени объявленных выражений и суммы оставшихся членов.

51: **factor z;** (4.67)

52: **tt;** (4.68)

$$(z^2 * (x^2 + 2 * x * y + y^2) + 2 * z * y * (x^2 + 2 * x * y + y^2) + y^2 * (x^2 + 2 * x * y + y^2)) / (z * (x^2 + 2 * x * y + y^2)) \quad (4.69)$$

Команда FACTOR часто используется при поднятом флаге RAT. В этом случае алгебраическое выражение при печати раскладывается на сумму алгебраических дробей (см. (4.34)–(4.36)).

Команда REMFAC используется для отмены действия команды FACTOR. Аргументом этой команды является тот же список идентификаторов или выражений, который был указан ранее в команде FACTOR.

53: **remfac z;** (4.70)

После ее ввода, опять распечатав выражение для **tt** командой (4.68), мы получим исходную форму его представления (4.28).

4.1.3. Краткое описание режимов работы REDUCE

В заключение приведем достаточно полный список флагов с краткими пояснениями. Для удобства ориентировки они расположены в алфавитном порядке.

В скобках указано положение, в котором флаг ***находится при исходном состоянии системы:***

- (ON) – поднят,
- (OFF) – опущен.

ALLFAC (ON) – обеспечивает поиск общих множителей выражения и выносит их за скобки при выводе выражений.

BIGFLOAT (OFF) – обеспечивает использование в многочленах вещественных коэффициентов произвольной точности.

DEFN (OFF) – дает указание системе вывести лисп-эквивалент входных выражений, не проводя вычислений.

DEMO (OFF) – обеспечивает паузу после каждой команды в файле до тех пор, пока не будет выдана команда “ввод”.

DIV (OFF) – сокращает простые множители при выводе, так что могут появиться отрицательные степени и рациональные дроби.

ECHO (OFF) – определяет вывод на печать входной информации или ее отсутствие.

Он автоматически поднимается на время чтения из файла при использовании после команды IN ограничителя “;” и опускается, если используется ограничитель “$”.

По окончании ввода из указанного в команде IN файла он возвращается в то состояние, в котором он был перед вводом.

В исходном состоянии флаг ECHO:

- в пакетном режиме (при вводе из файла) *поднят*;
- в диалоговом – *опущен* (нет необходимости повторять на дисплее только что набранный текст).

Даже если в команде IN в качестве ограничителя используется точка с запятой, повторение всего содержимого файла ввода или его части можно предотвратить, опустив флаг ECHO в файле ввода.

EXP (ON) – обеспечивает раскрытие выражений при их вычислениях.

FACTOR (OFF) – в положении ON обеспечивает разложение выражения на множители с целочисленными коэффициентами.

FAILHARD (OFF) – в положении ON приводит к прерыванию работы алгоритма с сообщением об ошибке, если интеграл не может быть вычислен в замкнутом виде.

В положении OFF в этом случае выдается выражение интеграла в формальном виде.

FLOAT (OFF) – допускает в преобразованиях числа с плавающей точкой.

FORT (OFF) – обеспечивает вывод выражений в форме Фортрана.

GCD (OFF) – обеспечивает сокращение наибольших общих делителей в рациональных выражениях.

INT (OFF) – в положении ON устанавливает интерактивный (диалоговый) режим работы.

Если входные данные поступают из внешнего файла, то система обрабатывает их в пакетном режиме. При желании в этом случае установить диалоговый режим работы, в файле следует поднять флаг INT. Если не требуется продолжать диалог с системой, то флаг нужно опустить.

LOM (OFF) – в положении ON вычисляет наименьшие общие кратные знаменателей при сложении рациональных выражений.

LIST (OFF) – обеспечивает печать каждого члена на новой строке.

MCD (ON) – обеспечивает объединение знаменателей при сложении выражений.

MSG (ON) – в положении OFF подавляет печать предупреждающих сообщений. Сообщения об ошибках продолжают печататься.

NAT (ON) – обеспечивает «естественный» способ вывода.

NERO (OFF) – запрещает печать нулевых значений элементов.

NOLNR (OFF) – в положении ON позволяет использовать в алгоритме линейные свойства интегрирования в тех случаях, когда интеграл не может быть взят в замкнутом виде.

NUMVAL (OFF) – устанавливает режим вычисления значений элементарных функций. Функции SIN, COS, TAN, ASIN, ACOS, SQRT, EXP, LOG с числовым аргументом и зарезервированные переменные E и PI принимают численные значения в форме с плавающей запятой с текущей степенью точности.

OUTPUT (ON) – в положении OFF подавляет печать значения любого верхнеуровнего выражения.

PERIOD (ON) – обеспечивает печать точки после каждого целочисленного коэффициента при выводе на Фортране.

PRET (ON) – обеспечивает печать входных команд с синтаксисом REDUCE в стандартном формате.

PRI (ON) – определяет печать выходного выражения в фиксированном формате.

RAISE (ON) – обеспечивает преобразование при вводе строчных букв в прописные, за исключением знаков и строк, перед которыми стоит знак "!".

RAT (OFF) – используется вместе с FACTOR. Обеспечивает печать общего знаменателя в выражении с каждым выделенным подвыражением.

RATIONAL (OFF) – в положении ON позволяет использовать рациональные числа в качестве коэффициентов полинома.

RESUBS (ON) – обеспечивает после выполнения подстановки проверку полученного преобразованного выражения на возможность выполнения новой подстановки. Этот процесс продолжается, пока имеется возможность выполнить еще какую-нибудь подстановку, которую иногда делать нежелательно. В этом случае следует опустить флаг RESUBS, в результате чего система не будет просматривать выражения на возможность дальнейших подстановок.

TRFAC (OFF) – устанавливает подробную печать в алгоритме факторизации.

TRINT (OFF) – устанавливает подробную печать действий в алгоритме интегрирования.

Обратим ваше внимание, что ***при ошибке в имени флага при его поднятии или опускании, система не сообщает об этом.*** В этом случае не возникает изменения состояния того флага, которое предполагалось достичь вводимой командой.

4.2. Подстановки и преобразование выражений

4.2.1. Применение для подстановок оператора присваивания

Наиболее часто встречающимися при проведении аналитических выкладок операциями являются различные замены и подстановки. С целью обеспечения возможности подобных действий в REDUCE введен весьма развитой аппарат подстановок. Рассмотрение его будем проводить на примерах сеансов работы в системе REDUCE, которые также будем начинать сначала, как и ранее в пп. 1.1 и 4.1. В отличие от них, где один сеанс работы продолжался на протяжении всего раздела, сейчас будем обращать внимание на автоматическую нумерацию вводимых команд:

- если номер первой команды приводимого фрагмента *начинается с цифры 1*, то сеанс работы начинается и *все впервые используемые в этом фрагменте переменные* ***являются свободными***;
- нумерация *продолжается – все впервые используемые в этом фрагменте переменные* ***могут быть связанными*** и иметь значения, заданные им в течение текущего сеанса работы;
- если номер шага *не указывается* – то приводимый фрагмент является локальным: он не зависит от текущего сеанса работы и все впервые используемые в нем переменные ***являются свободными***.

В своей основе аппарат подстановок использует идею *свободных переменных* (см. п. 1.2.4), которые в любой момент могут быть заменены на связанные переменные и сложные выражения, а затем снова сделаны свободными командой CLEAR (см. (1.13)). Поэтому при выполнении подстановок сначала выясняется, являются ли переменные связанными или нет.

Если в правой части оператора присваивания стоит *свободная переменная*, то в замене *используется ее имя*:

```
1: x := u;                                   (4.71)
x := u                                       (4.72)
```

Если в правой части оператора присваивания стоит *связанная переменная*, то в замене *используется ее значение*:

```
2: y := x;                                   (4.73)
y := u                                       (4.74)
```

Оператор (4.73) присваивает Y (левой части) то значение переменной X (правой части), которое она имеет в данном месте программы.

Изменим теперь значение переменной X оператором (4.75), а затем распечатаем командой (4.77) значение переменной Y (4.78):

3: **x := p + q;** (4.75)

x := p + q (4.76)

4: **y;** (4.77)

u (4.78)

После выполнения операторов (4.71) и (4.73) значения X (4.72) и Y (4.74) равны U. Изменение значения X (4.75) на P+Q (4.76) не отражается на величине Y (4.78): оно остается равным ранее вычисленному значению U (4.74).

Таким образом, при изменении одной из двух равных величин значение второй не изменяется. Это позволяет заключить, что

- в системе REDUCE распечатываются только вычисленные значения;
- оператор присваивания имеет ***локальный*** характер и все последующие изменения значения переменной X не будут отражаться на значении Y, хотя оператор (4.73) установил равенство их значений.

Поэтому для присваивания одинаковых значений переменным X и Y следует, например, опять использовать оператор присваивания (4.79). При его выполнении произойдет новое вычисление переменной Y, которое будет равно значению переменной X (4.75). Поэтому распечатано будет это новое вычисленное значение Y (4.80):

5: **y := x;** (4.79)

y := p + q (4.80)

По одному и тому же оператору Y:=X величине Y присваиваются разные значения (U и P+Q), которые связанная переменная X имеет в данном месте сеанса работы (или программы).

Теперь ***повторим заново*** сеанс работы из операторов (4.71) – (4.78) и изменим не величину переменной X, а ее значение U. При этом ***полностью повторяющиеся операторы в разных сеансах работы всегда будем обозначать штрихом над соответствующим номером в скобках, измененные и новые – буквой после номера*** (в нижеследующем примере i):

1: **x := u;** (4.71')

x := u (4.72')

2: **y := x;** (4.73')

y := u (4.74')

Изменим теперь значение переменной U оператором (4.75i), а затем распечатаем командами (4.77') и (4.79i) значения переменных Y (4.78i) и X (4.80i):

3: **u := p + q;** (4.75i)

u := p + q (4.76i)

4: **y;** (4.77')

p + q (4.78i)

5: **x;** (4.79i)

p + q (4.80i)

После выполнения операторов (4.71') и (4.73') значения X (4.72') и Y (4.74') так же, как и ранее, равны U. Оператор (4.75i) заменяет теперь не величину переменной X (4.75) на P+Q, а ее значение U, которое имеют обе переменные X и Y. Теперь новое значение U (4.76i) также имеют переменные Y (4.78i) и X (4.80i), что и выясняется после их распечатывания командами (4.77') и (4.79i).

Таким образом, при изменении некоторого значения, которое имеют обе равные переменные, величины последних также одинаково изменяются.

Аппарат подстановок в системе REDUCE позволяет в любой момент ***заменять свободные переменные на сложные выражения.*** Заменим свободную переменную Q на сложное выражение:

6: **q := (r + s)**2;** (4.81)

$q := r^2 + 2*r*s + s^2$ (4.82)

Распечатав затем величины U, X и Y командой (4.83) увидим, что их значения (4.84) – (4.86) представлены с учетом новой величины Q (4.82):

7: **u; x; y;** (4.83)

$p + r^2 + 2*r*s + s^2$ (4.84)

$p + r^2 + 2*r*s + s^2$ (4.85)

$p + r^2 + 2*r*s + s^2$ (4.86)

Если бы мы повторили заново *второй сеанс работы* и после выполнения операторов (4.71') – (4.76i) ввели оператор (4.81) и команду (4.83), то результаты для U (4.84), X (4.85) и Y (4.86) также бы не изменились. Величины X и Y также оказались бы равными, так как подстановка происходила бы для значения U (4.75i), которому равны обе переменные X и Y.

Однако. если бы мы повторили заново *первый сеанс работы* и после выполнения операторов (4.71) – (4.76) ввели оператор (4.81) и команду (4.87), то получили бы иной результат для значения Y (4.89):

5: **x; y;** (4.87)

$p + r^2 + 2*r*s + s^2$ (4.88)

u (4.89)

Здесь изменение значения X (4.75) на P+Q (4.76), как и в *первом сеансе работы,* не отражается на величине Y (4.89): оно остается равным ранее вычисленному значению U (4.74). Поэтому новая величина Q (4.81) подставляется и изменяет значение только X (4.88), которое от него зависит.

Мы рекомендуем заново выполнить оба сеанса работы. Разобраться в них и получить самостоятельно вышеописанные результаты при использовании подстановки по оператору (4.81) или любому сложному выражению.

4.2.2. Понятие ядра подстановки

В качестве свободных переменных могут также рассматриваться и более сложные выражения определенного вида, называемые ядрами. Их использование определяет дальнейшие возможности аппарата подстановок.

Ядром *может быть выражение, удовлетворяющее следующим требованиям:*

- оно должно представлять собой произведение степеней имен переменных и операторов со скобками;
- в аргументе ядра оператора может стоять лишь один знак сложения;
- знак "минус" может стоять лишь перед началом аргумента оператора;
- в ядре не допускается числовой множитель;
- знак деления может быть использован лишь в аргументе оператора. Он также может присутствовать в показателях степеней в ядрах (дробные степени), но при этом соответствующие подстановки выполняются только в простейших случаях: замена происходит лишь в случае точного совпадения значений степеней в ядре и заменяемом выражении.

Поэтому следующие выражения *могут быть ядрами подстановок:*

```
X, X**3*Y*Z**2, P(-X,Y), COS(A)**2, SIN(X+Y),
X*Y(X+Z)**K, Y**3*U(X*Y+Z*P)**(Z*P), V(-X)**L,
```

а нижеследующие — *не могут:*

```
X+Y, -X, X/Y, 3*X, 3*P(A+B), X**(-A), A-B.
```

Ядру может быть присвоено некоторое значение точно так же, как и обычной переменной или оператору:

7: **r*s := z$** (4.90)

После этого в вычислительный процесс включается поиск в вычисляемом значении соответствующего ядра с его последующей заменой.

Это производится в самом конце вычисления значения, т. е. когда все остальные операции (раскрытие скобок, приведение подобных и т. п.) уже выполнены.

Поэтому, распечатав, например, величину X командой (4.91) увидим, что ее значение (4.85) представлено теперь в форме (4.92) с заменой ядра R*S (4.90) его значением Z:

8: **x;** (4.91)

$p + r^2 + s^2 + 2*z$ (4.92)

Ядрам, как и простым переменным, в любой момент могут быть присвоены новые значения. Они освобождаются от присвоенных значений командой CLEAR. Освободить от значения ядро R*S можно командой

9: **clear r*s;** (4.93)

Можно повторить заново сеансы работ (4.71) – (4.80) и (4.71') – (4.80i), использовав везде вместо переменной X ядро X*X или X**2 (в операторах (4.71), (4.73), (4.75), (4.79) и соответственно в (4.71'), (4.73'), (4.77i)). При этом оба набора результатов не изменятся:

- значения ядра X*X или X**2 в соответствующих местах будут также совпадать со значением переменной X;
- значения второй переменной Y также не изменятся.

В обоих случаях *ядро* X*X или X**2 *ведет себя как переменная* X, что покажем на примере сеанса работы из операторов (4.71) – (4.80). При этом измененные и отличающиеся операторы будем обозначать буквой j после номера:

1: **x * x := u;** (4.71j)

x * x := u (4.72j)

2: **y := x * x;** (4.73j)

y := u (4.74')

3: **x * x := p + q;** (4.75j)

x * x := p + q (4.76j)

4: **y;** (4.77')

u (4.78')

5: **y := x * x;** (4.79j)

y := p + q (4.80')

Результаты второго сеанса работы из операторов (4.71') – (4.80i) с использованием вместо переменной X ядра X*X или X**2 предлагаем проверить самостоятельно.

Хотя *ядро ведет себя как переменная*, однако по отношению к операции присваивания между ними есть одно важное отличие. Перед присваиванием сложному ядру (сложнее одного имени) некоторого значения производится вычисление не только правой, но и левой части: ***присваивание производится не тому ядру, что записано, а его значению.*** Поясним сказанное на следующем примере.

1: **x := v$** (4.94)

2: **y := w$** (4.95)

3: **x * y := a;** (4.96)

v*w := a (4.97)

Значение A присвоено здесь оператором (4.96) не ядру X*Y, а ядру V*W, поскольку именно V*W было значением выражения X*Y в момент присваивания, что и подтверждается его распечатанным значением (4.97). Поэтому попытки освободить его с помощью команды CLEAR X*Y (4.98) ничего не дают: ядро V*W (4.101) сохраняет свое значение (4.102), равное A:

4: **clear x * y;** (4.98)

5: **x * y;** (4.99)

x * y (4.100)

6: **v*w;** (4.101)

a (4.102)

В параметрах команды CLEAR вычисление значений не производится. Поэтому для освобождения ядра из приведенного примера нужно использовать команду

7: clear v*w; (4.103)

Таким образом, при присваивании ядру какого-либо значения последнее присваивается вычисленному значению ядра. Это не всегда бывает удобным и может приводить к ошибкам за счет того, что вычисленное значение ядра может отличаться от его записанной формы.

Поэтому для эффективной работы с аппаратом подстановок система REDUCE имеет возможность присваивать значение непосредственно записанному ядру, а не его значению.

Это осуществляется при помощи специальных подстановок, использующих команды LET и SUB.

4.2.3. Команда LET

Одним из важных видов подстановок в системе REDUCE является присвоение с помощью команды LET. В нем используется обычный знак равенства (без двоеточия). Такую конструкцию мы будем называть ниже равенством.

Переменная, которой присваивается значение, записывается слева от знака равенства, а присваиваемое выражение — справа. Таким образом, параметром команды LET является определенное правило подстановки.

Формат команды LET имеет вид:

```
LET переменная=выражение;                                    (4.104)
```

или

```
LET переменная1=выражение1, …, переменнаяN=выражениеN;
```

В момент присвоения по команде LET не происходит вычисления ни левой, ни правой частей равенства, т. е. присвоение производится в соответствии с тем, как записаны выражения после слова LET. Замены с использованием команды LET носят **глобальный** характер: Они выполняются в любом месте программы.

Повторим заново сеанс работы (4.71) – (4.78), использовав вместо оператора (4.73) соответствующее LET–присвоение. При этом измененные и отличающиеся операторы будем обозначать буквой p после соответствующего номера в скобках:

1: **let x = u;** (4.71p)

2: **let y = x;** (4.73p)

Поскольку LET – команда, то вывода на печать после выполнения шагов 1 и 2, естественно, не последует. Однако система помнит установленные соотношения и ***в любом месте программы использует их.***

Одновременно с этим правила простой команды LET находятся на том же самом логическом уровне, что и оператор присваивания ":=", который поэтому отменяет действие команды LET, введенной ранее (и наоборот).

Таким образом, изменить значение переменной X можно также оператором (4.75') на P + Q (4.76'). Однако теперь сразу автоматически изменится и значение Y, став равным тому же значению P + Q (4.78p).

3: **x := p + q;** (4.75')

x := p + q (4.76')

4: **y;** (4.77')

p + q (4.78p)

Это происходит потому, что ранее командой (4.73р) установлено равенство X и Y с использованием команды LET, которое носит ***глобальный*** характер. Теперь при оценивании любого выражения, содержащего Y в любом месте программы или сеанса работы, вместо Y будет осуществляться подстановка текущего значения переменной X. В этом основное отличие LET-присвоения от оператора присваивания, имеющего ***локальный*** характер (см. сеанс работы из операторов (4.71)– (4.78)). Этим и вызвано отличие результата работы для Y (4.78) и (4.78р).

Отметим, что в левой части равенства вместо переменной может быть ядро или оператор, а в правой части вместо выражения — переменная или число. Команда (4.105)

let u * v = w; (4.105)

означает, что в любом выражении, где только ни встретятся переменные U и V в виде сомножителей, их произведение будет заменено на идентификатор W. Например, после ввода выражения (4.106)

u3 * v**5 *a;** (4.106)

оно предстанет в виде

$a*v^2 * w^3$ (4.107)

Команда (4.108)

let u3 = 3* v - 27;** (4.108)

устанавливает правило, которое будет применяться к любой степени переменной U, большей или равной третьей степени.

Аналогичная команда широко используется на практике для пренебрежения слагаемыми заданного порядка малости. Для этого нужно величину переменной в той степени, начиная с которой ею можно пренебрегать, приравнять нулю. Например, после ввода команды (4.109)

let x7 = 0;** (4.109)

все степени переменной X, начиная с 7–й, заменяются нулями, поскольку содержат в себе в виде сомножителя ядро, равное нулю.

4.2.4. Команда FOR ALL … LET

Если нужно выполнить подстановку для всевозможных значений данного параметра некоторой операции, то следует использовать подстановку FOR ALL, после которого указывается список варьируемых переменных, являющихся формальными параметрами. Это называется установлением правила подстановки.

При выполнения предложения, содержащего конструкцию FOR ALL … LET …; никаких вычислений ни в левых, ни в правых частях стоящих в них равенств не производится. При выполнении соответствующей подстановки формальные параметры заменяются на фактические переменные. Они должны соответствовать формальным параметрам своим ***порядком следования, типом, количеством***: первому формальному параметру соответствует первый фактический, второму формальному параметру — второй фактический и т.д. Используемые обозначения для идентификаторов не играют здесь абсолютно никакой роли (кроме, конечно, указания типа величины), так как соответствие проводится по порядку следования параметров.

Например, ввод команды (4.110)

1: **for all u,v let p(u,v) = u - v;** (4.110)

и положительный ответ на появляющийся запрос

Declare P operator ? (Y or N)

y

приведет к тому, что все левые части нижеследующих равенств, входящие в преобразуемые выражения, будут заменены их правыми частями независимо от того, как фактически обозначены переменные, входящие в левые части равенств: буквами X, Y или любыми другими. В результате значение P(X,Y) (4.111) будет равно величине X–Y (4.112), значение P(U+A, U+B) (4.113) соответственно будет равно A–B (4.114):

2: **p(x,y);** (4.111)

x – y (4.112)

3: **p(u+a, u+b);** (4.113)

a – b (4.114)

Введенные в команде (4.110) формальные параметры U и V при выполнении соответствующей подстановки заменяются на любые фактические аргументы: переменные X и Y в (4.111) и выражения U+A и U+B в (4.112). Формальные параметры U и V в (4.110) никак не связаны с такими же именами фактических переменных, встречающихся в других местах сеанса работы ((4.113), (4.115) и (4.116)), так как ***соответствие проводится по их порядку следования, типу и количеству параметров***:

4: **p(u+a, v+a);** (4.115)

u – v (4.116)

Если символ операции P будет использован с большим (4.117) или меньшим (4.119) числом параметров, чем указано при установлении правила

подстановки в (4.110), то команда LET не будет действовать. При этом сообщения об ошибке не выдается, а просто распечатывается вводимое выражение (4.118) или (4.120):

5: **p(u+a, v+a, c);** (4.117)

p(a + u, a + v, c) (4.118)

6: **p(u+a);** (4.119)

p(a + u) (4.120)

Отметим, что все формальные параметры соответствующих операций должны быть описаны произвольным образом с помощью команды FOR ALL. Если этого не сделать, то неуказанный параметр будет считаться фактическим, а не формальным.

Это приведет к тому, что устанавливаемое правило будет работать только при совпадении соответствующих идентификаторов. Например, если ввести команду (4.110) в начале рассматриваемого сеанса работы в виде (4.110')

1: **for all u let p(u,v) = u - v;** (4.110')

то ни одна из последующих команд (4.111)–(4.120) не будет выполнена, так как ни в одной из них на втором месте не стоит отдельно фактический параметр V. При этом просто будут распечатаны вводимые выражения.

Команда FOR ALL … LET широко используется для введения дополнительных правил дифференцирования, тригонометрических тождеств и желательных правил упрощений, отсутствующих в системе REDUCE.

В случае, если требуется, чтобы подстановка осуществлялась лишь для тех значений переменной, которые удовлетворяют некоторому условию, используется конструкция FOR ALL SUCH THAT... LET. Например, после ввода команды (4.121)

7: **for all x such that x<0 let h(x) = x**2;** (4.121)

значение H(X) будет вычисляться как X^2 только для отрицательных значений X

8: **h(–5);** (4.122)

25

Для положительных же значений X будет просто распечатываться вводимое выражение (4.123), так как для него никаких правил подстановки установлено не было

9: **h(5);** (4.123)

h(5)

4.2.5. Отмена присваиваний и правил подстановок

С помощью команды CLEAR можно отменить все установленные ранее присваивания и правила подстановок.

Для этого после команды CLEAR следует указать список левых частей введенных ранее подстановок, которые необходимо отменить, что для рассмотренных примеров п. 4.2.3.1 будет иметь вид:

clear x, y, u * v, u3, x**7;** (4.124)

При отмене предложения, содержащего конструкцию FOR ALL … LET …, команда CLEAR ставится на место слова LET, после которого также указываются только левые части введенных ранее подстановок.

Здесь следует иметь в виду одно важное обстоятельство. Хотя установленное ранее правило подстановок (4.110) выполняется для любых используемых фактических параметров (примеры (4.111)– (4.119)), *для его отмены оно должно быть представлено в команде CLEAR с использованием именно тех формальных переменных, указанных при установлении правила:*

10: **for all u,v clear p(u,v);** (4.125)

4.2.6. Встроенная процедура SUB

При выполнении подстановок не разрешается присваивать *свободным переменным и ядрам* значения, включающие их собственные имена, так как это может привести к зацикливанию программы.

Для полной уверенности в том, что используемые во фрагменте переменные являются свободными, здесь и далее они освобождаются командой CLEAR.

Поэтому недопустима, например, следующая подстановка, так как после ее выполнения для X выражение опять будет содержать X, для которого определена подстановка, и т. д.

В результате чего при распечатывании значения X возникает ошибка, связанная с переполнением памяти (стека данных) и выдачей соответствующего диагностического сообщения:

clear x; x: = x+1; x; (4.126)

```
x := x + 1
Data stack overflow [left-ss:514bytes, left-ds:254bytes]
ERROR 14 (Stack overflow)
```

Однако допустимы замены с одним и тем же именем в разных сторонах равенства, если соответствующая переменная является *связанной* и ей уже присвоено некоторое значение.

Поэтому ошибки во фрагменте (4.126) не произойдет, если предварительно переменной X присвоить некоторое значение:

x:= 7; x: = x+1; x; (4.127)

```
x := 7
x := 8
8
```

Также запрещены присваивания вида (4.128)

clear x*y; x*y := x*y*z; x*y; (4.128)

```
x*y := x*y*z
System stack overflow [left-ss:254bytes, left-ds:352bytes]
ERROR 14  (Stack overflow)
```

где правая часть содержит в себе то ядро, которому производится присваивание, что приводит к появлению системного сообщения об ошибке.

В то же время присваивания вида (4.129) вполне допустимы, ибо они не содержат значения, включающие их собственные имена (ядро X*Y в левой части и не зависящие от него переменные X и Y в правой):

clear x, y, x*y; x*y:= x; x*y:= x + y; (4.129)

```
x*y := x
x*y := x + y
```

Согласно вышеизложенному, также недопустима подстановка (4.130) с использованием команды LET, сразу приводящая к появлению сообщения о синтаксической ошибке:

clear x; let x: = x+1; (4.130)

```
***** Syntax error: X := X + 1 invalid
```

Однако довольно часто требуется заменить в отдельном выражении свободную переменную на содержащее ее же саму выражение.

В качестве примера можно привести рассмотренную выше операцию сдвига, когда вместо переменной X надо подставить, скажем, X+1 и т. п.

Для подстановок, содержащих в обеих частях равенств имена свободных переменных и ядер, в системе REDUCE предусмотрена специальная

встроенная процедура (подпрограмма) SUB. В скобках, следующих за именем процедуры SUB, стоят равенства, определяющие соответствующие замены и подобные равенствам в LET-присваивании.

В левых частях равенств могут стоять простые имена или операторы, в правых частях фигурируют алгебраические выражения, в которые могут входить имена и операторы из левых частей равенств.

На последнем месте перед закрывающейся круглой скобкой стоит алгебраическое выражение, которое вычисляется с учетом указанных ранее замен.

Вычисленное значение этого выражения и является результатом выполнения процедуры SUB:

clear x, y, x*y; sub(y=y3, x=x+1, x*y);** (4.131)

$y^3 *(x + 1)$

которое также можно сразу присвоить любой переменной

clear x, y, x*y; z:=sub(y=y3, x=x+1, x*y);** (4.132)

$z := y^3 *(x + 1)$

Отличия между LET и SUB заключаются в следующем:

- SUB выполняет подстановку локально, один раз в одном выражении, а подстановки с LET будут применяться глобально ко всем выражениям до тех пор, пока не будут уничтожены командой CLEAR;
- SUB может выполнять только замену простой переменной выражением, LET же допускает широкий класс левых частей подстановок.

Подчеркнем, что значения переменных, входящих в параметры процедуры SUB, после ее выполнения не меняются

Если нужно выполнить несколько подстановок для одной и той же переменной, например, заменить сначала X на X+1, а затем - X на X**3, то это можно сделать следующим образом:

clear x, y; z:=sub(x=x3, sub(x=x+1, y*x));** (4.133)

$z := y*(x^3 + 1)$

Отметим, что для подобных замен в REDUCE также предусмотрено выключение из процесса вычисления повторных подстановок, что достигается опусканием поднятого в исходном состоянии флага RESUBS.

В этом режиме все замены при вычислении любых выражений проводятся только один раз.

4.3. Логические выражения и условные операторы

Для разветвления вычислительных процессов часто используется условный оператор IF, когда разветвление выполняется при помощи логических выражений.

Простейшим логическим выражением является отношение, являющееся предположением о двух выражениях, которое может быть либо *истинным* (т.е. удовлетворяться для входящих в него величин), либо *ложным* (т.е. не удовлетворяться).

В зависимости от этого вычислительный процесс может быть направлен либо по одной, либо по другой ветви. В первом случае говорят, что отношение имеет значение "истина" (TRUE), во втором — "ложь" (FALSE). Сравниваемое выражение может быть любым, в него могут включаться пробелы и 0. По принятому соглашению только имени NIL соответствует значение "ложь" и его численное значение равно нулю.

Отношение состоит из двух выражений, соединенных знаком операции сравнения: = (равно), > (больше), < (меньше), >= (больше или равно), <= (меньше или равно), NEQ (не равно). При помощи перечисленных операций сравниваются значения выражений. Операции «равно» или «не равно» могут сравнивать любые значения, в том числе и символьные, а остальные — только целочисленные.

Условными выражениями называются такие, которые зависят от значений некоторых логических выражений, формат которых имеет следующий основной вид:

```
IF логическое_выражение THEN выражение1 или команда1
ELSE выражение2 или команда2                          (4.134)
```

При обработке условного выражения сначала вычисляется логическое выражение:

- если его значением является "истина", то обрабатывается лишь выражение1 или команда1, стоящие после служебного слова THEN, а выражение2 или команда2, стоящие после слова ELSE, игнорируются, при этом, если после слова THEN стоит:
  - *выражение1*, то оно вычисляется и значение всего выражения в этом случае полагается равным значению выражения1;
  - *команда1*, то она выполняется, а значение условного выражения при этом — NIL;

- если значение логического выражения есть “ложь”, то обрабатывается лишь выражение2 или команда2, стоящие после служебного слова ELSE, а выражение1 или команда1, стоящие после слова THEN, игнорируются, при этом, если после служебного слова ELSE стоит:
  - *выражение2*, то оно вычисляется и значение всего выражения в этом случае полагается равным значению выражения2;
  - *команда2*, то она выполняется, а значение условного выражения при этом также NIL.

Поэтому при A>5 после ввода условного выражения (4.135)

if a>5 then x:= y+z else x:= y*z; (4.135)

значение X будет равно Y+Z, а при A<5 — значение X будет равно Y*Z.

Заметим, что если вместо логического выражения по ошибке появится скалярное или численное выражение, то значение этого выражения всегда будет восприниматься как «истина».

Условную конструкцию (4.134) часто применяют в упрощенном виде, формат которого можно представить так:

```
IF логическое_выражение THEN выражение1 или команда1 (4.136)
```

Если значением логического выражения: является “истина”, то конструкция (4.136) выполняется точно так же, как и ее полная форма (4.134). Если же значением логического выражения является “ложь”, то никаких действий не выполняется, а условному выражению присваивается значение NIL.

Упрощенную форму (4.136) часто применяют, если необходимо распечатать что-либо только при выполнении некоторого условия. Тогда используется команда WRITE, стоящая внутри условного выражения. Поэтому при любом значении X, отличном от нуля, выполнение предложения (4.137)

if x neq 0 then write 5/x3;** (4.137)

приведет к распечатыванию значения 5/X**3, и не вызовет никаких действий, если X равно нулю.

Более сложные логические выражения, входящие в оператор IF, могут быть построены с помощью операций над выражениями отношения:

- NOT — отрицание (нет);
- AND — логическое умножение (и);
- OR — логическое сложение (или).

Операция отрицания NOT применяется к одному операнду и меняет значение логического выражения ЛВ на противоположное. Например, если ЛВ имеет значение истинно, то NOT ЛВ ложно и наоборот.

Операции AND и OR применяются к двум операндам, т.е. к двум выражениям отношения. Их результат выполнения имеет значение истинно, когда оба операнда (для AND) или хотя бы один из них (для OR) имеют значение истинно. Поэтому при 5<A<8 после ввода условного выражения (4.138)

```
if a>5 and a<8 then x:= y+z else x:= y*z;          (4.138)
```

значение X будет равно Y+Z, а при A<5 или A>8 — значение X будет равно Y*Z.

В логических выражениях используются также следующие операторы: FIXP(U), FREEOF(U, V), NUMBERP(U), ODRP(U, V). Они имеют значение "истина", если:

- в FIXP(U) выражение U имеет целочисленное значение;
- в FREEOF(U, V) в выражении U отсутствует ядро V;
- U является числом в NUMBERP(U);
- U имеет более высокий порядок, чем V в ODRP(U, V).

Заметим, что не следует без необходимости использовать сложные конструкции с IF, как и любые другие. Их лучше представить в виде нескольких более простых выражений.

Рассмотрим некоторые дополнительные возможности. Поскольку LET-присвоения производятся без вычисления правой части, можно делать условным само значение переменной. Это позволяет, например, провести вычисление модуля M значения переменной X:

```
let m = if x >= 0 then x else -x;                  (4.139)
x:=3$ m;
3
x:=-7$ m;
7
```

Значением переменной M в (4.139) становится само условное выражение целиком, поэтому вычисленное значение этой переменной зависит от текущего значения X, всегда оставаясь равной ему по модулю.

Отметим также, что выражения, стоящие внутри условных выражений, сами могут быть условными. В таких случаях нужно брать внутренние условные выражения в круглые скобки.

Подчеркнем, что между словами THEN и ELSE может находиться только одно выражение или одна команда. Для обхода этого ограничения используются специальные конструкции, называемые блоками и рассмотренные ниже.

4.4. Организация блоков

4.4.1. Простой блок или групповой оператор

Простой блок, называемый также групповым оператором или составным предложением, объединяет несколько утверждений или команд, которые должны выполняться одновременно. Он образуется внутри операторных скобок, открывающихся двумя знаками "меньше" (<<) и закрывающихся двумя знаками "больше" (>>). Например, присвоив с помощью команды LET переменной FLAG значение группового оператора:

```
let flag = <<on exp, allfac, mcd, resubs;
off gcd, rat, div, list>>;                    (4.140)
```

мы приведем сразу все указанные в нем флаги в исходное состояние с того момента, где появится предложение (4.140).

Использование такого блока полезно при формировании новых вычислительных задач в рамках одного сеанса работы или большой программы, где данные флаги уже поднимались и опускались произвольным образом.

Обратим ваше внимание, что в команде (4.140) перед закрывающимися операторными скобками (двумя знаками "больше" >>) не стоит никакого ограничителя (; или $), а только в конце самой команды. Наличие ограничителя перед ">>" необходимо только в том случае, если это одна из команд: LET, CLEAR, IN, OUT или SHUT. В остальных случаях это приводит к тому, что значением блока становится нуль (NIL).

Значение простого блока равно значению выражения, стоящего перед скобками >>, или 0, если там стоит команда. Поэтому значением блока в примере (4.141) является $7*z^3$:

```
u:= <<x:=7; y:=z**3$ on exp; x*y>>;           (4.141)
```

u:= $7*z^3$

4.4.2. Составной оператор или жесткий блок

Часто необходимо сохранять не только окончательные результаты, но и промежуточные. В таких случаях последовательность операций заключается между словами BEGIN и END, которые выполняют ту же роль, что и операторные скобки в простом блоке. Служебное слово BEGIN представляет собой открывающуюся операторную скобку, служебное слово END — закрывающуюся.

Такая конструкция образует составной оператор или жесткий блок. Чтобы в результате выполнения он получил какое–либо значение, необходимо воспользоваться оператором RETURN, вслед за которым должны следовать переменная или выражение, выводимые в качестве значения блока. Если же оператор RETURN опущен (или после слова RETURN ничего не написано), то значением блока будет 0 (NIL).

Рассмотренные выше свойства простого блока справедливы и для жесткого блока. Они оба также являются выражениями, т.е. имеют значения и могут использоваться в любом месте программы, где должно стоять выражение. Однако жесткий блок имеет свои особенности, которые мы рассмотрим ниже.

Внутри жесткого блока, в отличие от простого, можно определить внутренние переменные, имеющие *локальный* характер. Для объявления переменных ***локальными*** их идентификаторы должны появиться в соответствующих списках объявлений SCALAR, REAL или INTEGER сразу после слова BEGIN. После этого:

- *они будут действовать только внутри данного блока* и стираются из памяти, как только операции в блоке заканчиваются, что очень полезно при написании сложных программ с большим количеством переменных, имена которых должны быть различными;
- *они всегда связанные:* первоначальные их значения сразу после описания командой SCALAR — нули (в алгебраическом режиме , NIL — в символьном режиме), поэтому попытка очистить их командой CLEAR оценивается как ошибка;
- *их имена не связаны с такими же именами*, используемыми вне блока.

Все остальные переменные в блоке, отличные от локальных, являются ***глобальными***:

- *они определены по всей программе* и обращение к ним такое же, как и к переменным с тем же именем вне блока;
- *их значения могут непрерывно меняться*, даже если эти переменные не объявлены глобальными вне блока на более высоком уровне.

Иначе можно сказать, что глобальными являются переменные, не описанные ни в каком блоке, поэтому они являются внешними по отношению ко всем блокам. Также участок программы, не содержащийся ни в каком блоке, часто называют ее *верхним уровнем*. Поэтому можно, например, сказать, что терминатор ";" вызывает распечатывание значения выражения лишь на *верхнем уровне программы.*

Переменные, идентификаторы которых появились в соответствующих списках объявлений ARRAY или OPERATOR, всегда будут являться *глобальными,* даже если их описание было произведено в любом месте внутри блока. Поэтому массивы и операторы — всегда глобальные объекты.

После объявлений следуют операторы, которые требуется выполнить. *Последний оператор в блоке (перед END) записывается без ограничителя. Однако после END ограничитель указывается обязательно.*

Поэтому пример (4.141) для представления его в виде жесткого блока примет вид: (4.142):

```
u:= begin scalar x,y; x:=7; y:=z**3$ on exp;
return x*y end;                                    (4.142)
```

u:= $7*z^3$

Обратим ваше внимание, что если переменную Z в примере (4.142) также поместить в список объявлений команды SCALAR, то результат примет совсем другой вид:

```
u:= begin scalar x,y,z; x:=7; y:=z**3$ on exp;
return x*y end;                                    (4.143)
```

u:= 7*0

Это объясняется тем, что значение переменной Z внутри блока не изменяется и остается равным своему первоначальному значению сразу после описания командой SCALAR — нулю.

Внутри жесткого блока BEGIN ... END возможно использование не только операторов присваивания, но и более сложных структур. В них могут содержаться условные операторы, конструкции WHILE и REPEAT, которые могут присутствовать в составе других вложенных блоков, а также команда передачи управления GO TO.

Отметим, что эта команда имеет несколько имен, которые эквивалентны: GO, GO TO, GOTO.

Команда GO TO используется с единственным параметром, которым должно быть имя без скобок. Оно называется *именем метки* (M в примере (4.145)).

Внутри блока, где использована команда GO TO, должно стоять предложение, помеченное тем же именем и отделяемое от этого предложения двоеточием.

По команде GO TO происходит передача управления к предложению с указанной меткой, после чего оно станет выполняться.

Рассмотрим жесткий блок с использованием команды GO TO для вычисления факториала от N для заранее заданного любого его значения, например, для N=100:

n:=100$ begin scalar k; k:=1; (4.144)

m: if n=0 then return k; k:=k*n; n:=n –1; go to m end; (4.145)

93326215443944152681699238856266700490715968264381621468592963895217599993229915608941463976156518286253697920827223758251185210916864000000000000000000000000 (4.146)

В предложении (4.144):

- сначала присваивается значение переменной N, факториал от которой вычисляется;
- затем служебное слово BEGIN открывает составной оператор или жесткий блок, в котором:
  - переменная K объявляется локальной, присутствуя в списке из одного элемента в объявлении SCALAR;
  - после чего переменной K присваивается значение, равное единице.

  Предложение (4.145) продолжает жесткий блок, в котором:
  - сначала указывается имя метки M, отделяемое от дальнейшей части предложения двоеточием, к которой по команде GO TO каждый раз происходит передача управления;
  - затем следует условное выражение IF N=0 THEN RETURN K;, в котором оператор RETURN K будет выполнен только при N=0, прекращая выполнение жесткого блока и выводя в качестве его значения текущую величину переменной K. При любых других значениях N он будет просто пропускаться, затем будут выполняться следующие за ним операторы жесткого блока;
  - операторы присваивания (K:= K*N; N:= N – 1;) каждый раз определяют новое значение переменной K в виде произведения текущих значений переменных K и N. После этого значение N сразу уменьшают на единицу, присваивая его в качестве нового значения той же переменной N;
  - команда GO TO используется с единственным параметром, которым является имя метки M, стоящее также внутри блока, отделяемое от его остальной части двоеточием и определяющее место, куда по команде GO TO каждый раз происходит передача управле-

ния, после чего оно начинает выполняться. Таким образом операторы предложения (4.145) образуют внутри жесткого блока цикл, который будет выполняться до тех пор, пока значение N не станет равным нулю, после чего значение переменной K будет равно факториалу от N, что и будет выведено в качестве значения блока по оператору RETURN K;

- служебное слово END закрывает рассмотренный составной оператор или жесткий блок. Последний оператор в блоке (GO TO M) записывается, как и полагается, без ограничителя. Однако после END ограничитель указывается обязательно.

Выражение (4.146) представляет собой значение факториала от N для N=100, распечатываемое в результате работы рассмотренного жесткого блока.

4.5. Операторы цикла

Наиболее сложными и мощными из числа операторов управления являются операторы цикла, позволяющие многократно повторить нужную последовательность выражений или команд при различных значениях некоторой переменной, называемой *счетчиком цикла:* Операторы цикла могут быть представлены в различных формах:

- ***безусловные FOR-циклы*** – реализуют заранее задаваемое в *заголовке цикла* многократное выполнение каких-либо однотипных операций (выражений или команд). В зависимости от выполняемых действий бывают трех типов, определяемых служебными словами DO, SUM и PRODUCT:
  - FOR заголовок цикла DO выражение или команда – при каждом выполнении указанное выражение или команда соответственно вычисляется или исполняется при разных значениях счетчика цикла, изменяющихся с некоторым приращением от начальной до конечной величины. Сам оператор FOR … DO имеет своим значением 0;
  - FOR заголовок цикла SUM выражение – при заданных значениях параметров цикла *происходит определение* ***суммы*** *указанного выражения,* которую имеет своим значением сам оператор FOR … SUM;
  - FOR заголовок цикла PRODUCT выражение – при заданных значениях параметров цикла *происходит определение* ***произведения*** *указанного выражения,* которое имеет своим значением сам оператор FOR … PRODUCT;

- ***условные циклы*** – многократно выполняются до тех пор, пока используемое в них *логическое выражение:*
  - *истинно* – WHILE … DO-цикл;
  - *ложно* – REPEAT … UNTIL-цикл.

  Рассмотрим их подробнее.

4.5.1. Безусловные FOR-циклы

Заголовок FOR-цикла располагается после служебного слова FOR и представляет собой следующую конструкцию:

```
I:= N1 STEP N2 UNTIL N3                                    (4.147)
```

где I — *переменная цикла или счетчик цикла:* простой идентификатор без скобок, имеющий целочисленное значение, которое не связано с его значением вне цикла. Поэтому в качестве имени переменной цикла можно использовать I, хотя вне цикла I есть мнимая единица;

`N1, N2` и `N3` – соответственно *начальное значение N1, приращение N2* (шаг изменения при каждом прохождении цикла) *и конечное значение N3* переменной цикла I: любые выражения, не являющиеся блоками и обладающие целочисленными значениями, причем значение приращения N2 не должно быть нулевым.

Если приращение переменной цикла N2=1, то возможна упрощенная форма (4.148) записи заголовка цикла (4.147), в которой участок STEP 1 UNTIL заменяется двоеточием:

```
I:= N1:N3                                                  (4.148)
```

Оператор цикла FOR … DO используется для организации программных циклов с заранее известным числом повторений и может быть записан в одной из двух форм:

- *общей* — с использованием заголовка цикла в виде (4.147):

```
FOR I:= N1 STEP N2 UNTIL N3 DO выражение или команда  (4.149)
```

- *упрощенной* — с использованием заголовка цикла в виде (4.148), если приращение переменной цикла N2=1:

```
FOR I:= N1:N3 DO выражение или команда                     (4.150)
```

Отметим, что значение приращения N2 не должно быть нулевым, а `выражение или команда` может быть любой, в частности, это может быть еще один цикл, условное выражение или блок.

Стоящее после служебного слова DO выражение или команда выполняется сначала при значении целой переменной I=N1, затем при I=N1+N2, затем при I=N1+2*N2, I=N1+3*N2 и т.д. При этом каждый раз значение переменной цикла I сравнивается по величине с конечным значением цикла N3. Если значение I окажется больше N3 при положительном приращении N2 или меньше N3 при отрицательном N2, то выполнение цикла заканчивается и программа переходит к выполнению следующего оператора. Это называется нормальным выходом из цикла.

Выражение или команда не выполняются ни разу, если:

- начальное значение цикла N1 *больше* конечного значения N3 при положительном приращении N2;
- начальное значение цикла N1 *меньше* конечного значения N3 при отрицательном приращении N2.

В качестве примера использования цикла FOR … DO приведем фрагмент сеанса работы или программы для заполнения массива с именем P значениями первых восьми ортогональных многочленов Лежандра, вычисляемых по рекуррентной формуле:

$$P_M(X) := ((2*M-1)*X*P_{M-1}(X) - (M-1)*P_{M-2}(X))/M. \qquad (4.151)$$

Опишем массив P нужной размерности командой ARRAY и для более компактной формы представления результатов опустим флаг NAT:

```
1: array p(7)$ off nat;                                    (4.152)
```

Заполним известными значениями нулевой P(0) и первый P(1) многочлены Лежандра для возможности использования рекуррентной формулы (4.151):

```
3: p(0):= 1; p(1):= x;                                     (4.153)
p(0) := 1$
p(1) := x$
```

Организуем цикл FOR … DO с начальным N1=2 и конечным N3=7 значениями переменной цикла M с шагом N2=1, использовав упрощенный вид заголовка цикла (4.146). Поскольку присвоение элементам массива P в примере производится не на верхнем уровне программы, а в цикле, то для печати результатов используется оператор WRITE:

```
5: for m:=2:7 do write p(m):=
   ((2*m-1)*x*p(m-1) - (m-1)*p(m-2))/m;                    (4.154)
p(2) := (3*x**2 – 1)/2$
p(3) := (x*(5*x**2 – 3))/2$
p(4) := (35*x**4 – 30*x**2 + 3)/8$
```

```
p(5) := (x*(63*x**4 – 70*x**2 + 15))/8$
p(6) := (231*x**6 – 315*x**4 + 105*x**2 – 5)/16$
p(7) := (x*(429*x**6 – 693*x**4 + 315*x**2 – 35))/16$
```

Оператор цикла FOR … SUM используется для определения при заданных значениях параметров цикла ***суммы*** *указанного выражения,* которую имеет своим значением сам оператор FOR … SUM, и может быть записан в одной из двух форм:

- *общей* — с использованием заголовка цикла в виде (4.147):

```
FOR I:= N1 STEP N2 UNTIL N3 SUM выражение          (4.155)
```

- *упрощенной* — с использованием заголовка цикла в виде (4.148), если приращение переменной цикла N2=1:

```
FOR I:= N1:N3 SUM выражение                        (4.156)
```

Поэтому для получения, например, суммы степеней для $(X - A)^I$ для четных значений I, изменяющихся от 0 до 10, следует ввести:

```
off exp; for i:= 0 step 2 until 10 sum (x - a)**i; on exp;   (4.157)
```

$(((((a - x)^2 + 1) + (a - x)^4) + (a - x)^6) + (a - x)^8) + (a - x)^{10}$

Здесь флаг EXP опущен, чтобы не проводилось раскрытие выражений при их вычислениях, после чего он опять приводится в исходное состояние.

Оператор цикла FOR … PRODUCT используется для определения при заданных значениях параметров цикла ***произведения*** *указанного выражения,* которое имеет своим значением сам оператор FOR … PRODUCT, и может быть записан в одной из двух форм:

- *общей* — с использованием заголовка цикла в виде (4.147):

```
FOR I:= N1 STEP N2 UNTIL N3 PRODUCT выражение      (4.158)
```

- *упрощенной* — с использованием заголовка цикла в виде (4.148), если приращение переменной цикла N2=1:

```
FOR I:= N1:N3 PRODUCT выражение                    (4.159)
```

Поэтому для получения, например, произведения степеней для $(X - A)^I$ для четных I, изменяющихся от 0 до 10, следует ввести:

```
off exp; for i:= 0 step 2 until 10 product (x - a)**i; on exp;(4.160)
```

$(a - x)^{30}$

Здесь флаг EXP также опущен, чтобы не проводилось раскрытие выражений при их вычислениях, иначе в обоих примерах (4.157) и (4.160) будут получаться весьма громоздкие выражения. Мы рекомендуем убедиться в этом, запустив оба примера (4.157) и (4.160) с поднятым флагом EXP.

Рассмотрим также вариант реализации с помощью оператора цикла FOR … PRODUCT примера (4.144) – (4.145) для вычисления факториала от N для заранее заданного любого его значения, например, для N=100:

for n:= 1 : 100 product n; (4.161)

933262154439441526816992388562667004907159682643816214685929638952175999932299156089414639761565182862536979208272237582511852109168640000000000000000000000000 (4.162)

Конечно, результаты вычислений (4.146) и (4.162) полностью совпадают, однако получены они в последнем случае несравненно более простым способом.

Чтобы присвоить переменной NF факториал 100 следует пример (4.161) представить в виде:

nf:= for n:= 1 : 100 product n; (4.163)

После выполнения предложения (4.163) переменной NF будет присвоено значение (4.162).

Рассмотрим также пример совместного использования циклов FOR … SUM и FOR … PRODUCT, для чего определим при N=5 выражение (4.164), представляющее собой разложение функции e^X в степенной ряд:

$$\sum_{k=0}^{n} x^k \Big/ k! \qquad (4.164)$$

Это можно сделать, например, следующим образом:

on div; for k:=0:5 sum xk/for i:=1:k product i; off div;** (4.165)

$$\frac{1}{120} * x^5 + \frac{1}{24} * x^4 + \frac{1}{6} * x^3 + \frac{1}{2} * x^2 + x + 1 \qquad (4.166)$$

В предложении (4.165) флаг DIV поднят для получения результатов в более привычной для нас форме, чтобы проводилось сокращение простых множителей в выражениях при их вычислениях, после чего он опять возвращается в исходное состояние.

Обратим ваше внимание, что *под выражением в FOR-цикле понимается все выражение, стоящее между словом DO (PRODUCT, SUM) и ограничителем,* поэтому в примере (4.165):

- переменная K внешнего цикла FOR … SUM будет иметь каждый раз определенное значение в качестве конечного значения переменной цикла FOR … PRODUCT;

- цикл FOR … PRODUCT является внутренним. Он будет полностью выполняться при каждом значении переменной K внешнего цикла FOR … SUM;
- при вычислении суммы (4.166) на значение факториала от K делится каждый член суммы в отдельности.

Если начальное значение, шаг и ограничитель цикла таковы, что он не выполняется ни разу, то значением цикла FOR … PRODUCT будет единица, а значением FOR … SUM — нуль. Поэтому цикл FOR … PRODUCT из предложения (4.165) при K=0 не выполняется ни разу и его значение для 0! будет равно единице.

