Министерство образования Республики Беларусь

БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Кафедра «Технология бетона и строительные материалы»

ИНФОРМАТИКА

Лабораторный практикум
«Программирование в среде Visual Basic»
для студентов специальности 1-70 01 01

Под редакцией Ж.Л. Зеленковской, О.Б. Сенько

Электронный учебный материал

Минск 2014

УДК
ББК

Авторы

О.Б. Сенько (введение, лабораторные работы 1,2,3), Ж.Л. Зеленковская (лабораторные работы 4,5,6).

Под редакцией О.Б. Сенько

Рецензенты:

Ю.В. Лях, заместитель декана СФ по учебной работе БНТУ.

Лабораторный практикум включает шесть лабораторных работ с необходимым для их выполнения теоритическим материалом с примерами. Пособие предназначено для преподавателей и студентов по курсу «Информатика», способствует усвоению студентами теоретических знаний и получению практических навыков программирования в среде Visual Basic.

Белорусский национальный технический университет
пр-т Независимости, 65, г. Минск, Республика Беларусь
Тел.(017)2659587 факс(017)2659587
E-mail: janna_zelenaja@mail.ru
Регистрационный № БНТУ/СФ70-20.2014

Оглавление

Введение

Язык программирования Visual Basic все шире используется в современном образовании. Одна из проблем, с которыми сталкивается преподаватель, работающий с этим языком, - недостаток методической литературы. Данное учебное пособие может быть полезно преподавателям и студентам, подготовленным пользователям, изучающим объектно-ориентированное программирование на языке Visual Basic. Опыт работы с другими языками программирования не обязателен, хотя, конечно, полезен.

В современном VB, конечно же, очень сложно узнать его прародителя, созданного преподавателями Дартмутского колледжа (США) почти 40 лет назад, тем не менее связь времен легко прослеживается.

Долгие годы Бейсик был довольно примитивной системой, и о его использовании в качестве средства разработки серьезных программ не могло быть и речи. Однако нужно подчеркнуть, что Бейсик фактически никогда и не был языком программирования типа Фортрана или Алгола. Ведь он изначально представлял собой качественно новую технологию создания программ (с использованием режима "позднего связывания"!) в режиме интерактивного диалога между разработчиком и компьютером, то есть был прообразом современных систем быстрой разработки программ. Другое дело, что решение подобной задачи с помощью техники тех лет было возможно только за счет максимального упрощения языка программирования и использования транслятора типа "интерпретатор".

Сейчас вряд ли кто-то рискнет заявить, что сегодня Бейсик практически стал аналогом системы Microsoft Visual Basic, которая, в свою очередь, является самым популярным в мире инструментом разработки приложений. Конечно, у нее есть свои недостатки, многие относятся к ней с иронией, но факт остается фактом: не менее 80-90% программистов если не используют эту систему в работе, то, по крайней мере, знакомы с ней и при желании могут легко сотворить в ней что-то полезное. Существенный рост числа VB-программистов начался несколько лет назад, после появления MS Office 97/VBA — осваивать программирование в среде этого пакета стали многие его продвинутые пользователи.

Использованию данного учебного пособия должно предшествовать изучение теоретического материала по тематике практических работ.

Перед выполнением первой практической работы учащийся должен создать свою папку на диске и в дальнейшем свои приложения сохранять в ней.