4.5.2. Условные циклы

Цикл WHILE ... DO многократно выполняется до тех пор, пока используемое в нем *логическое выражение истинно,* и. может быть записан в следующей основной форме

```
WHILE логическое выражение DO выражение или команда (4.167)
```

Соответствующая проверка на истинность логического выражения производится *перед* очередным шагом цикла. Таким образом, если значение логического выражения с самого начала — ложь, выражение или команда не выполняются ни разу. После служебного слова DO должно стоять только одно выражение или команда. Если же их должно быть несколько, то применяют групповой оператор.

В качестве примера использования цикла WHILE ... DO приведем фрагмент сеанса работы или программы для распечатки квадратов первых восьми чисел.

```
n:=0$ while n<=7 do <<n:=n+1; write n**2>>$ (4.168)
1  4   9   16   25   36   49   64 (4.169)
```

Сначала оператор присваивания задает переменной N нулевое значение. Затем перед очередным шагом цикла производится проверка на истинность логического выражения. Так как при N=0 логическое выражение N<=7 истинно, то следующий за DO групповой оператор начинает выполняться. Сначала он увеличивает текущее значение N на единицу (оно становится равным N=1) и распечатывает квадрат этого нового значения.

Так продолжается на каждом шаге до N=8 и квадраты чисел распечатываются с новой строки (здесь для экономии места результаты (4.169) представлены в одной строке). При N=8 логическое выражение оказывается ложным и выполнение цикла заканчивается.

Цикл REPEAT ... UNTIL многократно выполняется до тех пор, пока используемое в нем *логическое выражение ложно,* и. может быть записан в следующей основной форме

```
REPEAT выражение_или_команда UNTIL логическое_выражение (4.170)
```

Соответствующая проверка на истинность логического выражения производится *после* выполнения каждого шага, что и отражает его соответствующее расположение в конце оператора. Таким образом, если значение логического выражения с самого начала — истина, то и в этом случае выражение или команда выполняются хотя бы один раз до остановки выполнения цикла.

Причина остановки цикла REPEAT ... UNTIL противоположна WHILE ... DO. Поэтому рассмотрим реализацию конструкции REPEAT ... UNTIL на том же примере (4.168) для распечатки квадратов первых восьми чисел, что может иметь, например, следующий вид:

```
n:=0$ repeat <<n:=n+1; write n**2 >> until n>=7$          (4.171)
1  4   9   16   25   36   49
```

Здесь при достижении N=7 происходит прекращение выполнения цикла, ибо логическое выражение N>=7 становится истинным. Для получения полного вывода (4.169) логическое выражение N>=7 в (4.171) нужно представить в виде N>7.

Обратим ваше внимание, что в REDUCE *невозможно* ни из какого цикла выйти за счет передачи управления командой GO TO на метку, находящуюся вне цикла, ибо метка по определению должна находиться внутри жесткого блока или цикла.

4.6. Процедуры

Все встроенные в систему REDUCE операторы и функции оформлены в виде тех или иных процедур, предназначенных для неоднократного выполнения некоторого алгоритма, вызов которого осуществляется по имени процедуры. В алгоритме используются обозначения переменных, массивов, операторов и других процедур, называемых формальными параметрами, имена которых при каждом обращении к процедуре можно легко заменять соответствующими фактическими параметрами.

Процесс задания алгоритма вычислений для данной процедуры называется ее *описанием* и производится при помощи команды PROCEDURE.

Описание процедуры производится на верхнем уровне программы (вне блоков и циклов) и должно предшествовать ее использованию. Оно занимает два предложения, называемых *заголовком* и *телом процедуры*:

- в *заголовке* за командой PROCEDURE указывается ее *имя*, за которым в скобках ставится список простых имен, называемых *формальными параметрами*:
  - *имя* обозначает данную процедуру, используется для ее вызова и не должно применяться ни для каких иных целей;
  - *формальные параметры* — это идентификаторы, которые внутри процедуры используются как имена простых переменных, массивов и других процедур. Список формальных параметров через запятую помещается в скобках после указания ее имени:

```
PROCEDURE имя (список формальных параметров через запятую);
```

- во втором предложении содержится выражение, определяющее алгоритм вычислений и называемое *телом процедуры*. Оно представляет собой связанную определенным образом совокупность различных блоков, циклов, условных выражений и т. п., составляющую один оператор. Если тело процедуры ограничено терминатором ";" (точка с запятой) и в нем не обнаружено синтаксических ошибок, то после описания процедуры произойдет печать ее имени. При использовании терминатора "$" никакой печати не происходит.

Если процедура с таким же именем уже описана ранее, то произойдет ее переопределение, в результате которого старое описание процедуры теряется и заменяется новым, а на печать выдается сообщение:

**** имя_процедуры is redefined.

Отметим, что в самом REDUCE существует большое количество внутренних процедур с простыми именами. Поэтому на этапе отладки программы следует использовать после тела процедуры ограничитель ";",чтобы случайно не переопределить какую-нибудь процедуру REDUCE.

Например, процедура Fl(N), вычисляющая факториал от N с использованием цикла FOR … PRODUCT, может быть описана следующим образом:

```
1: procedure fl(n)$ for k:=2:n product k;                    (4.172)
fl
```

Здесь FL — имя процедуры, N — формальный параметр, предложение после терминатора "$" — тело процедуры.

Существенно, что при выполнении приведенного выше примера, как и при описании любой процедуры, никаких вычислений не производится. На этапе описания происходит лишь синтаксический контроль и «запоминание» алгоритма. Если бы это было не так, то при выполнении (4.172) выдавалось бы сообщение об ошибке, поскольку ограничитель цикла должен иметь целочисленное значение, а формальный параметр N вообще каким-либо значением не обладает.

Для проведения вычислений по алгоритму, заложенному в теле процедуры, ее необходимо вызвать. Параметры, фигурирующие при обращении к процедуре, называются *фактическими:*

```
имя_процедуры (список фактических параметров через запятую);
```

При вызове процедуры формальные параметры заменяются на фактические параметры. В первую очередь происходит вычисление их значений, а затем они используются в процедуре.

Фактические параметры должны соответствовать формальным параметрам вызываемой процедуры своим порядком следования, типом, количеством, размерами массива: первому формальному параметру соответствует первый фактический, второму формальному параметру соответствует второй фактический и т.д. Используемые обозначения для идентификаторов не играют здесь абсолютно никакой роли (кроме, конечно, указания типа величины), так как *соответствие проводится по порядку следования параметров.*

Поэтому для вычисления факториала от N для заранее заданного любого его значения, например, для N=100, можно:

- непосредственно указать нужное значение фактического параметра при вызове процедуры:

```
2: fl(100);                                                    (4.173)
```

```
933262154439441526816992388562667004907159682643816214685929638952175999932299156089414639761565182862536979208272237582511852109168640000000000000000000000000000 (4.174)
```

- предварительно присвоить нужное значение фактического параметра любой переменной, используемой при вызове процедуры:

```
3: k:=100$ fl(k);                                              (4.175)
```

В последнем случае также будет напечатан результат 100! (4.174). Значение фактического параметра можно задавать также в виде выражения, например, для 20!:

5: **fl(k/5);** (4.176)

2432902008176640000 (4.177)

Чтобы представить это же вычисление факториала от N, описанное в примере (4.144)–(4.145), в виде процедуры с жестким блоком, нужно впереди него просто поставить имя процедуры FL с указанием используемого формального параметра:

6: **procedure fl(n)$ begin scalar k; k:=1;**

m: if n=0 then return k; k:=k*n; n:=n -1; go to m end; (4.178)

Теперь, после ввода предложения (4.178) и получения сообщения о переопределении процедуры, по любому ее вызову вида (4.173) или (4.175) также будет напечатан результат для 100! (4.174) или для любого произвольно заданного значения.

Обратим ваше внимание, что простейшие процедуры, используемые для выполнения определенных действий, могут не содержать формальных параметров. В качестве примера опишем процедуру KP контрольной печати текущих значений переменных X, Y и Z:

8: **procedure kp; begin write x, " ", y, " ", z; end;** (4.179)

kp

Присвоим теперь численное значение X := 7, символьное — Y := B и оставим свободной переменную Z. Поскольку список формальных параметров, используемых при описании процедуры KP, пуст, то таким же должен быть и соответствующий ему список фактических параметров. Это и указывается при обращении к процедуре KP( ) в предложении (4.180), после чего она выполняется и распечатывает текущие значения переменных X, Y и Z:

9: **x:=7$ y:=b$ kp( )$** (4.180)

7 b z

Отметим, что для описания процедур может быть также использовано LET-присвоение. Так, процедура FL (4.172) может быть записана в следующем операторном виде:

12: **for all n let fl(n) = for k:=2:n product k;** (4.181)

Теперь, после положительного ответа на запрос системы:

"Declare FL operator ? (Y or N)"

вызов теперь уже оператора процедуры (4.181) также будет осуществляться по любому обращению вида (4.173) или (4.175). При этом будет напечатан результат для 100! (4.174) или для любого произвольно заданного значения, что мы и рекомендуем проверить.

В примере (4.181) использована возможность команды LET ставить в соответствие переменным или операторам различные сложные конструкции. Выполнение такой конструкции начинается после обращения в ходе выполнения программы к соответствующей переменной или оператору. Поэтому очень удобно связать одну переменную с набором, например, процедур громоздких тригонометрических подстановок, определенных с помощью команды FOR ALL … LET, которые начнут действовать сразу после обращения к имени соответствующей переменной, а другую – с их отменой командой FOR ALL … CLEAR:

14: **let tp = <<for all a, b let sin(a)*sin(b) = (cos(a-b)-cos(a+b))/2,**
cos(a) * cos(b) = (cos(a-b) + cos(a+b))/2,
sin(a) * cos(b) = (sin(a-b) + sin(a+b))/2, (4.182)
cos(a) **2 = (1+cos(2*a))/2, sin(a)2 = (1-cos(2*a))/2;>>;**

15: **tpS cos(z)**8;** (4.183)

$$\frac{\cos(8*z) + 8*\cos(6*z) + 28*\cos(4*z) + 56*\cos(2*z) + 35}{128}$$

Конструкция LET TP = <<FOR ALL … LET …>> (4.182) описывает необходимые тригонометрические подстановки. Они начнут действовать сразу после обращения к имени переменной TP в начале предложения (4.183). Теперь любые степени функций COS или SIN любого аргумента, а также их произведения будут разлагаться в ряды Фурье:

16: **sin(z)**8;** (4.184)

$$\frac{\cos(8*z) - 8*\cos(6*z) + 28*\cos(4*z) - 56*\cos(2*z) + 35}{128}$$

17: **cos(z)**8*sin(z)**8;** (4.185)

$$\frac{\cos(16*z) - 8*\cos(12*z) + 28*\cos(8*z) - 56*\cos(4*z) + 35}{32768}$$

Для отмены введенных предложением (4.182) подстановок используется конструкция LET CTP = <<FOR ALL … CLEAR …>> (4.186). Тригонометрические соотношения перестанут действовать сразу после обращения к имени переменной CTP в начале предложения (4.187).

Теперь любые степени функций COS или SIN любого аргумента, а также их произведения станут свободными и будут распечатываться в неизменном виде (4.188):

18: **let ctp = <<for all a, b clear sin(a)*sin(b), cos(a) * cos(b), sin(a) * cos(b), cos(a) **2, sin(a)**2; >>;** (4.186)

19: **ctp$ cos(z)**8; sin(z)**8; cos(z)**8*sin(z)**8;** (4.187)

$\cos(z)^8$

$\sin(z)^8$ (4.188)

$\cos(z)^8 * \sin(z)^8$

Отметим, что имена используемых переменных для установления или отмены соответствующих подстановок, конечно, могут быть любыми. В некоторых версиях системы REDUCE внутри жесткого блока установление или отмена подстановок не вводится простым обращением к переменным TP и CTP. В этом случае, чтобы ввести или отменить подстановку, нужно другим вспомогательным переменным присвоить используемые соответствующие переменные TP и CTP: например, VTP:= TP$ и VCTP:= CTP$.

4.7. Операции дифференцирования DF и интегрирования INT

Операция дифференцирования DF была ранее описана в п. 3.1 и широко применялась в пп. 3.3–3.5 для определения скорости и ускорения точек при различных видах движения. Рассмотрим ее дополнительные возможности, а также операцию интегрирования INT, возможности которой также уже показывались в п. 1.1.

Синтаксис операторов дифференцирования DF и интегрирования INT можно представить соответственно в следующих общих формах:

`DF(функция,аргумент1,порядок1,…,аргументN,порядокN)` (4.189)

`INT(функция,аргумент)` (4.190)

Только первый параметр (функция) операторов дифференцирования DF и интегрирования INT может иметь значением алгебраическое выражение. Значениями последующих параметров должны быть:

- *простого вида переменные* для обозначения аргументов;
- *целые неотрицательные числа*, задающие соответствующий порядок производной, которую требуется взять по стоящей перед цифрой переменной.

Если число, выражающее порядок любой частной производной (порядокI) равно единице, то оно может быть опущено.

Производные от встроенных функций система вычисляет автоматически. Напомним, что в систему REDUCE встроены следующие функции: SIN, COS, TAN, SINH, COSH, TANH, соответствующие тригонометрическим и гиперболическим функциям, и обратные им функции ASIN, ACOS, ATAN, ASINH, ACOSH, ATANH, а также EXP, LOG, SQRT, и некоторые другие (COT, DILOG, ERF, EXPINT).

REDUCE может интегрировать выражения, состоящие из полиномов, логарифмов (LOG) и экспонент (EXP), тангенсов (TAN) и арктангенсов (ATAN). Рациональные функции интегрируются, если знаменатель факторизуется системой. Результатом действия INT является неопределенный интеграл от выражений, состоящих из полиномов, LOG, EXP, TAN и других элементарных функций без произвольной постоянной.

Могут также интегрироваться выражения, в состав которых входят неэлементарные функция ошибок (ERF), двойные логарифмические функции (DILOG) и некоторые тригонометрические выражения, отличные от тангенса. Однако здесь система не всегда добивается успеха в получении решения, даже если оно и существует.

Отметим, что программа интегрирования обязательно проверяет, является ли используемый аргумент в интеграле действительно свободной переменной или нет; в последнем случае выдается сообщение об ошибке.

Поскольку взятие неопределенных интегралов на ПК является одним из наиболее впечатляющих успехов систем аналитических вычислений, то рассмотрим их применение более подробно. Опустим сразу перед началом сеанса работы флаг NAT для удобства машинной обработки получаемых в REDUCE выражений:

```
1: off nat;                                                    (4.191)
2: y:= x**2((x+a)/(x+b))**2$ int(y,x);                         (4.192)
– (6*log(b + x)*a**2*b**2 + 6*log(b + x)*a**2*b*x –
18*log(b + x)*a*b**3 – 18*log(b + x)*a*b**2*x +
12*log(b + x)*b**4 + 12*log(b + x)*b**3*x – 6*a**2*b*x –       (4.193)
3*a**2*x**2 + 18*a*b**2*x + 9*a*b*x**2 – 3*a*x**3 –
12*b**3*x – 6*b**2*x**2 + 2*b*x**3 –x**4) / (3*(b + x))$
```

Интеграл от выражения для Y (4.192) является табличным и имеется в справочнике [20, с. 33, 17], однако приведенное там его значение отличается по форме от полученного вида (4.193). Не следует отчаиваться, если применение различных сочетаний флагов не приведет к желаемому результату.

Здесь мы имеем дело с достаточно серьезной проблемой машинных аналитических вычислений — приведения равных в математическом отношении объектов к единой форме представления. Поэтому в качестве проверки будем дифференцировать получаемые первообразные для возврата к исходным выражениям.

Напомним, что результаты каждого вычисленного выражения запоминаются системой в переменной WS и сохраняются до следующего вычисления, в котором этот идентификатор можно использовать. Поэтому, чтобы продифференцировать выражение (4.193), можно просто ввести:

```
4: df(ws,x);                                                    (4.194)
(x**2*(a**2 + 2*a*x + x**2))/(b**2 + 2*b*x + x**2)$             (4.195)
```

Полученное выражение (4.195) несколько отличается по форме от выражения для Y (4.192). Однако здесь легко заметить, что в его числителе и знаменателе просто раскрыты квадраты сумм, которые являются в математическом отношении равными.

Следует также отметить, что привычное аналитическое представление выражений, представляющее его в компактной форме с минимальным числом членов, оказывается не всегда удобным с машинной точки зрения. Например, дифференцирование и интегрирование легкого табличного интеграла от степенной функции:

```
5: df(x**p,x); int(x**p,x);                                     (4.196)
(x**p*p) / x$                                                   (4.197)
(x**p*x) / (p + 1)$                                             (4.198)
```

при использовании всевозможных сочетаний флагов не приведет к привычной для нас форме представления результата с измененной формой представления степени X. Широкие возможности системы по изменению формы представления вида выражений просто в принципе не могут представить степень переменной в виде разности (4.199) или суммы (4.200) свободной переменной и числового значения. Такого разделения понятий не существует при ручных аналитических расчетах, в которых эта степень просто обозначает уменьшенное (4.199) или увеличенное (4.200) на единицу значение числа P:

$$p * x^{p-1} \tag{4.199}$$

$$\frac{x^{p+1}}{p+1} \tag{4.200}$$

И эти отличия в формах представления равных в математическом отношении выражений резко возрастают с их усложнением.

Поэтому используемый нами метод проверки путем дифференцирования получаемых первообразных для возврата к исходным выражениям окажется весьма удобным и развеет всяческие сомнения по этому поводу. К тому же эта проверка очень легко выполняется для любых получаемых первообразных любой степени сложности введением одного очень простого оператора DF(WS,X), который всегда означает дифференцирование по X полученного ранее системой результата, хранящегося до следующего вычисления в переменной WS:

5: **y:= (exp**(a*x)-1)/ (exp**(a*x)+1)$ int(y,x);** (4.201)

(2*log(exp**(a*x) + 1) – log(exp)*a*x)/(log(exp)*a)$ (4.202)

7: **df(ws,x);** (4.203)

(exp**(a*x) – 1)/(exp**(a*x) + 1)$ (4.204)

8: **int(((a+tan(x))**3),x);** (4.205)

(3*log(tan(x)**2 + 1)*a**2 – log(tan(x)**2 + 1) +

tan(x)**2 + 6*tan(x)*a + 2*a**3*x – 6*a*x)/2$ (4.206)

9: **df(ws,x);** (4.207)

tan(x)**3 + 3*tan(x)**2*a + 3*tan(x)*a**2 + a**3$ (4.208)

10: **int(((log(x))**2)/x**3,x);** (4.209)

– (2*log(x)**2 + 2*log(x) + 1)/(4*x**2)$ (4.210)

11: **df(ws,x);** (4.211)

log(x)**2/x**3$ (4.212)

Здесь в этих примерах проверка путем дифференцирования получаемых первообразных для возврата к исходным выражениям полностью совпадает соответственно между результатами (4.204), (4.212) и исходными выражениями (4.201) и (4.209), и отличается лишь раскрытием куба суммы в результате дифференцирования (4.208) по сравнению с исходным выражением (4.205).

Обратим ваше внимание, что здесь для нас важен не столько результат получаемого совпадения, сколько реализация самого методического принципа "*Подвергай все сомнению*", завещанного нам Декартом.

Он остается справедливым и в применении к ПК (а может быть и особенно к ним), ведь и системы аналитических вычислений могут ошибаться (правда, довольно редко, но с очень тяжелыми возможными последствиями).

Поэтому и сам ПК, и системы символьной математики это не идолы, как к ним, к сожалению, очень часто относятся. Они только очень мощные орудия и инструменты исследования, очень сильно расширяющие наши возможности: отличные помощники выполняемой нами умственной работы. Но ***проверка, осмысливание и анализ получаемых результатов всегда при этом остается нашей задачей.***

Если интеграл не берется REDUCE в аналитическом виде, то ответ содержит формальный интеграл в одной из двух форм:

- *введенный оператор INT* от соответствующих параметров без каких–либо изменений:

12: **int(((log(x))**p)/x**q,x);** (4.213)

int(log(x)**p/x**q,x)$ (4.214)

- *выражение, включающее интегралы (INT)* от некоторых других функций (иногда более сложных, чем исходные):

13: **int(tan(x)/sqrt(a**2-b**2*(tan(x))**2),x);** (4.215)

–int((sqrt(– tan(x)**2*b**2 + a**2)*tan(x))/(tan(x)**2*b**2 – a**2),X)$ (4.216)

Таким образом, результатом обработки системой выражений (4.213) и (4.215), содержащих оператор INT, является просто переписывание их в принятой для системы внутренней форме.

Однако нетрудно расширить возможности системы с помощью команды LET для задания правил интегрирования или дифференцирования функций, для которых операторы INT или DF в системе REDUCE не определены.

При этом также можно использовать возможность команды LET ставить в соответствие переменным или операторам различные сложные конструкции.

Выполнение такой конструкции, как уже отмечалось выше, начинается после обращения в ходе выполнения программы к соответствующей переменной или оператору.

Поэтому также очень удобно связать одну переменную с набором, например, правил дифференцирования или интегрирования необходимых выражений, определенных с помощью команды FOR ALL … LET, которые начнут действовать сразу после обращения к имени соответствующей переменной, а другую – с их отменой командой FOR ALL … CLEAR (см, например, аналогичные примеры (4.182) и (4.186) использования процедур громоздких тригонометрических подстановок).

Замечание. Если в подынтегральное выражение входят функции, отличные от перечисленных выше (от TAN, ATAN, EXP, LOG, DILOG, ERF), то система REDUCE может произвести вычисление, если производные от этих функций выражаются через уже известные функции. Эти входящие в подынтегральное выражение функции могут быть введены самим пользователем в виде некоторого оператора, при этом производные от этих функций через известные функции также должны быть заданы самим пользователем. Например, если ввести новую функцию, производная от которой для любых переменных будет удовлетворять условию:

14: **for all x let df(dlg(x),x) =log(x)/(x+1);** (4.217)

то можно будет брать интегралы от выражений, содержащих эту функцию:

15: **int(x*dlg(x),x);** (4.218)

```
(4*dlg(x)*x**2 – 4*dlg(x) – 2*log(x)*x**2 + 4*log(x)*x +
x**2 – 4*x)/8$                                          (4.219)
```

В заключение отметим, что в работе [9] содержится большое количество процедур, а в пособии [24] — программ, которые удобно использовать в различных задачах механики и математики. Если при этом вы встретите трудности, почувствовав недостаток теоретического багажа, то в монографии [7] наиболее подробно описаны возможности операторов и команд базовой версии REDUCE.

ЧАСТЬ 2. ПРАКТИЧЕСКАЯ РАБОТА НА ПК В СРЕДЕ MS-DOS

ГЛАВА 5. ОБЩЕЕ ОПИСАНИЕ ПК

Приведем необходимые сведения, достаточные для самостоятельной работы на IBM-совместимых ПК.

5.1. Описание клавиатуры

1. В центральном поле клавиатуры находятся ***алфавитно-цифровые и знаковые клавиши***, расположенные примерно как на пишущей машинке.

Закрепление символов за конкретными алфавитно-цифровыми клавишами называется *раскладкой* клавиатуры. Обычно на каждом компьютере действуют как минимум две раскладки: русскоязычная и англоязычная. Переключение между этими двумя раскладками выполняется нажатием специальной комбинации клавиш, которая задается при настройке операционной системы (ОС), установленной на компьютере.

На ПК, работающих в ОС Windows 95/98/Me, используются следующие комбинации: левая клавиша <Alt>+<Shift> или <Ctrl>+<Shift>.

Для набора комбинаций клавиш следует нажать одну из них и, не отпуская ее, нажать другую (или другие). Подобные совместные нажатия клавиш будут обозначаться с помощью знака "+" между ними.

В среде MS-DOS переключение шрифтов определяется использованием специальной программы — драйвера клавиатуры. В зависимости от его вида для переключения алфавитов может быть использованы:

- *клавиши* <Caps Lock>, <F11>, правые клавиши <Ctrl>, <Shift>;
- *одновременное нажатие обеих клавиш:* <Shift>+<Shift> (правой и левой), <Ctrl>+<Alt>, <Ctrl>+правый <Shift> (в этом случае <Ctrl>+левый <Shift> переключает клавиатуру в режим латинских букв) и т.п.

2. Верхний ряд (над центральным полем) содержит ***функциональные клавиши*** <F1>–<F12>, обычно разделенные на три группы по 4 клавиши в каждой. Поскольку каждая функциональная клавиша при нажатии передает уникальный (непечатаемый) код, прикладные программы могут присваивать им специальные значения: удобный вызов функций или последовательность обычно используемых операций.

Отметим, что функциональные клавиши могут выполнять разные задачи для различных прикладных программ. Однако действие клавиш <F1> и <F10> стало традиционным.

Клавиша <F1>, как правило, используется для вызова справок и дополнительной информации. Если программа имеет встроенную справочную систему, то обычно она вызывается нажатием клавиши <F1>.

Клавиша <F10> обычно используется для вызова меню программ. Во всех приложениях Windows, имеющих в верхней части рабочего окна строку меню, вход в эту строку можно выполнить нажатием клавиши <F10>.

3. Слева и справа центральное светлое поле обрамляют ***стандартные управляющие клавиши***. Рассмотрим их функциональное назначение, указывая через запятую основные модификации названий.

<Enter>, <↵> — ввод текущей строки и перевод курсора в начало следующей строки. Ввод каждой команды также должен оканчиваться нажатием клавиши <Enter>. Пока клавиша <Enter> не нажата, команда не исполняется и ее можно отменить или изменить. При работе с элементами управления (например, с пунктами экранных меню), нажатие этой клавиши также служит для исполнения команды, связанной с текущим выделенным пунктом.

<Esc> — сокращенное написание слова “Escape” (“Уйти”). Отменяет текущие символы командной строки, которая в некоторых случаях при этом стирается. Используется для выхода из текущего режима в прикладных программах. Этой клавишей можно также закрывать экранные меню и другие экранные элементы управления.

<Shift>, <↑> — смена регистра (на время нажатия клавиши). Обычно используются для ввода прописных (заглавных) букв и специальных символов верхнего регистра клавиатуры, для чего нужно нажать клавишу <Shift> и, не отпуская ее, нажать клавишу с обозначением соответствующей буквы или символа. Также используется для модификации действия других клавиш, которые используются совместно с ней.

<Caps Lock> — нажатие этой клавиши фиксирует клавиатуру в режиме прописных (заглавных) букв, о чем свидетельствует зажигание соответ-

ствующей лампочки на клавиатуре. Этот режим удобен при вводе заголовков или текста, состоящего из таких букв, так как теперь для их набора не требуется одновременного нажатия клавиши <Shift>.

Отметим, что при нажатии клавиши <Shift> в режиме заглавных букв одновременное нажатие любой буквенной клавиши приводит к появлению на экране соответствующей строчной буквы нижнего регистра.

При повторном нажатии клавиши <Caps Lock> происходит возврат к режиму строчных букв.

Действие клавиши <Caps Lock> охватывает только клавиши с изображением букв.

Для ввода же специальных символов верхнего регистра всегда нужно нажимать клавишу <Shift> совместно с соответствующей клавишей.

<Ctrl>, <Control> и <Alt> — ("Control" – "Управление" и "Alternate" – "Изменение", "Альтернатива") используются только совместно с другими клавишами, изменяя их действие.

Отметим, что клавиши <Shift>, <Ctrl> и <Alt> часто называют регистровыми клавишами. Они, как и клавиша <Enter>, используются наиболее часто. Поэтому они продублированы, имеют увеличенный размер и расположены в наиболее удобных местах клавиатуры.

<Tab> — (табуляция) перемещение курсора по строке на 8 позиций вправо на нижнем регистре и на 8 позиций влево на верхнем регистре (<Shift>+<Tab>).

За ней также закрепилось представление, как об управляющей клавише. Ее основной смысл состоит в *переключении*: между объектами, элементами управления, режимами работы программы и даже между программами. Так, например, в ОС Windows 95/98/Me переключение между задачами осуществляется нажатием клавиш <Alt>+<Tab>.

<Back Space>, <BS>, <Back sp>, <BKSP>, <←> (стрелка влево над клавишей <Enter>) — удаление символа слева от курсора с одновременным смещением курсора на одну позицию влево.

4. Группа из четырех клавиш, обозначенных стрелками, используется для перемещения курсора по тексту влево <←>, вверх <↑>, вниз <↓> и вправо <→>.

Они называются *клавишами управления курсором* или просто *курсорными клавишами*.

Курсор — это экранный символ, отмечающий на экране *позицию ввода*, то есть то место, в которое происходит ввод текста при наборе.

В графических операционных системах, таких как Windows 95/98/Me, клавишами управления курсором также можно управлять *указателем мыши*, который служит для выбора экранных объектов или элементов управления.

5. Над курсорными располагаются *клавиши редактирования*. Эта группа состоит из шести клавиш. Они используются в при работе с текстом в текстовых редакторах или процессорах как клавиши редактирования или как более мощные клавиши управления курсором.

<Insert>, <Ins> — переключение между двумя режимами ввода символов: вставка или замещение (замена). В режиме вставки на месте курсора появляется набираемый символ, сдвигающий всю остальную часть строки вправо. В режиме замены новые символы занимают место тех, которые существовали ранее. При вводе нового текста эти режимы одинаковы.

В некоторых программах, управляющих операциями с файлами (копирование, перемещение, удаление и т.п.), клавишу <Insert> используют для группового выделения файлов.

<Delete>, <Del> ("Вычеркивание") — удаление символа в той позиции, в которой находится курсор (при этом все символы справа от курсора сдвигаются на одну позицию влево для заполнения освободившегося места). В графических операционных системах, таких как Windows 95/98/Me, клавиша <Delete> используется для удаления предварительно выделенных объектов или групп объектов.

<Home> и <End> — быстрое перемещение курсора соответственно в начало или конец строки. Общеприняты также такие сочетания: <Ctrl>+<Home> и <Ctrl>+<End> — быстрое перемещение курсора соответственно в начало или конец документа.

<PageUp>, <PgUp> — переход на предыдущую экранную страницу изображаемого на экране текста.

<PageDown>, <PgDn> — переход на следующую страницу изображаемого на экране текста.

6. Малое поле в правой части клавиатуры, где расположены цифровые и управляющие клавиши, образуют *двухрежимную малую цифровую клавиатуру*, называемую также *дополнительной цифровой панелью*.

При включенном режиме NumLock (Цифровая блокировка) эта самая правая группа клавиш дублирует цифровые клавиши (о чем свидетельствует зажигание соответствующей лампочки на клавиатуре), а при выключенном — клавиши управления перемещения курсора. Переключение между режимами осуществляется нажатием клавиши <Num Lock>.

С помощью дополнительной цифровой панели можно также обеспечить ввод любых символов, для которых вообще не предусмотрены клавиши. Для этого при включенном режиме NumLock в любой программе следует войти в режим ввода текста, нажать клавишу <Alt> и, не отпуская ее, набрать на дополнительной цифровой панели код нужного символа. Его можно узнать для любых шрифтов из специальной стандартной программы Таблица символов.

Имена накопителей на дисках (дисководов)

Отметим, что НГМД в MS DOS имеют обычно имена A: и B:. Чтобы облегчить управление огромной памятью НЖМД (винчестера), ее для удобства работы подразделяют на несколько томов или разделов — условных дисков, каждый из которых имеет свою собственную метку наподобие дисковода, которым присваивают имена C:, D: и т.д.. При этом машина не делает различия между разделом накопителя на жестком диске и накопителем на дискетах.

5.2. Действия при “зависании” компьютера

Если в процессе работы ПК возникнет ситуация, что компьютер не реагирует на нажатия клавиш или реагирует на них неадекватно, то предварительно желательно убедиться в том, что компьютер действительно завис. Для определения состояния ПК можно применять нажатие клавиш <Num Lock> или <Caps Lock>: если оно сопровождается сменой состояния соответствующей лампочки, то компьютер реагирует на нажатие клавиш и он не завис. Он может выполнять весьма длительную операцию или работать с очень медленным источником информации. Поэтому ваши действия должны быть очень осторожными. При прерывании операции вы можете потерять не сохраненную на носителе информацию или испортить сохраненную, если при зависании компьютер производит работу с накопителем.

Убедившись в том, что ПК действительно “завис”, нужно постараться выйти из этого состояния одним из следующих возрастающих по силе действий:

1. Нажать <Esc>.
2. Воспользоваться комбинацией <Ctrl>+<Pause Break>, следствием чего должно быть прекращение работы выполняемой команды или программы и перевод ПК в режим ожидания ввода команды с клавиатуры.

3. Произвести перезагрузку, нажав одновременно <Ctrl>+<Alt>+<Del>.

   При работе в Windows 95/98/Me при этом только откроется диалоговое окно Завершение работы программы (Close Program) со списком выполняемых в настоящее время программ. Те из них, которые не реагируют ни на какие обращения. будут иметь пометку [Не отвечает] ([Not responding]):

   - поочередно закройте каждую такую программу, выделив ее щелчком мыши и нажав кнопку Завершить задачу (End Task);
   - дождитесь открытия диалогового окна, в котором еще раз нажмите такую же кнопку Завершить задачу (End Task), подтвердив свое намерение закрыть программу. При этом вы потеряете все несохраненные в этой программе данные. Однако есть вероятность того, что после завершения всех отказавших программ продолжение работы на компьютере станет возможным.

   Если после закрытия всех отказавших приложений работоспособность компьютера не восстановлена, то, вновь открыв диалоговое окно Завершение работы программы (Close Program), придется предпринять одну из двух более жестких мер, ведущих к потере всех несохраненных данных:

   - нажать кнопку Завершить работу (Shut Down), чтобы завершить работу на компьютере, после чего снова загрузить Windows 95/98/Me;
   - еще раз нажать клавиши <Ctrl>+<Alt>+<Del>, чтобы перезагрузить систему.

4. Если ни на одно из этих действий компьютер не реагирует, то следует нажать кнопку <Reset> на корпусе системного блока.
5. Выключить компьютер, а затем включить снова.

При выведении компьютера из состояния “зависания” следует также помнить следующее правило. Если при подозрении на зависание ПК наблюдается интенсивное обращение к дисковым накопителям (что видно по свечению лампочек на дисководах и лампочки жесткого диска на корпусе компьютера), то выполнение п.п. 3, 4 и 5 нежелательно, так как может быть повреждена информация на магнитном диске (гибком или жестком).

Если проблемы постоянно вызывает одно и то же приложение, переустановите его. Когда нарушения в работе системы наблюдаются достаточно часто, переустановите Windows 95/98/Me. Надежнее всего — в новую папку с последующей переустановкой всех приложений.

ГЛАВА 6. ОСНОВЫ ОПЕРАЦИОННОЙ СИСТЕМЫ MS-DOS

Операционная система (ОС) служит для создания среды, в которой происходит диалог пользователя с компьютером. Она представляет собой совокупность программ, осуществляющих перевод вводимых команд на язык, понятный для ПК.

6.1. Основные термины и понятия

Файл (от слова "File" — "Массив") — это именованный упорядоченный массив однородной информации, размещенной на внешнем носителе (жестком или гибком диске, магнитной ленте), имеющей определенное функциональное назначение и воспринимаемой операционной системой как единое целое. В виде файлов на диске хранится вся информация. Файлом может быть текст программы на алгоритмическом языке, например, Фортран; набор исходных данных; объектный модуль программы, пригодный для ее понимания ЭВМ и т.п.

Имя файла состоит обычно из собственно имени и расширения, часто называемого типом файла, разделенных между собой точкой. Имя ИФ и тип файла ТФ могут содержать от 1 до 8 и от 0 до 3 символов соответственно, в качестве которых могут использоваться алфавитно-цифровые и некоторые специальные символы.

Наличие расширения в полном имени файла не является обязательным, однако оно обычно используется для описания типа информации, записанной в файле, т.е. для описания типа файла. Обычно файлы, относящиеся к одной основной программе, имеют одинаковые имена, но разные типы. Для некоторых типов информации обычно применяют стандартные или типовые расширения:

.BAT — командный файл DOS для пакетной обработки, представляющий собой последовательность команд, выполняемых автоматически;

.EXE — перемещаемая программа, готовая к выполнению под управлением DOS;

.COM — команда или программа в машинном коде, пригодные для непосредственного выполнения под управлением DOS (командный файл);

.DAT — файл данных;

.SYS — системные файлы, являющиеся частью операционной системы и используемые для расширения ее возможностей (например, драйверы), и др.

Каталог (Directory) — это таблица содержимого диска или его части, в которой находится список имен файлов, объединенных по некоторому критерию, с указанием их атрибутов (размера, даты и времени последней зарегистрированной модификации каждого файла). Каталоги файлов создаются для удобства организации файловых систем, так как каждый каталог можно рассматривать как раздел внешней памяти, с содержимым которого можно работать достаточно независимо. При создании каждого файла его имя, тип и другие атрибуты записываются в каком-то каталоге. При этом говорят, что файл находится в данном каталоге. Для создания нового каталога на диске в DOS используется команда *MKDIR* (или *MD*).

Имена каталогов подчиняются тем же требованиям, что и имена файлов, однако возможность расширения имени для каталогов обычно не используется. Имена каталогов указываются в оглавлении большими буквами, тогда как имена файлов маленькими.

Корневой каталог, не имеющий имени, всегда имеется на каждом магнитном диске. В нем регистрируются файлы и каталоги (называемые в этом случае обычно подкаталогами). Корневой каталог обозначается с помощью указателя дисковода, за которым следует обратная наклонная черта "\" (в некоторых версиях DOS вместо знака "\" используется знак "/").

Например, A:\ — корневой каталог гибкого диска на дисководе A, C:\ — корневой каталог жесткого диска (или его раздела — условного диска), расположенного на дисководе C.

Многоуровневые каталоги. В каждом подкаталоге (или каталоге 1-го уровня) корневого каталога, кроме находящихся в нем файлов, также можно создать произвольное количество других подкаталогов или каталогов 2-го уровня, для которых каталог 1-го уровня будет надкаталогом или родительским каталогом. В каждом созданном подкаталоге (или каталоге 2-го уровня) также можно создать произвольное количество других подкаталогов или каталогов 3-го уровня и т.д.

Каждый подкаталог представляет собой обычный файл, содержащий список имен относящихся к нему других файлов, часть которых (или все) также может быть в свою очередь подкаталогами, содержащими соответственно свои файлы и другие подкаталоги.

Цепочки включенных друг в друга каталогов обозначаются их именами, разделяемыми знаком обратной наклонной черты "\". Если знак "\" стоит перед первым именем каталога, значит выше "по иерархии" находится только корневой каталог данного диска. В этом случае знак "\" как бы отделяет несуществующее имя корневого каталога от имени каталога первого уровня, подчиненного корневому. Примеры цепочек подчиненных каталогов: \P-R — каталог P-R находится на 1-м уровне; \P-R\UPR — каталог UPR находится на 2-м уровне; \P-R\UPR\STATIKA — каталог STATIKA находится на 3-м уровне.

Так образуется иерархическая древовидная файловая структура, называемая многоуровневой системой каталогов. Слово "иерархическая" ни в каком случае не следует понимать буквально. Ни одному каталогу не присваивается приоритет по отношению к другому. Каждый каталог любого уровня также можно рассматривать как раздел внешней памяти, с содержанием которого можно работать абсолютно независимо, поэтому употребляемые часто обозначения "подкаталог", "надкаталог" или "родительский каталог" имеют только значение для указания их относительного расположения.

Текущий или рабочий каталог. Каталог (или подкаталог), в котором вы находитесь, называется текущим или рабочим каталогом. В каждый данный момент во внимание принимается только рабочий или текущий каталог, который и становится главным на время работы в нем. Когда вы создаете файл или подкаталог, он помещается в текущий каталог. Можно представить, что каждый диск разделен на столько секций, сколько в нем подкаталогов. Командам доступны только файлы, содержащиеся в текущем подкаталоге, чем достигается возможность независимой работы разных пользователей. В текущем подкаталоге также должны быть расположены все файлы и подпрограммы, необходимые для работы имеющихся там же программ (или в команде *PATH* должны быть указаны пути доступа к соответствующим каталогам, в которых содержатся необходимые файлы).

Пути и полные имена файлов. Принцип многоуровневой организации памяти на диске приводит к тому, что для указания файла уже недостаточно указать только его имя с соответствующим расширением: одноименные (и притом различные по содержанию) файлы могут находиться в нескольких каталогах. Для точной идентификации файла необходимо указать механизм определения местоположения файла в многоуровневой системе каталогов, в качестве чего указывается путь доступа к файлу. Он представляет собой маршрут по именам каталогов, содержащий последовательный указатель перечня имен каталогов на пути к файлу, разделенных между собой символом "\".

Путь может начинаться или от корневого каталога, или от текущего. В зависимости от этого маршрут поиска называется соответственно абсолютным или относительным. Если путь доступа начинается с обратной косой черты "\", то DOS осуществляет поиск файла, начиная с корневого каталога. В противном случае она просматривает указанный путь в поисках файла, начиная от текущего (рабочего) каталога.

Путь доступа к файлу PR2-1.LIS, находящемуся в каталоге 3-го уровня STATIKA, из корневого каталога можно указать так:

```
\P-R\UPR\STATIKA\pr2-1.lis                    (6.1)
```

Если файл расположен на жестком диске (дисковод С), а текущим является дисковод А, то полное описание файла, требуемое для доступа к файлу PR2-1.LIS, выглядит следующим образом:

```
C:\P-R\UPR\STATIKA\pr2-1.lis                  (6.2)
```

Путь с включенным в него именем файла полностью идентифицирует расположение файла на диске и образует полное описание файла или идентификатор файла (ИДФ). Отметим, что имя файла также должно быть отделено обратной косой чертой от последнего имени подкаталога. Полное описание файла имеет следующий формат:

```
[диск:][путь\]имя_файла[.расширение]          (6.3)
```

Здесь элементы, заключенные в квадратные скобки, являются необязательными. В этом случае, если в описании не указывается дисковод или путь, то подразумевается текущий дисковод или текущий каталог.

Так в примере (6.1) описатель дисковода не указан, так как текущим является дисковод С и маршрут поиска указывается от корневого каталога.

При работе в текущем каталоге STATIKA идентификатором находящегося в нем файла ИДФ будет только имя файла с его расширением (если оно имеется, в нашем примере PR2-1.LIS).

Последняя возможность очень удобна. Поэтому всегда в дальнейшем будем подразумевать, что работаем с файлами, находящимися в текущем каталоге.

Подстановочные символы "*" и "?" разрешается использовать в обозначениях имен файлов и их расширений для обозначения сразу нескольких файлов, что называется шаблоном имени файла ("групповым", "родовым" именем или обозначением имени по маске).

Знак "*", встречаемый в имени или типе файла, обозначает любое число произвольных символов, которые допускаются в именах и типах фай-

лов, количество которых ограничено диапазоном от нуля (т.е. отсутствует вовсе) до значения, дополняющего число указанных в шаблоне символов до их максимально допустимого значения (8 для имени и 3 для типа).

Вопросительный знак "?", встречающийся в имени или типе файла, соответствует любому одному произвольному допустимому символу или его отсутствию.

Примеры шаблонов: `ABC*.D*` — определяет подмножество файлов из существующих на диске в текущем каталоге, имена которых начинаются с ABC и имеют длину от 3-х до 8-ми символов, и расширениями, начинающимися с D и имеющими длину от одного до 3-х символов;

`*.` — все файлы текущего каталога, у которых отсутствует тип;

`A??B.*` — все файлы текущего каталога, начинающиеся с A, заканчивающиеся B и имеющие длину от 2-х до 4-х символов.

Обратим внимание, что DOS *игнорирует все символы в имени и типе файла после подстановочного символа звездочки,* поэтому обозначения `*abc.*d` и `*.*` трактуются одинаково, как все файлы текущего каталога.

Командные файлы. Для ввода часто повторяющейся последовательности команд используются специальные файлы, имеющие расширение .bat и называемые командными файлами. Нужная последовательность команд помещается в образовываемый командный файл с помощью любого текстового редактора. Она будет выполнена после ввода из командной строки имени этого файла (при этом его расширение можно не указывать).

Эта процедура называется пакетной обработкой и позволяет использовать командные файлы для создания собственных команд, что широко используется при трансляции программ, их сборке и т.п. (см. дополнение 3.4).

Особую роль для создания удобной рабочей обстановки для пользователя ПК играют два специальных командных файла *config.sys* и *autoexec.bat.* Данные файлы должны находиться в корневом каталоге системного диска. Они отрабатываются каждый раз при любом включении или перезагрузке системы и производят конфигурирование и начальную настройку системы.

С помощью файла конфигурации *config.sys* можно расширять операционную систему, добавлять новые внешние устройства или нестандартно использовать имеющиеся. Эти программы, называемые драйверами внешних устройств, можно включить в систему при загрузке ОС, инициализировав их из файла *config.sys*.

Файл автозапуска *autoexec.bat* содержит набор последовательных команд и является обычным пакетным командным файлом, выполняющем не-

обходимую настройку системы. Например, можно включить в приглашение DOS указание на текущий каталог, задавать путь поиска внешних команд с помощью команды *PATH* и т.д. В списке каталогов, задаваемых командой *PATH*, обычно перечисляются через точку с запятой те каталоги, в которых находятся исполняемые программы общего назначения. Рекомендуется [25] имена каталогов в команде *PATH* указывать полностью от корневого каталога соответствующего диска, что позволит командному процессору DOS правильно их находить из любого текущего каталога и диска (см., например, команду *PATH* (3.30)).

6.2. Основные сведения о командах DOS

6.2.1. Соглашения об обозначениях форматов команд

Здесь и далее по тексту при записи форматов команд приняты следующие правила:

- ключевые слова, записанные латинскими буквами, должны использоваться в команде без изменений и сокращений;
- параметры, записанные русскими буквами, должны быть заменены необходимыми конкретными значениями в соответствии с описанием команды;
- атрибуты команды или часть ее имени, заключенные в квадратные скобки "[...]", являются необязательными элементами и могут быть опущены при использовании команды;
- фигурные скобки "{...}", ограничивающие собой какое-либо используемое обозначение или его часть, даже при использовании латинского алфавита указывают на необходимость его замены во всех случаях соответствующими конкретными значениями. Сами символы фигурных скобок указывают только на необходимость замены заключенных в них выражений в формате команды при ее описании их конкретными значениями, поэтому в реальной команде они не указываются;
- сокращения используются для указания в форматах команд часто встречающихся переменных значений, например: ИДФ — идентификатор файла, ИФ — имя файла, ТФ — тип файла;
- группа слов, определяющих одну переменную, связывается символом "_".

6.2.2. Структура команд DOS

Будем считать все командные файлы находящимися в каталоге DOS диска C, путь к которому указан в команде *PATH*, включенной в состав файла autoexec.bat.

Тогда для запуска любой команды достаточно будет указать только имя соответствующего файла, обозначаемого нами далее как ИМЯ_КОМАНДЫ.

Теперь формат любой команды DOS в общем случае можно представить в виде:

```
ИМЯ_КОМАНДЫ [ОПЕРАНДЫ] [/РЕЖИМЫ]                    (6.4)
```

где ОПЕРАНДЫ и РЕЖИМЫ (или ОПЦИИ) могут включать в себя следующее: идентификатор файла, над которым данной командой совершается определенное действие, аргумент и переключатель.

Идентификатор файла ИДФ (6.3) следует в команде сразу за указанием ее имени и относится к числу позиционных операндов, которые должны задаваться в определенной последовательности.

Будем считать все обрабатываемые файлы (т.е. те, с которыми нам приходится работать) находящимися в текущем каталоге текущего диска.

Тогда идентификатором файла {ИДФ}, однозначно определяющим его местоположение, будет только имя файла (при необходимости с указанием соответствующего типа), а диск и путь можно будет не указывать.

Опция "аргумент" предоставляет команде некоторую дополнительную информацию.

Обычно приходится выбирать один из нескольких вариантов значений аргумента, например, `ON (ВКЛ)` или `OFF (ВЫКЛ)`, который также указывается после имени команды и относится к числу позиционных операндов.

Режимы выполнения команды, называемые также переключателем, начинаются с символа косой черты "/", за которой сразу без пробела указывается соответствующий режим.

Они представляют собой ключевые операнды и могут указываться в произвольной последовательности, например, `/f/g/q/s`.

Часть их принимается по умолчанию, поэтому указание режимов при обычном действии команды необязательно.

Все вводимые команды печатаются в командной строке после соответствующего приглашения DOS и после нажатия клавиши <Enter> вместе с ним вводятся в ПК.

6.3. Основные команды DOS

Использующему ПК для практической работы с применением сервисных программ желательны понятия о следующих наиболее часто встречающихся командах:

1) создание нового каталога на диске (*MKDIR* или *MD*);

2) переход в другой каталог (*CHDIR* или *CD*);

3) удаление каталога с диска (*RMDIR* или *RD*);

4) просмотр дерева подкаталогов на диске (*TREE*);

5) просмотр содержимого файла (*TYPE*);

6) копирование файлов (*COPY*);

7) удаление файлов из каталога (*DEL* или *DELETE*);

8) изменение имени файла (*REN* или *RENAME*);

9) подготовка дискеты к использованию (*FORMAT*);

10) копирование с дискеты на дискету (*DISKCOPY*).

Команды 1) – 4) предназначены для работы с каталогами, 5) – 8) для работы с файлами, а 9) – 10) представляют собой команды для работы с дисками:

Команда *MKDIR* или ***MD*** производит создание нового каталога на диске. Формат команды *MD*:

```
MD [диск:][путь\]имя_каталога                    (6.5)
```

Создание нового каталога может быть произведено в любом уже существующем каталоге на указанном [диск:] или текущем диске в указанном [путь\] или текущем каталоге.

Если вы в команде *MD* не указываете [диск:], то каталог создается на текущем диске по указанному пути. Если не указан и путь: *MD имя_каталога*, то каталог с указанным именем создается в текущем каталоге (в котором вы находитесь) и будет являться его подкаталогом.

Команда *CHDIR (CD)* выводит на экран имя текущего каталога или осуществляет переход в другой каталог (т.е. изменяет текущий каталог). Формат команды *CD*:

```
CD [диск:][путь\]имя_каталога                    (6.6)
```

При переходе из текущего каталога в подчиненный в команде *CD* достаточно указать его имя. Для перехода в каталог предыдущего уровня [родительский каталог] используются символы ".." [две точки]: *CD .. *, а для перехода в корневой каталог текущего диска достаточно ввести команду *CD *.

Команда *RMDIR (RD)* удаляет указанный каталог или подкаталог. Формат использования команды:

```
RD [диск:][путь\]имя_каталога                    (6.7)
```

Перед удалением каталога (или подкаталога) нужно удалить в нем все файлы и подкаталоги (он должен быть пуст). Нельзя использовать *RMDIR* для удаления текущего каталога (нужно перейти сначала в другой каталог).

Команда *TREE* графически показывает на экране структуру каталога. Формат использования команды:

```
TREE [диск:][путь\] [/F]                         (6.8)
```

Параметр [диск:][путь\] задает диск и маршрут, для которого вы хотите вывести структуру каталога. Параметр `/F` выводит имена файлов в каждом каталоге. Для вывода имен всех подкаталогов на текущем диске дайте команду *TREE *.

Команда *TYPE* используется для вывода на экран содержимого указанного текстового файла. Вывод можно приостановить нажатием клавиш <Ctrl>+<S> и продолжить их повторным нажатием. Формат использования команды:

```
TYPE [диск:][путь\]имя_файла                     (6.9)
```

Команда *COPY* осуществляет копирование файлов. Формат использования команды:

```
COPY ИДФ1 [ИДФ2] [/ключ]                         (6.10)
```

Здесь `ИДФ1` задает источник копирования: имя файла (или файлов — при этом допускается использование шаблона) или имя каталога, если необходимо скопировать все файлы каталога.

Если `ИДФ2` — назначение — не задано, то файлы копируются в текущий каталог с сохранением имен. Если `ИДФ2` — имя каталога, то файлы копируются в указанный каталог также с сохранением имен. Если `ИДФ2` — имя файла, то источник копируется в файл с указанным именем.

Например, команда *COPY* stat9n.dat stat9n.d22 позволяет создать копию существующего файла stat9n.dat под идентификатором stat9n.d22 в том же текущем каталоге.

Обратим внимание, что команда *COPY* всегда выполняется в режиме замены существующего файла (если он есть на диске) выводным файлом, если у них одинаковые идентификаторы.

В командах DOS значения неуказанных операндов или их частей выбираются системой принимаемыми по умолчанию. Поэтому для копирования всех файлов с жесткого диска C на A без изменения имен и типов файлов достаточно ввести *COPY C: A:*.

При копировании допускается использование логических устройств в качестве источника или назначения копирования, например:

CON — консоль (клавиатура — при вводе, дисплей — при выводе), что позволяет непосредственно вводить в нужный файл необходимую информацию с клавиатуры (*COPY CON* имя_файла) или получать вывод файла на экран (*COPY* имя_файла *CON*).

Отметим, что ввод информации в файл осуществляется построчно после ее набора в командной строке и нажатии клавиши <Enter>, а символ признака конца создаваемого файла (код 26) записывается в файл при одновременном нажатии клавиш <Ctrl>+<Z> (на экране этот символ отображается в виде ^ Z);

PRN — принтер (только как выходной файл):

```
COPY имя_файла PRN                                    (6.11)
```

Последняя возможность позволяет удобно распечатывать файлы с использованием оболочки Norton Commander (см. п. 7.1), что обычно удобнее применения команды *PRINT*.

Команду *COPY* можно также использовать для комбинирования файлов. В этом случае формат использования команды имеет вид:

```
COPY ИДФ1 [+ИДФ2+...] [ИДФ]                           (6.12)
```

Источник представляет собой перечень файлов `ИДФ1 [+ИДФ2+...]`, соединенных знаком “+” (плюс), а приемником (или файлом назначения) является идентификатор выводного файла `[ИДФ]`.

Чтобы скопировать несколько файлов в один, перечислите в источнике любое число файлов (разделив их плюсом) и задайте имя выводного файла: *COPY* stat.d1+stat.d2+stat.d3 stat.dsu.

При этом файлы текущего диска и каталога stat.d1, stat.d2 и stat.d3 объединяются и помещаются в файл stat.dsu (также в текущем каталоге).

Допускается использование подстановочных символов. Команда *COPY* stat.d* stat.dsu комбинирует все файлы в текущем каталоге с именем stat и расширениями, начинающимися с d и имеющими длину от одного до 3-х символов, в один файл stat.dsu.

Команда *DEL* или *DELETE* удаляет файл имя_файла из текущего или указанного каталога. Формат использования команды:

```
DEL [диск:][путь\]имя_файла [/P]                      (6.13)
```

Ключ /P перед удалением файла выводит запрос на подтверждение. Например, чтобы удалить файл stat9n.dat из текущего каталога, вы можете воспользоваться командой: *DEL* stat9n.dat.

Команда *REN* или ***RENAME*** изменяет имена заданных файлов (файла). Не допускается применять команду *REN* для переименования файлов с указанием другого диска или для перемещения файлов в другой каталог. Формат использования команды:

```
REN [диск:][путь\]имя_файла1 имя_файла2          (6.14)
```

Параметр `[диск:][путь\]имя_файла1` задает расположение файла или набора файлов, которые нужно переименовать. Параметр "`имя_файла2`" задает новое имя файла (или новые имена файлов при использовании шаблона). Новый диск и маршрут вы указать не можете.

Например, команда *REN* stat9n.dat stat9n.d22 позволяет изменить имя существующего файла stat9n.dat на stat9n.d22 без изменения содержания в том же текущем каталоге.

Если файл "`имя_файла2`" уже существует, *REN* работать не будет, и выводится аварийное сообщение: "Duplicate file name or file not found" ("Имя файла дублируется, или файл не найден").

Команда *FORMAT* производит подготовку (специальную разметку) гибких дисков [диск:] к дальнейшему использованию. Если на диске была записана какая-либо информация, то после форматирования она будет безвозвратно потеряна. Основной формат команды:

```
FORMAT диск: [/ключ]                             (6.15)
```

Команда *FORMAT* (например, *FORMAT* A:) по типу дисковода определяет нужные параметры и на какую емкость будет отформатирован диск на данном дисководе. В команде имеются многочисленные ключи, позволяющие изменять параметры форматирования по умолчанию (обычно в сторону их увеличения) после отработки соответствующих драйверов (например, 800.COM или 900.COM). Значения параметров команды *FORMAT* можно узнать после запуска драйвера 900.com в режиме запроса: 900.com /?. Перед использованием таких нестандартно отформатированных дискет также должен отрабатываться соответствующий драйвер (если его запуск не включен в файл *autoexec.bat*).

Команда *DISKCOPY* [диск1:] [диск2:] производит копирование (по дорожкам) всей информации с указанного [диск1:] на [диск2:]. Если копирование производится на одном и том же устройстве (*DISK COPY* A: A:), то на экран выводятся сообщения о необходимости неоднократной замены гибких дисков. Оба диска должны быть одного и того же формата, т.е. содержать одинаковое количество дорожек и секторов. Если диск, на который производится копирование, не форматирован, то он автоматически форматируется с параметрами диска источника.

ГЛАВА 7. ПРАКТИЧЕСКАЯ РАБОТА НА ПК С ИСПОЛЬЗОВАНИЕМ NORTON COMMANDER

Для удобства практической работы в DOS разработаны специальные операционные программы-оболочки, позволяющие существенно увеличить простоту и наглядность выполнения большинства наиболее употребительных ее команд. Вся предыдущая глава нужна в основном для понимания того, что будут делать эти программы-оболочки, а также для оценки достоинств работы с ними. К наиболее популярным сервисным программам относится Norton Commander.

7.1. Краткие сведения о Norton Commander (NC)

Запуск Norton Commander осуществляется набором в командной строке NC. Эта команда обычно включается в файл автозапуска AUTOEXEC.BAT, в результате чего после загрузки DOS пользователь сразу оказывается в среде Norton Commander.

Экран при этом оказывается разделенным на два окна, с каждым из которых пользователь может работать независимо. Переход курсора из одного окна в другое осуществляется с помощью клавиши табуляции <Tab>. Эти окна обычно называют панелями и в них содержатся оглавления каталогов, имена которых (и пути доступа к ним), изображаются сверху каждой панели.

Одна строка в каждом окне содержит необходимую информацию о каком-либо файле или каталоге. При этом первая колонка символов соответствует имени и типу файла, вторая — размеру файла.

У подкаталога во второй колонке находится указатель SUB-DIR. Имена файлов в оглавлении каталога выводятся строчными буквами, а подкаталоги для их отличия — заглавными.

Обычно сначала выводятся сведения о подкаталогах, а затем — о файлах. В оглавлении подкаталога самую верхнюю строку занимает ссылка на родительский каталог в виде многоточия "..", справа от которых изображается UP-DIR.

Подсвеченный (выделенный) файл или каталог

Будем называть подсвеченным (временно выделенным) такой файл или каталог, имя которого на экране выделено серым цветом (на монохромном дисплее — инверсным изображением) с помощью полосы выделения. Иногда говорят, что на этом имени стоит указатель, оно выбрано и называют это выделение курсором.

С помощью клавиш перемещения курсора можно передвигать выделенный участок по панели в пределах текущего каталога, имя которого выведено сверху панели. Нажав клавишу табуляции, можно перевести выделенный участок в другую панель Norton Commander, т.е. сделать текущим находящийся там каталог. С помощью мыши можно подсветить (выделить) любой файл, просто щелкнув на его названии в панели.

Если нажать клавишу <Alt> и, не отпуская ее, начать набирать первые буквы имени файла, то Norton Commander выделит нужный файл в текущем каталоге, когда будет введено достаточное для его отличия количество букв.

Если выполнить набор имени файла после нажатия клавиш <Alt>+<F7>, то поиск файла будет проведен во всех каталогах текущего диска. В имени файла можно использовать шаблоны "*" и "?". Если будет найдено несколько файлов, то клавишами вертикального перемещения курсора среди найденных необходимо выделить нужный файл. Затем следует в меню выделить ChDir и нажать <Enter> для перехода в тот каталог, где находится нужный файл.

Если выделить файл с расширением .COM, .EXE или .BAT и нажать <Enter>, то начнется выполнение этого файла.

Выделенная группа файлов

Постоянное выделение файла (т.е. помещение его в группу выделенных) осуществляется нажатием клавиши <Insert>. Для снятия выделения файла следует также нажать <Insert>.

Для выделения группы файлов можно также нажать <+> (плюс на функциональной клавиатуре), называемый для сокращения "серый плюс", и задать шаблон для выбора имен файлов. В нем можно использовать подстановочные символы "*" и "?", смысл которых тот же, что и в командах DOS. После нажатия клавиши <Enter> выделится группа файлов, имена которых соответствуют указанному шаблону (при вводе шаблона, установленного по умолчанию *.*, выделится весь подкаталог). Для снятия выделения у этой группы файлов нужно нажать <–>, называемый для сокращения "серый минус", (минус на функциональной клавиатуре) и ввести тот же шаблон.

Работа с дисками и каталогами

Для перехода на другой диск следует нажать <Alt>+<F1> для левой панели или <Alt>+<F2> для правой панели.

Затем клавишами горизонтального перемещения курсора следует выбрать из появившегося списка доступных дисков нужный и нажать <Enter>.

Для перехода в содержащийся в оглавлении подкаталог нужно совместить курсор с его именем и нажать клавишу <Enter>. Norton Commander "войдет" в этот каталог и выведет его оглавление.

Для перехода в родительский каталог надо выделить ".." и нажать <Enter> или при любом положении курсора нажать одновременно <Ctrl>+<PageUp>.

Для перехода в другой каталог на том же диске следует нажать комбинацию клавиш <Alt>+<F10>, выделить клавишами перемещения курсора среди появившихся на экране нужный каталог и нажать <Enter>.

Командная строка

Ниже панелей располагается командная строка с приглашением DOS, в которой можно вводить обычные команды или запускать программы. После их выполнения происходит возвращение в среду Norton Commander к ее панелям.

Снятие или возвращение панелей, осуществляемое для просмотра выводимых на экран результатов выполнения команд, достигается совместным нажатием клавиш <Ctrl>+<O>.

Для вывода в командную строку имени выделенного файла на панели следует нажать <Ctrl>+<Enter>, а для очистки командной строки — <Esc>.

Введенные команды запоминаются в памяти в виде последовательного списка. Нажатием клавиш <Ctrl>+<E> или <Ctrl>+<X> в командную строку каждый раз выводится команда, которая была введена соответственно перед или после ранее выполненной команды в списке.

Выбранная нужная команда после необходимых изменений может быть отправлена на выполнение.

Для выполнения одной из ранее введенных команд без всяких изменений следует нажать <Alt>+<F8>, выделить с помощью клавиш перемещения курсора в появившемся на экране списке нужную команду и нажать <Enter>.

Назначение функциональных клавиш

<F1> — выводит подсказку;

<F2> — вызывает меню пользователя;

<F3> — просмотр содержимого файла (на нем должен стоять указатель);

<F4> — редактирование текстового файла (на нем должен стоять указатель);

<F5> — копирование файла, директории или группы файлов;

<F6> — переименование или пересылка файлов, директорий или групп файлов;

<F7> — создание поддиректорий;

<F8> — удаление файлов, директорий или групп файлов;

<F9> — выводит в верхней части экрана управляющее меню, позволяющее устанавливать режимы работы и производить некоторые другие действия;

<F10> — выход из Norton Commander.

7.2. Norton Commander подробнее

7.2.1. Управление панелями и запуск файлов

Рассмотрим подробнее управление панелями Norton Commander и работу с файлами. На экране обычно отображаются две панели (левая и правая). Вверху расположена строка заголовка, указывающая полный путь к директории, отображаемой в панели, ниже отображается имя диска и подзаголовки NAME (Имя). Далее следует список директорий (большие буквы) и файлов (маленькие буквы), причем если он не помещается в одном столбце, то его продолжение выводится в следующем столбце и т.д. У файлов с атрибутами "системный" и (или) "скрытый" между именем и расширением выводится символ " ".

Если выводимая директория не корневая, то в начале списка помещается символ "..", позволяющий при совмещении с ним подсветки курсора и нажатии клавиши <Enter> перейти в наддиректорию данной директории. Внизу расположена информационная строка, в которой выводится имя указываемого файла, его размер, дата и время создания (для директории только имя, дата и время создания), или количество и суммарный объем всех выделенных файлов.

Активная панель. управление указателем

Одна из панелей всегда является активной, у нее заголовок (полный путь) выделен инверсным цветом (светлый фон, темные символы). Если в данной директории есть хотя бы один файл или директория (включая указатель на наддиректорию), то одно из имен также выделено инверсным цветом (иногда говорят, что на этом имени стоит указатель).

Указатель можно двигать. Для этого используются клавиши управления курсором (<↓>, <↑>, <→>, <←>), а также следующие клавиши редактирования:

<Home> — перемещает указатель на первую позицию в списке;

<End> — перемещает указатель на последнюю позицию в списке;

<PgUp> — перемещает указатель на экран вверх (используется в тех случаях, когда все файлы не могут из-за своей многочисленности поместиться на экране, так что конец или (и) начало списка оказывается как бы за экраном);

<PgDn> — перемещает указатель на экран вниз.

Действия при нажатии клавиши <Enter>

Действие Norton Commander при нажатии пользователем клавиши <Enter> зависит от того, имеется что-либо в командной строке внизу экрана или нет.

Если командная строка не пуста, то при нажатии клавиши <Enter> будет выполнена команда, содержащаяся в командной строке.

Если же командная строка пуста, то действие Norton Commander зависит от того, что выделено на панели:

- если указатель стоит на имени директории, то мы войдем в эту директорию;
- если указатель стоит на символе "..", то мы выйдем из текущей директории в ее наддиректорию;
- если указатель стоит на имени выполняемого файла (с расширением .com или .exe), то произойдет его запуск;
- если указатель стоит на имени пакетного файла (с расширением .bat), то начнется его выполнение;
- если нажать комбинацию клавиш <Ctrl>+<Enter>, то имя, на котором стоит указатель, будет помещено в командную строку, однако никаких действий выполняться не будет. Данная возможность используется в том случае, когда программу необходимо запустить с параметрами. Вы просто добавляете недостающие параметры и нажимаете <Enter>.

Вместо нажатия <Enter> можно дважды щелкнуть мышью. В этом случае вышеуказанные действия будут выполнены независимо от того, содержится что-либо в командной строке внизу экрана или нет.

Приведем сводный список команд управления панелями и работы с файлами.

Управление панелями

<Tab> — осуществляет переход между панелями, при этом активной становится другая панель (естественно, происходит и смена текущей поддиректории);

<Ctrl>+<U> — поменять панели местами;

<Ctrl>+<L> — вывести/убрать в неактивной панели сводную информацию о диске и директории активной панели;

<Ctrl>+<Q> — вывести/убрать в неактивной панели режим быстрого просмотра содержимого файла, указанного на активной панели;

<Ctrl>+<Z> — вывести/убрать в неактивной панели сводную информацию о директории или файле, указанном в активной панели.

Для просмотра результатов выполнения программы, скрытых панелями Norton Commander, можно воспользоваться следующими комбинациями клавиш:

<Ctrl>+<O> — убрать/вывести панели на экран (при первом нажатии панели убираются с экрана, при повторном нажатии появляются обратно);

<Ctrl>+<P> — убрать/вывести не текущую панель на экран;

<Ctrl>+<F1> — убрать/вывести левую панель на экран;

<Ctrl>+<F2> — убрать/вывести правую панель на экран.

Работа с файлами

<Enter> — осуществляет запуск указанного файла или вхождение в указанную директорию;

<Ctrl>+<Enter> — переносит имя выделенного файла (директории) в командную строку;

<Insert> — выделить/отменить выделение файла;

Серый <+> — выделить группу файлов;

Серый <-> — отменить выделение группы файлов.

Сортировка файлов

Файлы внутри панели могут располагаться в порядке зависимости от имени, расширения, размера, даты, времени создания или порядке их записи на диск. Переход от одного типа сортировки к другому в активной панели осуществляется с помощью клавиш:

<Ctrl>+<F3> — сортировать файлы по имени;
<Ctrl>+<F4> — сортировать файлы по расширению;
<Ctrl>+<F5> — сортировать файлы по дате и времени создания;
<Ctrl>+<F6> — сортировать файлы по размеру;
<Ctrl>+<F7> — не сортировать файлы (выводить в порядке их записи на диск).

Выполнение команд

<Alt>+<F10> — вывести дерево каталогов;
<Esc> — очистить содержимое командной строки или отменить выполнение команды;
<Alt>+<F8> — просмотр ранее введенных команд DOS;
<Ctrl>+<E> — просмотр ранее введенных команд DOS в командной строке;
<Ctrl>+<X> — вывести в командную строку DOS следующую после текущей команду;
<Ctrl>+<PgUp> — перейти в родительский каталог;
<Ctrl>+<PgDn> — перейти в подкаталог.

Дисковые операции

<Alt>+<F1> — смена текущего диска для левой панели;
<Alt>+<F2> — смена текущего диска для правой панели;
<Ctrl>+<R> — обновить содержимое текущей панели.

7.2.2. Редактирование текстового файла

Norton Commander имеет чрезвычайно полезную функцию — возможность редактирования не очень больших текстовых файлов (объемом до 64 Кбайт).

Если вы хотите отредактировать уже существующий текстовый файл, то нужно подвести к нему подсветку курсора и нажать клавишу <F4>. Если же вы хотите написать новый текстовый файл, то нажмите <Shift>+<F4>,

при этом на экране появится запрос на имя нового файла. Рассмотрим возможности при редактировании.

Ввод текста осуществляется обычным образом. Для перехода на другую строку необходимо нажать <Enter>.

Перемещение курсора

<← > (<→>) — на позицию влево (вправо);
<↑> (<↓>) — на позицию вверх (вниз);
<Ctrl>+<← > — на слово влево;
<Ctrl>+<→> — на слово вправо;
<Ctrl>+<Home> — в начало текста;
<Ctrl>+<End> — в конец текста;
<Alt>+<F8> — устанавливает курсор на строку с определенным номером (вам будет предложено его ввести).

Удаление символов

<BkSp> (<Del>) — удалить символ слева (под курсором);
<Ctrl>+<BkSp> — удалить слово слева (справа);
<Ctrl>+<T> — удалить слово справа;
<Ctrl>+<Y> — удалить текущую строку;
<Ctrl>+<K> — удалить все, начиная от текущей позиции и до конца строки.

Операции с файлами

<F2> — записать файл с тем же именем;
<Shift>+<F2> — записать файл под новым именем (на экране появится соответствующее окно запроса);
<F9> — распечатать файл на принтере;
<F10> — выйти из текстового редактора (перед выходом, если файл был изменен, редактор предложит вам записать его);
<Shift>+<F10> — записать файл с тем же именем и выйти из текстового редактора.

Встроенный редактор Norton Commander позволяет производить операции с блоками текста.

Блок — это одна или несколько строк текста, причем строка может входить в блок только целиком.

Для выделения блока нужно установить курсор на первую строку блока и нажать <F3>, затем переместить курсор на последнюю строку блока и еще раз нажать <F3>. Блок при этом будет выводиться инверсным цветом (светлый фон, темные символы).

Команды работы с блоком

<F3> — пометить блок;
<Shift>+<F3> — отменить выделение блока;
<F5> (<F6>) — копировать (переместить) блок (необходимо установить курсор в то место, куда вы хотите скопировать (переместить) блок, и нажать <F5> (<F6>));
<F8> — удалить блок;
<Alt>+<F10> — сохранить блок как файл (при этом на экране появится вопрос, под каким именем вы хотите сохранить блок).

Встроенный редактор позволяет осуществлять ***поиск строки символов в тексте***. При нажатии <F7> появляется окно запроса, в которое вы должны ввести интересующую вас последовательность символов. Если флажок **Case sensitive** (**Различать прописные и строчные**) сброшен, то регистры при поиске не учитываются (т.е. большие и маленькие буквы не различаются). Если же флажок установлен, то поиск идет с учетом регистра. При успешном завершении поиска курсор устанавливается на найденный фрагмент; если же текст не найден, то выдается соответствующее сообщение.

<F7> — осуществляет поиск строки символов от текущего положения курсора и до конца файла (поиск вперед);
<Shift>+<F7> — осуществляет поиск строки символов от текущего положения курсора и до начала файла (поиск назад);
<Alt>+<F7> — повторяет поиск с текущими параметрами.

Возможен также ***поиск и одновременная замена текста***. При нажатии <F4> появляется окно запроса, в которое вы должны ввести интересующую вас последовательность символов. Поиск также идет с учетом регистра или нет в зависимости от установки или сбрасывания флажка **Case sensitive**.

При успешном завершении поиска найденный фрагмент текста выделяется, выдается запрос на замену и варианты ответов:

- **Replace** — заменить данный фрагмент;
- **Skip** — не заменять данный фрагмент;
- **Replace All** — заменить везде без дальнейших запросов;
- **Replace One** — заменить только здесь и прекратить поиск.

7.3. Сеанс работы на ПК и запуск программы в CAB REDUCE

Рассмотрим последовательность действий для создания и редактирования файла программы с последующим его выполнением в CAB REDUCE.

7.3.1. Сеанс работы на ПК с выполнением программы для CAB REDUCE

Для определенности допустим, что это будет программа 2.1 (2.12), расположенная, например, в файле PR2-1 в подкаталоге \C8 корневого каталога текущего диска, на котором находится и CAB REDUCE.

Конечно, имена подкаталога и файла могут быть *любыми допустимыми именами* (до восьми символов), а имя файла может еще иметь расширение до (трех символов). Эти *ваши имена* нужно подставлять везде ниже соответственно вместо имен подкаталога \C8 и файла PR2-1.

Сначала нужно сделать текущим подкаталог, в котором будет находиться файл программы. Для этого следует подвести подсветку курсора к его имени (\C8) и нажать клавишу <Enter>. Если подкаталог не находится на текущем диске, то предварительно следует нажать <Alt>+<F1> для левой или <Alt>+<F2> для правой панели, клавишами горизонтального перемещения курсора перейти на нужный диск и нажать <Enter>.

Если выделенного для вашей работы подкаталога вообще не существует, то его необходимо создать. Для этого в корневом каталоге текущего диска следует нажать функциональную клавишу <F7> и в выводимой на экран строке следует указать имя нового подкаталога (например, C8). После нажатия клавиши <Enter> в оглавлении текущего каталога появится новый пустой подкаталог с указанным именем C8. Далее действия выполняются согласно приведенного сеанса работы.

1. Для создания или редактирования файла программы в текущем каталоге \C8 необходимо при любом положении курсора нажать клавиши <Shift>+<F4>. Затем в появившемся на экране приглашении набрать имя файла (PR2-1) и нажать клавишу <Enter>.

Если файл с таким именем существует, то он вызывается на редактирование, если нет — то в появившемся на экране сообщении об этом выделенным окажется вариант *New-file (Новый файл)* и нужно еще раз нажать <Enter>.

2. Начать набор текста программы или исправление ошибок. Для *удаления символа* используются клавиши: *над курсором* — <Delete>, *влево от курсора* — <Back Space>.

Для разделения строки на две (добавление пустой) надо поместить курсор в месте разбиения (в начале или конце строки) и нажать клавишу <Enter>.

Для соединения двух строк (удаления пустой), надо поместить курсор правее последнего символа первой строки (над пустой) и нажать <Delete>.

3. После окончания ввода программы, исправления ошибок и проверки набранной информации необходимо ее сохранить. Для этого нужно:

- нажать <F2> (запись на диск с именем редактируемого файла);
- нажать <Shift>+<F2> и в появившейся строке набрать *нужное имя* и нажать клавишу <Enter> (запись на диск с новым именем файла).

Для выхода из режима редактирования следует нажать клавишу <F10>. На экране опять появляются панели Norton Commander.

7.3.2. Запуск программы в CAB REDUCE

Чтобы запустить на выполнение программу в CAB REDUCE, нужно:

- выйти из подкаталога \C8 в корневой каталог;
- подвести подсветку курсора к имени подкаталога \REDUCE и нажать клавишу <Enter>;
- подвести подсветку курсора к системному файлу с именем REDUCE.EXE или REDUCE.BAT и нажать клавишу <Enter>.

После появления приглашения в командной строке нужно набрать

1: **in ''\c8\ pr2-1'';**

После этого все результаты будут записаны в подкаталоге \C8 в файл с именем PR2-1.LIS (см. команды 10 и 95 программы 2.1 (2.12)).

Для выхода из REDUCE необходимо набрать в командной строке:

2: **bye;**

На экране опять появляются панели Norton Commander.

Для просмотра результатов следует войти в подкаталог \C8 и после совмещения подсветки курсора с именем файла результатов PR2-1.LIS и нажать клавишу <F3>.

Листание файла выполняется с помощью клавиш <PageUp> — вперед к концу файла и <PageDn> — назад к его началу.

После окончания просмотра необходимо нажать клавишу <F10>.

ЧАСТЬ 3. ПРАКТИЧЕСКАЯ РАБОТА В WINDOWS

ГЛАВА 8. MICROSOFT WINDOWS 3.1/3.11

Система Windows реализует удобный графический пользовательский интерфейс, одинаковый для всех используемых программ, называемых Windows-приложениями.

Аналогичные действия в различных программах для Windows выполняются аналогичными командами, что очень удобно. При этом команды для Windows не набираются с клавиатуры, как для DOS, а обычно выбираются мышью, например, из списка, который называется меню. Для выбора команды следует только установить указатель мыши, который представлен на экране стрелкой, на нужную команду меню и нажать левую кнопку мыши.

Windows поставляется с большим набором полезных программ, которые помогают выполнять различную работу. Каждая программа в Windows имеет хотя бы одну прямоугольную область, называемую *окном*, предназначенную для связи пользователя с данной программой. Экран монитора представляется в Windows как *рабочий стол*, на котором располагаются окна работающих в данный момент программ.

Если в DOS каждой программе ставится в соответствие имя файла (не превышающее длиной 8 символов), то в Windows для каждой программы используется понятие *программного элемента:* комбинации небольшой картинки, называемой *пиктограммой, значком* или чаще *иконкой* (icon), и надписи под ней. Такая комбинация удобна в использовании и гораздо более информативна. Поэтому для запуска программ совсем не надо помнить имена файлов, как это делается в DOS, просто нужно указать стрелкой на иконку на экране.

Современные программные продукты имеют в своем составе до нескольких сотен файлов. В Windows им будут соответствовать 2–3 ссылки на них в виде ярлыков со своими иконками (запуск самой программы, ее настройка, файл Read Me или Прочти Меня). Ярлыки однородных по своему назначению программ обычно помещаются в программные группы. Все они, а также любые документы и объекты Windows, в любой момент могут

быть сжаты в иконку, чтобы не занимать места на экране, или восстановлены в реальных размерах, перемещены в различные места экрана, скопированы в любой каталог или удалены.

В любой версии Windows, в отличие от DOS, можно работать с несколькими программами одновременно, легко и быстро переходить из одной в другую и при этом обмениваться информацией между ними. Это позволяет совместимость форматов данных, что очень удобно, например, для вставки рисунка из графического редактора в текстовый и используется при оформлении многочисленных документов (и настоящей книги тоже).

Следует еще отметить возможность организовать совместную работу программ для Windows и DOS, а также нескольких DOS-программ между собой, чего без помощи Windows вообще никак не добиться. Так, например, при написании пп. 2 и 3 настоящей книги с описанием практического использования DOS-версий CAB REDUCE 3.3 и 3.5:

- из-под Windows запускался:
  - текстовый редактор Word 97, в котором подготавливался текст;
  - система символьной математики DERIVE 4.11 для построения графиков по передаваемым из REDUCE через буфер обмена аналитическим соотношениям;
  - графический редактор CorelDRAW 9 для обработки графиков и выполнения рисунков схем конструкций и механизмов;
- одновременно с ними запускались DOS-программы:
  - изучаемая система аналитических вычислений REDUCE версий 3.3 и 3.5;
  - сервисная программа Norton Commander.

К недостаткам Windows следует отнести более высокую требовательность к объему свободного места на диске и размеру оперативной памяти.

Microsoft Windows обычно устанавливается в каталог с именем WINDOWS или WIN и для его запуска необходимо отправить на выполнение программу WIN.COM (достаточно набрать win и нажать <Enter>, если запуск не выполняется автоматически).

Windows Help (Справка) позволяет быстро получить справочную информацию о работе с Windows нажатием кнопки мыши или клавиши <F1>. Следующий раздел поможет вам лучше понять смысл и описание приводимых там процедур.

8.1. Основные элементы Windows

Иконка (Icon), значок или пиктограмма — графическое представление различных элементов в Windows (программы или окна документа, которые были минимизированы).

Указать (Point) — переместить мышь, использование которой обеспечивает максимальные удобства при работе с Windows, чтобы ее указатель (острие стрелки) показывал на нужный элемент. С ее помощью выполняются три следующие стандартные операции.

Операция Click — установив курсор на нужный элемент изображения, быстро нажать и отпустить (обычно левую) кнопку мыши (сокращенно обозначают "нажмите или щелкните мышкой"). Используется для выделения различных элементов (имя файла и т. п.) или выбора команд (позиций меню), выполняющих какие-то действия.

Операция Double-click — установив курсор на нужный элемент изображения, быстро дважды нажать и отпустить кнопку мыши (сокращенно обозначают "дважды щелкните мышкой"). Используется обычно для запуска программ на выполнение. Если нажатия происходят недостаточно быстро, то вместо одной операции Double-click получается две операции Click.

Операция Drag (Переместить) — установив курсор на нужный объект или элемент изображения (граница окна экрана и т.п.), нажать и удерживать кнопку мыши при ее перемещении. При этом объект станет передвигаться по экрану синхронно с перемещением курсора и после отпускания кнопки мыши зафиксируется на новом месте.

Окно (Window) — прямоугольная область экрана, содержащая программу или файл документа. Ее можно открывать, закрывать, перемещать, изменять ее размеры, сжимать в пиктограмму или увеличивать до размеров всего экрана. Каждое окно имеет ряд общих элементов.

Две кнопки в правом верхнем углу окна, будучи нажатыми с помощью клавиши мыши, позволяют увеличить до размеров всего экрана (правая кнопка с треугольником острием вверх) или минимизировать до пиктограммы (левая с треугольником острием вниз) активное окно. После увеличения окна на правой кнопке появляется два треугольника, расположенных один над другим. Нажав на эту кнопку, можно вернуть окну его первоначальные размеры.

В правой (и нижней) части окна расположен часто используемый в Windows орган управления, называемый полосой прокрутки (скролл-баром

("Scroll Bar") или слайдером). В верхней и нижней части полосы (или правой и левой) расположены кнопки в виде стрелок.

После нажатия на них мышкой в соответствующую сторону происходит движение расположенного между ними движка слайдера (в виде выделенного прямоугольника), а с ним и перемещение информации, не помещающейся целиком в выделенном месте. Его положение между стрелками соответствует положению нужного места в документе (начало, середина или конец файла или списка), а перемещение в любую позицию осуществляется мышью (операция Drag).

Нажатие кнопки мышки в любом месте слайдера выше или ниже движка (или слева или справа для горизонтальной полосы) приведет к перемещению (скроллингу) информации на одно окно.

Заголовок (Title bar), расположенный посередине вверху каждого окна, содержит название программы (и документа, если он открыт этой программой). Если документ создается и еще не был сохранен, то на его месте появляется надпись untitled (безымянный).

Системное меню (Control-menu box) расположено в верхнем левом углу каждого окна в минимизированном виде и представляет из себя небольшой квадратик с прямоугольником. Для его вызова достаточно нажать кнопкой мыши по этому квадратику (или воспользоваться клавишами <Alt>+<Пробел>). Команды этого меню позволяют изменять размеры окна, перемещать, увеличивать, минимизировать и закрывать окна, а также переключаться на список активных в настоящий момент программ.

Строка меню (Menu bar) расположена ниже заголовка и в ней перечислены имеющиеся в наличие меню. Большинство программ имеют меню File (Файл), Edit (Редактировать), Help (Справка), а также специфические для данной программы меню. Для их открытия следует подвести курсор на имя меню в строке меню и нажать кнопку мыши.

После открытия меню делается выбор из него нужной команды, производящей выполнение соответствующего действия, для чего также следует нажать кнопкой мыши имя нужного элемента. Для закрытия меню следует нажать кнопкой мыши его имя или любое место вне меню.

Записи элементов меню подчиняются следующим правилам:

- *если имя команды выглядит нечетко (серого цвета)*, то в данный момент вы не можете ею воспользоваться;
- *многоточие после выбора элемента меню* означает, что после его выбора появится диалоговое окно, запрашивающее дополнительную информацию для выполнения команды;

- *галочка рядом с именем элемента меню* определяет, что команда является активной (используется для команд, переключающихся между двумя состояниями);
- *комбинация клавиш после имени элемента меню* является сокращением команды и позволяет ею воспользоваться без предварительного открытия меню;
- *треугольник, расположенный справа от команды меню* означает, что она приводит к открытию каскадного меню, в котором содержатся дополнительные команды.

8.2. Запуск, настройка и завершение работы Windows

Microsoft Windows обычно устанавливается в каталог с именем WINDOWS или WIN и для его *запуска* необходимо отправить на выполнение программу WIN.COM (достаточно набрать WIN и нажать клавишу <Enter>, если запуск не выполняется автоматически). При запуске происходит вывод заставки, загрузка самой оболочки, а также запуск Диспетчера Программ (Program Manager).

Практически все операции по *настройке* Windows выполняются с помощью Панели управления (Control Panel) Диспетчера программ (Program Manager). Как правило, панель управления расположена в программной группе Главная (Main).

При запуске Панели управления (Control Panel) открывается соответствующее окно. В этом окне расположен ряд иконок, дважды щелкнув по которым можно вызвать соответствующую функцию:

- Цвет (Color) — задает цвета Windows;
- Шрифты (Fonts) — позволяет установить новые и убрать ставшие ненужными шрифты;
- Порты (Ports) — позволяет настроить последовательные (COM) порты;
- Мышь (Mouse) — позволяет установить скорость отслеживания мыши и скорость двойного щелчка;
- Оформление (Desktop) — задает оформление Windows;
- Клавиатура (Keyboard) — позволяет установить параметры автоповтора клавиш;
- Принтеры (Printers) — позволяет установить новые и убрать ставшие ненужными драйверы принтеров, а также настроить принтеры под ваши нужды;

- Стандарты (International) — задает стандарты страны;
- Дата / Время (Date / Time) — позволяет установить текущие дату и время;
- Расширенный (Enhanced) — задает параметры расширенного режима Windows;
- Драйверы (Drivers) — позволяет установить новые и убрать ставшие ненужными драйверы;
- Звук (Sound) — позволяет назначить звуки событиям (при наличии звуковой платы);
- Сеть (Network) — позволяет установить параметры сети.

В зависимости от установленного оборудования в Панели управления могут появиться и другие пиктограммы.

Чтобы *завершить* работу в Windows, нужно перейти в окно диспетчера программ (<Ctrl>+<Esc>) и дважды щелкнуть по кнопке системного меню диспетчера программ. Можно также нажать <ALT>+<F4>. На экран будет выдан запрос на окончание работы. Если ответ на запрос будет утвердительным, то оболочка прекратит свою работу.

Если в момент выхода запущены Windows-приложения и в них есть не сохраненная информация, то перед выходом Windows предложит ее сохранить.

Если в момент выхода выяснится, что в системе есть незавершенные DOS-приложения, то Windows потребует вначале завершить эти приложения, а потом повторить попытку выхода. В этом случае вы должны перейти в DOS-приложение (<Ctrl>+<Esc>) и завершить его стандартным для него способом. После этого надо будет снова дважды щелкнуть по кнопке системного меню диспетчера программ.

8.3. Управление окнами

8.3.1. Изменение размера окна

Гибкость оболочки Windows не в последнюю очередь определяется гибкостью управления окнами. Рассмотрим приемы этого управления.

Для того чтобы *изменить размер окна*, маркер мыши необходимо установить на границу окна. Если данное окно допускает изменение размера (такие окна имеют более толстую границу), то маркер принимает вид “двойной” стрелки.

Стрелки будут показывать возможное направление. Если вы действительно хотите изменить границу окна, то необходимо просто переместить ее на новое место (нажать на левую кнопку мыши и, не отпуская ее, передвинуть маркер на нужное место). Если в окне отсутствовали линейки прокрутки, а вы уменьшили его размер, то может появиться одна или обе эти линейки.

Помимо этого, вы можете воспользоваться кнопками изменения размера. Они расположены справа от заголовка в правом верхнем углу экрана и представляют собой изображения треугольника соответственно вершинами **вниз** (свернуть) или **вверх** (развернуть). Если одна из этих кнопок или обе отсутствуют, то, значит, данная операция для этого окна невозможна.

Если вы нажмете на кнопку развернуть (с изображением треугольника *вершиной вверх*), то окно развернется на полный экран. При этом кнопка изменит свой вид и на ней появится изображение двух треугольников соответственно с *вершинами вверх и вниз*, расположенных один под одним. Нажатие на этой кнопке вернет экран в прежнее состояние (функция восстановления размера).

Кнопка свернуть (с изображением треугольника *вершиной вниз*) помогает свернуть окно в пиктограмму (чтобы не занимала лишнего места). Чтобы восстановить прежний размер, нужно дважды щелкнуть мышью по этой пиктограмме.

8.3.2. Перемещение окна или пиктограммы, их сворачивание и восстановление

Переместить окно достаточно просто. Для этого достаточно ухватить заголовок окна и переместить его в нужное место (при перемещении новое положение будет показываться контурами окна). Переместить иконку также просто. Необходимо ухватить ее мышью и передвинуть на новое место.

Чтобы свернуть открытое окно в пиктограмму, нужно воспользоваться кнопкой свернуть (справа от заголовка). Чтобы восстановить прежний размер, нужно дважды щелкнуть по пиктограмме.

8.3.3. Прокрутка содержимого окна

Если содержимое окна не “влезает” в текущие размеры окна, то слева и внизу появляются линейки прокрутки (или одна из них). Рассмотрим, как ими пользоваться. Самый простой вариант - это перетащить на новое место бегунок. При этом, соответственно, изменится и содержимое окна.

Бывают случаи, когда необходимо прокрутить окно всего на одну строчку (столбец). Тогда нужно воспользоваться кнопками со стрелками, расположенными по концам линейки прокрутки. Щелчек по такой кнопке и осуществляет скроллинг (перемещение) окна на одну строку.

Если же необходимо переместить окно на один экран вверх или вниз (вправо или влево), то вы можете поступить следующим образом. Установите маркер мыши на линейку прокрутки выше или ниже бегунка и щелкните левой кнопкой мыши. При этом произойдет скроллинг на один экран.

8.3.4. Переключение между окнами и их закрытие

Еще одной концептуальной особенностью Windows является возможность в любой момент переключиться в любое окно. Как мы уже говорили, приложение, находящееся в активном окне, само является активным, все остальные запущенные приложения находятся в фоновом режиме.

Windows очень легко позволяет сделать активным любое окно. Если окно или кусочек интересующего вас окна видна на экране, то достаточно щелкнуть мышью в любое место этого окна. Приложение, работающее в этом окне, тут же станет активным. Если интересующее вас окно свернуто в пиктограмму, то нужно дважды щелкнуть по ней мышью.

Чтобы закрыть окно (если это возможно) нужно дважды щелкнуть по кнопке системного меню. Если данное окно можно закрыть (окна Windows-приложений, часть вторичных окон, часть окон запроса), то после двойного щелчка данное окно исчезнет с экрана. При этом:

- если это было окно Windows-приложения, то данная задача будет завершена;
- если это было вторичное окно Windows-приложения, то обработка данного документа будет прекращена;
- если это было сообщение Windows-приложения, то данное окно исчезнет с экрана ;
- если это было окно с запросом Windows-приложения, то это будет означать отмену запрашиваемого действия.

Если в настоящий момент в окне находится не сохраненная информация, то при попытке закрыть данное окно Windows в начале предложит сохранить последние изменения.

При двойном щелчке по кнопке системного меню вторичного окна для ряда Windows-приложений происходит сворачивание окна в пиктограмму (например, у Диспетчера Программ (Program Manager)).

При двойном щелчке по кнопке системного меню DOS-приложения, запущенного в окне, никаких действий не происходит. Для закрытия такого окна необходимо вначале сделать это окно активным, а затем завершить DOS-программу стандартным для нее способом. После этих действий данное окно автоматически исчезнет с экрана.

8.3.5. Работа с диалоговыми окнами

Windows использует диалоговые окна (dialog boxes) для запроса нужной или выдачи дополнительной информации. Они появляются как при выборе команды, после имени которой в меню стоит многоточие (...), так и при использовании опций, запрашивающих дополнительную информацию. Для выбора кнопки команды, опции, элемента из поля списка или переключателя нужно нажать на кнопку с соответствующим именем мышкой. Если в нем содержится подчеркнутая буква, то эту команду или опцию можно выбрать также из любого места диалогового окна, нажав <Alt> и эту подчеркнутую букву. Выделенная кнопка имеет более темный контур по сравнению с остальными или черную точку (в списках взаимно исключающих вариантов выбора), помеченный переключатель содержит **x** и их может быть несколько, а недоступные в данный момент выглядят нечетко. Чтобы изменить выбор, пометьте другую кнопку или для выделенного элемента списка вновь нажмите его мышкой.

Если требуется ввести дополнительную информацию, то для этого в окне существует прямоугольная область, называемая полем текста (text box). Если оно пусто, то при перемещении указателя мыши в нем появляется курсор ввода (мерцающая вертикальная черта), после чего можно печатать. Если поле уже содержит текст, то при перемещении в него он автоматически выделяется и любой набираемый текст заменит его. Перед этим следует нажать кнопку мышки в том месте, где нужно поместить начало текста, для появления в нем курсора ввода.

После ввода всех необходимых данных (имени и места расположения файла и т.п.) необходимо отправить команду на выполнение. Для этого следует нажать мышкой кнопку **OK** (операция Click) или дважды нажать кнопку с именем нужной команды (операция Double-click). После этого диалоговое окно закрывается и команда выполняется. Закрытие диалогового окна без выполнения команды достигается нажатием клавиши <Esc>, или мышкой кнопки Cancel, либо двойным нажатием кнопки Системного Меню в верхнем левом углу экрана (последнее вызовет также закрытие окна программы или документа).

8.4. Работа с программами и файлами

Программы, написанные только для использования с системой Windows, использующие графический интерфейс, меню со стандартными элементами и диалоговые окна, подчиняются одним и тем же правилам и называются Windows-приложениями. Они не могут работать в MS DOS.

Одновременно с этим накоплено огромнейшее программное обеспечение еще до выпуска системы Windows, разработчики которой были вынуждены предусмотреть его использование. Они рассчитаны на работу с MS-DOS, называются DOS-программами и могут работать в графическом окружении Windows, хотя и не подчиняются его правилам (не используют меню, диалоговые окна и т.п.).

К последним относятся и описываемые в настоящем пособии программы для работы в системе REDUCE, которые являются универсальными и могут работать на любых IBM-совместимых ЭВМ с различными операционными системами.

В стандартном или полноэкранном режиме все DOS-программы работают во всем экране. Это означает, что программа занимает весь экран и выглядит точно так, как при выполнении в MS DOS, без Windows.

Для открытия файла следует выбрать команду Open (Открыть) из меню File программы, затем переместиться в поле списка Directories (Каталоги) и дважды нажать кнопкой мышки имя каталога, содержащего нужный файл. Потом в появившемся после этого поле списка Files (Файлы) следует указать нужный файл и дважды нажать мышкой его имя и кнопку OK. Для создания нового файла в меню File следует выбрать команду New.

Положение курсора (мерцающая вертикальная черта) помечает место, где будут появляться набираемые символы. При работе с новым файлом можно сразу начать печатать (при необходимости установив курсор клавишами <Enter> или <Пробел> в нужное место). Перед внесением исправлений в существующем файле следует нажать кнопку мышки в нужном месте, после чего там появится указатель курсора.

Для сохранения изменений в уже существующем файле используется команда Save (Сохранить) из меню File (Файл). Чтобы сохранить новый или уже существующий файл под новым именем из меню File следует выбрать команду Save As, указать нужное имя и путь в поле текста Filename (Имя файла), после чего нажать кнопку OK. Вместо указания полного пути предварительно в поле списка Directories (Каталоги) можно выбрать каталог, в котором нужно сохранить файл.

Запустить программу на выполнение в Windows можно тремя способами.

1. Открыв окно Диспетчера Программ (Program Manager), следует открыть окно группы, которая содержит запускаемую программу. После этого нужно дважды нажать кнопкой мышки пиктограмму программы.

2. Запустив Диспетчер Файлов (File Manager) (см. п. 8.7), следует открыть окно каталога, содержащего файл программы (с расширениями .COM, .EXE, .BAT или .PIF). Теперь также нужно дважды нажать кнопкой мышки пиктограмму программы.

3. Выбрать команду Run (Запустить) из меню File в окне Диспетчера Программ (Program Manager) или Диспетчер Файлов (File Manager). В появившемся диалоговом окне Run следует набрать полное имя запускаемого файла (с указанием пути к нему, если он находится вне текущего каталога). После этого следует нажать клавишу <Enter> или мышкой кнопку OK. Способ удобен для запуска редко используемых программ, которые не добавлены в группу в Диспетчере Программ (Program Manager).

На экране часто открыто много окон программ. Активное окно, в котором вы работаете, всегда появляется на переднем плане и его заголовок имеет другой цвет или интенсивность. Оно может перекрывать неактивные окна, частично или полностью загораживая их (поэтому некоторые из них могут быть невидимыми). Чтобы сделать активным видимое окно, следует нажать кнопкой мышки в любом его месте. Для неактивного окна сначала нужно открыть диалоговое окно Task List (Список Задач). Этого можно достичь двойным нажатием кнопкой мышки в любом свободном от икон или пиктограмм месте экрана, а также выбором Switch To (Переключиться На) из Системного Меню программы. Теперь двойное нажатие кнопкой мышки имени нужной программы в поле списка в Task List позволит на нее переключиться.

Отметим, что нажатие кнопок Cascade (Каскад) и Tile (Мозаика) в Task List приводит к соответствующему упорядочиванию окон программ или файлов, а Arrange Icons — к переупорядочиванию пиктограмм (вдоль нижнего края экрана).

Для выхода из программ нужно выбрать команду Exit из меню File программы (или набрать стандартную команду выхода для соответствующей DOS-программы, например, команду **bye** для REDUCE). Можно также дважды нажать мышкой кнопку системного меню в левом верхнем углу экрана или выбрать команду Close из системного меню, а также воспользоваться клавишами <Alt>+<F4>.

8.5. Использование меню Help для получения справочной информации

В системе Help (Справка) приведены описания основных приемов, необходимых для работы с Windows, клавиатурный эквивалент действий без мышки, а также содержится информация об используемых командах. В зависимости от этого имеются различные варианты вызова Help.

Если выбрать мышкой кнопку Help в строке меню, а затем нужный раздел, то получим или информацию по данному разделу, или список его тем. При вызове Help клавишей <F1> на экране появляется Index (Оглавление) справочной информации программы, с которой вы работаете.

Для поиска информации с помощью Help Index (Оглавления Help) следует нажать клавишу <F1> или выбрать нужный раздел из меню Help и переместиться к нужной теме. Когда указатель мышки станет похожим на указательный палец, нужно нажать кнопку мышки. Это означает, что информация по данной теме имеется в наличии, и при выборе этой темы появляется содержащее ее окно. Для более полного ответа в конце текста справки часто приводится список связанных с ней тем, выбрать которые можно также нажатием кнопки мышки.

Можно получить нужную информацию, соответствующую ключевому слову. Для этого следует открыть Help и нажать мышкой кнопку Search (Поиск). В появившемся диалоговом окне в текстовом поле Search For (Что искать?) нужно набрать ключевое слово или его часть. Ниже в поле списка приводятся ключевые слова, соответствующие набираемым буквам. При нажатии кнопки Search выдается список тем, имеющих отношение к набранному ключевому слову или фразе, из которых следует выбрать нужную. При следующем обращении к Search сохраняется название темы, найденной в прошлый раз.

Большинство меню Help содержит следующие основные опции:

- Index (Оглавление) — список всех тем справочной информации в алфавитном порядке;
- Keyboard (Клавиатура) — таблицы комбинаций клавиш для работы с клавиатурой в активной программе;
- Commands (Команды) — объяснения всех используемых команд;
- Procedure (Процедуры) — пошаговые инструкции по использованию активной программы;
- Using Help (Работа со Справкой) содержит небольшой урок и информацию о том, как работать с Windows Help. С ним рекомендуется ознакомиться при первоначальном использовании справочной системы.

8.6. Диспетчер Программ (Program Manager)

Центральной программой в Windows является Диспетчер Программ (Program Manager). Она стартует автоматически при запуске Windows и прекращает свою работу только вместе с ним. Предназначена для запуска программ и выполняет организационные функции (объединяет программы в группы по каким-либо признакам).

Окна групп содержат пиктограммы заголовков программ, запускающих связанную с ней программу. Команды строки меню Диспетчера Программ (Program Manager) воздействуют на все окна групп, имеющих только свое собственное Системное Меню.

Пиктограммы групп — это минимизированные окна групп. Они различаются только подписями под каждой пиктограммой и расположены в нижней части окна Диспетчера Программ (Program Manager), за пределы которого их, как и окна групп, невозможно переместить. Пиктограммы заголовков программ находятся внутри окон групп. Их можно перемещать из окна одной группы в окно другой, но не непосредственно в окно Диспетчера Программ (Program Manager) или за его пределы.

При первом запуске Windows Program Manager появляется на экране с открытым окном группы Main (Главная), которое содержит прикладные программы системы: File Manager (Диспетчер Файлов), Control Panel (Панель Управления), Print Manager (Диспетчер Печати), Clipboard Viewer (Буфер обмена), DOS Prompt (Интерпретатор команд, сеанс работы в DOS), Windows Setup (Установка Windows), PIF Editor (Редактор PIF) — редактор файлов программной информации для DOS-программ.

Группа Accessories (Реквизиты) включает текстовый процессор, программу рисования, коммуникационную программу и несколько простых сервисных программ (калькулятор, часы, календарь, блокнот, картотеку), а также Macro Recorder (генератор макрокоманд Windows).

Группы Windows Applications (Windows-программы) и Non-Windows Applications (DOS-программы) содержит соответствующие программы, которые были найдены Setup на винчестере во время процедуры установки Windows.

Чтобы запустить программу из группы нужно дважды нажать кнопкой мышки соответствующую пиктограмму группы, из-за чего ее окно откроется. Затем следует также дважды нажать мышкой пиктограмму нужной

программы. При запуске программы Program Manager обычно делает текущим каталог, ее содержащий. Чтобы вернуться обратно из какой-либо программы, следует дважды нажать мышкой пиктограмму Program Manager.

Отметим, что в Program Manager можно добавлять или удалять необходимое количество групп или программ, чтобы организовать работу наиболее удобным образом.

Чтобы создать новую группу следует выбрать команду New (Новый) из меню File (Файл). В появившемся диалоговом окне New Program Object (Новый Программный Объект) нужно выбрать опцию Program Group (Программная группа) и нажать OK.

После этого появится следующее диалоговое окно Program Group Properties (Реквизиты Программной Группы), в поле Description (Описание) которого следует ввести имя группы и нажать OK.

Чтобы добавить программу в группу нужно открыть ее окно (дважды нажав мышкой пиктограмму группы), выбрать также команду New (Новый) из меню File (Файл), но в том же диалоговом окне New Program Object (Новый Программный Объект) следует выбрать опцию Program Item (Программа) и нажать OK.

Затем в следующем диалоговом окне Program Item Properties (Реквизиты Программы) в поле текста Description (Описание) следует ввести имя программы, а в поле Command Line набрать полное имя командного файла (и путь к нему, если соответствующий каталог не указан в команде Path в файле AUTOEXEC.BAT), после чего нажать OK.

Если имя программного файла неизвестно, то в появившемся на экране после нажатия кнопки Browse (Просмотр) списке файлов и каталогов следует выделить нужное имя файла (и каталог, если он не является текущим), щелкнув по нему мышкой, и нажать OK, чтобы ввести его в поле Command Line.

Для удаления группы нужно нажать кнопкой мышки ее пиктограмму, выбрать команду Delete (Удалить) из меню File (Файл) и в диалоговом окне выбрать Yes (Да). В результате выделенная группа и все пиктограммы заголовков программ в ней будут удалены, хотя файлы соответствующих программ все же останутся на диске.