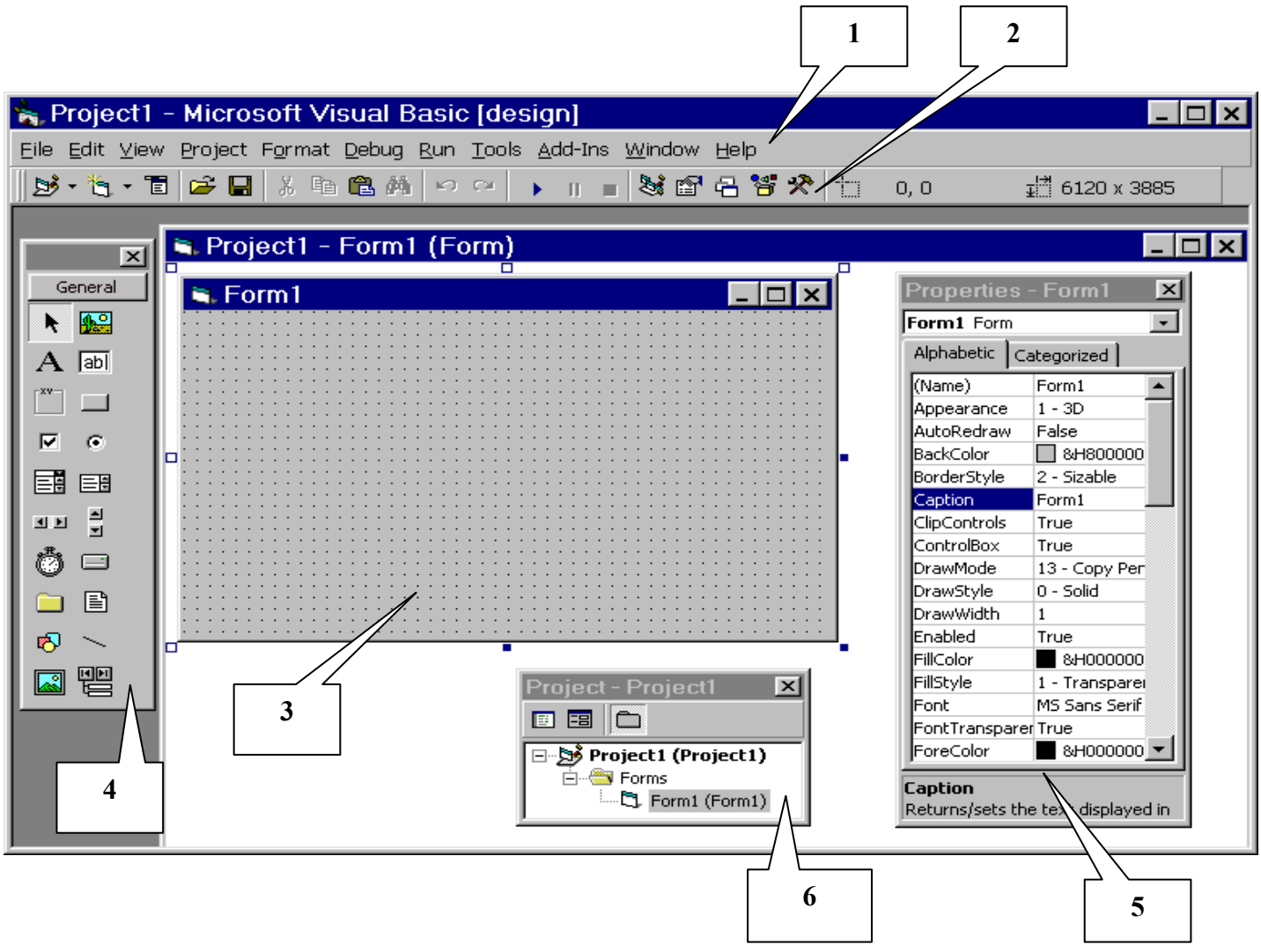
Контроль правильности выполнения практических работ и заданий для самостоятельного выполнения осуществляет преподаватель в режиме проектирования и в режиме выполнения.

1. Лабораторная работа №1

Проектирование в системе Visual Basic

Экран содержит окна:

1. *Строка меню*
   Меню содержит команды, используемые при работе Visual Basic . кроме стандартных меню **File, Edit, View, Window, Help**, здесь расположены меню, обеспечивающие доступ к функциям программирования, например, **Project, Format, Debug**
2. *Панель инструментов (***Toolbars Standard***)*
   Предоставляет быстрый доступ к наиболее часто используемым командам среды программирования
3. *Форма (* **Form** *)*– окно будущего приложения
4. *Панель элементов управления* (**Toolbox**)
   Панель обеспечивает проектировщика набором инструментов, необходимых во время разработки для размещения элементов управления на форме
5. *Окно свойств* (**Properties**)
   Перечисляет установленные свойства для выбранной формы или элемента управления.
6. *Окно проводника проекта* (**Project Explorer**)
   Представляет список форм и модулей текущего проекта. Проект – это набор файлов, используемых для построения приложения.



Объекты управления и их свойства

Объектом называется некая сущность, которая, во-первых, четко проявляет свое поведение, а во-вторых, является представителем некоторого класса подобных себе объектов.

Классом объектов называется общее описание объектов, для которых характерно наличие множества общих свойств и общих действий. Примером класса может служить класс Командная кнопка – общее описание кнопок в окнах приложений. Эти кнопки могут быть всех цветов и размеров, но должны иметь множество общих свойств и других характеристик, например, событий, которые для этих объектов одинаковы.