Для удаления программы из группы следует выполнить аналогичные действия с пиктограммой ее заголовка (предварительно открыв окно нужной группы), что также не приведет к ее удалению с диска.

Для перемещения программ или их копирования из одной группы в другую следует выполнить операцию Drag над пиктограммой программы, причем копирование будет проводиться при нажатии и удерживании клавиши <Ctrl> во все время перемещения.

Чтобы изменить реквизиты заголовка программы, нужно выделить пиктограмму программы, выбрать Properties (Реквизиты) из меню File, внести нужные изменения и нажать кнопку OK.

Чтобы выйти из Windows и из Диспетчера Программ (Program Manager) сначала следует выйти из всех выполняемых программ, выбрать Exit Windows (Выход из Windows) из меню File и после подтверждения завершения сеанса работы в появившемся диалоговом окне нажать кнопку **OK**. Если при этом переключатель Save Changes (Сохранить Изменения) будет помечен, то при следующем запуске Windows рабочая область Диспетчера Программ (Program Manager) будет иметь тот вид, который она имела во время последнего сеанса.

8.7. Диспетчер Файлов (File Manager)

Программа Диспетчер Файлов (File Manager) является инструментом, помогающим содержать файлы и каталоги в порядке. Из нее можно также запускать программы. При первом запуске Windows пиктограмма Диспетчера Файлов (File Manager) появляется на первом месте в группе Main (Главная) в Диспетчере Программ (Program Manager).

В Диспетчере Файлов (File Manager) используются окна двух типов. В окне Дерева Каталогов (Directory Tree) показана общая структура каталогов диска, в окне каталога — перечислены файлы и подкаталоги (таких окон можно сделать несколько при помощи меню Window и команды New Windows (Окно и Новое Окно)). Чтобы оба окна расположились без перекрытий, надо нажать <Shift>+<F4>. Тогда File Manager станет похожим на Norton Commander.

Чтобы выдать на экран структуру каталогов другого дискового накопителя (или содержание другого каталога) нужно выбрать мышкой пиктограмму с нужной буквой дисковода (или именем каталога). По умолчанию наличие подкаталогов ничем не отмечается. Если в меню Tree (Дерево) включен переключатель **Indicate Expandable Branches (Отмечать Расширяемые Ветви)**, то на пиктограммах каталогов появится индикация "+" и "–". Знак "+" означает, что в данном каталоге имеются подкаталоги.

Если нажать по нему мышкой или воспользоваться клавишей <+>, то Диспетчер Файлов (File Manager) покажет подкаталоги 1-го уровня этого каталога. Он станет "раскрытым" и на его пиктограмме вместо "+" появится знак "–". Аналогичные действия над знаком "–" (мышкой или клавишей <–>) уберут с экрана подкаталоги, каталог снова станет "закрытым" со знаком "+". Чтобы увидеть всю ветвь полностью, следует выделить раскрываемый каталог и нажать клавишу <*> (звездочка). Совместное использование <Ctrl>+<*> позволит увидеть одновременно все уровни всех каталогов.

Основные команды для работы с файлами и каталогами (открыть, переместить, копировать, удалить, переименовать, поиск) одинаковы и содержатся в меню File (Файл).

Работать можно с каждым файлом или каталогом в отдельности, либо группируя их, для чего их нужно *выделить*. Для одного файла или каталога это можно сделать, *нажав кнопкой мышки его имя*. Если их несколько и они расположены последовательно, выделив обычным образом первый элемент группы, для последнего следует *нажать и удерживать клавишу <Shift> в момент его выбора мышкой.*

При их произвольном расположении отмечать каждый новый элемент мышью следует, удерживая при этом <Ctrl>. Для выделения всех файлов удобнее пользоваться клавишами <Ctrl>+</>, а для его снятия — <Ctrl>+<\>.

Отметим, что с помощью операции Drag (Переместить) удобно выполнять перемещение или копирование файлов, каталогов или их выделенных групп. Просто "берите" мышью файл, каталог или группу выделенных файлов и каталогов и "тащите" к целевой пиктограмме каталога или диска. В пределах диска по умолчанию происходит перемещение (а при удерживании <Ctrl> — копирование), с диска на диск по умолчанию выполняется копирование (а при удерживании <Alt> — перемещение).

Если разместить на дисплее Диспетчер Программ (Program Manager) и Диспетчер Файлов (File Manager) так, чтобы были видны окна обеих программ, то операция Drag позволит также устанавливать новые программные элементы в группах Диспетчер Программ (Program Manager). Для этого просто "возьмите" мышью пиктограмму исполняемого файла программы в Диспетчере Файлов (File Manager) и "перенесите" ее в ту группу Диспетчера Программ (Program Manager), где желательно поместить новый программный элемент. Таким же образом в Диспетчер Программ (Program Manager) можно установить и файл данных, связанный с прикладной программой (свернутой до пиктограммы), для ее загрузки с этим файлом.

ГЛАВА 9. MICROSOFT WINDOWS 95/98/MILLENIUM (ME)

Версия системы Windows 95 во многом отличается от рассмотренных выше версий 3.1-3.11. Она объединяет в себе DOS и Windows. У нее иная логика построения интерфейса (окон, меню, команд, настроек), поэтому нам придется ниже уточнить некоторые понятия. Ее 32-разрядная архитектура *позволила*:

- значительно увеличить ее быстродействие;
- использование новых 32-разрядных версий современных программ;
- оснастить систему драйверами для большинства дополнительных устройств и реализовать технологию Plug and Play (воткни и работай), в результате чего она:
  - опознает вновь подключенное устройство;
  - сама настраивается на него;
- устранить некоторые неудобства старых версий Windows и DOS, в результате чего теперь можно прямо из Norton Commander запускать windows-программы;
- реализовать концепцию длинных имен файлов (до 255 символов, включая пробелы и точки).

Все эти достоинства существенно повысили требовательность Windows 95 к быстродействию процессора, объему оперативной памяти компьютера и свободному месту на диске, ибо за все надо платить.

Windows 98 и Windows Millenium (Me) являются непосредственными преемницами соответственно Windows 95 и Windows 98, последовательно развивая достижения каждой версии:

- делая работу на ПК более удобной и продуктивной;
- предоставляя много дополнительных возможностей;
- поддерживая более совершенное современное оборудования и созданное для него соответствующее программное обеспечение.

Все эти версии имеют много общего, при рассмотрении которого ссылка на них указывается в виде Windows 95/98/Me. При описании отличительных особенностей указывается название соответствующей версии.

9.1. Запуск, осмотр Рабочего стола (Desktop) и завершение работы Windows 95/98/Me

Операционная система (ОС) Windows 95/98/Me автоматически запускается при старте компьютера. Обычно также сразу загружается и графическая оболочка; если этого по каким-либо причинам не произошло, то необходимо набрать в командной строке WIN.COM

Запуск Windows 95/98/Me сопровождается выводом заставки, а затем вы попадаете на Рабочий стол (Desktop).

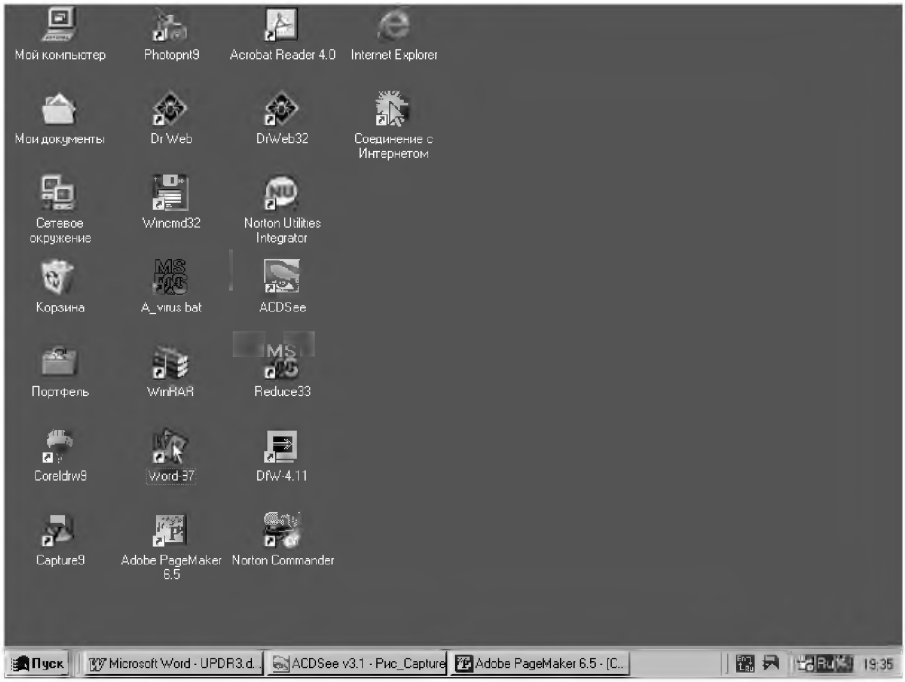

Рис. 9.1. Рабочий стол (Desktop) после загрузки Windows 98 (как он выглядит на компьютере автора)

Внизу Рабочего стола (Desktop) располагается Панель задач (Taskbar) с кнопкой Пуск (Start)  , предназначенная для вызова Главного меню (Start Menu), которое открывает доступ ко всем основным рабочим программам.

Кнопка Пуск (Start) — самая важная кнопка Рабочего стола. Практически при каждом включение компьютера пользователь будет нажимать эту кнопку (хотя бы для завершения работы).

Панель задач (Taskbar) представляет собой линейку, расположенную обычно горизонтально в нижней части Рабочего стола (Desktop). На ней также размещены кнопки активных приложений: названия открытых или свернутых окон и работающих программ.

Рис. 9.2. Панель задач (Taskbar)

Это оказалось чрезвычайно полезным и удобным: позволяет не только всегда видеть, какие приложения запущены, но также значительно облегчает переключение между ними. Для этого достаточно просто щелкнуть по кнопке нужного приложения.

Если полное название не умещается на кнопке и вам неясно, какая задача ей соответствует — задержите на несколько секунд мышь и вы получите ответ. Этот прием называется *зависанием указателя.*

Справа на панели задач расположена системная область ("System Tray"), называемая также панелью индикации, в которой обычно находятся:

- индикатор (он же переключатель) текущей кодировки клавиатуры (языка). Щелчок мыши по нему позволяет открыть небольшое меню для переключения между кодировками (выбора нужного языка);
- индикатор с показаниями системных часов компьютера (часы : минуты), при зависании указателя над которым всплывает подсказка с текущей датой, а двойной щелчок по нему открывает диалоговое окно, с помощью которого можно изменить настройку и системных часов и календаря;
- значки некоторых программ, при зависании указателя над которыми вы увидите на всплывающей подсказке их название.

Отметим, что размеры Панели задач (Taskbar), ее положение на экране, состав размещенных на ней элементов могут быть изменены по желанию пользователя. Например, вы можете, как и любое окно Windows:

- перенести панель задач в любое место экрана — ее нужно ухватить мышью и перенести на новое место;
- изменить ее размер — нужно ухватить границу панели задач и передвинуть на желаемое расстояние.

В Windows 98/Millenium между кнопкой Пуск и основным пространством панели задач, а также между ним и системной областью могут находиться настраиваемые панели инструментов и панель быстрого запуска, содержание которой также полностью определяется пользователем.

Основное пространство Рабочего стола (Desktop) занимают пиктограммы различных программ. Часть из них устанавливается по умолчанию, а большинство — в зависимости от индивидуальных настроек и потребностей пользователя. Поэтому некоторые из перечисленных ниже значков наиболее важных инструментов могут на вашем компьютере отсутствовать. Обязательными являются (см. рис. 9.1) Мой компьютер (My Computer) и Корзина (Recycle Bin):

- Мой компьютер (My Computer) — представляет для пользователя средство доступа ко всем ресурсам компьютера и сети (а также информацию о них): дискам, каталогам (папкам), файлам, принтерам. Для того, чтобы им воспользоваться, достаточно дважды щелкнуть мышью на его значке, после чего на Рабочем столе (Desktop) откроется соответствующее окно (см. рис. 9.3);
- Сетевое окружение (Network Neighborhood) — позволяет работать с ресурсами сети так же, как с ресурсами локального компьютера (если вы подключены к сети);
- Корзина (Recycle Bin) — позволяет удалять папки и файлы. Если вы перетащите какую-либо папку или файл на значок Корзины (Recycle Bin), то вы удалите его с занимаемого места на диске. Удаляемые пользователем папки и файлы на самом деле не уничтожаются, а последовательно складываются в корзину (переносятся в другое место на диске). Поэтому их можно при необходимости легко восстановить. Окончательное удаление происходит:
  - после принудительной очистки корзины, что время от времени выполняется пользователем;
  - при превышении общим объемом файлов в корзине некоторого значения (обычно 10% от объема диска), после чего их начинают вытеснять вновь помещаемые файлы.

Если вы хотите закончить работу Windows 95/98/Me, нажмите:

- кнопку Пуск (Start) на панели задач, а затем выберите опцию Завершение работы (Shut Down);
- сочетание клавиш <Alt>+<F4>.

В обоих случаях откроется диалоговое окно Завершение работы Windows (Shut Down Windows), в котором будет предложено:

- Выключить компьютер (Shut down the computer?) — завершить работу ОС Windows;

- Перезагрузить компьютер (Restart the computer?) — осуществить горячий рестарт ОС;
- Перезагрузить компьютер в режиме эмуляции MS-DOS (Restart the computer in MS-DOS mode?) — выйти в режим DOS, т.е. выгрузить графическую оболочку.

Рис. 9.3. Рабочий стол (Desktop) с открытым окном Мой компьютер (My Computer)

Заметим, что:

- версия Windows Millenium не имеет возможности эмуляции MS-DOS. Поэтому в ней могут работать только те программы, которые корректно запускаются в Windows в окнах DOS-приложений;
- в окне Завершение работы (Shut Down):
  - в версиях Windows 98/Me есть еще пункт Приостановить работу компьютера;
  - появляется строка Войти в систему под другим именем (Close all programs and log on as a different user), если компьютер подключен к локальной сети и на нем имеют право работать несколько зарегистрированных пользователей.

Если в момент выхода запущены Windows-приложения и в них есть не сохраненная информация, то перед выходом Windows предложит ее сохранить.

Если в момент выхода выяснится, что в системе есть незавершенные DOS-приложения, то Windows потребует вначале завершить эти приложения стандартным для них способом. После этого выгрузка ОС будет автоматически продолжена.

9.2. Файловая структура Windows 95/98/Me. Основные понятия

Операционные системы Windows 95/98/Me являются графическими. Поэтому все объекты, с которыми они работают, представляются графически в виде значков (называемых чаще иконками или пиктограммами).

Под *объектом* в Windows 95/98/Me понимается любой предмет, свойства которого различимы для Windows 95/98/Me и с которым оперирует система: программа, приложение или группа программ, файл, документ или его фрагмент, папка, иконка, компьютер при наличии сети, физическое устройство (жесткий диск, дисковод CD-ROM, принтер) и т.п.

Файлом называется то же самое, что подразумевалось под этим термином в MS-DOS: любой массив информации, сохраненный на диске и имеющий собственное имя. Файл, например, может быть программой, набором данных, текстовым документом.

Термин *документ (Document)* в Windows 95/98/Me приобрел расширенный смысл.

Под этим термином понимают не только текстовые файлы (как это было при работе в прежних версиях Windows), а практически любой файл, ассоциированный с программой для его просмотра или изменения и содержащий данные: текст, графическое изображение, электронную таблицу, звук, видеофильм.

Такой подход является основой реализованного в Windows 95/98/Me документо-ориентированного принципа работы: *документ является первичным по отношению к приложению, в котором он был создан или может быть использован.*

Если дважды щелкнуть мышью на значке нужного документа или на значке его ярлыка (Shortcut), то это приведет к вызову нужного приложения с последующей загрузкой выбранного документа.

Если рассмотреть все три указанных термина по отношению друг к другу, то получится, что *объект является наиболее общим* из них: в него могут входить и файлы, и группы файлов, и документы. Файл является более общим термином по отношению к документу.

Иконка, значок или *пиктограмма (Icon)* — графическое представление любого объекта Windows 95/98/Me. Ее внешний вид нельзя изменить без изменения внутренних параметров самой операционной системы. Иконки, как правило, сопровождаются подписями с именем того объекта, который они представляют.

Ярлык (Shortcut) — ссылка на некоторый объект Windows 95/98/Me, использующийся для удобства его вызова. Подобно всем объектам, ярлык также представляется соответствующей иконкой. Для отличия *иконка ярлыка имеет в своем левом нижнем углу стрелку* (см. рис. 9.1).

Ярлыков может быть любое количество и они могут находиться в любых местах. Независимо от того, где находится объект, для обращения к нему достаточно дважды щелкнуть по соответствующему ярлыку. После этого, если это был:

- *ярлык программы* — она запускается на выполнение;
- *ярлык документа* — запускается назначенная ему программа-редактор, которая загружает указанный документ.

Сам ярлык является специальным маленьким файлом, который выполняет указанные действия. Он имеет расширение pif (для DOS-программ) или lnk (для всего остального).

Особенностью ярлыка является то, что его графическое представление в виде иконки может быть *любым рисунком подходящего размера*. Наиболее часто такие рисунки содержатся в файлах с расширением ico, а также в Windows-программах (с типом exe) и программных модулях (с типом dll).

При удалении ярлыков соответствующий объект, на который он ссылается, не удаляется. Однако здесь, как и во всем, следует соблюдать мудрую колею осторожности, особенно для начинающего пользователя. Как мы далее увидим, операции копирования объекта и создания ярлыка очень похожи.

Поэтому при создании нескольких ярлыков для одного объекта на самом деле можно произвести ненужное размножение этого объекта, что приводит к сокращению свободного пространства на диске. Также после создания нескольких ярлыков следует быть внимательным, чтобы по неосторожности не удалить исходный объект.

В системах Windows 95/98/Me вместо понятия каталога (директория) используется термин *папка (Folder)*. Если мы говорим о способе хранения файлов на диске, то можно считать, что папка и каталог одно и то же. Однако папка — понятие более широкое:

- все каталоги диска являются папками;
- не все папки являются каталогами.

Каталог можно рассматривать как *физическое* понятие, с которым связана определенная область на диске. Папка представляет собой скорее *логическое* понятие. Существуют папки:

- которым не соответствуют области диска;
- в которых хранятся не файлы, а другие объекты.

Папки (Folders) в Windows 95/98/Me обозначают каталоги, по которым распределены все аппаратные и программные компоненты компьютера: диски, принтеры, документы, ярлыки, приложения, другие папки. Все папки образуют единую иерархическую систему, которая собственно и образует файловую систему компьютера. Для ее просмотра может быть использован Мой компьютер (My Computer) или приложение, называемое Проводник (Explorer).

Можно считать, что все компоненты компьютера размещены по своим папкам. Самая большая из них — это Рабочий стол (Desktop) (рис. 9.1). Каждый диск компьютера — тоже папка (см. рис. 9.3). В каждой из папок могут находиться другие папки, документы (Documents), приложения (Applications).

Для доступа внутрь папки нужно использовать двойной щелчок мышью на ее иконке. Поэтому для входа в папку диска C:

- сначала нужно дважды щелкнуть мышью на иконке Мой компьютер (My Computer) (см. рис. 9.3);
- затем в появившемся соответствующем окне также дважды щелкнуть на иконке папки C:.

На рис. 9.4 приведен пример окна, в котором представлены папки, размещенные на диске C:. Это только часть папок диска, поскольку остальные просто не уместились в окне. Для просмотра не уместившихся в окне папок нужно воспользоваться полосами прокрутки.

Кроме окон папок в операционной системе есть еще несколько других типов окон: окна приложений, диалоговое окна и окна справочной системы. С ними мы познакомимся в дальнейшем.

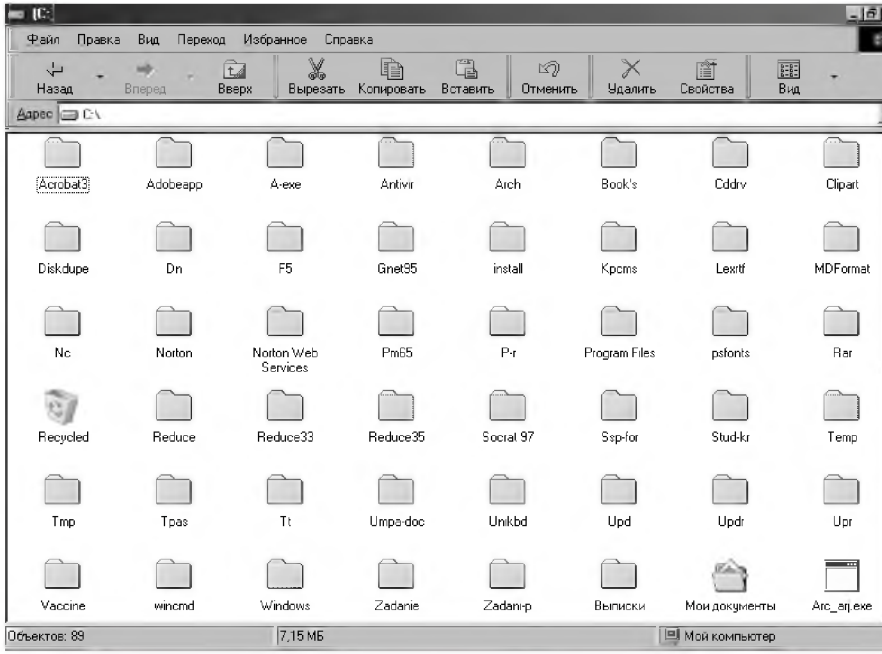

Рис. 9.4. Окно, в котором представлены значки папок (Folders), размещенных на диске C:

9.3. Окна и работа с ними

Основные принципы работы с окнами в Windows 95/98/Me остались без изменения. Поэтому можно настоятельно посоветовать ознакомиться с пп. 8.1, 8.3 и 8.5, где содержатся важнейшие сведения о системе, не повторяющиеся в данной главе. Они помогут вам лучше понять основные принципы Windows, которые в Windows 95/98/Me остались без изменения.

Внешний вид типового окна в Windows 95/98/Me представлен на примерах рис. 9.3 (окно Мой компьютер) и 9.4 (окно диска C:).

Строка заголовка (Title Bar) занимает самую верхнюю строчку в окне и содержит его название. Здесь же, в правом верхнем углу, находятся три кнопки управления окном. Они отличаются по внешнему виду и количеству от соответствующих кнопок Windows 3.1/3.11, но функции у них остались прежними:

- *кнопка свертывания* (Minimize Button), после щелчка по которой окно исчезает с экрана, превратившись в соответствующую кнопку на панели задач (Taskbar);

— *кнопка развертывания* (Maximize Button), после щелчка по которой окно разворачивается на весь экран. При этом *кнопка развертывания* заменяется на — *кнопку восстановления (Restore Button)*, щелчок по которой возвращает окно в свое первоначальное состояние;

— *кнопка закрытия* (Close Button), после щелчка по которой окно закрывается (клавиатурная комбинация для закрытия окна осталась прежней — <Alt>+<F4>).

Заметим, что если окно документа находится в нормальном состоянии (не развернуто на весь экран), то можно также *плавно* менять его размер. Для этого нужно:

- указать мышью один из углов или одну из границ окна, в результате чего указатель мыши примет вид двунаправленной стрелки;
- перетащить мышью этот угол или границу окна в нужное место экрана.

Отметим, что проще всего перетаскивать правый нижний угол окна, так как там есть специальная зона, избавляющая пользователя от необходимости точного позиционирования мыши.

В этой же Строке заголовка (Title Bar) в левом верхнем углу окна находится значок открытой папки или работающей программы, называемый *кнопкой системного меню*. После однократного щелчка по нему (или нажатия клавиш <Alt>+<пробел>) вызывается системное меню окна с набором команд, позволяющих изменить его размеры: свернуть, развернуть, восстановить или закрыть окно.

Необходимость в ее применении возникает достаточно редко, ибо пользоваться вышеописанными кнопками удобнее. Обычно пользуются им при отказе мыши, так как оно позволяет выполнить все эти операции с окном с помощью клавиатуры: нажатием клавиши <Alt> совместно с подчеркнутой буквой (см. рис. 9.5).

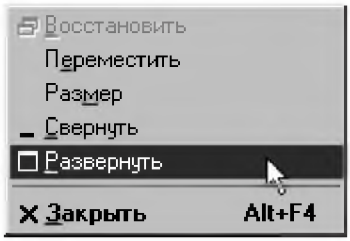

Рис. 9.5. Системное меню окна Windows 95/98/Me

Вторую строку окна Windows занимает строка меню (Menu Bar). Она содержит названия команд, при вызове любой из которых открывается доступ к подменю, соответствующему данной команде.

Состав команд, включенных в строку меню (Menu Bar), как правило, зависит от назначения окна.

Войти в строку меню можно и без помощи мыши — нажатием клавиши <Alt>, после чего перемещаться по нему можно с помощью клавиш горизонтального перемещения курсора, а для вызова соответствующего подменю использовать клавишу <Enter>. Быстрее можно сделать это, нажимая клавишу <Alt> совместно с буквой, подчеркнутой в нужной нам строке (например, <Alt>+<Ф> — вход в меню Файл).

Третью строку окна Windows занимает панель инструментов (Toolbar). На ней размещаются кнопки вызова наиболее часто исполняемых команд, специальные поля для ввода текста или выбора конкретного элемента из списка.

Следует отметить удобную особенность Windows 95/98/Me — возможность по каждой кнопке с помощью *зависания курсора* над ней получить справку в виде желтой всплывающей подсказки.

Справа и снизу вдоль окна могут быть расположены полосы (линейки, строки) прокрутки. Внешне они немного отличаются от полос прокрутки Windows 3.1/3.11, но используются аналогичным образом (см. п. 8.3.3).

Отметим также, что в зависимости от конкретной ситуации и типа окна все описанные его элементы могут частично или почти полностью отсутствовать. На экране может быть только одно активное окно, заголовок которого выделяется ярким цветом. Все остальные окна, которые могут находиться на экране, будут неактивны и их заголовки блеклого цвета.

Активное окно всегда выводится поверх других и все действия, совершаемые в данный момент, относятся именно к нему (приложение, которому оно соответствует, также называется активным).

Чтобы сделать неактивное окно активным, нужно щелкнуть мышью в любом его месте.

9.3.1. Меню Windows

Меню — это элемент управления, предоставляющий возможность выбора какого-то действия или команды из заранее подготовленного списка. Расположение и использование любых меню подчиняются единым правилам.

Для того чтобы выбрать один из пунктов меню, необходимо щелкнуть по нему мышью, после чего откроется соответствующее подменю. *При этом приняты следующие соглашения:*

- если пункт подменю написан блеклым цветом, то данная опция сейчас недоступна, ее выбор игнорируется;
- если рядом с названием пункта стоит *треугольник*, то при выборе данной опции раскроется соответствующее подменю;
- если рядом с названием пункта меню стоит *троеточие*, то при выборе данной опции раскроется соответствующее диалоговое окно (окно запроса соответствующих параметров);
- если справа от названия пункта меню указана *комбинация клавиш*, то этот пункт меню вы можете выбрать напрямую с помощью данной горячей комбинации клавиш;
- если перед названием пункта меню стоит *галочка*, то это означает, что режим, указываемый данной опцией, включен. Если вы выберете этот пункт меню, то режим будет выключен, и наоборот;
- если перед названием пункта меню стоит *кружочек*, то это означает, что из нескольких альтернативных режимов сейчас включен режим, указываемый данной опцией;
- группы опций со схожими функциями обычно отделяются горизонтальными чертами.

Практически у всех Windows-приложений существуют два пункта меню: Окно (Window) и ? (Справка, Help).

Пункт Окно (Window) отвечает за работу со вторичными окнами данного приложения. В его нижней части указывается список открытых в данный момент вторичных окон, в котором текущее активное окно отмечено галочкой. Если вы хотите сделать активным другое окно, то необходимо выбрать соответствующий пункт, щелкнув по нему мышкой. В верхней части этого меню располагаются опции, отвечающие за просмотр и позволяющие по-разному расположить вторичные окна. Их содержание у разных приложений различно.

Пункт ? (Справка, Help) позволяет получить дополнительные сведения о приложении и работе с ним.

9.3.2. Контекстное меню Windows 95/98/Me

Системы Windows 95/98/Me и программы, работающие под их управлением, спроектированы так, чтобы предсказывать ваши возможные действия при решении определенной задачи.

Как правило, команды, которые доступны в данной ситуации для данного объекта, собраны в *контекстное меню*, которое вызывается щелчком *правой* кнопки мыши.

Контекстное меню является динамическим. Его содержание зависит:

- только от конкретной ситуации (от выполняемой задачи), при которой происходит его вызов (связано с контекстом);
- от того, на каком объекте установлен указатель мыши.

Поэтому, *если вы не знаете, что делать дальше*, щелкните правой кнопкой элемент, который вызвал ваше сомнение. Есть очень большая вероятность того, что в контекстном меню вы увидите нужную команду. После этого вам стоит только ее выбрать, щелкнув по ней левой кнопкой мыши.

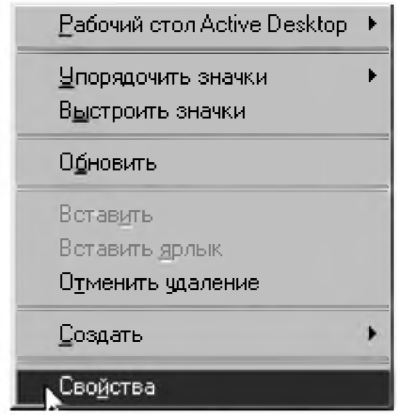

Рис. 9.6. Контекстное меню для свободного пространства Рабочего стола (Desktop)

9.3.3. Диалоговое окно и запрос Windows

Предварительно рассмотрим различные запросы Windows, которые являются еще одним средством диалога с пользователем. Это, наверное, наиболее многообразная форма диалога. Отметим, что если приложение генерирует запрос, без ответа на который продолжение работы невозможно, то:

- активным становится именно окно запроса;
- окно приложения делается неактивным.

После ответа на запрос:

- соответствующее окно запроса исчезает;
- окно приложения вновь становится активным.

В запросах и диалоговых окнах, как и в меню, *приняты аналогичные соглашения* (см. пп. 8.1 и 9.3.1).

Наиболее простым типом запроса является ***предупреждение или сообщение.*** В этом случае появляется только одна кнопка, как правило, OK. В окне слева располагается значок, говорящий о характере информации. После того как вы прочитаете данное сообщение, необходимо просто щелкнуть по кнопке OK.

Помимо простых запросов могут выдаваться окна, содержащие весьма большое количество полей. Одним из видов такого поля может быть ***запрос на ввод текста.*** Для того чтобы сделать такое поле активным (в этом случае в нем начнет мигать курсор), необходимо просто щелкнуть на нем мышью. После этого вы можете ввести с клавиатуры необходимый текст.

Если в окошке для ввода есть текст, выделенный цветом, то:

- при вводе нового текста он заменит выделенный;
- при нажатии на клавишу <BkSp> этот текст исчезнет и можно будет обычным образом вводить новый;
- для его редактирования следует щелкнуть по нему мышью — выделение исчезнет, курсор будет находиться в месте щелчка и можно будет вносить необходимые изменения.

В Windows 95/98/Me также часто используются ***диалоговые окна (Dialog Boxes)***, которые появляются на экране, когда операционной системе или какому-нибудь приложению требуется дополнительная информация для выполнения той или иной операции. Иногда ваш выбор ограничен кнопками Да (Yes), Нет (No) или OK. В других случаях вам придется задать немало параметров. Для их задания применяются разнообразные элементы управления. Рассмотрим их использование на примере диалогового окна Печать (Print) текстового редактора Microsoft Word, которое будет вам встречаться достаточно часто (см. рис. 9.7).

Предварительно отметим, что появляющиеся диалоговые окна можно перемещать обычным образом, перетащив курсором строку заголовка (верхнюю строку окна).

Однако изменить их размеры нельзя ни с помощью соответствующих кнопок (их нет на рис. 9.7), ни с помощью курсора, который не меняет свой формы при совмещении с границами окна или его углами.

Диалоговые окна можно только закрыть, щелкнув соответствующую кнопку:

- OK — внесенные изменения вступят в силу;
- × или Отмена — чтобы отказаться от изменения параметров и также закрыть окно.

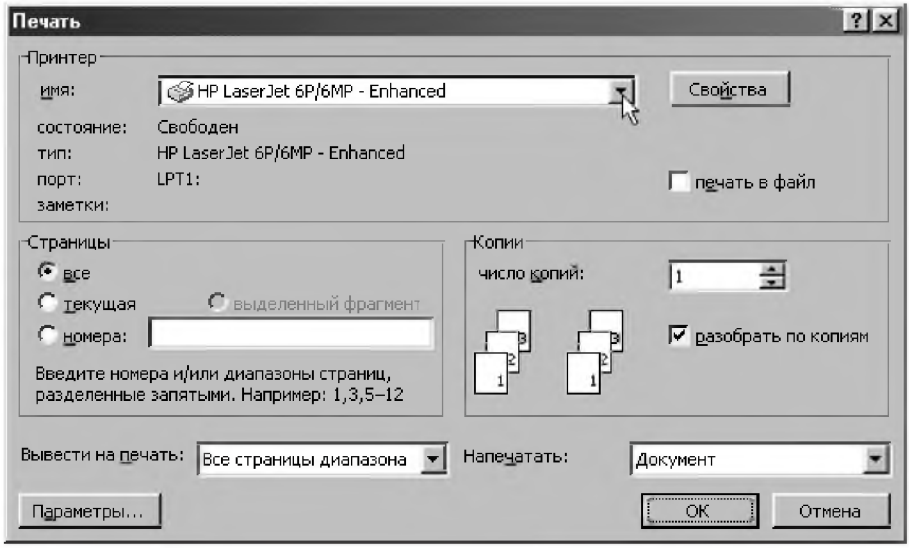

Рис. 9.7. Диалоговое окно Печать (Print) текстового редактора Microsoft Word 97

В строке заголовка (верхняя строка окна) находится также кнопка получения справочной информации об элементах окна ?.

Чтобы ею воспользоваться, следует щелкнуть эту кнопку (после чего к изображению курсора добавится восклицательный знак), а затем нужный элемент окна, чтобы прочитать краткую информацию о нем.

Раскрывающийся список (Drop-Down List Box) используется в диалоговом окне Печать (Print) текстового редактора Microsoft Word (см. рис. 9.7) трижды:

- для задания имени используемого принтера (вторая строчка сверху);
- в списках Вывести на печать и Напечатать (вторая строчка снизу).

Чтобы его раскрыть, необходимо щелкнуть мышью по кнопке со стрелкой, расположенной справа от списка.

Если в окно раскрытого списка не умещаются все строки списка, то, как обычно, появляется полоса прокрутки. Просмотр списка и выбор строки из него также производится с помощью мыши.

Одним из менее часто используемых вариантов раскрывающегося списка является просто ***список*** (List Box), который представлен сразу в раскрытом виде и предназначен также для выбора одного из включенных в этот список элементов.

Переключатель (Option Button) — это такой элемент, который позволяет щелчком мыши *выбрать одну из нескольких опций*, объединенных в группу. Он используется на рис. 9.7 в поле Страницы для указания распечатываемых страниц:

- **все**;
- **текущая**;
- **выделенный фрагмент** — эта опция в настоящий момент недоступна, так как в тексте нет выделенного участка. Поэтому она выглядит нечетко (серого цвета);
- **номера** — после этой опции находится ***строка ввода (Text Box)***, которая предназначена для непосредственного ввода текстовой информации (в данном случае для указания номера и/или диапазона страниц, разделенных запятыми). Обычно при вводе текста в этой строке можно использовать некоторые приемы редактирования (например, выделение, копирование в буфер, вставка, удаление символов).

Счетчик (Spin Box) позволяет установить нужное числовое значение. Для этого достаточно щелкнуть мышью по одной из кнопок  , расположенных в правой части счетчика:

- щелчок по *верхней* кнопке увеличивает значение, установленное в окне счетчика;
- щелчок по *нижней* — уменьшает.

Отметим, что многие счетчики допускают ввод нужного значения с помощью клавиатуры.

Для этого достаточно щелкнуть мышью непосредственно в окне счетчика, после чего можно вводить данные.

Счетчик используется на рис. 9.7 для указания нужного числа в поле число копий.

Флажки (Check Boxes) используются для того, чтобы устанавливать или отменять определенные режимы работы, для чего нужно установить или отменить соответствующую опцию, щелкнув по ее названию.

Отметим, что состояние любого флажка может меняться независимо от состояния других флажков (в отличие от переключателей).

Флажок используется на рис. 9.7 дважды:

- в поле разобрать по копиям соответствующая опция установлена;
- в поле печать в файл опция снята.

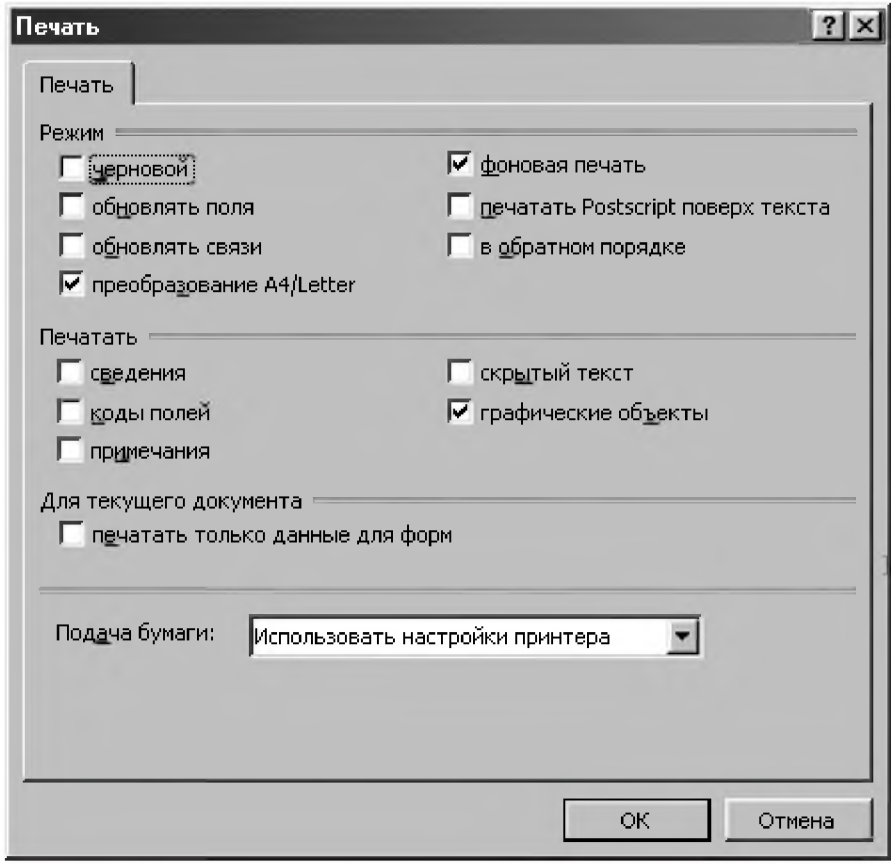

Рис. 9.8. Диалоговое окно кнопки Параметры окна Печать (Print) текстового редактора Microsoft Word 97

Чтобы показать возможность использования самого разнообразного сочетания флажков на рис. 9.8 показано диалоговое окно, которое появляется на экране после нажатия кнопки ***Параметры*** на рис. 9.7.

Если настраиваемых параметров слишком много, то они обычно располагаются по нескольким закладкам, объединенным в одно диалоговое окно. Вверху (во второй строке) выводятся названия закладок: щелкнув по любому из них, вы переместите соответствующую закладку наверх. Справа или внизу располагаются кнопки, позволяющие либо принять, либо отменить сделанные изменения.

С этой точки зрения можно рассматривать рис. 9.8 как окно с одной закладкой ***Печать***.

Чтобы увидеть их несколько, следует нажать на копку ***Свойства*** (Property) на рис. 9.7, что мы предлагаем сделать самостоятельно.

Замечание. Однако мы не предлагаем вам что-нибудь изменять там. Следует отметить, что существуют сотни опций, которые определяют текущий режим работы Windows и которые можно при желании изменять. Количество же различных их комбинаций просто не поддается учету.

В исходном состоянии операционная система функционирует в соответствии с настройками, сделанными разработчиками или поставщиками Windows *по умолчанию (default settings)*.

Начинающий пользователь не подозревает о практически безграничных возможностях, позволяющих ему изменять текущий режим работы системы. Поэтому проблем с настройками у него не возникает.

Более опытный пользователь часто, к сожалению, начинает работу с того, что принимается активно менять все, что только поддается модификации. Этот процесс становится самоцелью и продолжается до тех пор, пока он полностью не потеряет все исходные установки системы, не достигнув желаемой гармонии и потратив на это кучу бесценного времени.

И только профессионал один раз в начале работы сделает нужные для себя настройки, после чего вообще о них забывает, сосредоточившись на выполняемой работе.

Ему никогда не придет в голову длительное время исследовать различные варианты оптимальной работы.

Ему не нужно запоминать технические подробности, *нужно только эффективно работать*, а при необходимости логика и опыт всегда подскажут путь к решению вопроса.

Поэтому и мы возьмем пример с профессионального подхода к работе, сосредоточившись на освоении практической работы на ПК и уделяя внимание лишь действительно необходимым изменениям настроек.

9.4. Кнопка Пуск (Start) и Главное меню (Start Menu)

Получить доступ к ***главному меню*** можно:

- посредством щелчка по кнопке ***Пуск***, расположенной на панели задач;
- нажатием клавиатурной комбинации <Ctrl>+<Esc>.

Главное меню (Start Menu) является уникальным элементом управления для выполнения практически любого действия и любой операции, связанной с настройкой, поиском, запуском приложений и справочной системой Windows. Его использование существенно облегчает жизнь начинающим. Достаточно бегло изучить содержимое ***Главного меню*** (Start Menu), и все возможности Windows уже перед вами (рис. 9.9).

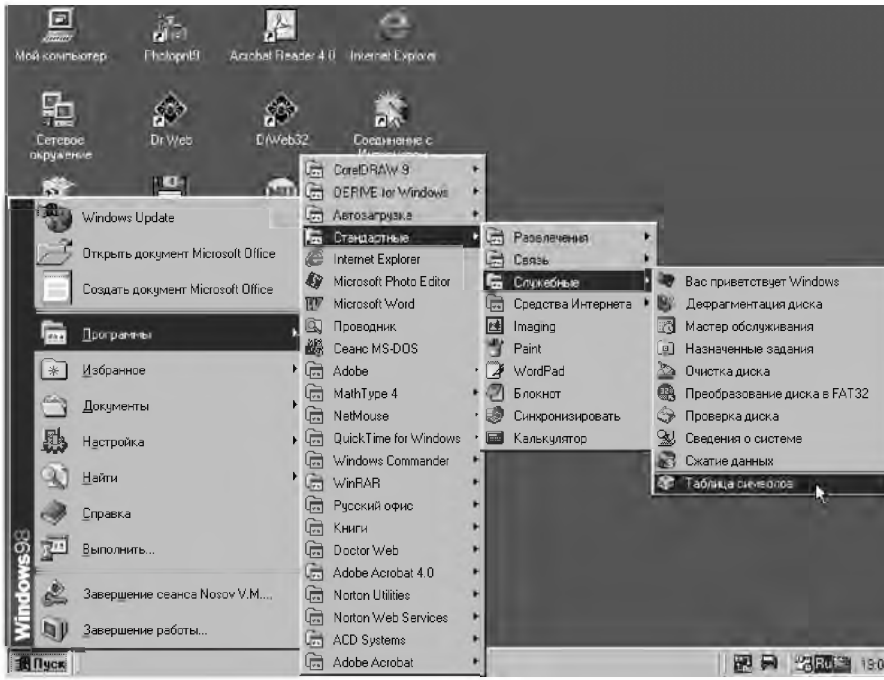

Рис. 9.9. Кнопка Пуск (Start) и Главное меню (Start Menu) являются основными элементами управления Рабочего стола (Desktop) Windows 95/98/Me

Главное меню (Start Menu) содержит набор основных команд Windows. Некоторые из них отмечены специальным маркером в виде стрелки. При выборе таких команд (помещении на них указателя мыши) автоматически открывается еще одно дополнительное меню, которое, в свою очередь, также может содержать команды, отмеченные таким же маркером.

Если же в команде меню стрелки нет, то ее можно сразу выполнить, нажав клавишу <Enter> или щелкнув мышью на названии команды.

На рис. 9.9 приведен пример четырех раскрытых меню, которые вложены одно в другое. Например, чтобы добраться до приложения Таблица символов (Table Symbols), необходимо открыть меню команды Программы (Programs), затем выбрать пункт Стандартные (Accessories), затем Служебные (System Tools) и в появившемся дополнительном меню найти нужное приложение Таблица символов (Table Symbols).

Напомним, что говоря о командах, доступ к которым осуществляется через вложенные меню, мы *будем указывать весь путь до этой команды*. Для рассматриваемого примера это будет иметь следующий вид: Пуск → Программы → Стандартные → Служебные → Таблица символов (Start → Programs → Accessories → System Tools → Table Symbols).

Это может показаться несколько громоздким, но зато оказывается очень полезным и удобным способом, так как кратко и точно описывает последовательность действий для вызова соответствующей команды.

Рассмотрим подробнее пункты главного меню и связанные с ними действия:

- Программы (Programs) — с помощью этого пункта меню можно запустить стандартные программы, входящие в состав Windows 95/98/Me, а также все приложения, установленные в вашей системе. При выборе этого пункта (зависании на нем указателя мыши) перед вами раскроется дополнительное подменю, в котором будут представлены отдельными подпунктами установленные в вашей системе приложения (чтобы запустить одно из них, необходимо открыть соответствующее подменю и затем щелкнуть на значке необходимой вам программы), а также стандартные программы, входящие в состав Windows 95/98/Me (их запуск осуществляется также);
- Документы (Documents) — этот пункт главного меню содержит последние пятнадцать документов, с которыми вы работали. Вы можете снова продолжить обработку одного из них, для этого достаточно щелкнуть на нем мышью;
- Настройка (Setting) — в этом пункте главного меню сосредоточены все инструменты по настройке Windows 95/98/Me (см. замечание в конце п. 9.3.3);
- Поиск (Find) — позволяет найти любую папку или файл;
- Справка (Help) — при выборе этого пункта (щелчка на нем мышью) осуществляется вызов встроенной справочной системы Windows 95/98/Me. Вызываемое диалоговое окно содержит три вкладки:

- Содержание (Contents) — позволяет обратиться к упорядоченному списку всех разделов справочной системы;
- Предметный указатель (Index) — дает возможность быстрого получения справки по термину, название которого пользователь может либо ввести с клавиатуры, либо выбрать из предложенного списка;
- Поиск (Find) — позволяет узнать, в каких разделах справочной системы встречается интересующее вас слово, и вывести на экран содержимое этих разделов;

- Выполнить (Run) — позволяет запустить любую программу;
- Завершение сеанса (Log Off) — если на компьютере зарегистрировано несколько пользователей со своими рабочими столами и личными настройками, то данный пункт меню позволяет выйти из одного стола для входа в другие;
- Завершение работы (Shut Down) — выбрав этот пункт, вы сможете закончить работу ОС Windows 95/98/Me.

9.5. Приемы управления и навигации

9.5.1. Приемы управления

Со всеми объектами и элементами управления в Windows можно справиться всего двумя кнопками мыши. Однако этими двумя кнопками можно сделать так много, что следует затратить время и еще раз уточнить приемы управления в Windows 95/98/Me.

В Windows 95 существовал только один стиль управления, называемый *классическим* в Windows 98/Me. В нем *одним щелчком (Click)* выполняют:

- все операции с элементами управления, включая запуск программ и открытие документов из Главного меню (Start Menu);
- подготовительные операции с объектами (выбор или выделение объекта перед дальнейшими операциями);
- запуск программ с панели быстрого запуска в Windows 98/Me.

Двойным щелчком (Double-click) выполняют:

- открытие окон папок, документов, запуск приложений с помощью значков Рабочего стола (Desktop);
- операции со значками, находящимися на панели индикации.

Для запоминания можно сформулировать простое правило: ***в классическом стиле с элементами управления работают одним щелчком, а с объектами — двумя щелчками.***

При этом под словом *щелчок* всегда понимается щелчок *основной кнопкой* мыши, под которой почти всегда подразумевается *левая* кнопка (без специальной перенастройки).

Левая кнопка используется также для выполнения операции *перемещения (Drag)*: установив курсор на нужный объект или элемент изображения, нажать и удерживать кнопку мыши при ее перемещении. При этом объект станет передвигаться по экрану синхронно с перемещением курсора и после отпускания кнопки мыши зафиксируется на новом месте.

При перемещении окна Windows указатель мыши должен быть установлен в пределах строки заголовка.

При перемещении объектов из окна одной папки в окно другой папки происходит изменение размещения файлов на жестком диске, так что этим методом можно работать с файловой системой компьютера.

Однако удобнее для этой цели использовать *специальное перемещение*, которое отличается от обычного только тем, что выполняется *правой кнопкой* мыши. По его окончании открывается меню, состоящее из трех пунктов:

- *Скопировать*;
- *Переместить*;
- *Создать ярлык*.

Теперь у нас есть не только более богатый выбор, но мы будем также четко осознавать, какую именно операцию мы выполняем.

Отметим, что ***все приемы управления и операции с объектами Windows работают не только в самой операционной системе, но и в ее приложениях***.

Поэтому, чтобы переместить или скопировать любой фрагмент текста (который также является объектом Windows), например, в текстовом редакторе Microsoft Word:

- его следует предварительно выделить;
- переместить выделенный фрагмент с использованием *специального перемещения* в нужное место;
- по окончании перемещения в появившемся меню выбрать себе тот пункт, который необходим в конкретном случае (при этом вам будет предоставлен даже гораздо более широкий выбор).

Это огромное достоинство определяется той формализацией системы, четкость определения понятий которой, возможно, раздражала вас не только в п. 9.2.

Стиль Web — это стиль управления, принятый в Интернете. Он может использоваться только в Windows 98/Me. Если в системе включен стиль управления *Web* — то *все подписи под значками* (на Рабочем столе (Desktop) и в окнах папок) *изображаются с подчеркиванием.*

Его отличия от классического стиля следующие:

- двойной щелчок не используется. Файлы открываются, а программы запускаются не двойным, а обычным щелчком на значке;
- выделение объектов и подготовка их к использованию выполняется вообще без щелчка. Достаточно просто навести указатель мыши на значок, и объект будет выделен, о чем свидетельствует изменение цвета его подписи.

Рассматривая основные отличия стиля *Web* от классического, можно сказать, что в нем нет разницы в работе с элементами управления системы и с ее объектами, а в классическом стиле управления такая разница есть.

В Windows 98/Me пользователь можно также создать свой персональный стиль управления на основе личных предпочтений, используя то, что ему кажется лучшим в *классическом* стиле и *Web*.

Мы же, из уважения к традиции и для единообразия изложения материала для Windows 95/98/Me, *будем везде использовать классический стиль*.

9.5.2. Запуск приложений

Для запуска приложений вы можете нажать кнопку ***Пуск***, выбрать пункт главного меню Программы (Programs), а затем щелкнуть по значку нужного вам приложения, найдя его в соответствующем подменю.

Если же ярлык нужного вам приложения находится на рабочем столе, то для запуска приложения следует дважды щелкнуть по этому ярлыку.

Если нужного приложения нет ни на рабочем столе, ни в меню Программы (Programs), то через Мой компьютер (My computer) или Проводник следует войти в нужную папку (обычно одноименную с приложением).

Затем следует дважды щелкнуть по значку нужного приложения (с типом .EXE), после чего оно запустится.

Интересный способ открытия документа одновременно с запуском соответствующего приложения связан с использованием операции *перемещения (Drag)*.

Для его реализации значок документа следует перетащить на значок соответствующего ему приложения. При этом документ откроется, причем сразу в том приложении, на значок которого его перетащили.

9.5.3. Настройка мыши и клавиатуры

Для настроек операционной системы и аппаратных устройств служит специальная папка — Панель управления (Control Panel) (рис. 9.10). Ее открывают любым из двух способов:

- непосредственно из окна Мой компьютер (My Computer);
- с помощью Главного меню (Start Menu) командой Пуск → Настройка → Панель управления (Start → Settings → Control Panel).

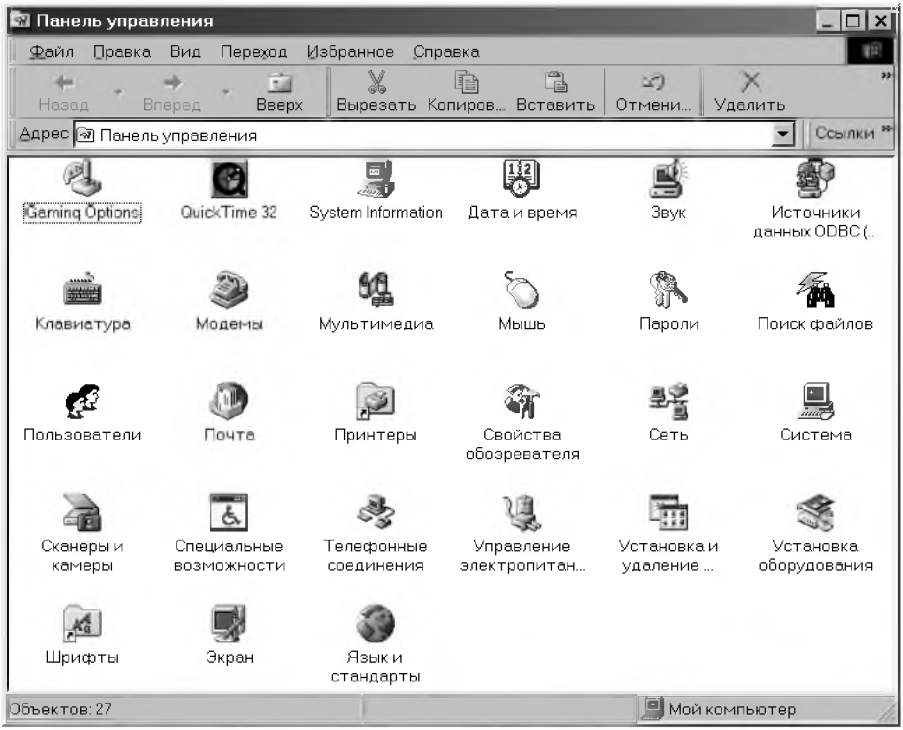

Рис. 9.10. Панель управления (Control Panel) — основное средство для настройки объектов Windows

Двойной щелчок на значке ***Мышь*** открывает соответствующее диалоговое окно для настройки ее свойств. В этом окне присутствует несколько вкладок. Основными для нас являются вкладки:

- ***Кнопки мыши*** — ее элементы управления служат для настройки действия кнопок мыши. Очень полезно правильно *настроить скорость двойного нажатия*. Система должна отличать два одинарных щелчка и один двойной щелчок по интервалу времени между ними, которым можно управлять с помощью движка Скорость двойного нажатия. Проверить установку можно здесь же, сделав двойной щелчок в Области проверки и посмотрев на результат;
- ***Перемещение*** — ее элементы управления позволяют регулировать чувствительность манипулятора, которым можно управлять с помощью движка ***Скорость перемещения указателя***.

Двойной щелчок на значке ***Клавиатура*** откроет соответствующее диалоговое окно ***Свойства: Клавиатура***. В этом окне присутствует несколько вкладок:

- ***Скорость*** — ее элементы управления служат для настройки параметров клавиатуры, которыми можно управлять с помощью перемещения трех соответствующих движков:
  - ***Задержка перед началом повтора символа***;
  - ***Скорость повтора*** (проверить установку можно здесь же, в специальном поле);
  - ***Скорость мерцания курсора***;
- ***Язык*** — ее элементы управления служат для настройки *языковых раскладок* (рис. 9.11), которые определяют способ закрепления символов национальных алфавитов за конкретными клавишами клавиатуры. Здесь же выбирают:
  - ***Установленные языки и раскладки клавиатуры*** — если в этом списке какой-то раскладки не хватает, то следует:
    - щелкнуть на кнопке Добавить;
    - вставить дистрибутивный диск с операционной системой Windows 95/98/Me;
    - нужная раскладка будет перенесена с этого диска. Лишние раскладки удаляются щелчком на кнопке Удалить;
  - ***Назначить используемым по умолчанию*** — с помощью этой кнопки можно одну из используемых раскладок сделать основной.

Для этого ее следует выделить в списке и нажать эту кнопку. Заметим, что при использовании большинства английских прикладных программ удобнее основной назначить английскую раскладку, для выпущенных в России или русифицированных — русскую. Если команды в меню будут отражаться непонятными символами, то обратите внимание на соответствие используемой программы с основной раскладкой;

- ***Переключение раскладок*** — в этом поле выбирают удобную для вас соответствующую пару клавиш, нажатием которой вы сможете переключаться с русского языка на английский и обратно;
- ***Отображать индикатор языка на панели задач*** — этот флажок обычно обязательно рекомендуют установить. Тогда на панели индикации, что на правом краю Панели задач, появится значок Ru, свидетельствующий о том, какая раскладка включена в данный момент:
  - щелчок *левой* кнопки мыши на значке индикатора открывает небольшое меню для переключения между раскладками, что особенно удобно при наличии на компьютере трех и более раскладок одновременно;
  - щелчок *правой* кнопки мыши на значке индикатора открывает небольшое меню для вызова диалогового окна Свойства: Клавиатура без необходимости вызывать окно Панель управления, что тоже очень удобно.

Замечание. В зависимости от модификации операционной системы Windows 95/98/Me возможны незначительные отличия во внешнем виде и надписях диалоговых окон. Например, в диалоговом окне ***Свойства: Клавиатура*** (рис. 9.11):

- вместо ***Используется по умолчанию*** может быть надпись ***Основной язык***. В этом случае кнопка ***Назначить используемым по умолчанию*** будет называться ***Сделать основным*** и располагаться не по центру, а в правой части диалогового окна;
- флажок ***Отображать индикатор языка на панели задач*** может называться ***Вывести индикатор*** и т. п.

Во всех подобных случаях, как и всегда в жизни, нас должен выручать обычный здравый смысл, подсказывающий нам, что все это просто разные названия одних и тех же понятий.

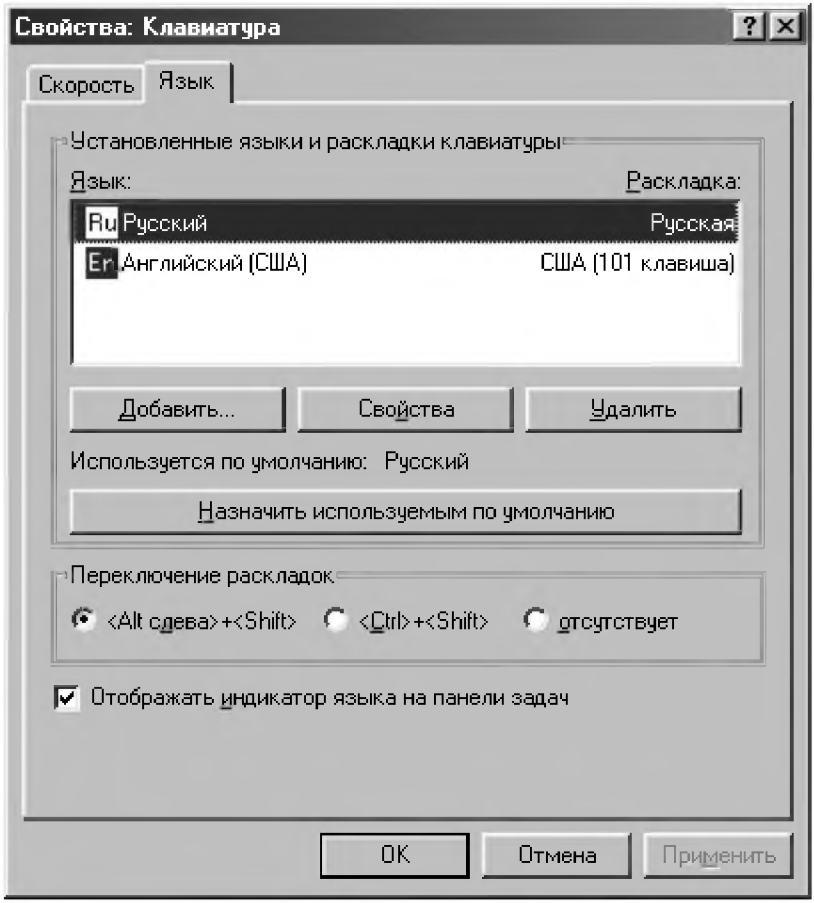

Рис. 9.11. Элементы управления для настройки раскладки клавиатуры

9.5.4. Навигация в структурах Мой компьютер (My Computer) и Проводник (Windows Explorer)

При работе в Windows 95/98/Me пользователь может применить два средства, каждое из которые позволяет получить представление о размещении объектов операционной системы. Это ***Мой компьютер (My Computer)*** и ***Проводник (Windows Explorer)***.

Мой компьютер (My computer) — это специальная папка, расположенная непосредственно на ***Рабочем столе (Desktop)***. Основное его назначение состоит в том, чтобы в любой момент предоставить пользователю доступ ко всем папкам и ресурсам компьютера. Он позволяет вам посмотреть содержание любой папки на дисках и найти любой нужный файл, вызвать ***Панель управления (Control Panel)***, открыть папку ***Принтеры (Printers)*** и т. д.

Можно сказать, что ***Мой компьютер (My computer)*** является самой главной папкой компьютера. Войти в эту папку можно с помощью двойного щелчка по значку ***Мой компьютер (My computer)***, вызвав на экран соответствующее окно (рис. 9.3). Щелкнув в нем на значке диска ***C:*** — откроется окно с его содержимым (рис. 9.4). Действуя и далее таким же образом, мы можем добраться до любого значка файла.

Это самый простой метод навигации, с которого удобнее всего начинать начинающим. Поместив на ***Рабочий стол (Desktop)*** нужные окна, легко и наглядно можно выполнять перемещение и копирование файлов между ними (проще всего специальным перетаскиванием — правой кнопкой мыши).

Однако режим, когда каждая папка открывается в новом окне, не всегда удобен. Если не надо заниматься копированием и перемещением, а файл достаточно найти и открыть, то есть смысл включить режим, когда каждая папка открывается в одном и том же окне.

В Windows 95 для изменения свойств окна папки служит команда ***Мой компьютер → Вид → Параметры*** , открывающая соответствующее диалоговое окно с несколькими вкладками. Выбор используемого режима осуществляется на вкладке ***Папка (Folder)*** в зависимости от выбора переключателя:

- **Открывать для каждой следующей папки отдельное окно;**
- **Просматривать содержимое отдельных папок в одном окне.**

В последнем случае затрудняется обратная навигация — возврат по пройденному маршруту, а также копирование и перемещение, поскольку все промежуточные окна оказываются закрытыми.

В этом случае для *обратной навигации* следует пользоваться:

- кнопкой Переход на один уровень вверх  , расположенной на панели инструментов;
- клавишей <BACKSPASE>, если панель инструментов скрыта.

Чтобы с помощью инструмента Мой компьютер (My Computer) *копировать, перемещать или создавать ярлыки объектов* нужно:

1. Открыть нужную папку (в которой находится копируемый или перемещаемый *объект-источник*), начиная от папки Мой компьютер (My Computer), расположенной на Рабочем столе (Desktop).
2. Также начиная от папки Мой компьютер (My Computer), открыть окно нужной папки–приемника, куда будет происходить копирование или перемещение.
3. Выполнить соответствующую операцию копирования или перемещения между двумя окнами, находящимися открытыми перед нами. Например, переместить специальным перетаскиванием (при нажатой правой кнопке мыши) *объект-источник* на любое свободное место папки-приемника и отпустить кнопку мыши.
4. В открывшемся контекстном меню выбрать нужный пункт:
   - Переместить (Move) — при этом исходный объект *удаляется и перемещается* в объект-приемник;
   - Копировать (Copy) — при этом исходный объект останется на прежнем месте, а его *копия* помещается в объект-приемник;
   - Создать ярлык(и) (Create Shortcut(s)) — при этом исходный объект останется на прежнем месте, а его *ярлык* помещается в объект-приемник.

В Windows 98 лишь в смешанном стиле управления, основанном на личных предпочтениях, можно настроить режим открытия окон, а в классическом стиле вообще нет возможности выбора: при всех условиях каждая папка открывается в новом окне. Поэтому для обратной навигации, копирования и перемещения следует также использовать вышеописанные методы.

Проводник (Windows Explorer) предназначен для отображения всех ресурсов компьютера (папок, документов, программ, других приложений) и доступа к ним.

Его удобно использовать, когда нужно просмотреть иерархическую структуру папок (дерево каталогов), быстро перейти к любому объекту, просмотреть его содержимое или вызвать его двойным щелчком на его значке.

При вызове нужного объекта, если им является:

- *папка* — то она раскрывается;
- *документ* — запускается создавшее его приложение с последующей загрузкой самого документа;
- *приложение* — происходит его запуск.

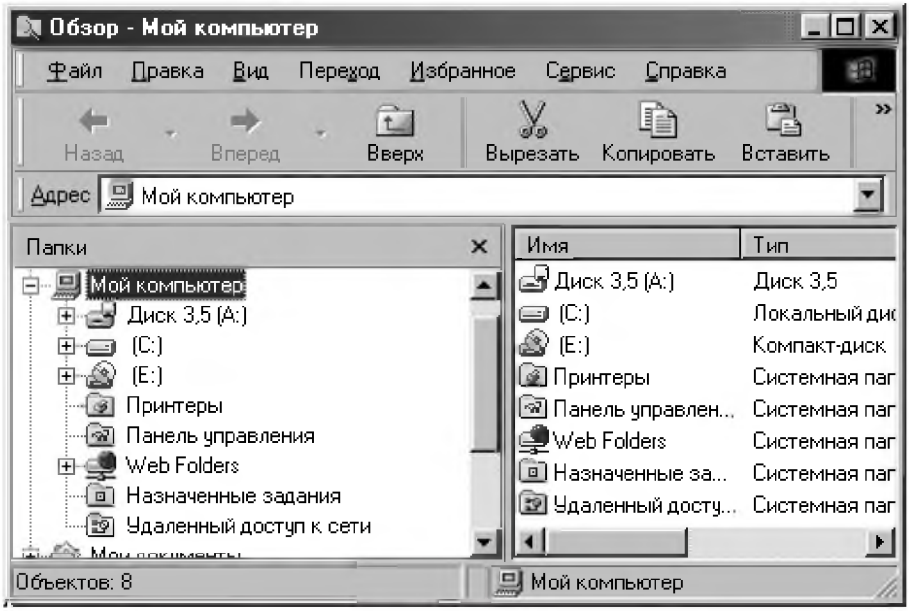

Рис. 9.12. Окно ***Проводник – Мой компьютер (Exploring – My computer)****, отображающее содержимое той же папки* ***Мой компьютер (My computer)****, что и на рис.* ***9.3***

Проводник (Windows Explorer) можно открыть разными способами:

1. Как и любую программу с помощью Главного меню (Start Menu) командой Пуск → Программы → Проводник (Start → Programs → Windows Explorer).
2. Щелкнуть *правой* кнопкой мыши (на кнопке Пуск (Start), на любом реквизитном значке Рабочего стола (Desktop) или на значке любой папки) и в появившемся контекстном меню выбрать пункт Проводник (Windows Explorer). В этом случае Проводник открывается таким образом, что объект, с помощью которого он был вызван, уже присутствует в окне программы и его не надо разыскивать. Это и представлено на рис. 9.12 после щелчка правой кнопкой мыши на значке Мой компьютер (My Computer) в Windows 98.
3. Двойным щелчком *левой* кнопкой мыши с нажатой клавишей <Shift>.

Функции Проводника (Windows Explorer) подобны функциям окна папки, которое вы открываете из окна Мой компьютер (My computer).

Однако окно Проводника разделено на две панели. Левая называется *панелью папок*, правая — *панелью содержимого*.

В его левой панели отображены компьютеры, диски и папки (то, что часто называют *деревом каталогов* или *папок* данного компьютера), к которым Проводник вас немедленно проведет.

В правой панели раскрывается содержимое объекта, выделенного в левой панели.

Если рядом с именем папки расположен узел в виде знака "+", то щелчком по нему можно "развернуть" эту ветвь дерева папок и показать все подпапки на той же левой панели, какие она содержит. При этом знак "+" сменится на "–". Щелкнув по нему, вы скроете подпапки.

Сворачивая и разворачивая ветви, мы можем добраться до любой папки на левой панели, после чего папку можно раскрыть или закрыть щелчком на ее значке.

Чтобы с помощью Проводника (Windows Explorer) *копировать, перемещать или создавать ярлыки объектов нужно:*

1. Найти на *правой* панели *объект-источник* (перемещаемый или копируемый файл) и выделить его однократным щелчком левой кнопкой мыши.
2. Отыскать на *левой* панели *объект-приемник* (диск и папку), куда его следует скопировать или переместить. При поиске *приемника* следует разворачивать папки, но не раскрывать. Это выполняется щелчками на узлах (знаках "+" и "–"), а не на значках папок, после которых папка-приемник раскроется и мы потеряем объект-источник.
3. Переместить специальным перетаскиванием (при нажатой правой кнопке мыши) *объект-источник на значок объекта-приемника*. Операция требует аккуратности. Отпускать кнопку мыши надо точно в тот момент, когда объект-приемник на левой панели изменяет цвет, то есть выделяется.
4. В открывшемся контекстном меню выбрать нужный пункт:
   - ***Переместить (Move)*** — при этом исходный объект *удаляется и перемещается* в объект-приемник;
   - ***Копировать (Copy)*** — при этом исходный объект останется на прежнем месте, а его *копия* помещается в объект-приемник;
   - ***Создать ярлык(и) (Create Shortcut(s))*** — при этом исходный объект останется на прежнем месте, а его *ярлык* помещается в объект-приемник.

Отметим, что весьма удобной альтернативой ***Проводнику (Windows Explorer)*** является программа **Windows Commander**. Она сочетает в себе:

- графический интерфейс Windows с развитыми возможностями и привычной компоновкой Norton Commander высоких версий;
- имеет высокую гибкость и настраиваемость.

9.5.5. Использование команд строки меню и панели инструментов

Непосредственно под строкой заголовка во второй строке сверху располагается *строка меню* (Menu Bar). Это типичный элемент управления, характерный для рабочих окон большинства приложений Windows и для всех окон папок. Каждый из пунктов *строки меню* (Menu Bar) открывает ниспадающее меню, пункты которого представляют команды для операций с объектами, представленными в окне.

Следует отметить, что это не самое удобное средство управления и без него можно обойтись, используя операции с мышью и контекстное меню. Однако средства *строки меню* (Menu Bar) обладают уникальной особенностью: *с их помощью можно сделать абсолютно все, что можно сделать в программе*. Приступая к изучению нового приложения Windows, можно больше никуда и не заглядывать, а сосредоточиться на содержании ниспадающих меню, открываемых из этой строки.

Пункт меню ***Файл (File)*** содержит команды для операций с объектами. Его содержание зависит от того, какой объект в данный момент выделен на панели содержимого. Он позволяет совершить различные действия с одним или несколькими объектами папки:

- ***Открыть*** (Open);
- ***Создать ярлык*** (Create Shortcut);
- ***Переименовать*** (Rename);
- ***Удалить*** (Delete) и т. д.

Если объектом является диск, то в списке будет присутствовать весьма специфическая команда ***Форматировать***, которой мы рекомендуем пользоваться только для гибких дисков.

Пункт меню ***Правка (Edit)*** содержит команды, имеющие непосредственное отношение к обслуживанию файловой системы (копирование объектов, перемещение, их выделение и т. п.). Он позволяет:

- отменить последнее действие (***Отменить*** (Undo));

- поместить объекты в буфер обмена с удалением (***Вырезать*** (Cut)) или сохранением (***Копировать*** (Copy)) оригинала соответственно;
- вставить содержимое из буфера обмена (***Вставить*** (Paste));
- выделить все объекты в папке (***Выделить все*** (Select All));
- сделать все выделенные объекты невыделенными, а невыделенные выделенными (***Обратить выделение*** (Invert Selection)).

Отметим, что в операциях, реализуемых с помощью этого меню, активное участие принимает *буфер обмена (Clipboard)* (см. п. 9.6.3).

Пункт меню ***Вид (View)*** содержит команды, управляющие внешним видом рабочего окна и отображением его содержимого. Кроме того, в меню ***Вид (View)*** расположены команды для включения или отключения некоторых элементов окна:

- ***Панель инструментов*** (Toolbar) позволяет скрыть/показать панель инструментов окна;
- ***Строка состояния*** (Status Bar) позволяет скрыть/показать строку состояния внизу окна (в строке состояния выводится различная служебная информация);
- ***Крупные значки*** (Large Icons), ***Мелкие значки*** (Small Icons), ***Список*** (List) и ***Таблица*** (Details) отвечают за размер и представление значков в окне;
- ***Упорядочить значки*** (Arrange Icons) и ***Выстроить значки*** (Line Up Icons) позволяют в определенном порядке расположить значки в окне;
- ***Обновить*** (Refresh) позволяет принудительно обновить содержимое окна;
- ***Параметры*** (Options) отвечает за различные варианты представления информации.

Пункт меню ***Сервис (Tools)*** содержит дополнительные средства, позволяющие более эффективно выполнять некоторые операции. Так, например, в программе ***Проводник (Windows Explorer)*** это меню содержит команду ***Найти (Find)***, с помощью которой запускается поисковая система Windows.

Пункт меню ***Справка (Help)*** присутствует в строке меню в виде вопросительного знака "?". Он стандартен для большинства приложений Windows.

Обычно в приложениях он ведет к справочной системе самого приложения, но в ***Проводнике (Windows Explorer)*** он действует шире и

открывает доступ ко в/сей справочной системе Windows, также как и пункт ***Справка (Help) Главного меню*** (Start Menu) (см. п. 9.4).

Панель инструментов (Toolbar) является характерным стандартным элементом окон папок и большинства приложений Windows. Она располагается непосредственно под строкой меню (в третьей строке сверху). На ней размещается комплект элементов управления. Панели инструментов приложений Windows нестандартны и состав их инструментов зависит от назначения приложения. Если в окне программы ***панель инструментов*** (Toolbar) не присутствует, то ее отображение можно включить в меню ***Вид (View)***.

Все кнопки и прочие элементы панели соответствуют командам строки меню (Menu Bar) и командам, доступ к которым можно получить из контекстного меню. Поэтому панель инструментов (Toolbar) не позволяет сделать ничего такого, чего нельзя было бы сделать другими средствами. Она просто представляет удобную возможность для работы с мышью.

Название элемента управления можно получить с помощью всплывающей подсказки, используя метод зависания курсора над соответствующей кнопкой. Подробнее использование панели инструментов (Toolbar) мы рассмотрим при изучении системы компьютерной математики DERIVE (см. п. 10).

9.6. Операции с объектами Windows 95/98/Me

9.6.1. Выделение объектов

Любой объект, перед тем как с ним будет выполнено какое-либо действие, должен быть выделен, что легко узнать по характерному изменению цвета значка и его метки. В Windows 95/98/Me есть несколько вариантов выделения объектов. Можно выделить (с использованием рассматриваемого классического стиля управления):

- *один объект*, щелкнув по нему левой клавишей мыши;
- *несколько объектов, размещенных в произвольном порядке*, для чего по ним следует щелкать с нажатой клавишей <Ctrl>;
- *несколько объектов, расположенных последовательно:*
  - для чего следует щелкнуть по первому и — с нажатой клавишей <Shift> по последнему;

- можно также растянуть мышью прямоугольную область в окне, и все попавшие в нее объекты выделятся. Однако начинать такое растягивание надо не со значка объекта, а с пустого места в окне, иначе Windows подумает, что вы хотите перетащить объект на другое место;

- *все объекты в папке:*
  - совместным нажатием клавиш <Ctrl>+<A> (<Ctrl>+<Ф>);
  - с использованием меню командой Правка → Выделить все (Edit → Select All).

Если вам нужно *отменить выделение*, щелкните мышью на любом свободном месте Рабочего стола (Desktop) или окна, в котором производится выделение объектов. Для того, чтобы снять выделение только с одного объекта помеченной группы, нажмите клавишу <Ctrl> и щелкните на этом объекте мышью.

Если вам нужно *инвертировать выделение* (невыделенные объекты сделать выделенными и наоборот) — выполните команду Правка → Обратить выделение (Edit → Invert Selection). Правда, последнее невозможно при выделении объектов на Рабочем столе (Desktop).

9.6.2. Запуск объектов и программ

Выделенные объекты можно разом запустить — двойным щелчком мыши или командой Открыть (Open) в меню Файл (File) (мы будем, как всегда, писать командой Файл → Открыть (File → Open)). После этого, если объектами будут:

- *программы* — то они все по очереди запустятся;
- *документы* — то загрузятся в свои соответствующие редакторы;
- *документы многооконного редактора* (такого, как Word) — загрузятся в разные его окна.

Следует отметить, что *для запуска выделенной группы объектов дважды щелкать надо не по самому значку, а возле него*. При этом нельзя отпускать клавишу <Ctrl> (или <Shift> в зависимости от используемого способа выделения — см. п. 9.6.1). Если этого не сделать, то выделение снимется и запустится всего одна программа.

Файлы неизвестного типа по двойному щелчку ни запуститься, ни загрузиться никуда не могут. После этого просто появится диалоговое окно

(типа изображенного на рис. 9.13). Это окно программы-регистратора документов.

Теперь в запросе на ввод текста в поле Описание файлов вводим название документа нового типа, под которым он будет в дальнейшем фигурировать в системе. Затем в списке Выберите программу для открытия файла выделяем нужную программу-редактор и нажимаем кнопку OK. После этого регистрация совершается: непонятные файлы становятся для системы вполне определенными документами.

Если же в списке нет нужной программы, нажимаем кнопку Другая... (Other...). В этом случае в появившемся диалоговом окне придется еще указать, где находится подходящий редактор.

Кроме двойного щелчка на значке программы или ее ярлыка, операционная система Windows 95/98/Me предоставляет несколько различных способов для запуска программ (см. также п. 9.5.2):

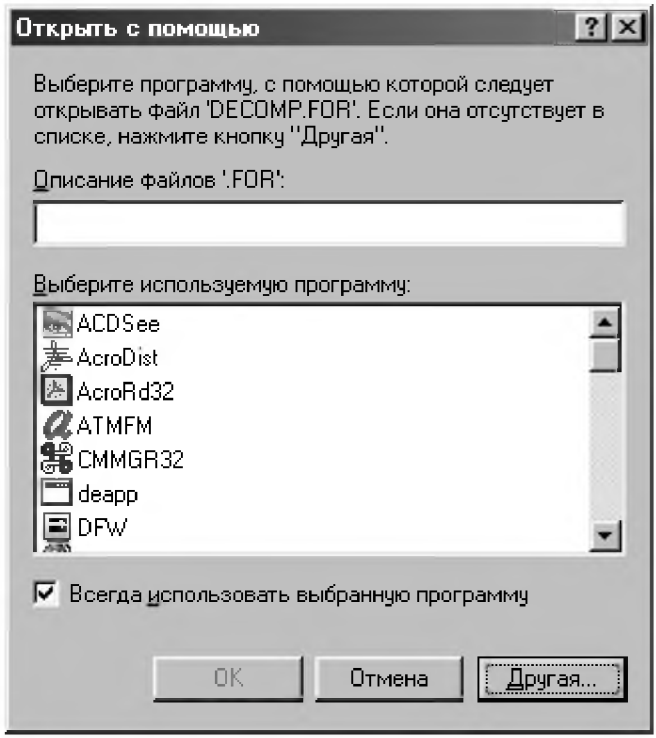

Рис. 9.13. Создание нового типа документов

1. Если на Панели задач (Taskbar) уже присутствует кнопка с названием программы, то для ее запуска достаточно щелкнуть один раз мышью на соответствующей кнопке.
2. Для запуска программ вы можете нажать кнопку Пуск (Start), выбрать пункт главного меню Программы (Programs), а затем щелкнуть по значку нужной вам программы, если он присутствует в появившемся меню. В противном случае откройте одну из папок, также включенных в это меню, найдите нужный значок и запустите программу.
3. Если значок нужной программы все-таки не найден, то используйте команду Пуск → Найти → Файлы и папки (Start → Find → Files or Folders). Затем после завершения поиска двойным щелчком мыши запустите программу прямо из окна поиска.
4. Если вы знаете имя и местоположение программы, но не можете найти ее значок, используйте команду Главного меню (Start Menu) Выполнить (Run), которое открывает пользователю доступ к окну Запуск программы (рис. 9.14). Это окно содержит раскрывающийся список Открыть: (Open:), который появляется после нажатия на кнопку . В данном списке вы найдете имена запускаемых ранее таким образом программ. Щелчок по любой строке из списка запускает соответствующую программу на выполнение. Если имя программы известно, можно просто ввести его в поле ввода Открыть: (Open:) и щелкнуть на кнопке OK.

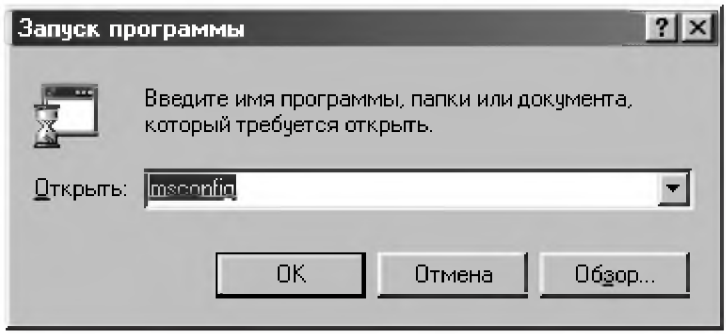

Рис. 9.14. Окно Запуск программы (Run)

Если имя программы в списке ***Открыть: (Open:)*** на рис. 9.14 отсутствует, то следует нажать на кнопку ***Обзор (Browse)***, окно которого представлено на рис. 9.15. Обнаружив в результате просмотра имя нужной программы, нажмите кнопку ***Открыть (Open)*** на рис. 9.15. После этого имя программы будет

помещено в окно ***Запуск программы (Run)*** и занесено в список. Программа будет запущена уже из этого окна, а ее имя останется в списке.

Не забывайте, что для запуска программы (приложения), которая работает с документами (создает или редактирует их), вовсе не обязательно вызывать саму программу, достаточно дважды щелкнуть на значке интересующего вас документа. При этом программа будет запущена, а документ окажется загружен в нее.

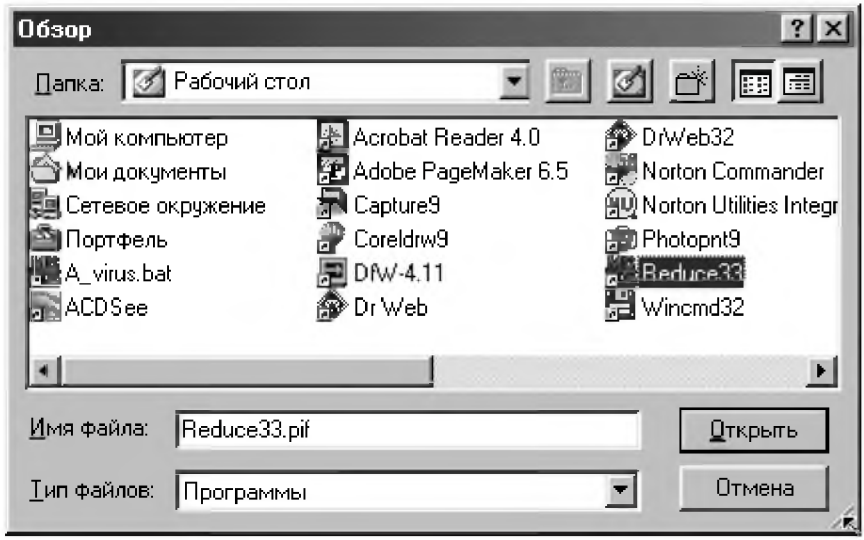

Рис. 9.15. Поиск программы в окне Обзор (Browse)

9.6.3. Копирование и перемещение объектов

Операции копирования и перемещения объектов относятся к числу наиболее распространенных, поэтому они связаны с максимальным количеством ошибок пользователей и могут быть выполнены большим количеством способов.

Копирование (Copying) — это операция создания копии объекта на новом месте. При этом объект как бы размножается и вместо одного объекта появляется два: один — на прежнем месте, а другой — на новом.

Перемещением (Moving) называется операция переноса объекта на новое место. При этом объект как был один, так и остался один, только местоположение его изменяется.

В п. 9.5.4 рассматривались операции копирования и перемещения объектов с использованием специального перетаскивания (правой кнопкой мыши) в структурах ***Мой компьютер (My Computer)*** и ***Проводник (Windows Explorer)***.

Отметим, что эти операции можно выполнять как с одним, так и с группой выделенных объектов (см. п. 9.6.1).

Если сразу после выполнения этих операций вы обнаружили результат, которого не ожидали, то нажмите сочетание клавиш <Ctrl>+<Z> (<Ctrl>+<Я>). Это вообще универсальная комбинация клавиш: почти во всех приложениях Windows 95/98/Me таким образом можно отменить результат выполнения нескольких последних операций.

Рассмотрим еще один универсальный способ для выполнения операций копирования и перемещения объектов с использованием буфера обмена.

Буфер обмена (Clipboard) — это место для временного хранения информации. Он расположен в памяти компьютера, поэтому его содержимое при отключении питания или при перезагрузке компьютера пропадает. Хранить в нем можно любые объекты: папки, документы, фрагменты текста, изображений.

Используется б*уфер обмена (Clipboard)* для того, чтобы временно сохраненную в нем информацию можно было вставить в другой объект. Он активно используется для переноса объектов из одного места документа в другое, а также для обмена информацией между различными приложениями Windows. При работе с файловой системой можно переносить с места на место файлы, группы файлов и папки

При работе с *буфером обмена (Clipboard)* операция производится в два приема с помощью специальных команд. Сначала объект копируется (или "вырезается") в *буфер обмена (Clipboard)*, а потом вставляется в другое место. Эти команды можно выполнять разными способами:

- с помощью клавиатурной комбинации клавиш <Ctrl>+<C>, <Ctrl>+<X>, <Ctrl>+<V>, что является наиболее удобным способом;
- из меню команды ***Правка*** (Edit), показанном на рис. 9.16;
- использовать для работы с *буфером обмена (Clipboard)* команды контекстного меню, вызываемого нажатием правой кнопки мыши;
- использовать соответствующие кнопки ***Копировать***  , ***Вырезать***  и ***Вставить***  панели инструментов (см., например, рис. 9.12) для работы с *буфером обмена (Clipboard)*.

Команда ***Копировать*** (Copy) — <Ctrl>+<C> предназначена для переноса копии выделенного объекта в буфер с сохранением оригинала на прежнем месте.

Команда ***Вырезать*** (Cut) — <Ctrl>+<X> предназначена для перемещения выделенного объекта в буфер (при этом объект на прежнем месте исчезает).

Команда ***Вставить*** (Paste) — <Ctrl>+<V> предназначена для копирования содержимого буфера обмена в позицию размещения указателя мыши или курсора.

Содержимое *буфера обмена (Clipboard)* при этом сохраняется и может быть использовано для вставки неограниченное количество раз.

Замена объекта в *буфере обмена* производится только при записи в него новой информации.

Вставить ярлык (Paste Shortcut) — это команда для размещения ярлыка со ссылкой на объект, помещенный1 в *буфер обмена*.

Размещение производится в той папке, из меню которой была выполнена эта команда.

Заметим, что команда ***Вставить ярлык*** (Paste Shortcut) появляется в меню ***Правка*** (Edit) только тогда, когда она имеет смысл.

Например, при работе в текстовом редакторе вставить в документ ярлык нельзя, поэтому в меню редактора эта команда отсутствует. Это замечание относится ко всем командам.

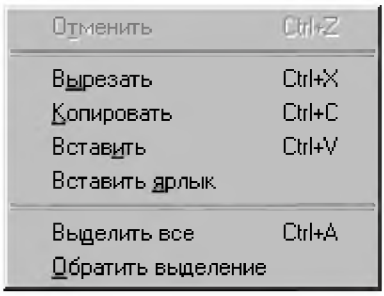

Рис. 9.16. Меню команды Правка (Edit) для работы с буфером обмена (Clipboard)

Для того, чтобы увидеть содержание *буфера обмена (Clipboard)*, нужно вызвать специальную программу для его просмотра.

Отметим, что *буфер обмена (Clipboard)* удивительно полезная вещь. Для многозадачной операционной системы он просто незаменим.

При работе с любыми приложениями Windows всегда следует придерживаться принципа: *как можно реже создавать что-то новое и как можно чаще копировать и размножать то, что было создано ранее, но может быть использовано в новом качестве после незначительных правок.*

При работе в Windows 95/98/Me часто бывает нужно перенести или скопировать какие-либо файлы (или папки с их содержимым) в другое место, что мы и рассмотрим в качестве примера конкретного использования *буфера обмена (Clipboard).*

Для *перемещения* файлов (папок) в другое место необходимо:

1. Открыть папку, содержащую перемещаемые объекты.
2. Выделить их (см. п. 9.6.1).
3. Переслать их в *буфер обмена (Clipboard)*, для чего следует из меню Правка (Edit) выдать команду Вырезать (Cut). Можно также для этого нажать одноименную кнопку на панели инструментов  или использовать клавиатурное сокращение <Ctrl>+<X>. Теперь папку с перемещаемыми объектами можно закрыть.
4. Перейти в папку-приемник (папку, в которую вы переносите объекты).
5. Поместите объекты в эту папку, для чего следует из меню Правка (Edit) выдать команду Вставить (Paste). Можно также для этого нажать одноименную кнопку на панели инструментов  или использовать клавиатурное сокращение <Ctrl>+<V>.
6. Дождитесь завершения операции перемещения.

Для *копирования* файлов (папок) необходимо:

1. Открыть папку, содержащую объекты-оригиналы.
2. Выделить эти объекты (см. п. 9.6.1).
3. Переслать их в *буфер обмена (Clipboard)*, для чего следует из меню Правка (Edit) выдать команду Копировать (Copy). Можно также для этого нажать одноименную кнопку на панели инструментов  или использовать клавиатурное сокращение <Ctrl>+<C>. Теперь папку с копируемыми объектами можно закрыть.
4. Перейти в папку-приемник (папку, в которую нужно поместить копии объектов).

5. Поместите объекты-дубликаты в эту папку, для чего следует из меню Правка (Edit) выдать команду Вставить (Paste). Можно также для этого нажать одноименную кнопку на панели инструментов  или использовать клавиатурное сокращение <Ctrl>+<V>.
6. Дождитесь завершения операции копирования.

Отметим, что можно осуществлять операции перемещения или копирования не только с одиночными файлами или папками, но и с группами, для чего их необходимо предварительно выделить (см. п. 9.6.1).

Существует другой способ перемещения (копирования) файлов и (или) папок — с помощью *левой* клавиши мыши. Вы должны ухватить интересующие вас файлы и перенести их в другое место. Если при перемещении файлов (папок) держать нажатой клавишу <Ctrl>, то вместо перемещения будет осуществлено копирование файлов и папок (об этом будет свидетельствовать плюс рядом с перемещаемым файлом).

9.6.4. Создание и переименование объектов

В процессе работы на компьютере вам обязательно придется создавать новые объекты: папки, документы, ярлыки. Windows 95/98/Me предлагает простые и единообразные способы для этого.

Порядок действий для создания нового объекта следующий:

1. Откройте окно диска или папки, в которой вы хотите создать новый объект (или установите указатель мыши на свободной части Рабочего стола (Desktop), если его нужно создать там).
2. Щелкните на пустом пространстве окна (или Рабочего стола) правой кнопкой мыши (или нажмите сочетание клавиш <Shift>+<F10>) для вызова Контекстного меню.
3. Раскройте подменю команды Создать (New).
4. Выберите в нем соответствующую команду в зависимости от вида создаваемого объекта (см. рис. 9.17) и выполните ее:
   - Папка (Folder) — после ее выполнения появится значок созданной папки со стандартным именем. В поле метки значка вписываем требуемое название и нажимаем клавишу <Enter>;
   - для создания документа выбираем тип документа и щелкаем мышкой (или нажимаем клавишу <Enter>), после чего в поле метки значка указываем нужное имя;

- Ярлык (Shortcut) — после ее выполнения появится диалоговое окно Создание ярлыка (Create Shortcut). В нем следует указать имя объекта, для которого следует создать ярлык, либо разыскать его самостоятельно с помощью кнопки Обзор (Browse). При выполнении этой процедуры Windows 95/98/Me последовательно выведет на экран несколько похожих окон с инструкциями, после выполнения которых в итоге получится нужный ярлык. Перетащите его в удобное для вас место:
  - на Рабочий стол (Desktop);
  - в папку;
  - на кнопку Пуск (Start), чтобы включить ярлык в Главное меню (Start Menu).

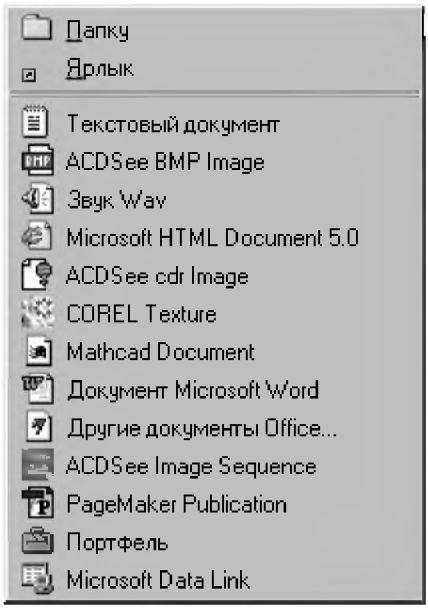

Рис. 9.17. Подменю команды Создать (New) Контекстного меню окна папки или Рабочего стола (Desktop)

Отметим, что п. 4 создания ярлыка для объекта можно существенно упростить, если в п. 2 *щелкнуть правой кнопкой мыши* не на пустом пространстве окна (или Рабочего стола), а *на самом объекте*. В этом случае вместо достаточно длительной процедуры указания объекта для ярлыка сразу появится сам ярлык.

Однако его все равно придется из папки с объектом-оригиналом перетаскивать в нужное место. Поэтому удобнее и проще всего совместить эти операции и для создания ярлыка *использовать специальное перетаскивание* с использованием инструментов Мой компьютер (My Computer) или Проводник (Windows Explorer) (см. п. 9.5.4).

В дополнение к описанным имеется по крайней мере еще пять способов создания ярлыков, ибо их использование существенно убыстряет работу. Если вы часто обращаетесь к определенному файлу или папке, создайте для них ярлыки и разместите их на Рабочем столе (Desktop). Так вы существенно ускорите доступ к этим элементам. Ярлык документа служит для его открытия, ярлык программы — для ее запуска. Щелкнув ярлык папки, вы откроете окно с его содержимым.

Для того, чтобы позиции значков после создания или перемещения их в новое положение автоматически выравнивались, следует использовать команду Упорядочить значки → Автоматически (Arrange Icons → Auto Arrange). Эта команда всегда помещена в *контекстное меню*.

Отметим также, что имена файлов и папок в Windows 95/98/Me могут содержать уже не 8, а до 255 символов всех установленных в системе алфавитов. При использовании длинных имен фактически создается еще одно имя (короткое) — для совместимости с MS-DOS. Оно формируется по старому правилу "8.3": 8 символов для имени файла и 3 символа для расширения.

Приложения, которые были созданы для работы в MS-DOS или для предыдущих версий Windows, используют короткие имена файлов по старому правилу "8.3".

Следует также предостеречь от неосторожного использования программ, не поддерживающих длинные имена, для операций с файловой системой. Например, при копировании файлов с помощью Norton Commander, их длинные имена наверняка будут потеряны.

Переименование (Renaming) объектов в Windows 95/98/Me можно производить практически в любой папке и в любом окне. Это одна из самых простых операций. Для этого нужно:

1. Открыть папку, содержащую переименовываемый объект.
2. Выделить объект, имя которого предполагается изменить, щелкнув по нему мышкой.
3. Щелкнуть после секундного интервала еще раз в поле метки его значка (на старом его названии). При этом не следует торопиться, чтобы это

не было воспринято системой как двойной щелчок, после которого объект просто запустится на выполнение.

4. Ввести новое имя объекта непосредственно в поле метки его значка (вместо старого названия).
5. Нажать клавишу <Enter>.

Для переименования объекта в п. 3 можно также использовать:

- команду Переименовать (Rename) *контекстного меню*, вызвав его правой кнопкой мыши;
- команду Файл → Переименовать (File → Rename);
- клавишу <F2>;

Отметим, что расширение имени файла менять не следует, если перед вами не стоит нестандартная задача.

9.6.5. Удаление и восстановление объектов

Удаление объекта также относится к числу самых простейших операций. Однако последствия от удаления нужных объектов столь значительны, что перед их удалением хорошо подумайте, действительно ли они вам больше не понадобятся. Для этой цели Windows каждый раз переспрашивает вас, действительно ли вы хотите от них избавиться. Даже после получения положительного ответа, удаляемый объект не уничтожается, а аккуратно складывается в Корзину (Recycle Bin), откуда при необходимости его можно достаточно просто восстановить.

Однако относитесь к предупреждениям Windows с максимальным вниманием: *восстановить файлы и папки, удаленные с дискеты или из сети, нельзя*.

Для удаления файлов, папок или ярлыков необходимо выделить ненужные объекты и нажать клавишу <Delete>. Другой вариант — щелкнуть по объекту правой кнопкой мыши и в появившемся контекстном меню выбрать Удалить (Delete). Можно также ухватить ненужный объект и перетащить его на значок Корзины (Recycle Bin) (значок при этом станет синим).

Затем нужно санкционировать удаление нажатием кнопки Да (Yes) в открывшемся диалоговом окне (в первом способе для этой цели удобнее нажать клавишу <Enter>, находящуюся рядом с клавишей <Delete>). Удаляя папку с файлами, щелкайте Да (Yes) в ответ на вопрос Windows, нужно ли удалять файлы некоторых типов.

При помещении файлов в Корзину (Recycle Bin) она изменяет свой вид — из пустой становится полной. При этом файлы не удаляются окончательно, а лишь переносятся в другое место на диске, и их легко можно восстановить.

Окончательное удаление файлов происходит через некоторое время, когда их вытесняют вновь помещаемые в Корзину (Recycle Bin) файлы (общий объем файлов в корзине не может превышать обычно 10% от объема всего диска).

Вы можете и принудительно очистить Корзину (Recycle Bin). Для этого необходимо:

- щелкнуть на значке Корзины (Recycle Bin) правой кнопкой мыши;
- в контекстном меню выбрать Очистить корзину (Empty Recycle Bin);
- подтвердить очистку Корзины (Recycle Bin).

После очистки Корзины (Recycle Bin) *помещенные в нее файлы окончательно удаляются с диска* (и восстановить их можно только теоретически с помощью специальных программ).

Вы можете также удалять файлы с диска сразу в обход Корзины (Recycle Bin), *если точно знаете, что они вам не нужны*.

Для этого просто удерживайте <Shift> при выполнении команды Удалить (Delete) в любой ее форме. Отпустите ее только после открытия диалогового окна, в котором предлагается подтвердить ваши намерения.

Напомним, что при удалении *ярлыков (Shortcuts)* сами объекты, на которые ссылаются эти ярлыки, не удаляются.

Для ***восстановления*** ошибочно удаленных файлов из Корзины (Recycle Bin) нужно:

- открыть Корзину (Recycle Bin), дважды щелкнув по соответствующему значку;
- в открывшейся папке Корзины (Recycle Bin) выбрать файлы, подлежащие восстановлению;
- воспользоваться командой Восстановить (Restore) пункта меню ***Файл (File)*** или соответствующей командой ***контекстного меню***, вызвав его правой кнопкой мыши;
- щелкнуть кнопку  , чтобы закрыть Корзину (Recycle Bin).

После этого выбранные для восстановления файлы будут помещены на диск на свое прежнее место.

Кроме того, Windows 95/98/Me позволяет *отменить последнюю из выполненных операций* с объектами (их удаление, копирование).

Для этой цели удобно использовать универсальную комбинацию клавиш <Ctrl>+<Z> или кнопки на панели инструментов (Toolbar).

9.6.6. Поиск папок, файлов или ярлыков

В Windows 95/98/Me имеется программа поиска папок и файлов по составному критерию, в котором может фигурировать имя, тип, свойства и даже содержимое объекта поиска.

Для того чтобы найти нужную вам папку или файл необходимо:

- в Главном меню (Start Menu) открыть команду Поиск (Find), затем пункт Файлы и папки (Files and Folders), после чего откроется окно Найти: Все файлы (Find: All Files);
- задайте в диалоговом окне Имя (Name) условия (имя, часть имени, шаблон) и пространство поиска (Папка (Folder)) данного объекта;
- инициируйте поиск нажатием кнопки Найти (Find Now);
- дождитесь окончания поиска или, если вы увидели нужный вам файл, прекратите поиск нажатием кнопки Остановить (Stop).

Теперь можно просмотреть список найденных объектов и выполнить с ними требуемые действия (например, открыть, скопировать, переместить или удалить).

Если вы нажмете кнопку Новый поиск (New Search), то программа поиска переводится в исходное состояние.

9.7. Dos-режим Windows 95/98/Me

Иногда в среде Windows необходимо использовать программы, написанные для ОС MS-DOS. Для этого можно применять разные способы:

- двойной щелчок либо выделение его с последующим нажатием клавиши <Enter>:
  - на иконке исполняемого файла;
  - на заранее созданном ярлыке;
- специальный режим работы: Пуск → Программы → Сеанс MS-DOS (Start → Programs → MS-DOS Prompt). Он позволяет получить доступ к интерфейсу и командам MS-DOS;

- режим эмуляции MS-DOS (если два первых способа вызывают ошибки): Пуск → Завершение работы (Start Shut down) по команде Перезагрузить компьютер в режиме MS-DOS (Restart in MS-DOS mode).

Следует отметить, что единственно правильный выход из DOS-режимов осуществляется по команде *EXIT*.

В Windows Millenium перезагрузка в режиме MS-DOS недоступна.

Неудобством использования DOS-режимов является то, что вам придется набирать все команды вручную (без помощи мыши).

Рассмотрим работу CAB REDUCE при использовании Сеанса MS-DOS (MS-DOS Prompt).

Чтобы загрузить REDUCE из Windows необходимо щелкнуть левой кнопкой мыши по ярлыку REDUCE (если он есть), иначе вы можете это сделать при помощи Проводника (Explorer).

Перед вами появится обычное окно Windows, в котором вы можете выполнять все возможные операции REDUCE.

Для того чтобы вернуться в Windows, не выгружая REDUCE, необходимо нажать клавиши <Alt>+<Tab> или щелкнуть по кнопке окна Свернуть.

Нажатие клавиш <Alt>+<Enter> позволяет включить полноэкранный режим (без органов управления окном) или вернуться к оконному режиму.

Если же вы загружали REDUCE через Главное меню (Main Menu), находясь, допустим, в Word, то достаточно только щелкнуть по кнопке открытого окна Microsoft Word на Панели задач (Taskbars).

ЧАСТЬ 4. МАТЕМАТИЧЕСКАЯ СИСТЕМА DERIVE 4.01–4.11 ПОД WINDOWS 95/98/ME

ГЛАВА 10. БЫСТРЫЙ СТАРТ

DERIVE является универсальной математической системой, ориентированной на решение весьма широкого круга математических и научно-технических задач.

Она предназначена для проведения арифметических, логических и символьных операций с различными числами, переменными, функциями, полиномами и матрицами практически любой степени сложности.

DERIVE может выполнять символьное и численное дифференцирование и интегрирование, разложение и возведение в степень полиномов и рациональных функций в различных формах, вычисление пределов функций, нахождение сумм и произведений элементов рядов, выполнять преобразования формул и просто работать с различными матричными выражениями. Она может строить двумерные и трехмерные графики функций в декартовой и полярной системах координат, а также заданных в параметрической форме.

Буквы при вводе в DERIVE можно писать строчные и прописные, так что sin(x) и SIN(x) означает одно и то же. Однако система всегда записывает имя функции, например SIN, прописными буквами. Операции с целыми числами осуществляются с произвольной точностью.