Форма – это окно будущего приложения. Форма обладает свойствами, определяющими ее внешний вид, методами, определяющими ее поведение, и событиями, которые определяют ее взаимодействие с пользователем.

Элементы управления – это объекты, содержащиеся внутри объекта - *Форма*. Каждый тип элемента управления имеет свой собственный набор свойств, методов и событий, что делает его пригодным для определенной цели.

Свойства в Visual Basic - количественно измеряемый атрибут объекта (элемента управления или формы). Значения свойств можно задать на стадии проектирования в окне свойств. Активизировать окно свойств, можно по-разному:

- в строке меню **View / Properties Window**
- клавиша **F4**
- кнопка на панели инструментов 

Установка значений свойств объектов

Значения свойств объектов можно менять двумя способами:

- *при проектировании*:
  В каждый момент проектирования только один объект является выделенным (активным). Он окружен рамкой из восьми маркеров. В окне свойств отображается список свойств именно активного объекта. Новое значение свойства вводится в окне свойств.
- *при выполнении приложения*:
  в программный код включается команда, имеющая следующий общий вид

ИмяОбъекта.Свойство=Значение

Получение значений свойств

Получают значение свойств тогда, когда хотят определить состояние объекта до выполнения каких-либо действий из кода. Общий вид команды следующий:

Переменная=ИмяОбъекта.Свойство

Основные свойства объектов управления

Свойство	Назначение
Name	Имя объекта
Caption	Заголовок
Visible	Видимость
BorderStyle	Стиль границ
FontBold	Полужирный шрифт
FontItalic	Курсив
FontName	*Тип шрифта*
FontSize	Размер шрифта
FontUnderline	Шрифт подчеркнутый
Enabled	Доступ
Left	Координата по горизонтали
Top	Координата по вертикали
Height	Высота объекта
Width	Ширина объекта
BorderColor	Цвет фона
ForeColor	Цвет шрифта
BorderColor	Цвет границ
FillStyle	Стиль заполнения
MousePointer	Вид курсора при наведении на объект

Свойства, используемые для управления формой

Свойство	Назначение
MinButton MaxButton	Наличие кнопки минимизации окна Наличие кнопки максимизации окна
KeyPreview	Определяет, вызываются ли процедуры обработки события клавиатуры формы перед событиями клавиатуры элементов управления
Left Top	Определяют местоположение формы по отношению к левому верхнему углу экрана монитора
Icon	Устанавливает отображаемый при сворачивании формы значок
WindowState	Состояние окна после загрузки приложения
Auto Redraw	Определяет возможность автоматического перерисования
ClipControls	Определяет необходимость перерисовки всего объекта или появляющейся части
ControlBox	Определяет наличие кнопки системного меню на форме
DrawWidth	Определяет ширину рисуемой линии (точки)
ScaleHeight, ScaleWidth	Определяет число единиц измерения по вертикали и горизонтали
ScaleLeft, ScaleTop	Определяет координаты верхнего левого угла

Основные объекты управления и их специфичные свойства

Пистограмма / Назначение		*Специфические свойства*
	*Командная кнопка* (**CommandButton**)	*Default* при значении True командная кнопка определена как кнопка по умолчанию, т.е. при нажатии Enter она будет нажата. *Cancel* определяет как кнопку отмены по умолчанию т.е. при нажатии Esc она будет нажата. Style *стиль, принимает два значения: стандартный и графический* Если стиль Graphical, то можно менять свойства: *Picture* картинка *DownPicture*- картинка внизу *DisabledPicture*- картинка если у кнопки нет доступа
	*Текстовое окно* (**TextBox**) – экранная область, в которое можно вводить текст	MaxLength *максимальная длина, если значение нуль, то можно вводить любое кол-во символов* *Multiline* значение False запрещает ввод более одной строки, значение True – разрешает ввод нескольких строк после нажатия Enter ScrollBars *наличие (1, 2, 3) или отсутствие (0) линеек прокруток в текстовом поле* *Text* текст, отображаемый в поле. *Locked* блокировка редактирования Следующие свойства доступны в режиме выполнения *SelStart* число, указывающее место вставки в строке текста *SelLength* количество выделяемых символов *SelText* определяет выделенный текст
	*Метка* (**Label**) – применяется для отображения текста, который пользователь не может редактировать	Alignment *выравнивание* *AutoSize* автоподстройка размера. При значении True размер метки подгоняется под размер текста, заданный свойством Caption. Если значение False метка сохраняет размер, установленный при проектировании *WordWpar* перенос слов *BorderStyle* стиль границ
	*Переключатель* (**OptionButton**) для организации выбора из нескольких возможностей. Выбор одного сбрасывает все другие переключатели.	Value *показывает выбран переключатель или нет* Style *стиль, изменение вида переключателя* Если стиль Graphical, то можно менять свойства: *Picture* картинка