Предваряя более полное описание и изложение возможностей математической системы DERIVE, покажем на примере нескольких операторов и команд, что это простой и удобный математический помощник пользователя ПК, что дословно и означает перевод полного названия этой системы.

Дальнейшее описание относится к работе в среде Windows 95/98/Me. Если вы с ней недостаточно хорошо знакомы, то следует обратиться к соответствующим разделам п. 9.

10.1. Первый сеанс работы

После запуска DERIVE, осуществляемого обычно двойным щелчком на соответствующей иконке на рабочем столе, появится основное окно, на фоне которого непродолжительное время видна красочная заставка системы (см. рис. 10.1).

Рис. 10.1. Основное окно математической системы DERIVE 4.01–4.11 под Windows

Сразу после исчезновения красочной заставки в основном окне останутся видными следующие строки:

- вверху:
  - заголовка:

  - главного меню:

- панели инструментов:

- внизу:
  - состояния программы:

  - панели задач

Для работы с системой DERIVE сначала необходимо задать математическое выражение, называемое авторским, которое вводится после активизации позиции главного меню Author. Для этого нужно подвести к нему курсор и щелкнуть по нему мышкой, после чего появится подменю позиции Author:

Рис. 10.2. Подменю позиции Author

Команды подменю позиции Author служат для вызова после щелчка на них мышью соответствующих диалоговых окон, на что указывают три точки после их имен:

- Expression... <Ctrl>+<A> — вызывает появление окна ввода (рис. 10.3), с помощью которого в режиме строчного редактирования можно вводить различные математические выражения;
- Vector... и Matrix... — вызывает появление окон для ввода размеров вектора или матрицы, после чего появляются окна для задания их элементов.

Указанные на рис. 10.2 клавиатурные сокращения <Ctrl>+<A> служат для непосредственного вызова окна ввода математических выражений (рис. 10.3) сразу из основного окна DERIVE (рис. 10.1).

Теперь в диалоговом окне в режиме строчного редактирования с клавиатуры можно вводить различные математические выражения — как численные, так и в виде математических формул. Греческие буквы и специальные математические символы, приведенные в специальной панели,

можно вставить в выражение, просто установив на соответствующей позиции курсор и щелкнув левой клавишей мыши.

Повторим в системе DERIVE основные этапы первого сеанса работы на REDUCE, приведенные в п. 1.1. Сначала возведем в 2000-ю степень число 2, набрав соответствующее выражение с клавиатуры, что и показано на рис. 10.3. Отметим, что в DERIVE для указания операции возведения в степень используется символ ^, в отличие от двух звездочек ** в REDUCE.

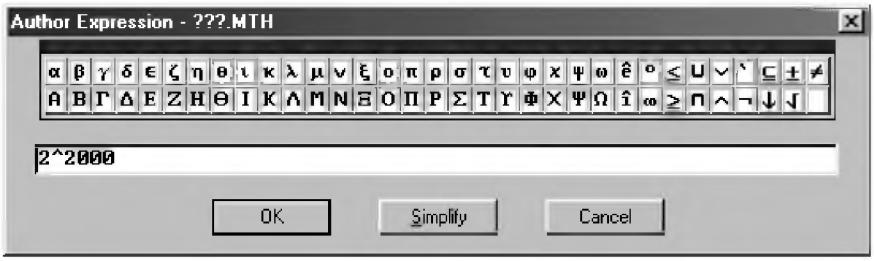

Рис. 10.3. Диалоговое окно Author Expression (<Ctrl>+<A>) для ввода математических выражений с примером возведения числа 2 в 2000-ю степень

Внизу диалогового окна ввода расположены три кнопки (рис. 10.3). После щелчка на них левой клавишей мыши будут соответственно выполнены следующие действия:

- OK — введенное в диалоговом окне ввода математическое выражение появится под номером #1 в отдельной строке в окне выражений (см рис. 10.4). Оно будет выделено цветом: синим для фона и ярко-белым для символов самого выражения. Такое же действие ввода выражения будет получено после нажатия клавиши <Enter>:

Рис. 10.4. Окно выражений с введенным примером возведения числа 2 в 2000-ю степень

- Simplify — введенное в диалоговом окне ввода математическое выражение будет вычислено и появится в отдельной строке в окне выражений в вычисленном виде и будет также выделено синим цветом фона и ярко-белым символов самого выражения. Строка в окне выражений не имеет

границ и все 603 цифры значения (10.1) числа 2^{2000} будут напечатаны после указания номера в одной строке (#2 рис. 10.9). Для ее просмотра используются клавиши горизонтального перемещения курсора на границах видимой части выражения, а также <Home> и <End>. Для экономии места покажем только содержательную часть окна выражений (после указания номера строки) в удобной для ее обзора форме (типа (10.1)):

#2:11481306952742545242328332011776819840223177020886952004776
42736825766261392370313856659486316506269918445964638987462773
44711896086305533142593135616665318539129989145312280000688779
14824004487142892699006348624478161546364638836394731702604046
63539709049965581623988089446296056233116495361642219703326813
44168908984458505602379484807914058900934776500429002716706625
83052200813223628129176126788331720659899539641812702177985840
40421598531832515408894339020919205549577835896720391600819572
16630582755380425583726015528348786419432054508915275783882625
17543552880082284277081796545376218485114902937 6 (10.1)

- Cancel — отказ от ввода расположенного в диалоговом окне (рис. 10.3) математического выражения и переход в исходное состояние окна выражений.

Отметим, что удобнее вводить выражения нажатием кнопки OK. Тогда его всегда не только можно будет вычислить, но и произвести над ним весь имеющийся у DERIVE комплекс действий простым нажатием кнопок быстрого управления системой, расположенных на панели инструментов, или позиций главного меню.

Поэтому одновременно нажмем клавиши <Ctrl>+<A>. В появившемся диалоговом окне ввода (рис.10.3) наберем новое выражение, например:

(x+y*x/2+x*z/3)^3 (10.2)

и нажмем кнопку OK, после чего оно запишется под следующим номером #3 в отдельной строке окна выражений и также будет выделено цветом.

#3: $\left(x+\frac{yx}{2}+\frac{xz}{3}\right)^3$ (10.3)

Обратите внимание, что в DERIVE знак умножения "*" заменяется пробелом, который на экране представляется точкой (см., например, строки #3-#8 на рис. 10.9), причем по умолчанию используются однобуквенные переменные. Поэтому выражение (10.2) в окне ввода (рис.10.3) может быть набрано и в виде (10.4):

(x+yx/2+xz/3)^3 (10.4)

После нажатия кнопки OK DERIVE преобразует **yx** и **xz** соответственно в **y x** и **x z** и будет готов вычислять их произведения. Поэтому внешний вид введенного выражения (10.3) при этом не изменится. Оно только записалось под номером #4:

$$\#4: \left(x+\frac{yx}{2}+\frac{xz}{3}\right)^3 \qquad (10.5)$$

Однако мы будем представлять вводимые выражения в привычной для нас форме (10.2) со знаками умножения, выделяя их также *в пособии* по аналогии с REDUCE для отличия жирным шрифтом, как и все, что вводится нами с клавиатуры. В окне же выражений DERIVE всегда при вводе автоматически выделяется цветом последнее введенное выражение (сейчас это выражение под номером #4).

Над выделенным выражением выполняются все действия, имеющиеся в распоряжении у DERIVE. Чтобы выделить другое нужное выражение, следует просто подвести курсор к любому его месту или соответствующему ему номеру и нажать левую клавишу мыши.

Для дальнейшей работы нам не нужно иметь в окне выражений две одинаковых записи (#3 и #4). Поэтому удалим последнюю из них (которая в настоящий момент является выделенной), нажав на клавишу <Delete> и после появления окна (рис. 10.5) — на клавишу <Enter> (или на кнопку OK).

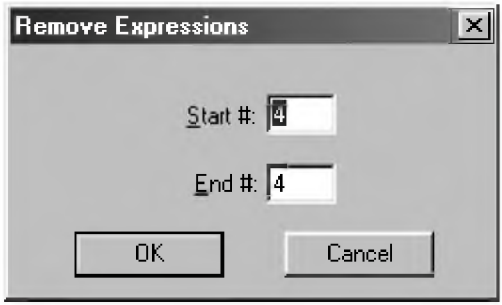

Рис. 10.5. Диалоговое окно Remove Expressions *для задания номеров стираемых строк с примером удаления 4-й строки*

Отметим, что в окне (рис. 10.5) можно указать любой нужный диапазон стираемых строк. Он там будет автоматически указан, если нужные строки любым обычным образом предварительно выделить в алгебраическом окне выражений.

Само диалоговое окно Remove Expressions также может быть вызвано:

- одновременным нажатием клавиш <Ctrl>+<R>;
- активизацией позиции главного меню Edit (Правка) с последующим нажатием команды Remove в появившемся подменю (рис. 10.6). Такой порядок действий, как и ранее, для краткости будем обозначать Edit → Remove:

Рис. 10.6. Подменю позиции Edit (Правка) *главного меню*

Здесь на примере удаления строк записи показана возможность многовариантного выполнения любых действий.

Любую операцию в DERIVE можно выполнить несколькими способами, как и в любом из приложений Windows.

Каждый раз будут описываться те из них, которые с точки зрения автора обеспечивают максимальную простоту и эффективность выполнения данной операции, чтобы нас при описании DERIVE не погубили его слишком широкие возможности.

Эти способы условно можно разбить следующим образом:

- *"клавиатурные" способы:*
  - *ассоциативные* — работают в DOS и в Windows, однако существуют не для всех операций (использование клавиши <Delete> для удаления в нашем случае);
  - *клавиатурные сокращения* (называемые также быстрыми или горячими клавишами) — очень быстро выполняют данную операцию (клавиши <Ctrl>+<R>, сразу вызывающие окно задания номеров стираемых строк на рис. 10.5). Однако они также

существуют не для всех операций и имеется проблема с их запоминанием; поэтому удобно пользоваться теми из них, которые очень часто вами используются или одинаковы в различных приложениях Windows (например, клавиши <Ctrl>+<C> и <Ctrl>+<V> соответственно для копирования в буфер и вставки из него);

- *"мышиные" способы:*
  - использование мыши *для активизации позиций главного меню*, затем подменю и выдачей соответствующих команд — может применяться для любых операций. Однако этот способ состоит из нескольких последовательных действий, поэтому более медленен (командой Edit → Remove из главного меню в нашем случае). Им особенно удобно пользоваться при начале обучения, когда вопросы быстроты несущественны, а правильности и надежности — весьма актуальны;
  - использование мыши *для активизации кнопок на панели инструментов* — достоинства быстроты выполнения данной операции соединяются с наглядностью их расположения на панели инструментов. Однако кнопки также существуют не для всех операций (пример их использования рассмотрен в следующем абзаце).

Чтобы вычислить выделенное в настоящее время выражение под номером #3 (10.3), подведем курсор к кнопке = на панели инструментов (при секундной задержке на ней курсора внизу всплывет знакомое нам уже ее название Simplify) и нажмем левую клавишу мыши. В окне выражений под следующим номером #4 появится его вычисленное и упрощенное выражение:

$$\#4:\ \frac{x^3(3y+2z+6)^3}{216} \qquad (10.6)$$

Сравнив выражение (10.6) с различными формами его представления (1.4) и (1.6), полученными REDUCE, убеждаемся:

- возможности REDUCE по представлению выражений в различных формах выше, чем у DERIVE;
- однако DERIVE сразу нашел максимально простую и удобную для дальнейших аналитических преобразований форму данного выражения, что является очень важным для пользователя.

Чтобы лучше понять те принципы, в соответствии с которыми работает система DERIVE, давайте рассмотрим эти простые примеры. Поскольку в алгебраических вычислениях нужны точные результаты, то арифметические операции с целыми числами выполняются практически с безграничной точностью (зависящей только от доступной памяти вашего ПК): значение (10.1) числа 2**2000 состоит из 603 цифр.

Теперь выделим выражение под номером #1 (рис. 10.4) и нажмем кнопку ≈ на панели инструментов (при секундной задержке курсора всплывет ее название Approximate) и нажмем левую клавишу мыши. В окне выражений под следующим номером #5 появится его вычисленное значение в виде действительного числа:

$$\#5:\ 1.14813\cdot 10^{602} \qquad (10.7)$$

На секунду поразимся возможностям DERIVE, легко оперирующим с числами порядка 10^{602} в целой или вещественной формах. Невозможно даже представить себе какую-нибудь аналогию для этого числа: ведь даже возраст Вселенной оценивается примерно в 10^{18} секунд. А тут 10^{602}...

Представим вычисленное выражение под номером #3 (10.3) в другом виде, для чего выделим его и нажмем кнопку ≈ (Approximate):

$$\#6:\ 0.00462962\cdot x^3\cdot(3\cdot y + 2\cdot z + 6)^3 \qquad (10.8)$$

Отметим. что найденная DERIVE максимально простая и удобная форма для дальнейших аналитических преобразований данного выражения сохраняется и при представлении его с использованием десятичных чисел. В нем также переменные упорядочены специальным образом. Однако и упорядочивание переменных, и формат вывода, и тому подобное, все это контролируется пользователем.

Возьмем теперь производную по x от выражения под номером (10.3), для чего выделим позицию под номером #3 в окне выражений и нажмем кнопку ∂ на панели инструментов (при секундной задержке курсора всплывет ее название Calculate derivative) и нажмем левую клавишу мыши. На экране появится окно команды дифференцирования (рис. 10.7), в котором будет задана:

- *дифференцируемая функция* в виде однострочного выражения с применением использующихся в DERIVE обозначений (например, символ ^ для указания операции возведения в степень);
- *переменная*, по которой осуществляется дифференцирование. Для активизации возможности ее выбора нужно:

- щелкнуть мышкой по кнопке ▼;
- в появившемся списке (если переменных в выражении несколько) подвести курсор к нужной из них, в результате чего она окажется выделенной;
- щелкнуть мышкой по выделенной переменной;

- *порядок* искомой производной (для его увеличения или уменьшения имеются соответствующие кнопки со стрелками вверх или вниз).

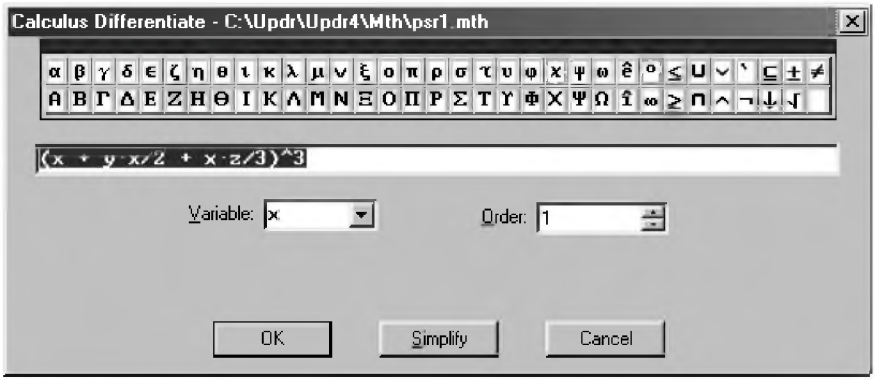

Рис. 10.7. Окно команды дифференцирования

Подведем курсор к кнопке Simplify (рис. 10.7) и нажмем левую клавишу мыши. Окно команды дифференцирования исчезнет и в окне выражений под следующим номером #7 появится производная от исходного выражения (10.3):

$$\#7:\ \frac{x^2(3y+2z+6)^3}{72} \qquad (10.9)$$

Обратим ваше внимание, что в (10.9) наряду с определением производной произошло также вычисление и упрощение исходного выражения. Поэтому результат будет точно таким же, если мы возьмем производную от предварительно вычисленного и упрощенного выражения (10.6), выделив его под номером #4 в окне выражений и нажав кнопку ∂ на панели инструментов, а затем кнопку Simplify в окне команды дифференцирования (рис. 10.7):

$$\#8:\ \frac{x^2(3y+2z+6)^3}{72} \qquad (10.10)$$

Для дальнейшей работы нам также не нужно иметь в окне выражений две одинаковых записи (#7 и #8). Поэтому удалим последнюю из них (которая в настоящий момент является выделенной), нажав последовательно на клавиши <Delete> и после появления окна (рис. 10.5) — на <Enter>.

Возьмем теперь неопределенный интеграл по x от выражения под номером (10.9), которое является в настоящее время выделенным, для чего нажмем кнопку ∫ на панели инструментов (при секундной задержке курсора всплывет ее название Calculate integral) и нажмем левую клавишу мыши. На экране появится окно команды интегрирования (рис. 10.8), в котором:

- будет задана интегрируемая функция в виде однострочного выражения с применением использующихся в DERIVE обозначений;
- переменная, по которой осуществляется интегрирование. Для активизации возможности ее выбора нужно:
  - щелкнуть мышкой по кнопке ▼;
  - в появившемся списке (если переменных в выражении несколько) подвести курсор к нужной из них, в результате чего она окажется выделенной;
  - щелкнуть мышкой по выделенной переменной.

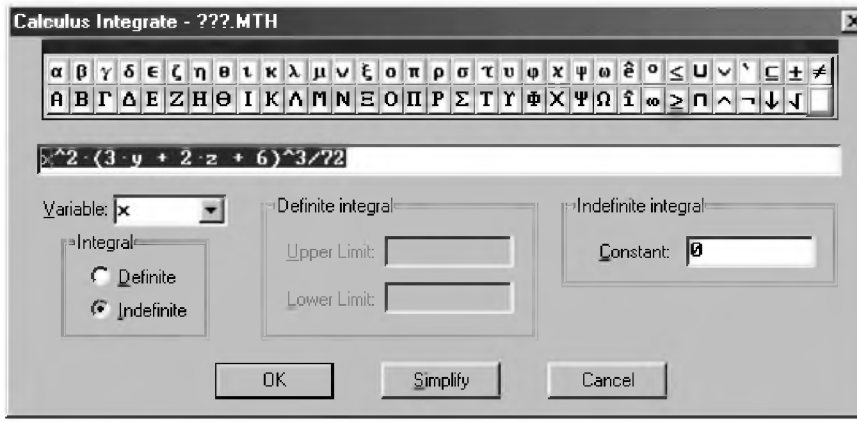

Рис. 10.8. Окно команды интегрирования

Подведем курсор к кнопке Simplify (рис. 10.8) и нажмем левую клавишу мыши. Окно команды интегрирования исчезнет и в окне выражений под следующим номером #8 появится первообразная от исходного выражения (10.9):

$$\#8:\ \frac{x^3(3y+2z+6)^3}{216} \qquad (10.11)$$

Сравнивая полученное выражение (10.11) с (1.8), убеждаемся, что они представляют собой одно и тоже выражение. Оно, конечно, также совпадает с (10.6), так как после взятия от него производной по x и вычисления интеграла по x мы должны опять получить исходное выражение. Однако, найденная DERIVE максимально простая и удобная форма для дальнейших аналитических преобразований данного выражения позволяет, по сравнению с соответствующими выкладками REDUCE (1.6) – (1.8), легко их проследить и даже выполнить устным образом.

Если вы последовательно выполняли все действия вышеприведенного сеанса работы и удаления лишних записей, то в итоге экран вашего монитора будет иметь примерно следующий вид:

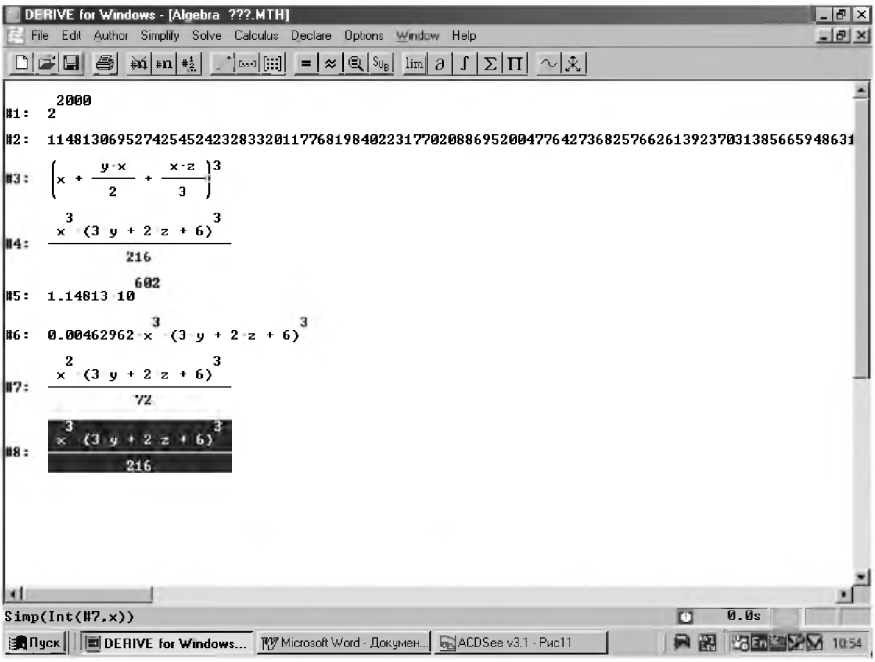

Рис. 10.9. Пример выполнения первого сеанса работы на DERIVE

DERIVE, в отличие от рассмотренных версий REDUCE, позволяет записать все результаты *диалогового* сеанса работы (все введенные и

вычисленные выражения) на магнитный диск в виде файла с любым задаваемым вами именем. Иначе это называют сохранение сессии на диске, что можно сделать командой File → Save As... из главного меню. Напомним, что эта краткая форма записи означает, что:

- надо активизировать позицию File из главного меню, щелкнув по ней мышкой;
- в возникающем подменю подвести курсор к команде Save As... (Сохранить как...), в результате чего она окажется выделенной, и нажать левую клавишу мыши.

После ее исполнения появится нижеследующее окно записи (рис. 10.10).

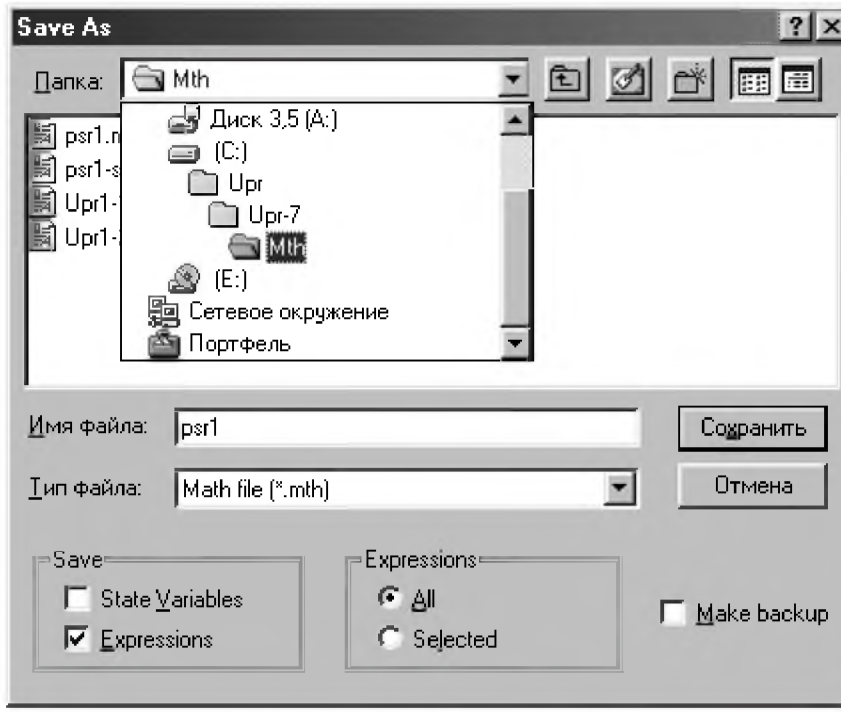

Рис. 10.10. Диалоговое окно записи файла с заданным именем с открытым подменю для выбора папки (подкаталога), в которую будет делаться запись

Отметим, что окно с выражениями вначале всегда имеет имя ???.mth. Команда Save As... (Сохранить как...) позволяет записать файл под любым

именем и в любом нужном месте. Для этого следует щелкнуть мышью в начале окна против надписи Имя файла, после чего следует набрать нужное имя (на рис. 10.10 набрано psr1). Тип файла mth изменять и задавать не следует: он будет добавлен автоматически в соответствие с форматом сохранения, с которым работает DERIVE.

Папка (подкаталог), в которой будет производиться запись, указывается в окне с надписью Папка: (на рис. 10.10 ее имя Mth при создании папки выбрано совпадающим с типом хранящихся в ней файлов), а ее содержимое показывается в расположенном ниже безымянном списке. Оба эти условия определяют, что папка находится в раскрытом состоянии. Нажав на кнопку Сохранить или клавишу <Enter> (<Ввод>), вы запишете представленный на рис. 10.9 ваш первый сеанс работы в раскрытую папку Mth под именем psr1.mth (при этом полный идентификатор файла будет иметь вид: C:\Upr\Upr-7\Mth\psr1.mth). Если файл с этим именем уже существует в этой папке, то в целях безопасности система предложит санкционировать замещение существующего файла.

Если вы хотите сохранить ваш документ в другой папке, то ее также предварительно нужно раскрыть. Для этого используются следующие приемы навигации:

- нажатие кнопки На один уровень вверх (Up One Level)  , чтобы открыть родительскую папку. В результате неоднократного применения этого приема нужная папка:
  - будет раскрыта;
  - окажется в безымянном списке; представляющем собой содержание папки, указанной в окне с надписью Папка, после чего она может быть раскрыта двойным щелчком мыши;
- открытие подменю для поиска нужной папки после нажатия кнопки  (рис. 10.10), после чего следует подвести к ней курсор (она окажется выделенной) и щелкнуть по ней мышкой. В подменю показаны:
  - корневые папки всех существующих на вашем ПК дисков;
  - все родительские папки от открытой в настоящий момент до корневой данного диска.

Поэтому для сохранения документа с тем же именем в другой папке, например, с идентификатором c:\Upr\Image\Image\psr1.mth:

- откроем подменю для поиска нужной папки нажатием кнопки  ;
- подведем курсор к папке Upr (она окажется выделенной) и щелкнем по ней мышкой;

- в появившемся списке содержания папки Upr раскроем дочернюю папку Image, дважды щелкнув по ней мышкой;
- в появившемся списке содержания папки Image раскроем ее дочернюю папку также с именем Image (одинаковые имена родительской и дочерней папок выбраны для иллюстрации такой возможности);
- нажав на кнопку Сохранить или клавишу <Enter> (<Ввод>), запишем документ на диск с идентификатором c:\Upr\Image\Image\psr1.mth, что и представлено на рис. 10.10.

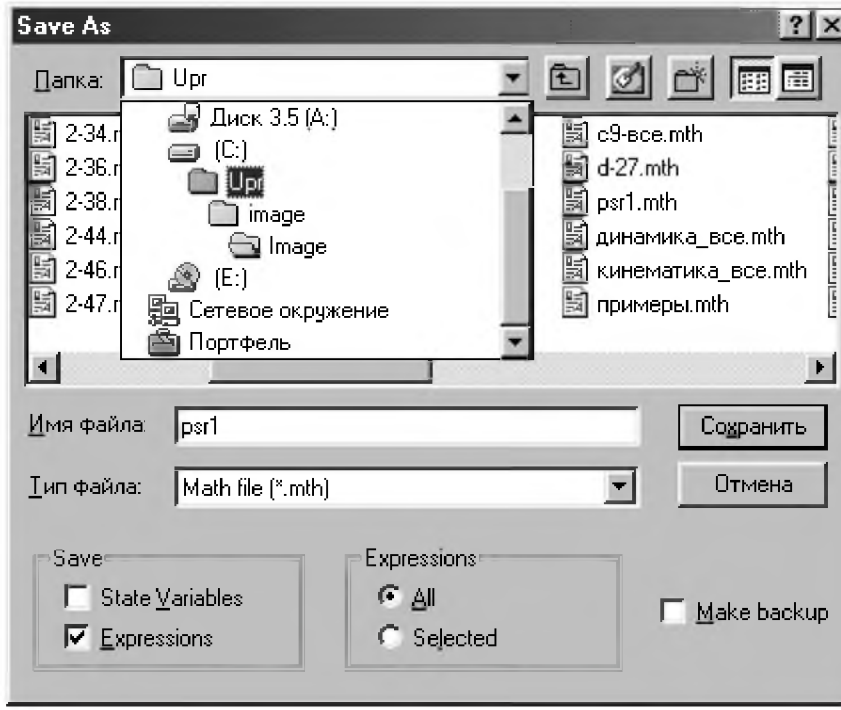

Рис. 10.11. Диалоговое окно записи файла с заданным именем с открытым подменю для выбора другой папки (подкаталога), в которую будет делаться запись

Отметим, что на рис. 10.10 и 10.11 впервые появилось много имен в окнах и списках, не совпадающих с вашими, после открытия соответствующего диалогового окна на вашем ПК. Это имена конкретных папок и файлов. Они могут быть любыми и зависят от конкретной файловой структуры

на вашем ПК. На эти отличия обращать внимание не следует. Важно овладеть вышеописанными приемами навигации, которые практически одинаковы при записи любых документов в любом из приложений Windows (см.. например, п. 10.3.3 и рис. 10.4).

Второй смущающей особенностью рис. 10.10 и 10.11 может явиться то, что в этом окне часть надписей дана на русском языке.

Здесь дело не в DERIVE, а в русифицированной системе Windows 98, стоящей на компьютере автора. Ее окна используются DERIVE, как и другими приложениями, поэтому изредка в них возникают надписи на русском языке.

Сама система DERIVE под Windows работает только с одним специальным набором символов, который вводится при установке системы и содержит также буквы латинского алфавита.

Русских букв в нем нет и при попытке их использования (даже в комментариях) начинают искажаться математические символы. Поэтому при работе с DERIVE мы будем пользоваться только латинским алфавитом.

Однако комплексное использование DERIVE и текстового редактора Word посредством буфера обмена (Clipboard) позволяет обойти это ограничение.

Часто используемым свойством системы является также возможность выдачи результатов в форматах файлов, совместимых с Фортраном и другими языками программирования (Бейсиком, СИ и Паскалем).

Это особенно полезно, если рассматривать работу системы, как некоторый способ генерации алгебраических формул, которые должны быть использованы в качестве основы для обширных численных расчетов.

Для записи выражений в форматах различных языков программирования командой File → Write To из главного меню вызывается соответствующее подменю с командами:

- Basic file...
- C file...
- Fortran file...
- Pascal file...

Выбор любой из них вызывает появление диалогового окна, практически аналогичного окну Save As..., позволяющему записывать выражения в соответствующем формате выбранного языка программирования с изменением имени или с изменением места расположения файла.

Если вы хотите записать в выбранном формате только выделенное выражение, а не всю сессию, то не забудьте перед нажатием кнопки OK поднять в диалоговом окне флаг **Selected**.

Итак, мы повторили в системе DERIVE основные этапы первого сеанса работы на REDUCE, приведенные в п. 1.1.

Рекомендуем также обратить внимание на **замечание** в конце п. 1.1. Надеемся, вы сами проверите более сложные примеры, передающие практически безграничные возможности DERIVE. И если при этой проверке благотворное чувство удивления посетит вас и приведет к изучению практического использования системы DERIVE на ПК, то ваши возможности также возрастут удивительным образом.

И в надежде на это мы пока завершим наш первый сеанс работы в системе DERIVE, нажав правую верхнюю кнопку ✕ в правом верхнем углу экрана, и приступим к более систематическому изучению.

10.2. Описание элементов системы DERIVE

10.2.1. Символы и выражения

В качестве основных символов системы символьной математики DERIVE используются:

- прописные и строчные буквы латинского алфавита;
- десятичные арабские цифры от 0 до 9;
- все доступные для ввода символы клавиш клавиатуры (знаки математических операций и спецзнаки).

Система допускает при вводе использование в произвольном порядке строчных и прописных букв. Для обычного режима работы DERIVE, который устанавливается пользователем, они внутренне неразличимы. Однако DERIVE всегда записывает имя функции (например, SIN, COS) прописными буквами. Этот же принцип будет применяться нами и при выводе результатов.

Обращаем ваше внимание на частое отличие вводимых данных от их представления на экране. Поэтому наряду с копиями экрана часто будут представляться вводимые выражения.

Также для лучшего отличия записей символов всегда будут использоваться:

- в тексте прописные буквы;
- строчными буквами жирным шрифтом с использованием абзацного отступа выделяется все, что должно находиться в диалоговом окне ввода и после чего должна нажиматься:
- клавиша <Enter> (<Ввод>) или кнопка OK;
- кнопка Simplify;
- без выделения обычными буквами и без использования абзацного отступа печатаются результаты преобразований и вычислений DERIVE.

Математические выражения составляются из *чисел* и *имен переменных*, называемых *операндами*, а также *операторов* и *функций*.

Числа в системе DERIVE могут быть *целые, рациональные* или *вещественные,* а также *комплексные*.

Операторами являются знаки математических операций, например +, –, * и /. Это арифметические операторы сложения, вычитания, умножения и деления. Они применяются совместно с операндами: числами или переменными. Например, в выражении 5+X число 5 и переменная X — это операнды, а знак + — оператор.

Особую роль играет арифметический оператор присваивания :=. Он используется для вычисления арифметических выражений и присваивания их значений переменным. Этот оператор заменяет значение переменной в левой части оператора на значение выражения, стоящего в его правой части, и имеет вид:

```
переменная := выражение                    (10.12)
```

Арифметический оператор присваивания означает для компьютера: взять содержимое ячейки памяти, соответствующее значению выражения справа и передать результат в ячейку памяти, соответствующую переменной, стоящей слева.

Обратим ваше внимание на универсальность арифметического оператора присваивания. Переменной слева в (10.12) присваивается значение, равное значению числа или арифметического выражения справа от оператора присваивания.

Отметим, что выражение в (10.12) может иметь не только *численное*, но и *символьное* значение, а также быть *функцией, вектором* или *матрицей*. Последнее имеет не только важное значение при решении систем линейных алгебраических уравнений, но и резко облегчает ввод в диалоговом окне Author Expression (<Ctrl>+<A>) больших матриц, которые могут быть

предварительно подготовлены с использованием, например, текстового редактора Word.

В DERIVE, в отличие от REDUCE, не все знаки арифметических действий могут присутствовать в выражении явно, ибо знак умножения заменяется пробелом. Также по умолчанию используются однобуквенные переменные.

Поэтому математическое выражение для умножения, например, значения I на значение J в DERIVE может быть написано в любой из форм: IJ или I*J. В обоих случаях оно после нажатия кнопки OK будет преобразовано к виду i j для вычисления произведения двух переменных (при установке режимов ввода, принятых по умолчанию — см. п. 10.2.3). Однако для удобства ориентировки мы будем сохранять при наборе привычную форму записи выражений.

Последовательность выполнения арифметических операций в выражении такая же, как и при обычной записи алгебраических формул. Сначала выполняется возведение в степень, затем умножение и деление, а в конце — сложение и вычитание.

Если в выражении записано несколько одинаковых операций подряд, то они выполняются слева направо. В случае необходимости иного порядка выполнения арифметических операций, как и при написании обычных формул, используются круглые скобки.

Использование круглых скобок желательно также в сомнительных случаях для ясности записи и уверенности, что выражение будет вычисляться правильно. Нужно только следить, чтобы количество открывающих скобок совпадало с количеством закрывающих скобок.

Математические выражения в DERIVE всегда вводятся в диалоговом окне Author Expression (<Ctrl>+<A>) в режиме строчного редактирования (рис. 10.3). Количество знаков в строке ввода не ограничено и если выражение не помещается в строке, то левая часть его уходит за пределы экрана и можно продолжать ввод. Для быстрого перемещения курсора в начало или конец строки также используются обычные клавиши <Home> и <End>.

Отметим, что ограничения и неудобства ввода длинных и сложных выражений, вызванные использованием режима строчного редактирования, можно практически исключить использованием буфера обмена (Clipboard), вставляя через него предварительно отредактированные выражения из текстового редактора Word.

10.2.2. Числа

Целые числа представляют собой последовательности десятичных цифр со знаком или без него, написанных без использования десятичной точки: 88, +5, –7. Они могут входить в любые выражения. Система DERIVE ориентирована на работу с любыми числами и также не накладывает ограничений на количество цифр для представления целых чисел с произвольной точностью (см., например, целое число, выражающее результат возведения 2**2000 и представленное под номером (10.1)).

Рациональные числа представляются в виде отношения двух целых чисел с сокращенными общими множителями. Так обычно представляются дробные числа (в виде отношения двух целых несократимых чисел). Поскольку длина целого числа в DERIVE ограничивается лишь размерами используемой оперативной памяти, то это позволяет производить вычисление рациональных значений без округления.

Например, нажмем комбинацию клавиш <Ctrl>+<A> и в появившемся диалоговом окне ввода (рис. 10.3) наберем:

aa:=(1/(-2)+1/7+1/11)^15 (10.13)

и нажмем кнопку OK, после чего оно запишется в строке #1 в окне выражений (рис. 10.12) и будет выделено цветом. Для его вычисления подведем курсор к кнопке = на панели инструментов (при секундной задержке на ней курсора внизу всплывет знакомое нам уже ее название Simplify) и нажмем левую клавишу мыши. В окне выражений (рис. 10.12) под следующим номером #2 появится его вычисленное и упрощенное выражение.

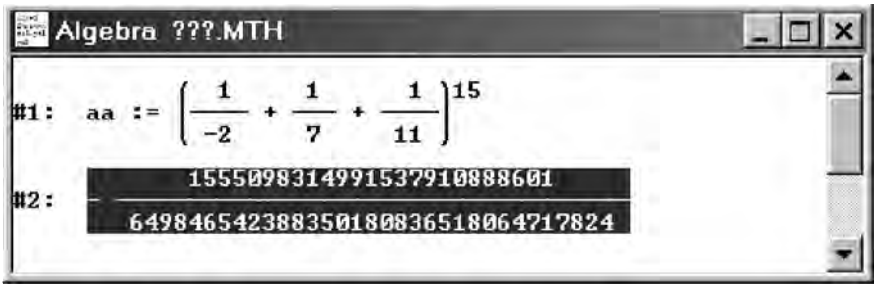

Рис. 10.12. Пример ввода набранного арифметического выражения (10.13) и его вычисления в виде рациональных чисел

Сравнивая полученный результат с аналогичным вычислением выражения (1.15) на REDUCE, убеждаемся в их тождественности.

Вещественные числа записываются в двух формах:

- *в основной форме* число записывается в виде целой и дробной частей, разделенных точкой, со знаком +, – или без знака: +151.2, 1.512, –0.01512;
- *в показательной форме* запись числа состоит из основной формы с указанием десятичного порядка, который обозначается числом 10, а следующая за ним целая константа со знаком или без него определяет величину порядка. Например, +0.1512 10^3, –0.1512 10^–1, 151.2 10^–2.

Однако результаты вычисления вводимых чисел в системе DERIVE зависит от установленного режима, который по умолчанию представляет числа в рациональной форме. Введем, например, число –0.1512 10^–1 в диалоговом окне ввода (рис. 10.3) и нажмем кнопку OK (шаг #3 на рис. 10.13). Постараемся представить это число в основной форме, но после нажатия кнопки = (Simplify) получим неожиданный результат (шаг #4 на рис. 10.13).

Обратимся к главному меню системы DERIVE и выберем там пункт Declare. В открывшемся подменю увидим различные возможности декларации (задания) переменных, функций, матриц и векторов, а также нужное нам задание формата ввода/вывода в подменю Algebra State.

Позиция Algebra State в подменю позиции Declare главного меню выводит свое подменю со следующими командами:

- Input... (<Ctrl>+<I>) — окно опций управления форматом ввода;
- Output... (<Ctrl>+<J>) — окно опций управления форматом вывода;
- Simplification... —окно опций упрощения;
- Reset All — отмена всех изменений опций, установленных предыдущими командами.

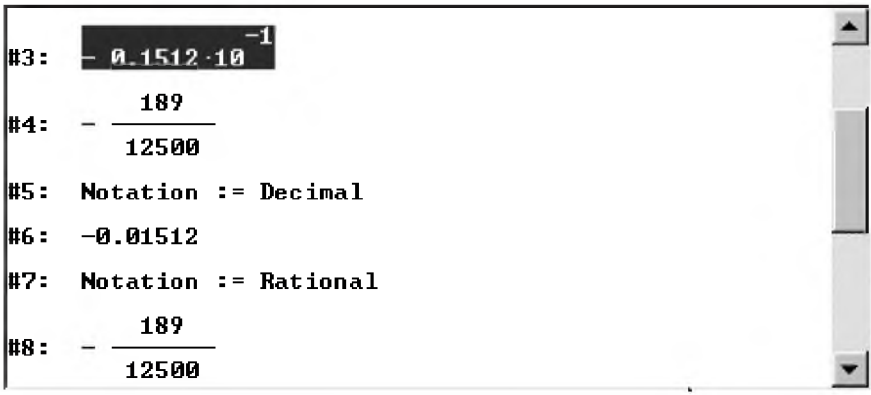

Рис. 10.13. Пример представления введенного числа в различных форматах

Выберем команду Output... (или сразу из основного окна DERIVE нажмем комбинацию клавиш <Ctrl>+<J>). В результате на экране появится диалоговое окно Output Options для установки опций вывода (рис. 10.14). Оно содержит две группы опций для отображения:

- Number display — чисел;
- Expression display —выражений.

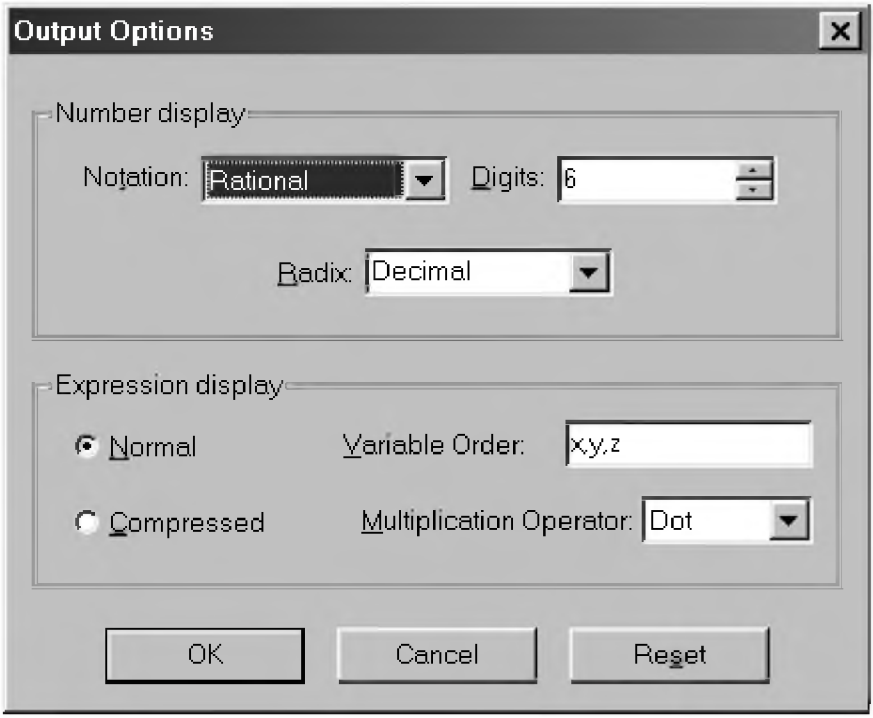

Рис. 10.14. Диалоговое окно Output Options для установки опций вывода

В опции отображения чисел входят:

- в поле Notation установки формата чисел в раскрывающемся списке:
  - десятичный — Decimal;
  - смешанный — Mixed;
  - рациональный — Rational;
  - комплексный — Complex;
- в поле Digits счетчик установки количества цифр при представлении чисел (по умолчанию 6);

- в поле Radix установки основания (двоичного, восьмеричного, десятичного и шестнадцатеричного).

Не увлекаясь пока слишком широкими открывающимися возможностями, выберем в поле Notation из раскрывающегося списка десятичный формат чисел — Decimal. Нажав кнопку OK, обнаружим в окне выражений (в строке #5 на рис. 10.13) соответствующую запись об изменении формата чисел: Notation := Decimal.

Выделим опять строку #3 на экране (рис. 10.13) и после нажатия кнопки = (Simplify) получим долгожданный результат представления этого числа в основной форме (строка #6 на рис. 10.13).

Стараясь без должной необходимости не изменять установки по умолчанию системы, возвратим ее в исходное состояние и проверим получающийся результат (строки #7 и #8 на рис. 10.13).

10.2.3. Идентификаторы и переменные

Идентификаторы используются в качестве переменных, имен массивов (векторов и матриц), а также функций пользователя.

Идентификаторы представляют собой набор букв, цифр и допустимых символов, начинающийся с буквы: X, Y, N, A1, B2, SIN_FI, COS_FI.

В качестве букв используются буквы латинского алфавита от A до Z, цифры: 0, 1, ..., 9.

Некоторые идентификаторы переменных имеют в DERIVE определенные значения:

- **pi**— число π, имеющее значение 3.14159265;
- **deg** — константа, равная π/180, для перевода углов из градусов в радианы;
- **inf** — положительная машинная бесконечность.

Для ввода в диалоговом окне Author Expression (<Ctrl>+<A>) основания натурального логарифма и мнимой единицы (квадратный корень из -1) используются спецсимволы: соответственно **#e** и **#i**. Для отличия от обычных символов в окне выражений они после ввода отображаются соответствующими буквами **e** и **i** с символом ^ над ними.

Переменные — это величины, которым присвоены наименования. Для каждой переменной в памяти машины отводится место, в котором хранится ее численное или символьное значение. Для обозначения переменных используются символические имена (идентификаторы).

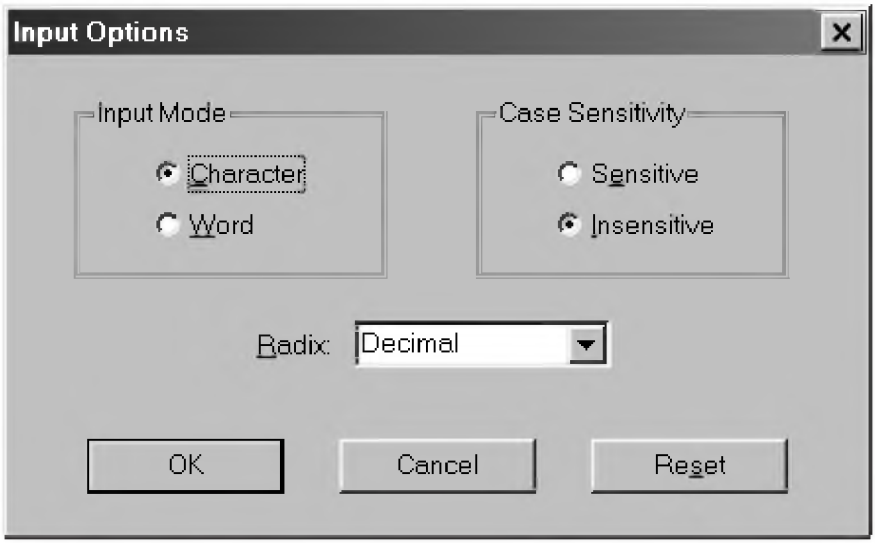

Рис. 10.15. Диалоговое окно Input Options *для установки опций ввода*

Выберем команду Declare → Algebra State → Input... (или сразу из основного окна DERIVE нажмем комбинацию клавиш <Ctrl>+<I>). В результате на экране появится диалоговое окно Input Options для установки опций ввода (рис. 10.15).

Оно содержит две опции для отображения режима ввода Input Mode:

- Character — воспринимает ввод слитных символов по отдельности (установлено по умолчанию);
- Word — воспринимает ввод слитных символов как слово.

Это имеет значение при вводе идентификаторов (имен) переменных. Как мы ранее обращали внимание, что в DERIVE по умолчанию используются однобуквенные переменные. Это происходит потому, что в диалоговом окне Input Options для установки опций ввода (рис. 10.15) по умолчанию установлен режим Character. Поэтому ввод слитных символов воспринимается системой по отдельности: после нажатия кнопки OK DERIVE разделяет их пробелами (например, **yx** и **xz** соответственно преобразует в **y x** и **x z)** и будет готов вычислять их произведения (см. выражение (10.4)).

Чтобы иметь возможность вводить переменные с многобуквенными именами (например, **yx, xz, sin_fi)**, нужно вызвать диалоговое окно Input

Options (рис. 10.15) и установить в нем режим Word. Необходимость в этом возникает при работе с DERIVE достаточно часто.

Другая пара опций группы Case Sensitivity устанавливает:

Sensitive — чувствительность к регистрам переключения размеров символов;

Insensitive — отсутствие чувствительности к размеру вводимых букв (установлено по умолчанию). Необходимость в изменении этого режима возникает при работе с DERIVE достаточно редко.

При переключении режимов ввода после нажатия кнопки OK в окне выражений появляется соответствующая запись: InputMode := Word, InputMode := Character, CaseMode := Sensitive или CaseMode := Insensitive.

В диалоговом окне Input Options имеется также переключатель основания чисел Radix с четырьмя позициями:

- **Binary** — двоичное основание;
- **Octal** — восьмеричное;
- **Decimal** — десятичное (установлено по умолчанию);
- **Hexadecimal** — шестнадцатеричное основание.

Клавиши внизу диалогового окна Input Options (рис. 10.15) выполняют следующие действия:

- OK — фиксирует ввод;
- Cancel — отменяет изменение опций ввода;
- Reset — уничтожает сделанные изменения и задает значения опций, принятые по умолчанию.

Первые две клавиши (OK и Cancel) имеют в DERIVE свое обычное действие. Клавиша Reset впервые появляется в диалоговых окнах системы DERIVE, задающих возможность изменения режимов работы системы, принятых по умолчанию (см., например, рис. 10.14).

Клавиша Reset:

- является новой по отношению к ранее рассмотренным диалоговым окнам Windows 95/98/Me и Word 7.0/97/2000;
- всегда уничтожает сделанные изменения и задает исходные значения опций, принятых по умолчанию.

Каждая переменная обозначается именем, которое является ее первоначальным значением. Необходимо различать имя переменной и ее значение, даже если они совпадают. Имя переменной — обозначение

"емкости", которую с помощью операторов присваивания наполняют содержимым (численным или символьным значением).

Необъявленной (свободной) называется переменная, которой ничего не присвоено и значение которой совпадает с ее именем.

Объявленной (связанной) — переменная, которой ранее было присвоено некоторое значение.

Присвоить значение переменной можно двумя способами:

1. С помощью оператора присваивания (10.12), для чего нужно:
   - вызвать диалоговое окно Author Expression (<Ctrl>+<A>);
   - в строке ввода с использованием оператора присваивания (10.12) задать нужное значение переменной;
   - нажать кнопку OK.

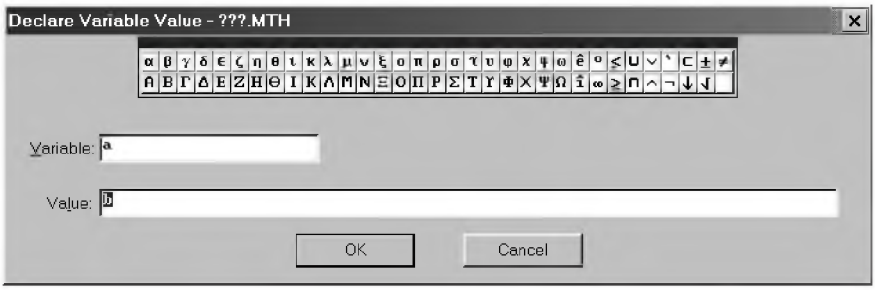

Рис. 10.16. Диалоговое окно Declare Variable Value для декларации переменной и присвоения ей значения

2. С помощью команды Variable Value..., вызывающей соответствующее диалоговое окно из подменю позиции Declare главного меню. Для этого нужно:
   - выбрать команду Declare → Variable Value:
   - в появившемся на экране диалоговом окне (рис. 10.16):
     - в поле Variable задать имя переменной с использованием набора латинских букв и цифр, а также доступных знаков на соответствующей панели этого окна;
     - в поле Value задать ее значение;
   - нажать кнопку OK.