	*Флажок* (**Check Box**) - для организации выбора типа да/нет. Работают независимо друг от друга, пользователь может установить любое их число одновременно.	*DownPicture*- картинка внизу *DisabledPicture*- картинка если у переключателя нет доступа
	*Рамка* (**Frame**) для объединения объектов в группы	
 	*Линейки прокрутки* (**Scroll bar**) горизонтальная и вертикальная действуют совершенно одинаково. Эти объекты позволяют узнавать о позиции движка (scrollbox), кроме того контролировать диапазон действия линейки прокрутки и дискретность перемещения движка	*LargeChange* определяет величину, которая добавляется или вычитается из значения Valuепри щелчке внутри линейки прокрутки *Max* число, определяющее крайнюю правую или нижнюю позицию Min *число, определяющее крайнюю левую или верхнюю позицию* SmallChange *-определяет величину, которая добавляется или вычитается из значения Value при щелчке на одной из стрелок на концах линейки прокрутки* *Value* число, которое отражает текущую позицию движка на линейке
	*Таймер (* **Timer***)* – это объект, способный инициировать события через регулярные промежутки времени	*Interval* число (от 0 до 65535), определяющее интервал времени в мс между двумя событиями. Интервал, равный нулю, отключает таймер
	*Линия (***Line***)* –для вычерчивания линий на поверхности формы. Не поддерживает никаких событий.	*X1, Y1* координаты левого края линии *X2, Y2* координаты правого края линии *BorderWidth* толщина линии *BorderStyle* стиль линии
	*Список (***ListBox***)*- предоставляет список возможных вариантов выбора, позволяет ограничить ввод элементами списка	*Style* стиль списка *Sorted* сортировка элементов списка *List* позволяет заполнить список на стадии проектирования *ListIndex* определяет положение выбранного элемента списка *ListCount* определяет количество элементов списка *Text* элемент списка или строка, введенная пользователем
	*Комбинированный список (***ComboBox***)* Совмещает возможности списка и текстового окна, содержит редактируемое поле	

	*Окно рисунка (***PictureBox***)*- для размещения графической информации в определенных участках формы. Требуют больше памяти и времени на обработку, больше подходят для динамических объектов Может выполнять функции контейнера для других элементов управления.	*Picture* позволяет выводить растровую картинку (.bmp), либо значок (.icon) *AutoSize* автоподстройка размера
	*Изображение (***Image***)* – для размещения графической информации в определенных участках формы. Удобно использовать в статической среде (не предполагается изменение)	*Picture* позволяет выводить растровую картинку (.bmp), метафайл, файлы JPEG или GIF либо значок (.icon) *Stretch* Растягивать. Если значение True картинка подгоняется под размер элемента управления.
	*Контур или фигура (***Shape***)* для вычерчивания контуров в виде прямоугольника, окружности, овала, квадрата, прямоугольника, квадрата с закругленными углами	*Shape* тип контура *FillStyle* стиль заполнения *BorderStyle* стиль границ контура *BorderWidth* толщина контура
	*Список файлов(* **FileListBox***)* позволяет узнать, какие есть файлы на дисках системы и выбрать один из них	*Pattern* определение шаблона для списка файлов Следующие свойства определяют тип отображаемых файлов *Archive* архивный *System* системный *Hidden* скрытый *ReadOnly* только для чтения
	*Список каталогов (***DirListBox***)* позволяет узнать, какие есть каталоги на дисках системы и выбрать один из них	*Path* позволяет установить или получить текущий каталог

Наименование объектов Visual Basic

При изменение имени (*Name*) объектов Visual Basic рекомендуется использовать следующую простую схему:

- начинать название с трехбуквенного префикса;
- использовать только буквы, цифры и знак подчеркивания (_);
- использовать не более 40 символов.

Создатели Visual Basic рекомендуют начинать название с трехбуквенного префикса в соответствии с типом объекта. Например, у вас может быть командная кнопка с названием **cmdCancel** и форма **frmMain**. Рекомендуемые префиксы перечислены в табл.

Объект	Рекомендуемый префикс
Форма	Frm
Флажок	Chk
Комбинированное окно	Cbo
Командная кнопка	Cmd
Окно данных	Dat
Список каталогов	Dir
Список дисков	Dsk
Рамка	Fra
Сетка	Grd
Горизонтальная линейка прокрутки	Hsb
Изображение	Img
Метка	Lbl
Линия	Lin
Список	Lst
Меню	Mnu
Переключатель	Opt
Окно рисунка	Pic
Фигура	Shp
Текстовое окно	Txt
Таймер	Tmr
Вертикальная линейка прокрутки	Vsb

События

Событием называется характеристика класса объектов, описывающая внешнее воздействие, на которое реагирует объект этого класса во время работы приложения.

Программы на Visual Basic управляются событиями, другими словами – действия пользователя вызывают выполнение различных процедур. Работает это примерно так: программа ждет, пока пользователь не сделает что-либо, т.е. пока не произойдет событие; затем программа реагирует на это событие, запуская соответствующую процедуру или процедуры, затем программа снова терпеливо ждет следующего события.

События, возникающие при работе с мышью

Событие	Описание	Параметры событий
Click	Щелчок	
DblClick	Двойной щелчок	
MouseDown	Кнопка мыши нажата	Shift – **определяет статус клавиш Shift(1), Ctrl(2), Alt(4) ни одна кнопка не нажата - 0** Button – **определяет статус нажатой кнопки (левая - 1, правая - 2, средняя– 4,)** X,Y **– позиция указателя курсора**
MouseUp	Нажатая кнопка мыши отпущена	
MouseMove	Мышь перемещается из своей текущей позиции	
DragDrop	Завершение перетаскивания	**Source** – ссылка на объект, который был перемещен, **X**,**Y** – позиция курсора **Stale** – принимает значения 0 –область занята, 1–область свободна
DragOver	«Буксируемый» объект попадает в область другого объекта	

События, возникающие при работе с клавиатурой

Событие	Описание	Параметры событий
KeyPress	Нажата клавиша, соответствующая символу ASCII	**KeyAscii**- значение ASCII-кода нажатой клавиши
KeyDown	Нажата любая клавиша на клавиатуре	**KeyCode** – указывает нажатую клавишу **Shift** – определяет статус клавиш Shift(1), Ctrl(2), Alt(4)
KeyUp	Отпущена любая клавиша	

События, связанные с фокусом

Объект, *имеющий фокус*, может получать вводимую пользователем информацию с помощью мыши и клавиатуры.

Событие	Описание
GotFocus	Получении фокуса
LostFocus	Потеря фокуса

События, специфичные для объектов

Событие	Описание	Для какого объекта характерны
Load UnLoad	Загрузка Закрыть форму	Форма
Resize	Изменение размера	Форма, Картинка
Change	Изменение	Текстовое окно, Полосы прокрутки, Список каталогов Комбинированный список,
Scroll	Прокрутка	Полосы прокрутки, Список, Комбинированный список, Список файлов, Список каталогов
Timer	Истечение интервала времени	Таймер