Оба способа задания значений переменной совершенно равнозначны:

- после нажатия кнопки OK строка с присваиванием записывается в соответствующем месте окна выражений (шаг #1 на рис. 10.17);

- заданные любым образом переменные являются *глобальными*:
  - присвоенное переменной значение сохраняется неизменным при дальнейшем выполнении программы или сеанса работы в DERIVE до тех пор, пока снова не будет изменено оператором присваивания (10.12) или в диалоговом окне Declare Variable Value (рис. 10.16);
  - значения переменных можно изменить и использовать в любом месте программы или сеанса работы с DERIVE. Поэтому *при выделении ранее введенных выражений* и нажатии кнопки = (Simplify) *они будут вычисляться* с ***последним набором значений***, присвоенных переменным.

Обратим ваше внимание на некоторое отличие в отражении на экране присваивания свободных и связанных переменных в REDUCE и DERIVE. Сказанное поясним на примерах (1.16)–(1.17), выполнив их в DERIVE (рис. 10.17).

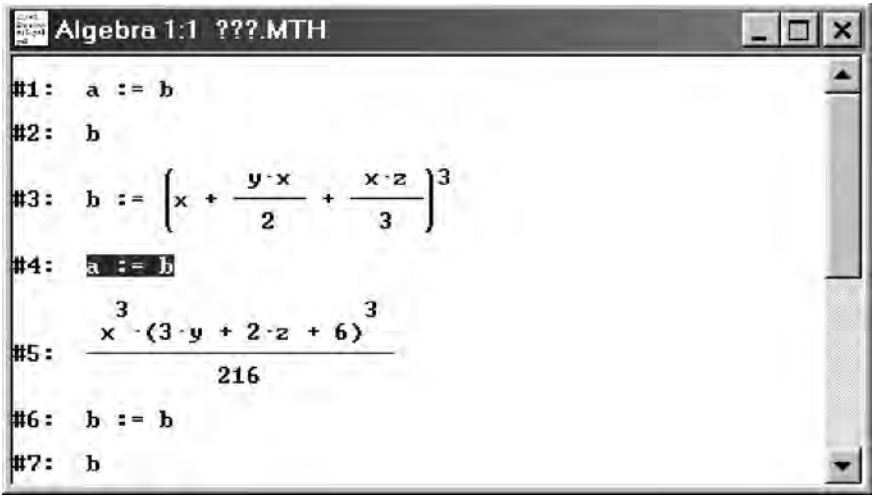

Рис. 10.17. Примеры присваивания свободных и связанных переменных в DERIVE

В DERIVE при вводе арифметического оператора присваивания в диалоговом окне Author Expression (<Ctrl>+<A>) он только записывается в окне выражений после нажатия кнопки OK в своем исходном виде. Это происходит *одинаково* независимо от того, находится в операторе присваивания справа свободная переменная или связанная (соответственно строки #1 и #4 на рис. 10.17).

В REDUCE после нажатия кнопки OK:

- в первом случае *имя* свободной переменной справа присваивается в качестве вычисленного значения переменной слева (пример (1.16));
- во втором случае переменной слева сразу присваивается *вычисленное ранее значение переменной* справа (пример (1.17));

В DERIVE для получения настоящего значения переменной слева, которое будет использоваться везде в вычислениях, нужно при выделенном соответствующем операторе присваивания (#1 или #4 на рис. 10.17) еще нажать кнопку = (Simplify). Результаты представлены в строках #2 и #5 соответственно для присваивания свободной или связанной переменных.

В REDUCE и в DERIVE всегда, когда происходит вычисление выражения, все входящие в него имена исследуются на предмет их совпадения с именами связанных переменных. Если такое совпадение обнаружится, то данное имя заменяется на значение связанной переменной. Имена свободных переменных, которыми являются в нашем примере x, y и z, участвуют в дальнейших преобразованиях.

В REDUCE команда Clear делает связанные переменные свободными, какими они считаются в момент начала выполнения программы. Эта же команда есть в версии DERIVE под DOS в подменю команды Transfer основного меню.

Однако команда Transfer не попала в основное меню версий DERIVE 4.01–4.11 под Windows. Поэтому для этой цели можно использовать любой из двух следующих простых способов.

Чтобы сделать любую связанную переменную свободной, какой она считается в момент начала выполнения программы, нужно:

- *присвоить ей в правой части оператора присваивания то же имя, что и в левой части;*
- *оставить правую часть оператора присваивания свободной.*

При использовании многобуквенных переменных предварительно нужно установить режим Word в диалоговом окне Input Options (<Ctrl>+<I>) для установки опций ввода (рис. 10.15).

Для рассматриваемого на рис. 10.17 примера это будет строка #6 для переменной *b*. После этого вычисление настоящего значения переменной слева при выделенном соответствующем операторе присваивания (#4 на рис. 10.17) даст результат #7, полностью совпадающий с шагом #2 для присваивания свободной переменной.

10.3. Функции или имена со скобками

Помимо переменных, описанных выше, в DERIVE используются также *функции* или *имена со скобками*, в которых указываются списки параметров, разделенных запятыми: Q(5), F(X), SIN(ALFA), COS(BETA), FY(A, F(X), SIN(ALFA)).

Функции или *имена со скобками* имеют следующий формат записи:

```
имя_функции(список параметров)                    (10.14)
```

где `имя_функции` — является ее идентификатором и подчиняется тем же правилам, что и имена переменных, т.е. имя должно содержать набор букв, цифр и допустимых символов, начинающийся с буквы;

`список параметров` — является набором имен переменных, разделенных запятыми. В качестве параметров в списке могут выступать не только простые имена, но и любые выражения, в том числе включающие в себя имена со скобками. При этом все переменные и выражения, входящие в список, являются:

- *формальными параметрами* при задании или описании функции;
- *фактическими параметрами* при обращении к функции для ее вычисления.

Формальные параметры — это идентификаторы, которые внутри функции используются как имена переменных или выражений. Они *локальны*, т. е. область их действия ограничена выражением, определяющим функцию при ее задании или описании.

Фактические параметры, использующиеся при обращении к функции для ее вычисления, являются *глобальными* переменными: имеют в программе или сеансе работы численное или символьное значение, которое может использоваться и изменяться в любом месте программы (документа). Они должны соответствовать формальным параметрам вызываемой функции порядком следования, типом и количеством: первому формальному параметру соответствует первый фактический, второму формальному параметру соответствует второй фактический и т. д.

Используемые обозначения для идентификаторов формальных параметров не играют здесь абсолютно никакой роли, так как соответствие проводится по порядку следования параметров.

Однако везде в пособии для ясности изложения и удобства чтения обычно используются обозначения, которые аналогичны или даже одинаковы

с обозначениями формальных параметров. Это повышает ясность записи и облегчает понимание работы программы.

Если список параметров состоит из одного элемента, то скобки, его ограничивающие, можно заменить пробелами. Однако этим нарушается ясность записи. Поэтому мы сами так никогда поступать не станем, и вам не советуем.

Первая операция, которую выполняет DERIVE, встречая в программе имя со скобками, — вычисление значений выражений, записанных в списке параметров. Иначе говоря, *список параметров всегда преобразуется к списку их значений*, и только после этого начинается анализ самого имени со скобками.

В DERIVE имена со скобками применяются для обозначения функций пользователя (см. п. 10.3.1), математических функций (см. п. 10.3.2), векторных и матричных функций.

10.3.1. Функции пользователя или оператор-функции

Функции, действующие в системе, называются операторами. В программировании под функцией понимают некоторую вычислительную операцию над заданными параметрами, возвращающую в ответ на обращение к ней единственный результат. Это может быть число того или иного типа (например, действительное или комплексное) или символьное выражение, полученное в ходе заданных операций.

Основным свойством оператора является то, что само его имя вместе со списком значений параметров, который у оператора принято называть списком аргументов, может быть его значением. Иными словами, *имя со скобкой, при использовании его в качестве оператора может условно рассматриваться в качестве свободной переменной.*

Это свойство свободных имен со скобками в DERIVE при использовании в их качестве системных функций, являющихся также разновидностью операторов, позволит нам в дальнейшем с легкостью проводить различные исследования, например, изучение вариаций различных геометрических факторов и действующих нагрузок в различных задачах.

Функции бывают встроенные (системные) и задаваемые пользователем. Несмотря на обилие встроенных в систему функций (экспоненциальных, показательных, тригонометрических, статистических и иных), всегда может потребоваться та или иная новая функция.

Задать функцию пользователя можно двумя способами:

1. С помощью оператора присваивания, для чего нужно:
   - вызвать диалоговое окно Author Expression (<Ctrl>+<A>);
   - в строке ввода задать имя функции пользователя со списком параметров и после оператора присваивания нужное выражение для ее определения по следующему формату:

```
имя_функции(список параметров) := выражение     (10.15)
```

   - нажать кнопку OK.

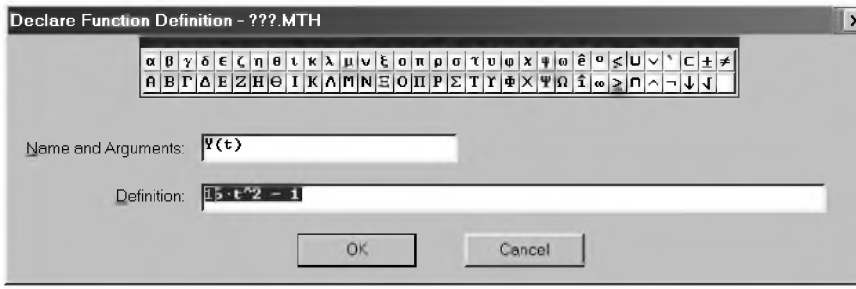

Рис. 10.18. Диалоговое окно Declare Function Definition *для декларации функции пользователя и ее определения*

2. С помощью команды Function Definition..., вызывающей соответствующее диалоговое окно из подменю позиции Declare главного меню. Для этого нужно:
   - выбрать команду Declare → Function Definition;
   - в появившемся на экране диалоговом окне (рис. 10.18):
     - в поле Name and Arguments задать по формату (10.14) имя функции со списком формальных параметров (через запятую), заключенных в скобки;
     - в поле Definition задать нужное выражение для определения функции пользователя;
   - нажать кнопку OK.

Оба способа задания функции пользователя совершенно равнозначны: после нажатия кнопки OK заданная функция будет введена в очередную строку окна выражений и станет доступна для использования.

Пример. Для возможности неоднократного вычисления при различных T значений координат X и Y, задающих уравнения движения

материальной точки (3.4), можно задать следующие функции пользователя или оператор-функции:

```
X(T):= 4*T                                    (10.16)
Y(T):= 16*T^2 - 1                             (10.17)
```

Здесь именами операторов-функций являются однобуквенные идентификаторы X и Y, а список формальных параметров состоит из одного аргумента T.

Формальные параметры не занимают никакого места в памяти. Они могут быть такими же, как и наименования переменных в другом месте программы, и совершенно несущественны (за исключением указания типа).

Они используются только для обозначения последовательности операций, которые нужно будет выполнить над соответствующими фактическими параметрами.

Поэтому ничего не изменится, если в (10.16)–(10.17) вместо формального параметра T использовать, например, X1 и X2 (предварительно установив в диалоговом окне Input Options (<Ctrl>+<I>) опцию Word (рис. 10.15) для возможности использования многобуквенных идентификаторов):

```
X(X1):= 4*X1                                  (10.18)
Y(X2):= 16*X2^2 - 1                           (10.19)
```

Обращение к операторам-функциям (10.16)–(10.17) или (10.18)–(10.19) будет одинаковым. Для этого следует в любом месте программы записать имя функции пользователя, указав при этом нужные фактические параметры, которые могут быть константами, простыми или индексными переменными, выражениями. Результат присваивается имени функции пользователя.

Например, для вычисления значений рассматриваемых функций (3.4) в трех различных точках программы при T+DT, T и T–DT, необходимо просто записать обращение к этим операторам-функциям в виде: X(T+DT), X(T) и X(T–DT) или Y(T+DT), Y(T) и Y(T–DT) соответственно.

При этом каждый раз указанный фактический параметр T+DT, T и T–DT, имеющий соответствующее значение в данном месте программы, будет подставлен вместо формального параметра (T в (10.16)–(10.17) или X1 в (10.18) и X2 в (10.19) — все равно).

С ним в этом месте программы будут проделаны все необходимые вычисления и определено значение функции:

```
AX:=(X(T+DT)-2*X(T)+X(T-DT))/DT^2             (10.20)
AY:=(Y(T+DT)-2*Y(T)+Y(T-DT))/DT^2             (10.21)
```

Отметим, что выражение в правой части определения функции пользователя может содержать переменные, которых нет в списке формальных параметров.

Например, операторы-функции (10.16)–(10.17) можно записать, используя вместо чисел переменные A1, B, C. Эти переменные являются общими (глобальными) для всей программы или документа (в отличие от формального параметра T).

Для выполнения численного расчета значения этих переменных должны быть определены последующими операторами, но обязательно до первого обращения к функции, где эти переменные используются:

```
X(T):=A1*T
Y(T):=B*T^2-C                                          (10.22)
.......................
A1:= 4
B:= 16
C:=1                                                   (10.23)
AX:=(X(T+DT)-2*X(T)+X(T-DT))/DT^2
AY:=(Y(T+DT)-2*Y(T)+Y(T-DT))/DT^2
```

Последние два оператора, иллюстрирующие обращение к операторам-функциям, остаются, конечно, без изменений и совпадают с операторами (10.20)–(10.21).

Выражение в правой части определения функции пользователя может содержать математические системные функции (см. п. 10.3.2), а также другие операторы-функции, определенные раньше. Например, нижеследующие функции пользователя эквивалентны операторам (10.16)–(10.17):

```
X(T):=4*T
Y(T):=X(T)^2-1                                         (10.24)
```

10.3.2. Математические функции

В систему DERIVE встроены многочисленные математические функции. Системе известны элементарные соотношения и свойства этих функций.

Для удобства пользования состав встроенных элементарных функций сделан достаточно полным.

Признаком функции является возврат значения, численного или символьного. Поэтому функции могут использоваться совместно с операторами для записи математических выражений, вычисляемых Derive.

При установках по умолчанию (режим Insensitive) функции отображаются, в отличие от переменных, прописными буквами, например SIN(x), но набираться могут буквами *любых* размеров.

Обратим ваше внимание, что при установке в диалоговом окне Input Options (<Ctrl>+<I>) опции Sensitive (рис. 10.15), устанавливающей режим чувствительности к регистрам вводимых символов, набираться имена функций должны также только прописными буквами.

Встроенные в ядро Derive основные математические функции можно квалифицировать по следующим группам:

1. ***Числовые функции*** возвращают (*x* и *y* имеют действительные значения):

ABS(x) — абсолютное значение *x*;

SIGN(x) — 1 при $x > 0$; 0 при $x = 0$ и –1 при $x < 0$;

MAX(x, y,...) — максимальное значение из списка значений аргументов;

MIN(x, y,...) — минимальное значение из списка значений аргументов;

STEP(x) — 1 при $x > 0$ и 0 при $x < 0$;

CHI(a, x, b) — 1 при $a < x < b$ и 0 при $x < a$ или $x > b$;

NEXT_PRIME(n) — следующее простое число, превосходящее *n*.

2. ***Степенные (экспоненциальные) функции*** возвращают:

SQRT(z) — квадратный корень из *z* (на мониторе отображается как $\sqrt{z}$ и может быть также вставлен в выражение щелчком по знаку $\sqrt{}$ на панели специальных символов диалогового окна ввода выражений (рис. 10.3));

$\hat{e}$ — основание натурального логарифма;

EXP(z) — число $\hat{e}$ в степени *z*.

3. ***Логарифмические функции*** возвращают:

LN(z) — главное значение натурального логарифма *z*;

LOG(z, w) — логарифм *z* по основанию *w*.

4. ***Тригонометрические функции*** возвращают соответствующие функции угла z: *SIN(z)* — синус, *COS(z)* — косинус, *TAN(z)* — тангенс, *COT(z)* — котангенс, *SEC(z)* — секанс, *CSC(z)* — косеканс угла z.

Угол z обычно задается в радианах. При задании в градусах его следует умножать на преобразующую константу *deg*, например *SIN(z*deg)*, где *z* — угол в градусах.

5. ***Обратные тригонометрические функции*** возвращают значение угла *z*, выраженного в радианах:

ASIN(z) — арксинус *z;*
ACOS(z) — арккосинус *z*;
ATAN(z) — арктангенс *z*;
ATAN(y, x) — арктангенс радиус-вектора точки (*x, y*), причем угол измеряется от положительной полуоси *x*;
ACOT(z) — арккотангенс *z*;
ACOT(x, y) — арккотангенс радиус-вектора точки (*x, y*);
ASEC(z) — арксеканс *z*;
ACSC(z) — арккосеканс *z*.

Результат этих функций — угол в радианах, приведенный в диапазоне от $-\pi$ до π. Исключением является угол, возвращаемый функцией *ATAN(y, x)*; он лежит в области значений от $-\pi/2$ до $\pi/2$. В этом же диапазоне углов лежат значения ASIN(x), если $-1 \le x \le 1$. При тех же значениях *x* значение *ACOS(x)* лежит в пределах от 0 до π.

Если желателен возврат угла этими функциями в градусах, следует поделить возвращаемое значение на константу *deg*.При преобразованиях *обратных тригонометрических* функций действуют следующие соотношения:

$$ACOT(z) = \pi/2 - ATAN(z), \quad ACOT(x, y) = ATAN(y, x),$$
$$ATAN(y, x) = ATAN(y/x), \quad ACOS(z) = \pi/2 - ASIN(z), \quad (10.25)$$
$$ASEC(z) = ACOS(l/z), \quad ACSC(z) = ASIN(l/z).$$

6. ***Гиперболические функции:***

SINH (z) — гиперболический синус *z*;
COSH (z) — гиперболический косинус *z*;
TANH (z) — гиперболический тангенс *z*;
COTH (z) — гиперболический котангенс *z*;
SECH (z) — гиперболический секанс *z*;
CSCH (z) — гиперболический косеканс *z*.

При упрощении гиперболических функций используются их определения через экспоненциальные функции:

$$SINH(z) = \frac{e^z}{2} - \frac{e^{-z}}{2}, \qquad COSH(z) = \frac{e^z}{2} + \frac{e^{-z}}{2},$$

TANH(z)=SINH(z)/COSH(z), *COTH(z) = COSH(z)/SINH(z),* (10.26)
SECH (z)=1/COSH(z), *CSCH (z) = 1/SINH(z).*

7. Обратные гиперболические функции:

ASINH (z) — обратный гиперболический синус *z*;
ACOSH (z) — обратный гиперболический косинус *z;*
ATANH (z) — обратный гиперболический тангенс *z*;
ACOTH (z) — обратный гиперболический котангенс *z*;
ASECH (z) — обратный гиперболический секанс *z*;
ACSCH (z) — обратный гиперболический косеканс *z*.

Обратные гиперболические функции упрощаются к их логарифмическим представлениям:

$$ASINH(z) = LN(z+\sqrt{z^2+1}),\quad ACOSH(z) = LN(z+\sqrt{z^2-1}),$$

$$ATANH(z) = \frac{LN(1+z)}{2} - \frac{LN(1-z)}{2}, \qquad (10.27)$$

$$ACOTH(z) = \frac{LN(z+1)}{2} - \frac{LN(z-1)}{2}.$$

8. Функции комплексного аргумента:

$\hat{i}$ — мнимая единица;
ABS(z) — модуль *z*;
SIGN (z) — упрощается к точке на единичной окружности с тем же углом, что у *z* (при *z* = 0 — угол любой и упрощения не происходит);
RE (z) — действительная часть *z*;
IM (z) — мнимая часть *z*;
CONJ (z) — комплексно-сопряженное с *z* число;
PHASE (z) — фаза (угол радиус-вектора *z*, определенный в диапазоне значений от –π до π).

9. Факториальные и комбинаторные функции:

z! — факториал *z*;
GAMMA (z) или Г(z) — гамма-функция;
PERM (n, m) — число размещений из *n* элементов по *m*;
COMB (n, m) — число сочетаний из *n* элементов по *m*.

ГЛАВА 11. ИНТЕГРИРОВАНИЕ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ В DERIVE НА ПРИМЕРАХ ЕГО ИСПОЛЬЗОВАНИЯ В ЗАДАЧАХ ДИНАМИКИ

Решение многих задач в математике, физике, теоретической механике и других областях науки и техники связано с решением дифференциальных уравнений и систем из них.

В частности, такие уравнения широко используются в задачах динамики.

DERIVE, в отличие от CAB REDUCE, содержит три библиотечных файла внешних расширений, ориентированных на решение дифференциальных уравнений различного типа:

- файл Ode1.mth содержит функции для точного решения различных дифференциальных уравнений *первого* порядка с применением команды Simplify ;
- файл Ode2.mth содержит ряд функций (включая тестовые со словом TEST в конце имени, проверяющие допустимость применения основных функций) для точного решения дифференциальных уравнений *второго* порядка (см. пп. 11.1–11.3);
- файл Ode_appr.mth содержит дополнительные расширения для приближенного решения дифференциальных уравнений. Из них наиболее важной является функция RK, обеспечивающая решение систем нелинейных дифференциальных уравнений численным методом Рунге – Кутта (см. п. 11.5).

11.1. Интегрирование в DERIVE обыкновенных дифференциальных уравнений второго порядка

В разделе динамика курса теоретической механики широко используются обыкновенные дифференциальные уравнения второго

порядка. Поэтому с них начнем наше рассмотрение интегрирования обыкновенных дифференциальных уравнений.

Решение дифференциальных уравнений второго порядка обеспечивается функциями файла Ode2.mth (рис. 11.1).

Их набор дает прекрасный пример техники программирования на языке системы DERIVE и свидетельствует о компактности программы. Он является наглядным примером функционального программирования, когда конечная функция определена через ряд промежуточных функций.

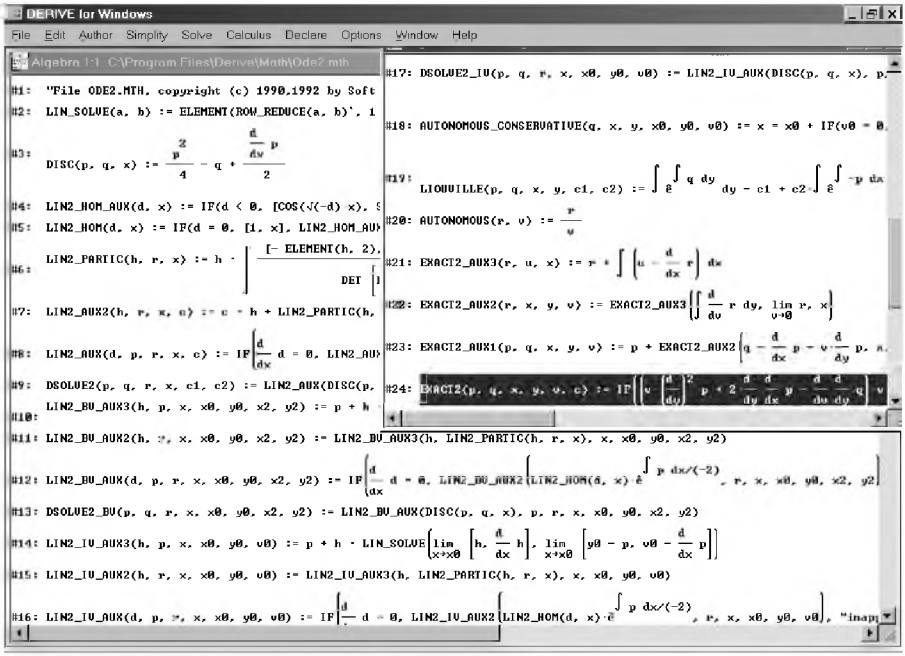

Рис. 11.1. Функции файла Ode2. mth для точного решения обыкновенных дифференциальных уравнений второго порядка

Файл Ode2.mth расположен по адресу C:\Program Files\Derive\Math\Ode2.mth. Чтобы использовать функции файла Ode2.mth, представленные на рис. 11.1, для точного интегрирования обыкновенных дифференциальных уравнений второго порядка, необходимо сначала его загрузить. Для этого нужно:

- запустить DERIVE;
- вызвать на выполнение файл Ode2.mth;

- записать его под вашим рабочим именем. Это можно сделать командой File → Save As... из главного меню, набрав в появившемся диалоговом окне нужное имя в поле Имя файла (рис. 10.10), а также указав нужную папку в раскрывающемся списке Папка (рис. 10.11);
- начать выполнять нужное вам интегрирование в этом рабочем файле.

Обычно решение уравнения можно осуществить несколькими способами.

Сначала в файле Ode2.mth описаны наиболее простые прикладные методы, в результате реализации которых приходят к решению исходного уравнения в явной форме.

Напротив, методы, описанные позднее, часто дают решения в неявной форме или включают большее количество шагов, требующих, например, последовательного решения двух дифференциальных уравнений первого порядка с помощью функций файла Ode1.mth.

Если решение невозможно или не удовлетворяет указанным условиям, то возвращается сообщение "*inapplicable*" ("*неприменимо*").

Решение дифференциальных уравнений второго порядка обеспечивается следующими функциями файла Ode2.mth.

DSOLVE2 (p, q, r, x, c1, c2) получает общее решение линейного дифференциального уравнения второго порядка

```
y'' + p(x)·y' + q(x)·y = r(x)                    (11.1)
```

в явной форме с использованием постоянных интегрирования c1 и c2.

Два последних формальных параметра могут быть опущены, если они переменные и нет необходимости в их переименовании. DSOLVE2 может легко находить решение и в том случае, когда функции p и q являются численными константами.

Полученное решение может содержать неопределенные интегралы, включающие r(x), для которых DERIVE не может определить первообразные. Для проверки такого результата следует убедиться в выполнении соответствующего тождества:

```
y'' + p(x)·y' + q(x)·y - r(x) = 0.
```

Если требуется отыскать определенное решение, удовлетворяющее символьным или числовым начальным или граничным условиям, то лучше использовать DSOLVE2_IV или DSOLVE2_BV, которые непосредственно возвращают необходимые решения.

DSOLVE2_BV (p, q, r, x, x0, y0, x2, y2) аналогична функции DSOLVE2, но получает определенное решение, которое удовлетворяет *граничным* условиям y = y0 при x = x0 и y = y2 при x = x2.

DSOLVE2_IV (p, q, r, x, x0, y0, v0) также подобна функции DSOLVE2_BV, но получает определенное решение, которое удовлетворяет *начальным* условиям y = y0 и y' = v0 при x = x0. По этой причине ей удобно пользоваться при интегрировании различных дифференциальных уравнений задач динамики.

AUTONOMOUS_CONSERVATIVE (q, x, y, x0, y0, v0) получает в неявной форме алгебраическое решение уравнения вида `y'' = q(y)`, которое имеет начальные условия y = y0 и y' = v0 при x = x0. Если вместо начального условия y' = v0 имеется второе граничное условие y = y2 при x = x2, то необходимо заменить x2 на x и y2 на y в решении и вычислить v0.

LIOUVILLE (p, q, x, y, c1, c2) получает в неявной форме алгебраическое решение уравнения

```
y'' + p(x)·y' + q(y)·y'^2 = 0.                    (11.2)
```

Это решение представляет собой линейную комбинацию двух произвольных постоянных c1 и c2. Следовательно, можно совмещать два набора начальных или граничных условий, и затем одновременно решать два линейных уравнения для c1 и c2.

AUTONOMOUS (r, v) получает выражение для dv/dy с учетом уравнения `y'' = r(y,v)`, в котором v представляет y'. Эта функция приводит заданное уравнение `y'' = r(y,v)` к двум последовательным уравнениям первого порядка Если r не зависит от v, то рекомендуется использовать функцию AUTONOMOUS_CONSERVATIVE, чтобы решить уравнение за один шаг.

EXACT2(p, q, x, y, v, c) понижает порядок дифференциального уравнения

```
p(x, y, v)·y'' + q(x, y, v) = 0,                  (11.3)
```

если оно в полных дифференциалах. В данном случае переменная *v* используется для представления *y': v = y'*.

В результате получается алгебраическое уравнение, связывающее переменные y, v, x и произвольную символьную константу c. Если уравнение не представляется в полных дифференциалах, то EXACT2 возвращает сообщение "*inapplicable*" ("*неприменимо*").

11.2. Интегрирование в DERIVE одномерных задач динамики в аналитическом виде

11.2.1. Использование функции DSOLVE2_IV для решения задач динамики

В теоретической механике обычно используются производные по времени *t*, которые обозначаются соответствующим числом точек над буквами. Поэтому общий вид линейного дифференциального уравнения второго порядка (11.1) примет следующую форму записи для этого случая:

$$\ddot{x} + p(t)\dot{x} + q(t)x = r(t) \qquad (11.4)$$

При интегрировании различных дифференциальных уравнений задач динамики обычно известны *начальные* условия $x = x0$ и $\dot{x} = v0$ при $t = t0$. По этой причине для их решения удобно использовать функцию *DSOLVE2_IV (p, q, r, t, t0, x0, v0)*. Она также может легко находить решение и в том случае, когда функции p и q являются численными константами.

Функция *DSOLVE2_IV* в файле *Ode2.mth* записана в строке #17 и определена с использованием восьми промежуточных функций, расположенных в строках #2–#6 и #14–#16, которые нужно загрузить в документ, в котором предполагается выполнять интегрирование. Проще всего это выполнить, как описано в п. 11.1: загрузив файл *Ode2.mth* и присвоив ему собственное имя, например, *Ris14-1.mth* (в вашей рабочей папке).

Теперь в файле *Ris14-1.mth* можно:

- удалить для рассматриваемых ниже трех примеров ненужные функции в строках #7–#13 и #18–#24, перенумеровав по порядку оставшиеся строки командой Edit → Renumber;
- оставить в файле *Ris14-1.mth* все функции файла *Ode2.mth* без изменения, повинуясь требованиям мудрой колеи осторожности:
  - при возникновении осложнений в ряде случаев нам может понадобиться использование других функций;
  - удаление принадлежит к числу опасных операций, в результате которых по неосторожности может нарушиться согласование в работе используемых промежуточных функций.

Мы всегда выбираем более простое второе решение. Если вы захотите удалить указанные функции и проверить результат, то отличие в приведенных ниже двух сеансах работы будет только в нумерации: у вас на рис. 11.3 она начнется не со строки #25, а с #10.

Напомним также, что любая функция в DERIVE задается с использованием формальных параметров (как и оператор-функция в Фортране или оператор в REDUCE). Поэтому не имеет никакого значения, какие идентификаторы использовались при ее задании в качестве параметров функции *DSOLVE2_IV (p, q, r, x, x0, y0, v0)* при ее описании в строке #17 файла *Ode2.mth*.

При вызове функции *DSOLVE2_IV (p, q, r, t, t0, x0, v0)* в файле *Ris14-1.mth* указанные в ней фактические параметры *(p, q, r, t, t0, x0, v0)* будут поставлены на место соответствующих формальных параметров *(p, q, r, x, x0, y0, v0)*, после чего с ними будут выполнены все необходимые действия.

Фактические параметры должны соответствовать формальным параметрам функции пользователя только своим порядком следования и типом: первому формальному параметру соответствует первый фактический, второму формальному параметру соответствует второй фактический и т. д.

Поэтому все будет правильно и вместо формальных параметров *x, x0, y0* будут соответственно использованы нужные фактические параметры *t, t0, x0*.

11.2.2. Интегрирование линейных одномерных дифференциальных уравнений

В качестве линейных одномерных дифференциальных уравнений рассмотрим типовые примеры задач по исследованию колебательного (Д-3 [23, с. 138-148]) и относительного (Д-4 [23, с. 148-155]) движения материальной точки, а также свободных колебаний механической системы с одной степенью свободы (Д-23 [23, с. 312-320]). Все перечисленные задания решаются аналитически на ПК в системе DERIVE и обычными методами.

Для аналитического решения на ПК нужно составить дифференциальное уравнение движения материальной точки и определить числовые значения его коэффициентов. Затем следует сеанс работы на ПК, рассмотренный ниже.

Одновременно с этим обычными методами механики студентом производится параллельное аналитическое решение указанных заданий (которое рассмотрено в сборниках [22], [23] и в пособии не повторяется).

Для типовых примеров рассматриваемых заданий интегрируемые дифференциальные уравнения после проведения нужных вычислений, могут быть представлены в следующих видах:

- для Д-3 ([23, с. 147]):

$$\ddot{x} + k^2 x = h\sin(pt) \quad \text{или} \qquad (11.5)$$

$$\ddot{x} + 300x = 6\sin(10t) \qquad (11.6)$$

где $k^2 = c/m_D = 600/2 = 300\ (c^{-1})$; $h = d\ c/m_D = 0{,}02\ 600/2 = 6\ (м/c^2)$; $p = 10(c^{-1})$;

- для Д-4 (см. в [23, с. 153] уравнение (2)):

$$\ddot{x} + (c/m - \omega^2 \sin^2\alpha)x = \omega^2 r\sin\alpha - g\cos\alpha + cl_0/m \qquad (11.7)$$

где $c = 1$ *Н/м*; $\omega = \pi$ *рад/с*$^{-1}$; $\alpha = \pi/6$ *рад*; $m = 0{,}01$ *кг*; $r = 0{,}2$ *м* $= l_0$;

- для Д-23 (см. в [23, с. 319] уравнение (2)):

$$\ddot{y} + k^2 y = 0 \qquad (11.8)$$

где $k = \sqrt{\dfrac{9c - 8G_6/l}{m_1 + m_2/2 + 16m_6/3 + 6m_4}}$; $m_1 = 1$ *кг*, $m_2 = 2$ *кг*, $m_4 = 1$ *кг*, $m_6 = 3$ *кг*, $l = 0{,}6$ *м*, $c = 20$ *Н/см* $= 2000$ *Н/м*, $y_0 = 0{,}2$ *см*, $\dot{y}_0 = 8$ *см/с*.

Три дифференциальных уравнения (11.6)–(11.8) являются частными случаями его общего вида (11.4), для решения которого удобно использовать функцию DSOLVE2_IV (p, q, r, t, t0, x0, v0). Эту функцию вместе с нужными промежуточными функциями нужно загрузить в документ, в котором предполагается выполнять интегрирование. Для этого загрузим файл Ode2.mth (см. п. 11.1 и рис. 11.1) и присвоим ему собственное имя, например, Ris14-2.mth (в вашей рабочей папке).

Установим опять принятое по умолчанию количество цифр при выводе результатов, выдав команду Declare → Algebra State → Output... (или сразу из основного окна DERIVE нажмем комбинацию клавиш <Ctrl>+<J>). В результате на экране появится диалоговое окно Output Options для установки опций вывода (рис. 10.14). В нем в поле Digits наберем число 6

для установки количества цифр при представлении чисел. После нажатия кнопки OK соответствующая запись появится в строке #25 рис. 11.3.

Строка #26 представляет собой введенный нами краткий комментарий с указанием номера выполняемого задания (Д-3), решение в DERIVE типового примера которого рассматривается нами ниже.

Выдадим команду Declare → Algebra State → Simplification… и в появившемся диалоговом окне Simplification Options (рис. 13.6) в поле Trigonometry установим направление преобразования Collect — сбор. Оно обеспечивает получение более компактной формы выражений после интегрирования дифференциального уравнения. После нажатия кнопки OK сообщение об установке нового режима появится в строке #27 рис. 11.3.

В строке #28 рис. 11.3 записывается обращение к функции *DSOLVE2_IV (p, q, r, t, t0, x0, v0)* со следующими фактическими параметрами *DSOLVE2_IV(0, 300, 6·SIN(10·t), t, 0, -0.0245, 0)* для решения дифференциального уравнения (11.6):

- функция *p* перед первой производной по *x* из общего вида дифференциального уравнения (11.4) имеет численное значение в (11.6), равное нулю;
- функция *q* перед координатой *x* из общего вида дифференциального уравнения (11.4) имеет численное значение в (11.6), равное *300*;
- функция *r(t)* в правой части общего вида дифференциального уравнения (11.4) имеет в (11.6) следующий вид: *6·SIN(10·t)*;
- на четвертом и пятом местах в списке фактических параметров указываются переменная интегрирования *t* и ее начальное значение *t0*, равное нулю;
- на шестом и седьмом местах в списке фактических параметров указываются начальные условия при *t=0*:
  - значение координаты *x0 = –0.0245* (м);
  - величина скорости *v0*, равная нулю.

Результат решения дифференциального уравнения (11.6) после выделения строки #28 и нажатия:

- кнопки = (Simplify) — записывается в строке #29;
- кнопки ≈ (Approximate) — #30.

Представим решение в другой форме, для чего выдадим команду Declare → Algebra State → Simplification… и в появившемся диалоговом окне Simplification Options (рис. 13.6) в поле Trigonometry установим направление преобразования Expand — расширение. После нажатия кнопки OK сообщение об установке нового режима появится в строке #31 рис. 11.3.

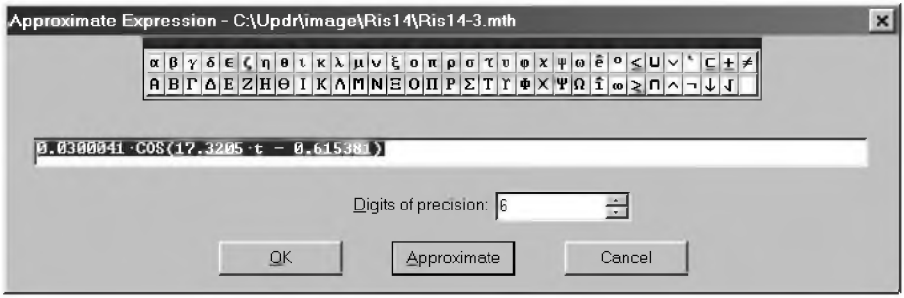

Рис. 11.2. Диалоговое окно Approximate Expression *(<Ctrl>+<G>)*

Теперь выделим правую часть выражения #29 или #30, дважды с небольшим интервалом щелкнув по ней мышкой.

Затем выдадим команду Simplify → Approximate... (<Ctrl>+<G>) и в появившемся диалоговом окне Approximate Expression (рис. 11.2) нажмем кнопку OK. В результате предполагаемая к выполнению команда будет записана в строке #32 рис. 11.3.

Повторим все действия предыдущего абзаца и в появившемся диалоговом окне Approximate Expression (рис. 11.2) нажмем кнопку Approximate.

В результате получим новую форму записи решения рассматриваемого дифференциального уравнения в строке #33 рис. 11.3.

Обратите внимание, что мы выделяли только правую часть выражения #29 или #30 на рис. 11.3, в которой был записан синус суммы в виде рациональных (#29) или вещественных (#30) чисел.

Однако, после нажатия кнопки Approximate, он не только был раскрыт с учетом установленного в строке #31 нового направления преобразования Expand, но в строке #33 оказалось записанным полностью решение дифференциального уравнения, только часть которого была выделена в строке #29 или #30. Отметим, что это является особенностью DERIVE.

В графическом окне рис. 11.3 построен график, который получается после выделения любой из форм записи рассматриваемого дифференциального уравнения #29, #30 или #33 на рис. 11.3 и выполнения следующих действий:

- открытия графического окна, например, с помощью комбинации клавиш <Ctrl>+<2>;

- нажатия кнопки  (*Plot expression*) на *панели инструментов* графического окна.

Для отличия кривая D_3 помечена названием темы рассматриваемого типового примера Д-3 и представляет собой результат сложения:

- трех колебательных движений без сдвига фаз в форме записи результатов #33;
- двух колебательных движений со сдвигом фаз в любой из форм представления результатов решения #29 или #30.

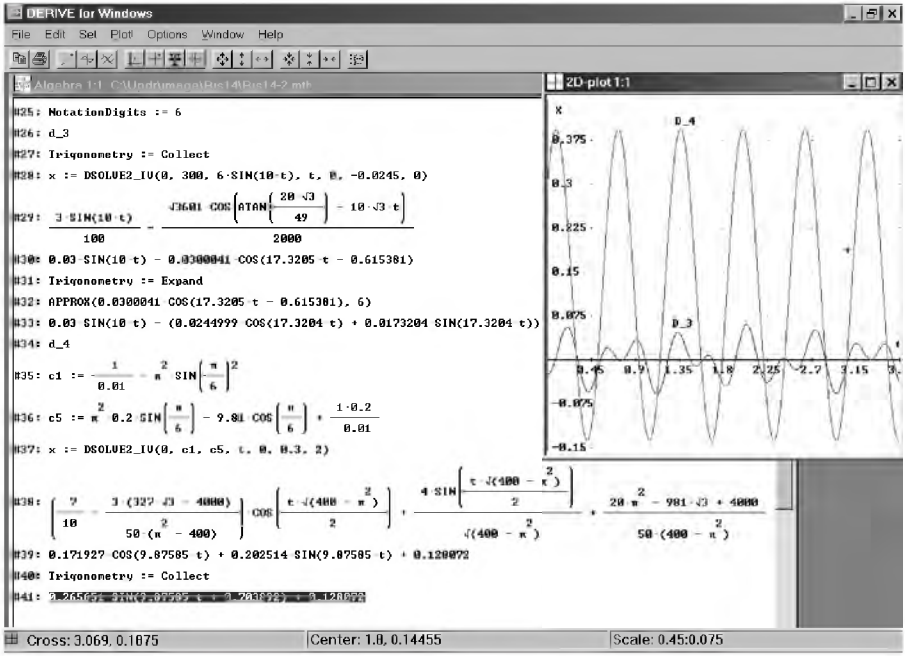

Рис. 11.3. Интегрирование дифференциальных уравнений для типовых примеров колебательного Д-3 ([23, с. 147]) и относительного Д-4 ([23, с. 153]) движения материальной точки с графическим представлением результатов

В строках #34–#41 рис. 11.3 и #42–#49 рис. 11.4 расположены аналогичные решения интегрирования линейных дифференциальных уравнений второго порядка соответственно для типовых примеров заданий Д-4 ([23, с. 148-155]) и Д-23 ([23, с. 312-320]).

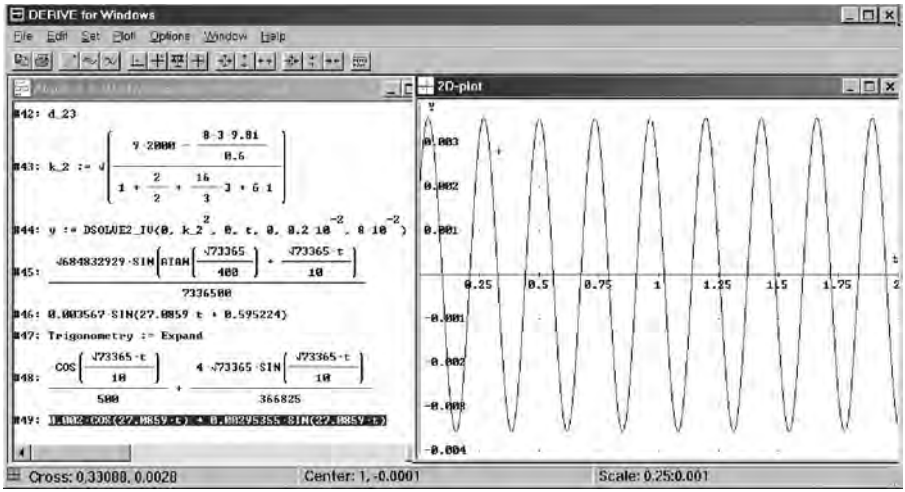

Рис. 11.4. Интегрирование дифференциального уравнения для типового примера свободных колебаний механической системы с одной степенью свободы Д-23 ([23, с. 319]) с графическим представлением результатов

Последовательность действий для их решения аналогична рассмотренной для типового примера Д-3.

Строки #34 рис. 11.3 и #42 рис. 11.4 представляют собой введенный нами краткий комментарий с указанием номеров выполняемых заданий (Д-4 и Д-23), решение в DERIVE типовых примеров которых рассматривается нами ниже.

В строках #35, #36 рис. 11.3 и #43 рис. 11.4 присваиваются значения переменным *c1, c5* (Д-4) и *к_2* (Д-23), которые используются при задании дифференциальных уравнений рассматриваемых типовых примеров в функции *DSOLVE2_IV*.

Обратите внимание, что в соответствующие формулы для циклической частоты (*c1 и к_2*) и постоянной *c5* в правой части дифференциального уравнения (11.7) просто подставляются данные условия в численном виде. Нужные вычисления для определения этих переменных будут выполнены самой системой DERIVE.

В строках #37 рис. 11.3 и #44 рис. 11.4 записываются обращения к функции *DSOLVE2_IV (p, q, r, t, t0, x0, v0)* для решения типовых примеров дифференциальных уравнений:

- Д-4 (11.7) — *DSOLVE2_IV(0, c1, c5, t, 0, 0.3, 2)*;
- Д-23 (11.8) — *DSOLVE2_IV(0, k_2², 0, t, 0, 0.2·10⁻², 8·10⁻²)*.

Используемые при обращениях фактические параметры имеют следующие значения:

- функция *p* перед первой производной по *x* из общего вида дифференциального уравнения (11.4) имеет численное значение в (11.7) и (11.8), равное нулю;
- функция *q* перед координатой *x* из общего вида дифференциального уравнения (11.4) имеет численное значение:
  - в (11.7), равное *c1*;
  - в (11.8), равное *к_2²*;
- функция *r(t)* в правой части общего вида дифференциального уравнения (11.4) является в (11.7) постоянной и обозначается через *c5*, а в (11.8) равняется нулю;
- на четвертом и пятом местах в списке фактических параметров указываются переменная интегрирования *t* и ее начальное значение *t0*, равное нулю;
- на шестом и седьмом местах в списке фактических параметров указываются начальные условия при *t=0*:
  - для типового примера Д-4 ([23, с. 149]):
    - значение координаты *x0 = 0,3* (м);
    - величина скорости *v0 = 2* (м/с);
  - для типового примера Д-23 ([23, с. 313]):
    - значение координаты *y0 = 0,2·10⁻²* (м);
    - величина скорости *v0 = 8 ·10⁻²* (м/с).

Результаты решения дифференциальных уравнений:

- (11.7) после выделения строки #37 на рис. 11.3 и нажатия кнопки:
  - = (*Simplify*) — записывается в строке #38;
  - ≈ (*Approximate*) — #39 рис. 11.3;
- (11.8) после выделения строки #44 на рис. 11.4 и нажатия кнопки:
  - = (*Simplify*) — записывается в строке #45;
  - ≈ (*Approximate*) — #46 рис. 11.4.

Обратите внимание, что результаты решения в строках:

- #38 и #39 на рис. 11.3 получены при установленном в строке #31 направлении преобразования *Expand* (*расширение*), поэтому они представлены в виде суммы синуса и косинуса одного аргумента;
- #45 и #46 на рис. 11.4 получены при установленном в строке #40 направлении преобразования *Collect* (*сбор*), поэтому они представлены в виде синуса суммы в амплитудной форме с начальной фазой колебаний $\beta = 0{,}595224$ *рад* (строка #46).

Представим оба решения в других формах, для чего выдадим команду Declare → Algebra State → Simplification... и в появившемся диалоговом окне Simplification Options (рис. 13.6) в поле Trigonometry установим направление преобразования:

- на рис. 11.3 *Collect — сбор*. После нажатия кнопки OK сообщение об установке нового режима появится в строке #40 рис. 11.3;
- на рис. 11.4 *Expand — расширение*. После нажатия кнопки OK сообщение об установке нового режима появится в строке #47 рис. 11.4.

Теперь выделим соответствующее обращение к функции *DSOLVE2_IV*:

- #37 на рис. 11.3 и нажмем кнопку ≈ (*Approximate*) — соответствующее решение дифференциального уравнения (11.7) в амплитудной форме будет записано в строке #41;
- #44 на рис. 11.4 и нажмем кнопку:
  - = *Simplify* — соответствующее решение дифференциального уравнения (11.8) в форме суммы синуса и косинуса одного аргумента будет записано в строке #48 в виде рациональных чисел;
  - ≈ (*Approximate*) — решение будет записано в строке #49 в виде вещественных чисел.

Графики результатов решения представляют собой простые синусоидальные кривые с начальной фазой колебаний:

- со смещением вверх по оси x на величину 0,128072 (м) от положения статического равновесия, относительно которого происходит колебательное движение с амплитудой 0,26565 (м) — кривая D_4 на рис. 11.3;
- без смещения от положения статического равновесия, относительно которого происходит колебательное движение с амплитудой 0,003567 (м) — график на рис. 11.4.

11.3. Двумерные (плоские) и трехмерные (пространственные) задачи

Интегрирование двумерных и трехмерных задач рассмотрим на примерах решения плоских и пространственных задач динамики материальной точки. Выберем для этой цели в качестве:

- *плоской* задачи — типовой пример задания Д-2 [23, с. 130, 136-138], дифференциальные уравнения движения которого с учетом силы сопротивления имеют вид (см. уравнение (9) [23, с. 137]):

$$\ddot{x} + 2\dot{x} + 4x = 0, \quad \ddot{z} + 2\dot{z} + 4z = -9,81 \qquad (11.9)$$

с начальными условиями при t=0:

  - для координат точки $x_0 = 0$ и $z_0 = 10$ (м);
  - для проекций скорости точки $\dot{x}_0 = 20$ и $\dot{z}_0 = 0$ (м/с);

- *пространственной* задачи — пример 28-го варианта задания Д-2 [23, с. 133, 135], дифференциальные уравнения движения которого после деления на массу точки имеют вид [23, с. 135]:

$$\ddot{x} + 0,5\dot{x} = 0, \quad \ddot{y} + 0,5\dot{y} = 0, \ddot{z} + 0,5\dot{z} = 9,81 \qquad (11.10)$$

с начальными условиями при t=0:

  - для координат точки $x_0 = 0$, $y_0 = 0$ и $z_0 = 0$ (м);
  - для проекций скорости точки $\dot{x}_0 = 1, \dot{y}_0 = 1$ и $\dot{z}_0 = 2$ (м/с).

Системы уравнений (11.9) и (11.10) состоят соответственно из двух и трех независимых друг от друга дифференциальных уравнений, в которых роль независимой переменной выполняет время *t*.

В подобных случаях DERIVE легко справляется с такими задачами, что достигается просто двукратным или трехкратным обращением к функции *DSOLVE2_IV* с заданием нужных фактических параметров. Соответствующая сессия работы представлена на рис. 11.5.

Строки #50 и #58 на рис. 11.5 представляют собой введенный нами краткий комментарий с указанием типа решаемой ниже в DERIVE задачи:

- *dvumernaia_zadacha* — плоская;
- *trexmernaia_zadacha* — пространственная.

Обратим ваше внимание, что из-за многочисленных затруднений, связанных с использованием в DERIVE букв русского алфавита, все комментарии записаны латинскими буквами в русской транскрипции.

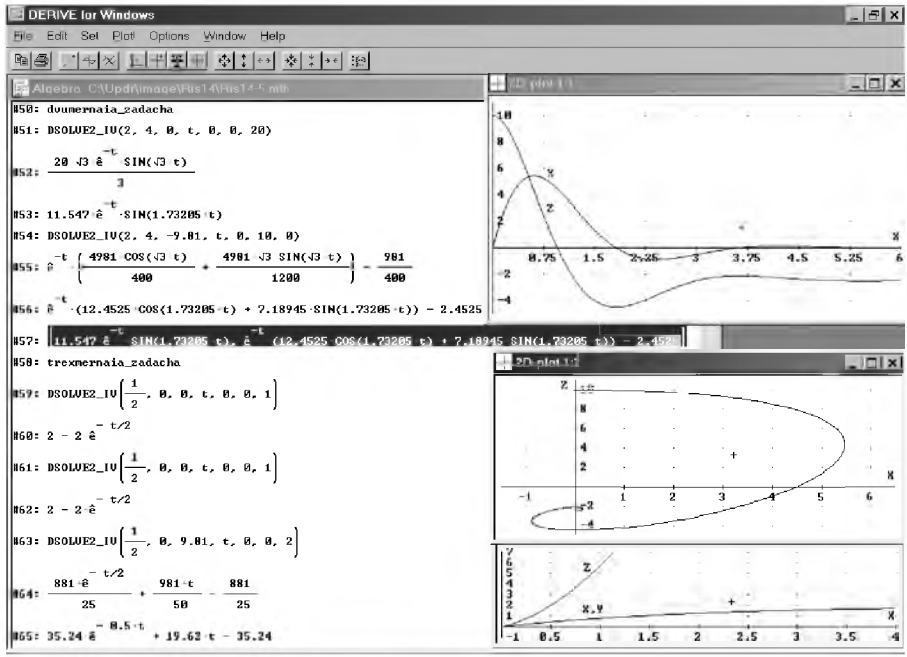

Рис. 11.5. Интегрирование дифференциальных уравнений движения материальной точки, находящейся под действием переменных сил, для типового примера задания Д-2 ([23, с. 130, 136-138]) и его 28-го варианта ([23, с. 135]) с графическим представлением результатов

Интегрирование подобных двумерной (11.9) и трехмерной задачи (11.10), как указывалось ранее, достигается просто двукратным или трехкратным обращением к функции *DSOLVE2_IV (p, q, r, t, t0, x0, v0)* с заданием нужных фактических параметров:

- для *плоской* задачи для первого и второго уравнения из (11.9) соответственно в строках:
  - #51 — *DSOLVE2_IV(2, 4, 0, t, 0, 0, 20)*;
  - #54 — *DSOLVE2_IV(2, 4, -9.81, t, 0, 10, 0)*,

где используемые при обращениях фактические параметры имеют следующие значения:

- функция *p* перед первой производной по *x* из общего вида дифференциального уравнения (11.4) имеет в строках #51 и #54 численное значение *2*, одинаковое для обоих уравнений (11.9);

- функция *q* перед координатой *x* из (11.4) имеет в строках #51 и #54 численное значение *4*, одинаковое для обоих уравнений (11.9);
- функция *r(t)* в правой части общего вида дифференциального уравнения (11.4) в первом из уравнений (11.9) равняется нулю, а во втором является постоянной и равняется *–9,81*;
- на четвертом и пятом местах в списке фактических параметров указываются переменная интегрирования *t* и ее начальное значение *t0*, равное нулю;
- на шестом и седьмом местах в списке фактических параметров указываются начальные условия при *t=0*:
  - для *первого* из уравнений (11.9) в строке #51:
    - значение координаты *x0 = 0* (м);
    - величина скорости *v0 = 20* (м/с);
  - для *второго* из уравнений (11.9) в строке #54:
    - значение координаты *z0 = 10* (м);
    - величина скорости *v0 = 0* (м/с);

- для *пространственной* задачи для первого, второго и третьего уравнения из (11.10) соответственно в строках:
  - #59 — *DSOLVE2_IV(1/2, 0, 0, t, 0, 0, 1)*;
  - #61 — *DSOLVE2_IV(1/2, 0, 0, t, 0, 0, 1)*;
  - #63 — *DSOLVE2_IV(1/2, 0, 9.81, t, 0, 0, 2)*,

где используемые при обращениях фактические параметры имеют следующие значения:

- функция *p* перед первой производной по *x* из общего вида дифференциального уравнения (11.4) имеет в строках #59, #61 и #63 численное значение *1/2*, одинаковое для трех уравнений (11.10);
- функция *q* перед координатой *x* из (11.4) имеет в строках #59, #61 и #63 численное значение *0*, одинаковое для трех уравнений (11.10);
- функция *r(t)* в правой части общего вида дифференциального уравнения (11.4):
  - в первом и втором из уравнений (11.10) равняется нулю;
  - в третьем является постоянной и равняется *9,81*;
- на четвертом и пятом местах в списке фактических параметров указываются переменная интегрирования *t* и ее начальное значение *t0*, равное нулю;

- на шестом и седьмом местах в списке фактических параметров указываются начальные условия при *t=0*:
  - для *первого* и *второго* из уравнений (11.10) в строках #59 и #61:
    - значение координат *x0 = 0* и *y0 = 0* (м);
    - величина скорости *v0 = 1* (м/с);
  - для *третьего* из уравнений (11.10) в строке #63:
    - значение координаты *z0 = 0* (м);
    - величина скорости *v0 = 2* (м/с).

Ниже каждого обращения к функции *DSOLVE2_IV* на рис. 11.5 располагаются соответствующие результаты решения дифференциальных уравнений.

Они появляются в алгебраическом окне выражений после выделения соответствующей функции *DSOLVE2_IV* на рис. 11.5 и нажатия:

- кнопки = (*Simplify*) — результаты решения с использованием рациональных чисел записываются в строках #52, #55, #60, #62 и #64;
- кнопки ≈ (*Approximate*) — результаты решения с использованием вещественных чисел записываются в строках #53, #56 и #65.

Первые два уравнения системы (11.10) имеют практически одинаковую форму представления результатов решения #60 и #62 после нажатия кнопок = Simplify и ≈ (Approximate), поэтому они повторно не приводятся.

В графических окнах рис. 11.5 находятся графики результатов решения соответствующих дифференциальных уравнений:

- *плоской* задачи (11.9) — правое верхнее окно:
  - кривая *X* соответствует уравнению в формах записи #52 или #53 и представляет собой быстро затухающее колебательное движение с положительной начальной скоростью из нулевого положения, асимптотически приближающееся к оси абсцисс *t*;
  - кривая *Z* соответствует уравнению в формах записи #55 или #56 и представляет собой так же быстро затухающее колебательное движение без начальной скорости из начального положения с координатой x_0 = 10 (м), асимптотически приближающееся к прямой, параллельной оси абсцисс *t* и смещенной вниз по оси ординат на расстояние 2,4525 (м);

- *пространственной* задачи (11.10) — правое нижнее окно:
  - совпадающие кривые для *X* и *Y* соответствуют уравнениям в формах записи #60 или #62 и представляют собой одну и ту же экспоненциальную кривую, асимптотически приближающуюся к соответствующей горизонтальной прямой $x = 2$ и $y = 2$ (м);
  - кривая *Z* соответствует уравнению в формах записи #64 или #65 и представляет собой экспоненциальную кривую, асимптотически приближающуюся к наклонной прямой, определенной уравнением $19{,}62\,t - 35{,}24$ (м).

В среднем графическом окне представлен график движения материальной точки в *плоской* задаче (11.9) в координатах *X* и *Z*. Для этого соответствующие параметрические функции для *X* #53 и *Z* #56 представлены, разделенными запятой, в строке #57 в виде вектора, заключенного в квадратные скобки.

Выделим строку #57 и, перейдя в новое графическое окно, предварительно открытое командой Window → New 2D-plot Window, нажмем кнопку  (*Plot expression*).

В появившемся диалоговом окне Parametric Plot Parameters зададим минимальное (*0*) и максимальное (*8*) значение параметра *t* параметрических функций *X* и *Z*, после чего нажмем кнопку OK. Теперь в новом графическом окне будет построен нужный график. После его небольшого редактирования видно, что он представляет собой кривую в виде расширяющейся спирали, навивающуюся на точку на оси ординат с координатой $z_0 = -2{,}45$ (м).

Для более сложных случаев, когда дифференциальные уравнения в двумерных и трехмерных задачах зависят друг от друга, требуется их представление в виде систем дифференциальных уравнений первого порядка с последующим их численным интегрированием.

Система DERIVE имеет для этих случаев соответствующий файл *Ode_appr.mth*. Он определяет функции для решения обыкновенных дифференциальных уравнений первого порядка и их систем, работа которых основана на свойствах рядов и приближенных численных методах, что рассмотрено в п. 11. 5.

Для численного интегрирования дифференциальных уравнений высших порядков их сначала следует представить в виде системы дифференциальных уравнений первого порядка, чему посвящен следующий раздел 11.4.

Однако, следует отметить, что для численного интегрирования дифференциальных уравнений гораздо лучше использовать алгоритмические языки высокого уровня, например, Фортран, на котором накоплено огромное программное обеспечение. Его использование для всех представленных в настоящем пособии примеров рассмотрено в работе [17].

11.4. Представление дифференциальных уравнений высших порядков в виде системы дифференциальных уравнений первого порядка

11.4.1. Замена дифференциального уравнения *N*–го порядка системой из *N* дифференциальных уравнений первого порядка

При интегрировании одного дифференциального уравнения его нужно представить в виде, в котором высшая производная (например, вторая) выделена и находится в левой части дифференциального уравнения:

$$\ddot{x} = f\ (t,\ x, \dot{x}\ ). \qquad (11.11)$$

Введем обозначение: $\dot{x} = xp$, тогда $\ddot{x} = \dot{x}p$ и уравнение (11.11) можно переписать в виде системы двух дифференциальных уравнений первого порядка:

$$\dot{x} = xp, \qquad (11.12)$$

$$\dot{x}p = f\ (t,\ x,\ xp). \qquad (11.13)$$

Таким образом, мы выполнили поставленную задачу и представили дифференциальное уравнение второго порядка (11.11) в виде системы из двух дифференциальных уравнений первого порядка (11.12)–(11.13).

Отметим, что оператор (11.12) будет одинаковым для всех линейных (одномерных) задач, ибо он только присваивает производной введенное для нее обозначение.

При записи правой части оператора (11.13) не следует обращать внимание на его довольно громоздкое выражение в общем виде, а просто записать конкретную для решаемой задачи соответствующую правую часть интегрируемого уравнения (11.11) по правилам DERIVE с учетом принятых обозначений.

Рассмотрим также подготовку для численного интегрирования одного дифференциального уравнения третьего порядка, представив его сначала в виде, в котором третья производная выделена и находится в левой части дифференциального уравнения:

$$\dddot{x} = f\ (t,\ x, \dot{x}, \ddot{x}). \qquad (11.14)$$

Введем обозначения $\dot{x} = xp$ и $\ddot{x} = \dot{x}p = xpp$, тогда $\dddot{x} = \dot{x}pp$ и уравнение (11.14) можно переписать в виде системы трех дифференциальных уравнений первого порядка:

$$\dot{x} = xp, \qquad (11.15)$$

$$\dot{x}p = xpp, \qquad (11.16)$$

$$\dot{x}pp = f\ (t,\ x,\ xp,\ xpp). \qquad (11.17)$$

Таким образом представляется дифференциальное уравнение третьего порядка (11.14) в виде системы из трех дифференциальных уравнений первого порядка (11.15)–(11.17).

Действуя аналогичным образом, из каждого дифференциального уравнения N-го порядка получится в общем случае система из N дифференциальных уравнений первого порядка.

11.4.2. Замена систем двух или трех дифференциальных уравнений второго порядка системами дифференциальных уравнений первого порядка

При численном интегрировании *двумерных* (плоских) задач динамики часто нужно заменить систему двух дифференциальных уравнений второго порядка системой дифференциальных уравнений первого порядка.

В этом случае оба интегрируемых уравнения сначала следует представить в форме с выделенной в левой части второй производной, что в общем виде можно записать так:

$$\ddot{x} = f1\ (t,\ x,\ y, \dot{x}, \dot{y}), \qquad (11.18)$$

$$\ddot{y} = f2\ (t,\ x,\ y, \dot{x}, \dot{y}). \qquad (11.19)$$

Вводим обозначения: $\dot{x} = xp$ и $\dot{y} = yp$, с использованием которых левые части уравнений (11.18)–(11.19) примут вид: $\ddot{x} = \dot{x}p$, $\ddot{y} = \dot{y}p$.

Теперь уравнения (11.18)–(11.19) можно переписать в виде системы из четырех дифференциальных уравнений 1-го порядка:

$$\dot{x} = xp, \quad (11.20)$$

$$\dot{xp} = f1\ (t,\ x,\ y,\ xp,\ yp), \quad (11.21)$$

$$\dot{y} = yp, \quad (11.22)$$

$$\dot{yp} = f2\ (t,\ x,\ y,\ xp,\ yp). \quad (11.23)$$

Таким образом, мы выполнили поставленную задачу и представили систему из двух дифференциальных уравнений второго порядка (11.18)–(11.19) в виде системы из четырех дифференциальных уравнений первого порядка (11.20)–(11.23).

Обратим ваше внимание, что:

- операторы (11.20) и (11.22) будут одинаковыми для всех плоских (двумерных) задач, ибо они только присваивают соответствующим производным введенные для них обозначения;
- последовательность, то есть алгоритм производимых действий при замене двух дифференциальных уравнений полностью совпадает с рассмотренным выше случаем замены одного уравнения. Он сохраняется и для случая трех уравнений.

При численном интегрировании *трехмерных* (пространственных) задач динамики часто нужно заменить систему трех дифференциальных уравнений второго порядка системой дифференциальных уравнений первого порядка.

Для этого также представляем интегрируемые уравнения в виде с выделенной в левой части второй производной:

$$\ddot{x} = f1\ (t,\ x,\ y,\ z, \dot{x}, \dot{y}, \dot{z}), \quad (11.24)$$

$$\ddot{y} = f2\ (t,\ x,\ y,\ z, \dot{x}, \dot{y}, \dot{z}), \quad (11.25)$$

$$\ddot{z} = f3\ (t,\ x,\ y,\ z, \dot{x}, \dot{y}, \dot{z}). \quad (11.26)$$

Вводим обозначения: $\dot{x} = xp$, $\dot{y} = yp$, и $\dot{z} = zp$ с использованием которых левые части уравнений (11.24)–(11.26) примут вид: $\ddot{x} = \dot{xp}$, $\ddot{y} = \dot{yp}$ и $\ddot{z} = \dot{zp}$.

Теперь уравнения (11.24)–(11.26) можно переписать в виде системы из шести дифференциальных уравнений 1-го порядка:

$$\dot{x} = xp, \qquad (11.27)$$

$$\dot{xp} = f1\ (t,\ x,\ y,\ z,\ xp,\ yp,\ zp), \qquad (11.28)$$

$$\dot{y} = yp, \qquad (11.29)$$

$$\dot{yp} = f2\ (t,\ x,\ y,\ z,\ xp,\ yp,\ zp), \qquad (11.30)$$

$$\dot{z} = zp, \qquad (11.31)$$

$$\dot{zp} = f3\ (t,\ x,\ y,\ z,\ xp,\ yp,\ zp). \qquad (11.32)$$

Таким образом, мы выполнили поставленную задачу и представили систему из трех дифференциальных уравнений второго порядка (11.24)–(11.26) в виде системы из шести дифференциальных уравнений первого порядка (11.27)–(11.32).

Обратим ваше внимание, что операторы (11.27), (11.29) и (11.31) также будут одинаковыми для всех пространственных (трехмерных) задач, ибо они только присваивают соответствующим производным введенные для них обозначения.

При записи правых частей операторов (11.28), (11.30) и (11.32) также не следует обращать внимание на их довольно громоздкие выражения в общем виде, а просто записать конкретные для решаемой задачи соответствующие правые части интегрируемых уравнений (11.24)–(11.26) по правилам DERIVE с учетом принятых обозначений.

11.5. Численное интегрирование дифференциальных уравнений в DERIVE

11.5.1. Интегрирование уравнений вида y' = r (x) с начальным условием y (x0) = y0

Файл *Ode_appr.mth* определяет функции для решения обыкновенных дифференциальных уравнений и их систем, работа которых основана на свойствах рядов и приближенных численных методах.

Чтобы использовать функции, описанные ниже, необходимо:

- сначала загрузить файл Ode_appr.mth, расположенный по адресу *C:\Program Files\Derive\Math\ Ode_appr.mth*;
- записать его под вашим рабочим именем.

Для этого нужно:

- запустить DERIVE;
- вызвать на выполнение файл *Ode_appr.mth* (рис. 11.6–11.7);
- записать его под вашим рабочим именем. Это можно сделать командой File → Save As... из главного меню, набрав в появившемся диалоговом окне нужное имя в поле *Имя файла* (рис. 10.10), а также указав нужную папку в раскрывающемся списке *Папка* (рис. 10.11);
- начать выполнять нужное вам интегрирование в этом рабочем файле.

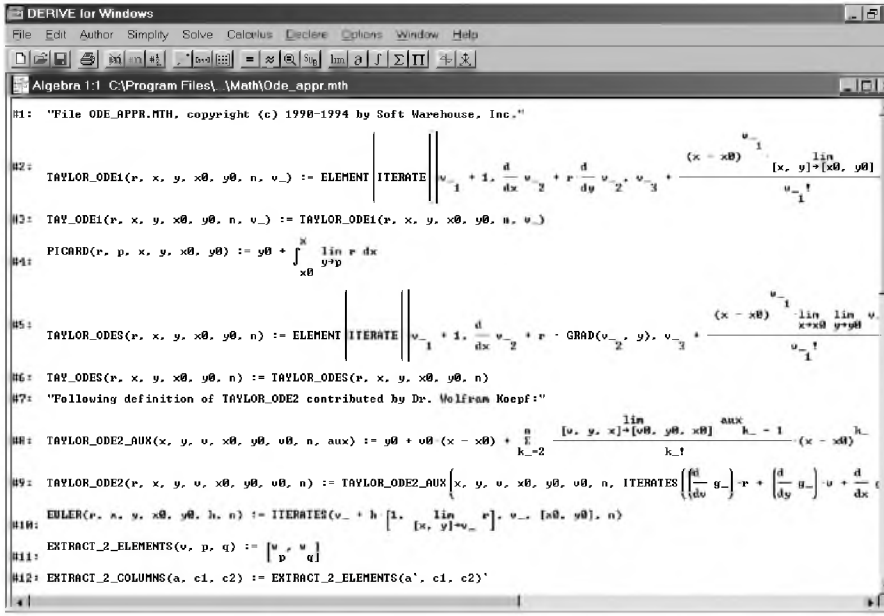

Рис. 11.6. Функции файла Ode_appr.mth *для приближенного решения нелинейных дифференциальных уравнений первого порядка и их систем*

Файл Ode_appr.mth содержит дополнительные расширения для решения одиночных дифференциальных уравнений вида y' = r (x) при y(x0)=y0. В нем заданы функции, представленные ниже.

Функция TAY_ODE1(r, x, y, x0, y0, n) определена в строках #2–#3 рис. 11.6. Она получает решение в виде ряда Тейлора n-го порядка, определяемого уравнением y' = r(x, y) с начальными условиями y = y0 и x = x0.

TAY_ODE1 возвращает знак вопроса "?" в том случае, если функция r недостаточное число раз дифференцируема для соответствующего ряда n-го порядка. В этой ситуации следует уменьшить порядок n, если это возможно.

Функция PICARD(r, p, x, y, x0, y0) описана в строке #4 рис. 11.6. Она получает приближенное решение в виде ряда, определяемого уравнением y' = r(x, y), учитывая его приближенное решение p(x). В результате происходит уточнение получающегося приближения с учетом заданного приближения p(x).

Этот метод использует объединенную функцию r(x,p(x)). Если в упрощенном результате не присутствует ни один из интегралов, можно осуществить следующее повторение, используя улучшенное приближение для p, и так далее. Если первое приближение является наилучшим, то желательно использовать постоянную y0.

Функция PICARD может также использоваться для нахождения решения в виде рядов, определяемых системами дифференциальных уравнений первого порядка. В этом случае она дает улучшение аппроксимации в виде вектора p(x).

На основе использования этих функций в файле Ode_appr.mth определены также и другие функции для решения систем дифференциальных уравнений первого порядка, заданных в виде векторов y' = r(x) и вектора начальных условий y0 = y(x0).

Функция TAY_ODES(r, x, y, x0, y0, n) описана в строках #5–#6 рис. 11.6. Она получает решение в виде рядов Тейлора n-й степени, если вектор r достаточное число раз дифференцируем.

Функция TAYLOR_ODE2 (r, x, y, v, x0, y0, v0, n) определена в строках #8–#9 рис. 11.6. Она получает решение в виде рядов Тейлора, определяемого уравнением y'' = r(x, y, v). В данном случае v представляет собой y', а v0 — v при x = x0.

TAYLOR_ODE2 возвращает знак вопроса "?" в том случае, если функция r недостаточное число раз дифференцируема для соответствующего ряда n-го порядка. В этой ситуации следует уменьшить порядок n, если это возможно.

Функция EULER(r, x, y, x0, y0, h, n) описана в строке #10 рис. 11.6. Она приближает к вектору n+1 решений уравнения y' = r(x, y) с y = y0 при x=x0, начиная с x=x0, изменяясь на шаг h. В результате получается матрица

[[x0, y0], ..., [xn,yn]] решения методом Эйлера при шаге h (используется команда APPROX).

Метод Эйлера имеет только образовательную ценность как самый простой числовой метод для решения обыкновенных дифференциальных уравнений. Он имеет погрешность решения, больше чем у любого стандартного метода.

Для серьезных вычислений должен использоваться более точный метод Рунге – Кутта, рассматриваемый в п. 11.5.2.

Функция EXTRACT_2_COLUMNS(A, j, k) определена в строках #11–#12 рис. 11.6. Она выделяет столбцы j,k матрицы *A* для графического представления результатов решения методом Рунге–Кутта.

Функция DIRECTION_FIELD(r, x, x0, xm, m, y, y0, yn, n) описана в строках #19–#21 рис. 11.7. Она получает матрицу, определяемую двумя составляющими векторов для уравнения y' =r(x, y), в котором x изменяется от x0 до xm с шагом m, а y — от y0 до yn с шагом n. В результате строится сетка с координатами (x0, y0) и (xm, yn) противолежащих углов (при этом используются команды APPROX и Plot).

11.5.2. Интегрирование систем нелинейных дифференциальных уравнений методом Рунге-Кутта

Функция RK(r, v, v0, h, n) определена в строках #13–#17 рис. 11.7. Она получает решение системы дифференциальных уравнений первого порядка yi' = ri(x, y), где ri — вектор [r1, r2..., rm], v — вектор [x, y1, y2..., ym], v0 — соответствующий вектор начальных условий, h — величина шага, n — число шагов.

Функция RK получает последовательно приближенные решения методом Рунге–Кутта в виде матрицы [[x0, y0(x0), ym(x0)], ..., [xn, y0(xn), ym(xn)]] с шагом решения h.

Она является важнейшей функцией файла Ode_appr.mth и является наглядным примером функционального программирования, когда конечная функция определена через ряд промежуточных функций.

Подробнее рассмотрим использование формальных параметров функции RK(r, v, v0, h, n) применительно к задачам динамики, где обычно используются производные по времени t, а дифференциальные уравнения обычно бывают второго порядка.

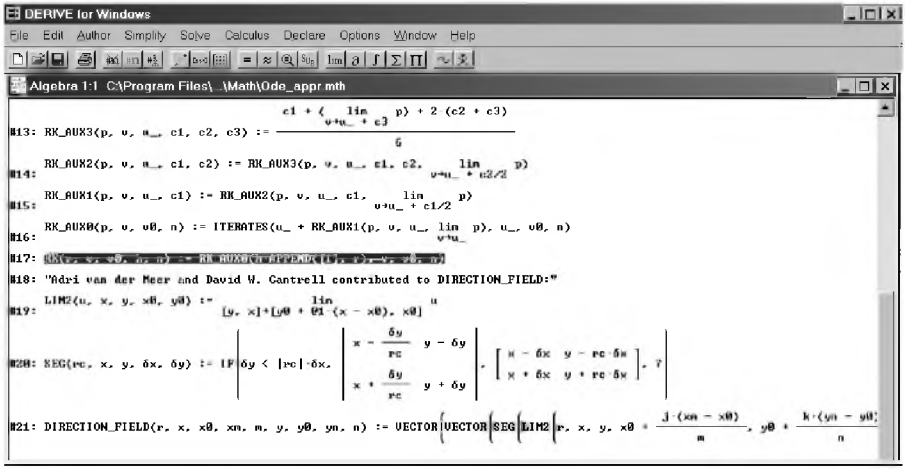

Рис. 11.7. Продолжение файла ***Ode_appr.mth*** *с описанием функций для приближенного решения систем нелинейных дифференциальных уравнений первого порядка*

Чтобы воспользоваться возможностью DERIVE по численному интегрированию дифференциальных уравнений или их систем, их нужно представить сначала в виде системы дифференциальных уравнений первого порядка, что подробно описано в п. 11.4. Напомним кратко алгоритм выполняемых действий.

Для одного дифференциального уравнения второго порядка (11.11) сначала вводится подстановочная переменная, например, *xp*, которой присваивается значение первой производной от соответствующей координаты по времени *t*:

$$\dot{x} = xp. \qquad (11.33)$$

С учетом уравнения (11.33), вторая производная от соответствующей координаты по времени *t* равняется первой производной от используемой подстановочной переменной *xp*:

$$\ddot{x} = \dot{xp}\ . \qquad (11.34)$$

Теперь, с учетом значения второй производной (11.11) и с использованием введенной подстановочной переменной *xp* (11.33), будем иметь:

$$\dot{xp} = f1(t,\ x,\ xp). \qquad (11.35)$$

В результате одно дифференциальное уравнение второго порядка (11.11) будет представлено в виде системы из двух дифференциальных уравнений первого порядка (11.33) и (11.35) (конечно, тождественно совпадающих с системой уравнений (11.12)–(11.13)).

Замена системы из двух (двумерная) или трех дифференциальных уравнений второго порядка (трехмерная задача) выполняется аналогичным образом. Для этого описанный алгоритм замены и введения подстановочной переменной для координаты *x* и ее производных следует повторить для всех используемых координат:

- *x, y* и их производных — *двумерная* задача. В результате два дифференциальных уравнения второго порядка (11.18)–(11.19) будут представлены в виде системы из четырех дифференциальных уравнений первого порядка (11.20)–(11.23);
- *x, y, z* и их производных — *трехмерная* задача. В результате три дифференциальных уравнения второго порядка (11.24)–(11.26) будут представлены в виде системы из шести дифференциальных уравнений первого порядка (11.27)–(11.32).

В функции RK(r, v, v0, h, n) первый формальный параметр r — вектор правых частей системы дифференциальных уравнений первого порядка. Поэтому вектор r при обращении к функции RK может быть представлен для различных задач в нижеследующих общих формах записи:

- *линейная* (одномерная) задача — правые части уравнений (11.12)–(11.13) (или (11.33) и (11.35)):

```
[xp, f (t, x, xp)]                                          (11.36)
```

- *плоская* (двумерная) задача — правые части уравнений (11.20)–(11.23):

```
[xp, f1 (t, x, y, xp, yp), yp, f2 (t, x, y, xp, yp)]        (11.37)
```

- *пространственная* (трехмерная) задача — правые части уравнений (11.27)–(11.32):

```
[xp,f1(t,x,y,z,xp,yp,zp),yp,f2(t,x,y,z,xp,yp,zp),zp,f3(t,x,y,z,xp,yp,zp)](11.38)
```

Напомним, что при записи функций *f, f1, f2 , f3* в выражениях (11.36)–(11.38) не следует обращать внимание на их довольно громоздкие выражения в общем виде. Нужно просто записать конкретные для решаемой задачи соответствующие правые части интегрируемых уравнений по правилам DERIVE с учетом принятых обозначений.

Второй и третий формальные параметры функции *RK(r, v, v0, h, n)* также являются векторами и должны содержать:

- *вектор v* — перечисление всех переменных, что может быть представлено для различных задач в нижеследующих общих формах записи:
  - *линейная* (одномерная) задача:

    [t, x, xp] (11.39)

  - *плоская* (двумерная) задача:

    [t, x, xp, y, yp] (11.40)

  - *пространственная* (трехмерная) задача:

    [t, x, xp, y, yp, z, zp] (11.41)

- *вектор v0* — перечисление всех начальных значений используемых переменных, которые ниже обозначаются символом 0 (нуля) после идентификатора соответствующей переменной в нижеследующих общих формах записи:
  - *линейная* (одномерная) задача:

    [t0, x0, xp0] (11.42)

  - *плоская* (двумерная) задача:

    [t0, x0, xp0, y0, yp0] (11.43)

  - *пространственная* (трехмерная) задача:

    [t0, x0, xp0, y0, yp0, z0, zp0] (11.44)

Четвертым и пятым *формальным* параметром функции RK(r, v, v0, h, n) являются:

- шаг решения по переменной *t*, который задается значением *h*;
- число точек решения — задается значением *n*.

Результаты численного решения представляются в виде матрицы, столбцы которой дают значения переменных:

- *линейная* (одномерная) задача — три столбца, соответствующие переменным:

  t, x, xp; (11.45)

- *плоская* (двумерная) задача — пять столбцов:

  t, x, xp, y, yp; (11.46)

- *пространственная* (трехмерная) задача — семь столбцов, соответствующих переменным:

  t, x, xp, y, yp, z, zp. (11.47)

Напомним, что обозначения аргумента *t*, переменных *x*, *y*, *z*, производных *xp*, *yp*, *zp* и их начальных условий, используемых нами при описании формальных параметров функции *RK(r, v, v0, h, n)*, также являются формальными параметрами. Поэтому они могут быть обозначены любыми символическими именами или сразу их численными значениями. В первом случае на их место при выполнении функции *RK* будут поставлены значения фактических параметров, определенные на данный момент выполнения функции в текущем сеансе работы в DERIVE, с которыми и будут произведены все действия.

Однако обратите внимание на строгость обязательного совпадения *порядка следования* используемых фактических параметров соответствующим формальным параметрам. Они должны указываться:

- при задании вектора *r* (правых частей системы дифференциальных уравнений первого порядка) — в зависимости от вида задачи в порядке, определенном формами записи (11.36)–(11.38);
- перечисление всех переменных в векторе *v* и их начальных условий в векторе *v0* также должно строго соответствовать:
  - в векторе *v* — в зависимости от вида задачи в порядке, определенном формами записи (11.39)–(11.41);
  - в векторе *v0* — в порядке, определенном формами записи (11.42)–(11.44).

При нарушении этого соответствия вы также получите итоговые матрицы чисел (11.45)–(11.47), столбцы которой дают значения переменных. По внешнему виду они будут очень похожи на правильное решение, однако значения этих чисел не будут соответствовать решаемой задачи. Поэтому удовольствие от такого вождения самого себя за нос прекратится при первом же соприкосновении с реальностью. Это будет сопровождаться большими неприятностями для вас, а также, к сожалению, для окружающих.

Рассмотрим решение с использованием функции *RK* типового примера задания Д-27 [23, с. 352–361] по исследованию колебательного движения механической системы, которое рекомендовано в [23] для выполнения на ЭВМ. Его нелинейное дифференциальное уравнение записано там же под номером (12) и имеет вид:

$$11.6\ \ddot{x} + (78 + 1944\ x^2)\ x\ /\ \mathrm{ABS}(x) + 684\ x + 3888\ x^3 = 0. \quad (11.48)$$

Начальные условия при времени *t=0* имеют следующие значения:

$$x_0 = 0{,}5 \text{ рад}, \quad \dot{x}_0 = 0. \quad (11.49)$$

Сначала запишем уравнение (11.48) в форме с выделенной в левой части второй производной. При этом также используем вспомогательные переменные:

$$\ddot{x} = -(b\_27 + c\_27\ x^2)\ x / ABS(x) - d\_27\ x - e\_27\ x^3, \quad (11.50)$$

где *a_27=11.6, b_27=78/a_27, c_27=1944/a_27, d_27=684/a_27, e_27=3888/a_27.* (11.51)

Теперь представим уравнение (11.50) в виде системы дифференциальных уравнений первого порядка (см. п. 11.4.1):

$$\dot{x} = xp, \quad (11.52)$$

$$\dot{xp} = -(b\_27 + c\_27\ x^2)\ x / ABS(x) - d\_27\ x - e\_27\ x^3. \quad (11.53)$$

Соответствующая сессия работы по численному интегрированию системы дифференциальных уравнений первого порядка (11.52)–(11.53) представлена на рис. 11.8. Она является продолжением работы в файле *C:\Program Files\Derive\Math\Ode_appr.mth*, состоящему из строк #1–#21 и представленному на рис. 11.6–11.7. После открытия файла *Ode_appr.mth* ему присвоено рабочее имя, например, *Ris14-8.mth* (в вашей рабочей папке).

В строках #22–#26 рис. 11.8 присваиваются значения переменным (11.51), которые используются при задании системы дифференциальных уравнений первого порядка (11.52)–(11.53) рассматриваемого типового примера в функции *RK*.

Обратите внимание, что используемые переменные выбраны индивидуальным образом для рассматриваемой задачи (везде после букв *a–e* используется символ *_27*). Это сделано для того, чтобы с надежностью исключить возможность совпадения вводимых в строках #22–#26 переменных с используемыми ранее в строках #2–#21 файла *Ode_appr.mth*.

Вы лучше поймете важность сделанного замечания, если после повторения представленного на рис. 11.8 сеанса работы, вы попробуете его еще раз повторить, убрав везде в строках #22–#27 используемый для отличия символ *_27*.

Значения самих переменных в строках #22–#26 задаются для простоты в численном виде, определенном ранее в пособии [23, с. 352–361].

Отметим, что при самостоятельном получении сложного дифференциального уравнения собственных вычислений лучше не проводить. Проще в правые части операторов присваивания, определяющих используемые переменные при выводе дифференциального уравнения, подставить их выражения в символьном виде из данных условия. Конечно,

сами значения данных условия в этом случае должны быть предварительно заданы в рассматриваемой сессии работы.

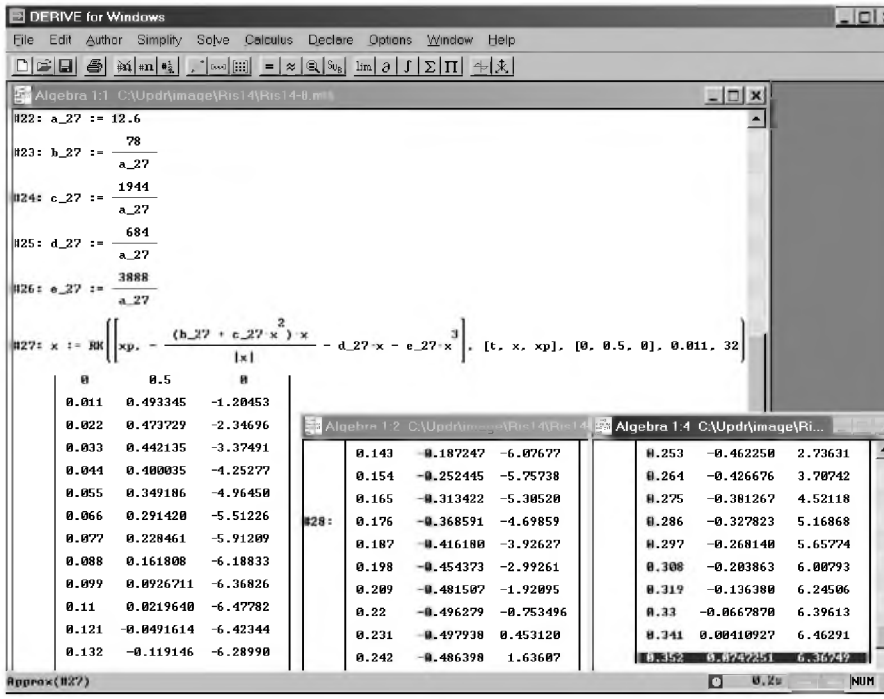

Рис. 11.8. Численное интегрирование нелинейного дифференциального уравнения второго порядка (11.48) для типового примера колебаний механической системы с одной степенью свободы Д-27 ([23, с. 352-361]) с представлением численных результатов

В строке #27 рис. 11.8 происходит обращение к функции *RK(r, v, v0, h, n)* со следующими фактическими параметрами:

- в качестве вектора *r* указывается вектор правых частей системы дифференциальных уравнений первого порядка для решаемого типового примера, представляющий собой правые части системы (11.52)–(11.53):

```
[xp, - (b_27 + c_27 x^2) x / ABS(x) - d_27 x - e_27 x^3]; (11.54)
```

- вторым фактическим параметром (в качестве вектора *v*) указывается вектор, содержащий перечисление используемых переменных в интег-

рируемой системе дифференциальных уравнений первого порядка (11.52)–(11.53): времени *t*, координаты *x* и скорости *xp*. Нами для удобства пользования фактические параметры выбраны совпадающими с формальными. В этом случае для линейной (одномерной) задачи форма записи вектора *v* совпадает с (11.39):

```
[t, x, xp];                                    (11.55)
```

- третьим фактическим параметром (в качестве вектора *v0*) указывается вектор, содержащий начальные значения используемых переменных (11.49) при *t = 0*, в порядке, определенном в перечислении (11.55):

```
[0, 0.5, 0];                                   (11.56)
```

- шаг приращения *h* переменной *t*, при различных значениях которой будут определяться численные значения координаты *x* и скорости *xp* интегрируемого дифференциального уравнения, задан *0,011* с. Это сделано для возможности сравнения результатов с приведенным численным решением в таблице 66 [23, с. 361];
- пятым фактическим параметром функции *RK(r, v, v0, h, n)* является: число точек решения *n*, которое задано своим значением (*32*).

После нажатия кнопки ≈ (Approximate) (при выделенной строке #27), результаты численного решения записываются в строке #28 рис. 11.8. Они представляются в виде матрицы, три столбца которой дают значения переменных: времени *t*, координаты *x* и скорости *xp*. Из сравнения результатов с приведенным численным решением в таблице 66 [23, с. 361] можно сделать вывод о их совпадении (с точностью до первых четырех значащих цифр).

11.5.3. Численное интегрирование двумерных и трехмерных задач динамики методом Рунге-Кутта

Численное интегрирование плоских (двумерных) и пространственных (трехмерных) задач динамики, описываемых соответственно двумя или тремя дифференциальными уравнениями второго порядка, изучим на том же типовом примере задания Д-27 [23, с. 352–361].

Он был ранее решен в одномерной постановке в п. 11.5.2. Его нелинейное дифференциальное уравнение имеет вид (11.48) с начальными условиями, определенными в (11.49).

Если задача поставлена в более общей двумерной или трехмерной постановке, то очень полезно предварительно проверить алгоритм ее

решения на частном одномерном случае. Это и позволит нам сделать выбор одномерного типового примера.

Предварительно укажем общие закономерности, представив одномерную задачу в более общей двумерной или трехмерной постановке, а затем в виде соответствующих систем дифференциальных уравнений первого порядка:

- любая одномерная задача в двумерной или трехмерной постановке при соответствующем выборе осей будет иметь соответственно одно (11.58) или два (11.60)–(11.61) дифференциальных уравнения движения 2-го порядка, равные нулю, с соответствующими нулевыми начальными условиями. Для рассматриваемого примера (11.48), после выделения в левой части второй производной и использования вспомогательных переменных (11.51), это примет вид:
  - в *двумерной* постановке:

$$\ddot{x} = -(b\_27 + c\_27\, x^2)\, x / ABS(x) - d\_27\, x - e\_27\, x^3, \quad (11.57)$$

$$\ddot{y} = 0; \quad (11.58)$$

  - в *трехмерной* постановке:

$$\ddot{x} = -(b\_27 + c\_27\, x^2)\, x / ABS(x) - d\_27\, x - e\_27\, x^3, \quad (11.59)$$

$$\ddot{y} = 0, \quad (11.60)$$

$$\ddot{z} = 0; \quad (11.61)$$

- при представлении одномерного дифференциального уравнения 2-го порядка в виде системы дифференциальных уравнений 1-го порядка:
  - для *двумерной* постановки будет только одно (11.65) нулевое дифференциальное уравнение (а не два, как можно сразу подумать, раз из одного дифференциального уравнения 2-го порядка получается 2 дифференциальных уравнений 1-го порядка):

$$\dot{x} = xp, \quad (11.62)$$

$$\dot{xp} = -(b\_27 + c\_27\, x^2)\, x / ABS(x) - d\_27\, x - e\_27\, x^3, \quad (11.63)$$

$$\dot{y} = yp, \quad (11.64)$$

$$\dot{yp} = 0; \quad (11.65)$$

  - для *трехмерной* постановки будет только два нулевых (11.69) и (11.71) дифференциальных уравнения (а не четыре):

$$\dot{x} = xp, \tag{11.66}$$

$$\dot{xp} = -\ (b\_27 + c\_27\ x^2)\ x\ /\ ABS(x) - d\_27\ x - e\_27\ x^3, \tag{11.67}$$

$$\dot{y} = yp, \tag{11.68}$$

$$\dot{yp} = 0, \tag{11.69}$$

$$\dot{z} = zp, \tag{11.70}$$

$$\dot{zp} = 0. \tag{11.71}$$

Последнее объясняется тем, что первое и третье дифференциальные уравнения 1-го порядка для плоской задачи (и плюс пятое для пространственной) только вводят подстановочные переменные для первых производных. Они не зависят от их конкретных значений (см. соответственно уравнения (11.62), (11.64) для плоской и (11.66), (11.68), (11.70) для пространственной задачи).

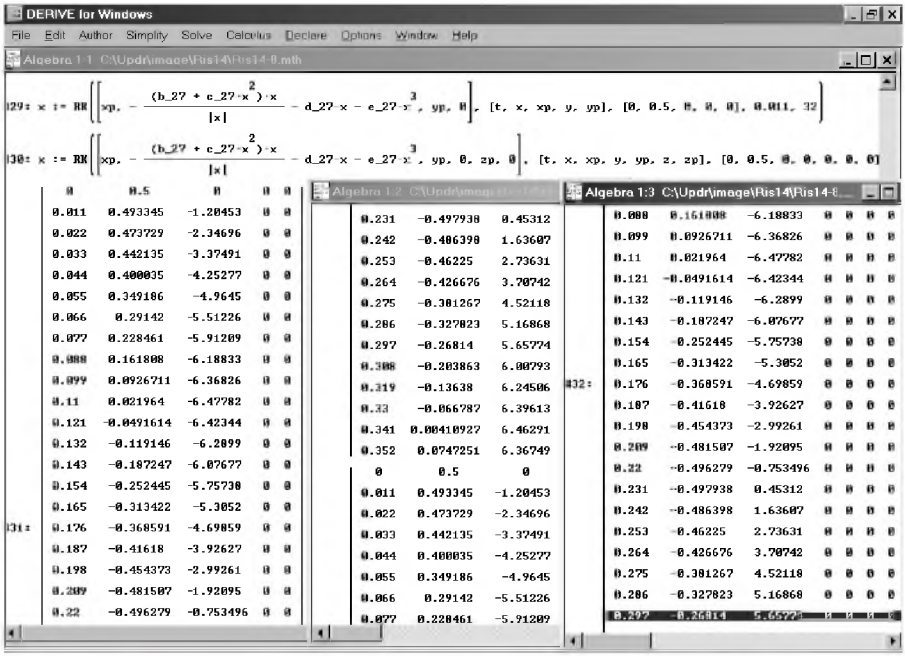

Рис. 11.9. Численное интегрирование одномерного нелинейного дифференциального уравнения второго порядка (11.48) в двумерной (строка #29) и трехмерной (строка #30) общих постановках с представлением численных результатов

В строке #29 рис. 11.9 происходит обращение к функции *RK(r, v, v0, h, n)* для численного интегрирования одномерного нелинейного дифференциального уравнения второго порядка (11.48) в двумерной общей постановке (11.62)–(11.65) со следующими фактическими параметрами:

- в качестве вектора r указывается соответствующий вектор правых частей системы дифференциальных уравнений первого порядка (11.62)–(11.65):

  [xp, – (b_27 + c_27 x^2) x / ABS(x) – d_27 x – e_27 x^3, *yp, 0*] (11.72)

- вторым фактическим параметром (в качестве вектора v) указывается вектор, содержащий перечисление используемых переменных в интегрируемой системе дифференциальных уравнений первого порядка (11.62)–(11.65): времени t, координаты x и скорости xp, координаты y и скорости yp. Также для удобства пользования фактические параметры выбраны совпадающими с формальными. В этом случае для плоской (двумерной) задачи форма записи вектора v совпадает с (11.40):

  [t, x, xp, y, yp] (11.73)

- третьим фактическим параметром (в качестве вектора v0) указывается вектор, содержащий начальные значения используемых переменных (11.49) при t = 0, в порядке, определенном в перечислении (11.73):

  [0, 0.5, 0, 0, 0] (11.74)

Четвертые и пятые фактические параметры функции RK(r, v, v0, h, n) для возможности сравнения решений в различных постановках оставлены без изменения в строках #29 и #30:

- шаг решения по переменной t также задан значением h = 0,011 (с);
- число точек решения задано значением n = 32.

В строке #30 рис. 11.9 происходит обращение к функции RK(r, v, v0, h, n) для численного интегрирования одномерного нелинейного дифференциального уравнения второго порядка (11.48) в трехмерной общей постановке (11.66)–(11.71) со следующими фактическими параметрами:

- в качестве вектора r указывается соответствующий вектор правых частей системы дифференциальных уравнений первого порядка (11.66)–(11.71):

  [xp, –(b_27+c_27 x^2) x/ABS(x) – d_27 x – e_27 x^3,*yp,0,zp,0*] (11.75)

- вторым фактическим параметром (в качестве вектора v) указывается вектор, содержащий перечисление используемых переменных в интегрируемой системе дифференциальных уравнений первого порядка (11.66)–(11.71): времени t, координаты x и скорости xp, ко-

ординаты z и скорости zp. Также для удобства пользования фактические параметры выбраны совпадающими с формальными. В этом случае для пространственной (трехмерной) задачи форма записи вектора v совпадает с (11.41):

[t, x, xp, y, yp, z, zp] (11.76)

- третьим фактическим параметром (в качестве вектора v0) указывается вектор, содержащий начальные значения используемых переменных (11.49) при t = 0, в порядке, определенном в перечислении (11.76):

[0, 0.5, 0, 0, 0, 0, 0] (11.77)

После нажатия кнопки ≈ (Approximate) при последовательно выделенных строках #29 и #30, соответствующие результаты численного решения записываются в строках #31 и #32 рис. 11.9. Они представляются в виде матрицы:

- пять столбцов которой в строке #31 при *двумерной* постановке задачи представляют значения переменных: времени t, координаты x и скорости xp, координаты y и скорости yp. Сравнив полученные результаты со строкой #28, можно сделать вывод о их полном совпадении для координаты x и скорости xp. Соответствующие значения для координаты y и скорости yp равны нулю при любом времени t, что и должно было получиться для одномерной задачи при ее решении в общей двумерной постановке;
- семь столбцов которой в строке #32 при *трехмерной* постановке задачи представляют значения переменных: времени t, координаты x и скорости xp, координаты y и скорости yp, координаты z и скорости zp. Сравнив полученные результаты со строками #28 и #31, можно также сделать вывод о их полном совпадении для координаты x и скорости xp. Соответствующие значения для координаты y и скорости yp, координаты z и скорости zp также равны нулю при любом времени t, что и должно было получиться для одномерной задачи при ее решении в общей трехмерной постановке.

Обратим ваше внимание, что нулевые столбцы при окончании строки #31 (после t = 0,231) и в начале строки #32 (перед t = 0,088), находящиеся во втором (среднем) алгебраическом окне, закрыты на рис. 11.9 третьим текущим окном для максимально большего представления полученных результатов. Также в третьем окне не поместились на экране последние пять строк (после выделенной строки при t = 0,297).

ЛИТЕРАТУРА

1. Алексеев А., Евсеев Г., Мураховский В., Симонович С. Новейший самоучитель работы на компьютере. – М.: ДЕСС КОМ , 2000.
2. Ветров С.И. Microsoft Office XP: справочник. – М.: СОЛОН–Р, 2002.
3. Гаевский А.Ю. Самоучитель работы на компьютере. – М.: Технолоджи, 2001.
4. Гурин Н.И., Скоморохов А.Г. Аналитические вычисления в системе REDUCE: Справ. пособие.– Мн.: Наука и техника, 1989.
5. Дьяконов В.П. Система компьютерной алгебры DERIVE. Самоучитель. – М.: СОЛОН–Р, 2001.
6. Дьяконов В.П. Справочник по системе символьной математики DERIVE. – М.: «СК Пресс», 1998.
7. Еднерал В.Ф., Крюков А.П., Родионов А.Я. Язык аналитических вычислений REDUCE. – М.: Изд–во Моск. ун–та, 1989.
8. Левин А. Самоучитель работы на компьютере. – Спб.: Питер, 2002.
9. Климов Д.М., Руденко В.М. Методы компьютерной алгебры в задачах механики. – М.: Наука, 1989.
10. Комягин В.Б. Настоящий самоучитель Windows 98/Me/2000. Практ. пособие. – М.: ТРИУМФ, 2001.
11. Корн Г., Корн Т. Справочник по математике для научных работников и инженеров. – М.: Наука, 1984.
12. Лобанова О.В. Практикум по решению задач в математической системе Derive: Учеб. пособие. – М.: Финансы и статистика, 1999.
13. Микляев А.П. Настольная книга пользователя IBM PC. – М.: Солон, 2001.
14. Носов В.М.. Практическое использование CAB REDUCE (на примерах теор. механики): Учеб. пособие. – Мн.: ТЕХНОПРИНТ, 2000.
15. Носов В.М. Выполнение заданий для курсовых работ по теоретической механике с применением ЭВМ: Метод. пособие по комплекс. преподаванию теор. механики, вычислит. мат. и программирования. – Мн.: БПИ – БГПА, 1989 – 1993. – Ч. 1 – 6.
16. Носов В.М. Определение скоростей и ускорений с использованием их аналогов для основных (базовых) механизмов: Учеб.–метод. пособие для мех. спец. – Мн.: БГПА, 1994.
17. Носов В.М. Программирование на персональных ЭВМ задач теоретической механики: Учеб. пособие. – Мн.: ТЕХНОПРИНТ, 1997.

18. Позняк Ю.В., Лучко Ю.Ф. Система аналитических вычислений REDUCE. – Мн.: БГУ, 1992.
19. Позняк Ю.В. Аналитические преобразования на ЭВМ в математике и механике: Учеб. пособие. – Мн.: БГУ, 1995.
20. Прудников А.П., Брычков Ю.А., Маричев О.И. Интегралы и ряды. – М.: Наука, 1981.
21. Сборник задач по теор. механике, решаемых с применением ЭВМ. Учеб. пос. для втузов/ [Е.М.Будин и др]. – СПб.: Политехника, 1995.
22. Сборник заданий для курсовых работ по теоретической механике: Учеб. пособие для техн. вузов/ [Яблонский А.А., Норейко С.С., Вольфсон С.А. и др.; Под ред. А.А. Яблонского]. – 3–е изд. – М.: Высш. шк., 1978.
23. Сборник заданий для курсовых работ по теоретической механике: Учеб. пособие для техн. вузов/ [Яблонский А.А., Норейко С.С., Вольфсон С.А. и др.; Под ред. А.А. Яблонского]. – 4–е изд. – М.: Высш. шк., 1985.
24. Теоретическая механика. Вывод и анализ уравнений движения на ЭВМ: Учеб. Пособие для вузов. Ч.1 / В.Г. Веретенников, И.И. Карпов, А.П. Маркеев и др.; Под ред. В.Г. Веретенникова. – М.: Высш. шк., 1990.
25. Фигурнов В.Э. IBM PC для пользователя. Изд.7-е.–М.: ИНФРА-М, 2001.
26. Фигурнов В.Э. IBM PC для пользователя. Краткий курс.– М.: ИНФРА-М, 2001.
27. Чуприн А., Титаренко Н. Эффективный самоучитель работы в Windows Me. – М.: ДиаСофт, 2001.
28. Шапошников А.С., Заботин Ю.Д. Самоучитель работы на ПК. Настольная книга пользователя. – М.: РИПОЛ КЛАССИК, 2001.
29. Berry Y.S., Graham E., Watkins A.J. Learning Mathematics Trough DERIVE. Ellis Horwood, 1993.
30. Bohm Josef. Teaching Mathematics with DERIVE. Great Britain: Chartwell-Bratt Ltd, 1992.
31. Brian H. Denton. Learning Linear Algebra Trough DERIVE. Ellis Horwood, 1993.
32. Hearn A.C. REDUCE USER's Manual. Version 3.0. CA 90406, RAND Co., Santa Monica, April, 1983.
33. Rayna G. REDUCE. Software for Algebraic Computation. London Springer Verlag, 1987.

Содержание

Учебное издание

Носов Валерий Михайлович

Практическое использование на персональном компьютере численных и аналитических методов в курсе теоретической механики

Редактор *В.М. Носов*

Технический редактор *В.М. Носов*

Ответственный за выпуск А.П. Аношко

Подписано в печать 25.07.2002. Формат $60\times90^{1}/_{16}$. Бум. офсетная.
Печать офсетная. Усл. печ. л. 23. Тираж 2000 экз. (1-й з-д: 1–1000).
Зак. 1082.

Издатель: УП «Технопринт». Лицензия ЛВ №380 от 28.04.1999 г.
220027, г. Минск, пр-т Ф. Скорины, 65, корп. 14, оф. 209.
тел./факс 231-86-93, тел. 239-91-57

Отпечатано с оригинал-макета на УП «Технопринт».
Лицензия ЛП №203 от 26.01.1998 г.
220027, г. Минск, пр-т Ф. Скорины, 65, корп. 14, оф. 209.
тел./факс 231-86-93, тел. 239-91-57