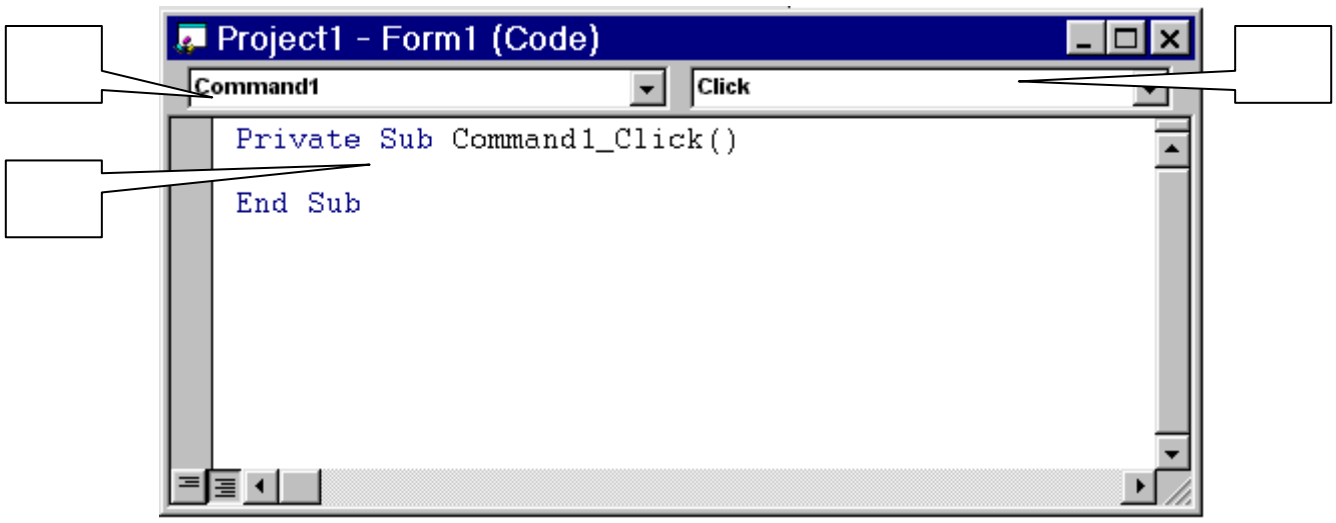
Окно кода процедуры обработки события

Любой объект можно связать с набором процедур, исполняемых в строго определенные моменты. Процедура (*procedure*) – это группа операторов языка. Исполняется процедура – исполняются ее операторы. Так или иначе, весь составленный вами исполняемый код обязательно помещается в какую-нибудь процедуру. Процедура, присвоенная объекту, связана с определенным событием и поэтому называется *процедурой обработки события*. Важно отметить, что с одним объектом могут быть связаны несколько событий.

Имя процедуры обработки события для элемента управления составляется из имени элемента управления (Name), знака подчеркивания (_) и имени события.

Для открытия окна кода процедуры существует три способа:

1. двойной щелчок по объекту
2. клавиша **F7**
3. в меню выбрать **View / Code**



1. список объектов формы
2. список событий объекта
3. процедура

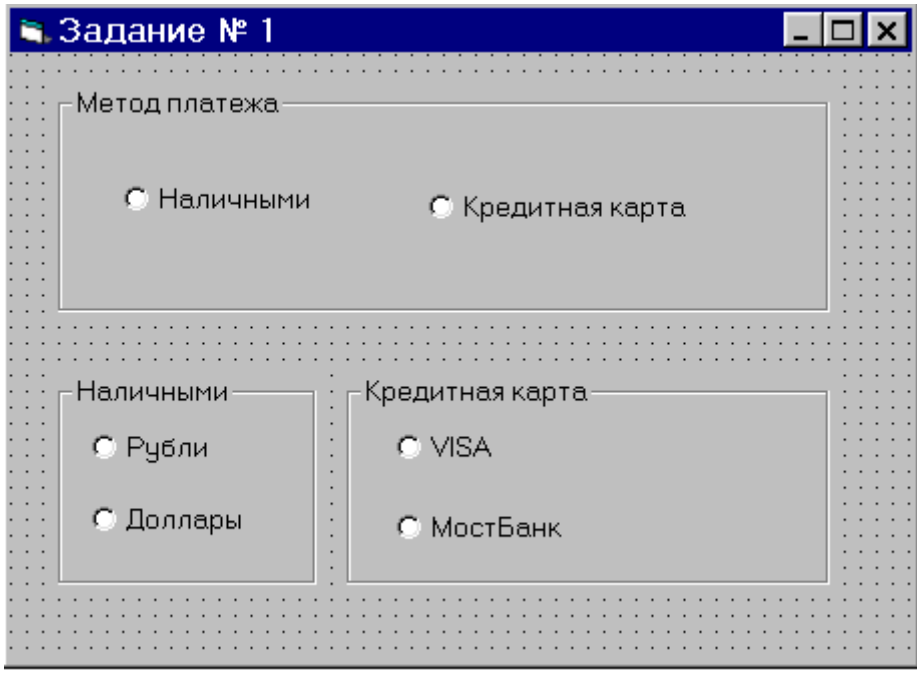
Процедуры по умолчанию не делают ничего; они состоят лишь из объявления процедуры (*Sub*)и оператора, помечающего конец процедуры (*End Sub*)
Программный код вводится между строками **Private Sub** и **End Sub**

Задания

1. форма приложения должна полностью соответствовать приведенному образцу;
2. имена объектов управления должны иметь трехбуквенный префикс в соответствии с типом объекта;
3. после разработки приложения создать исполняемый файл.

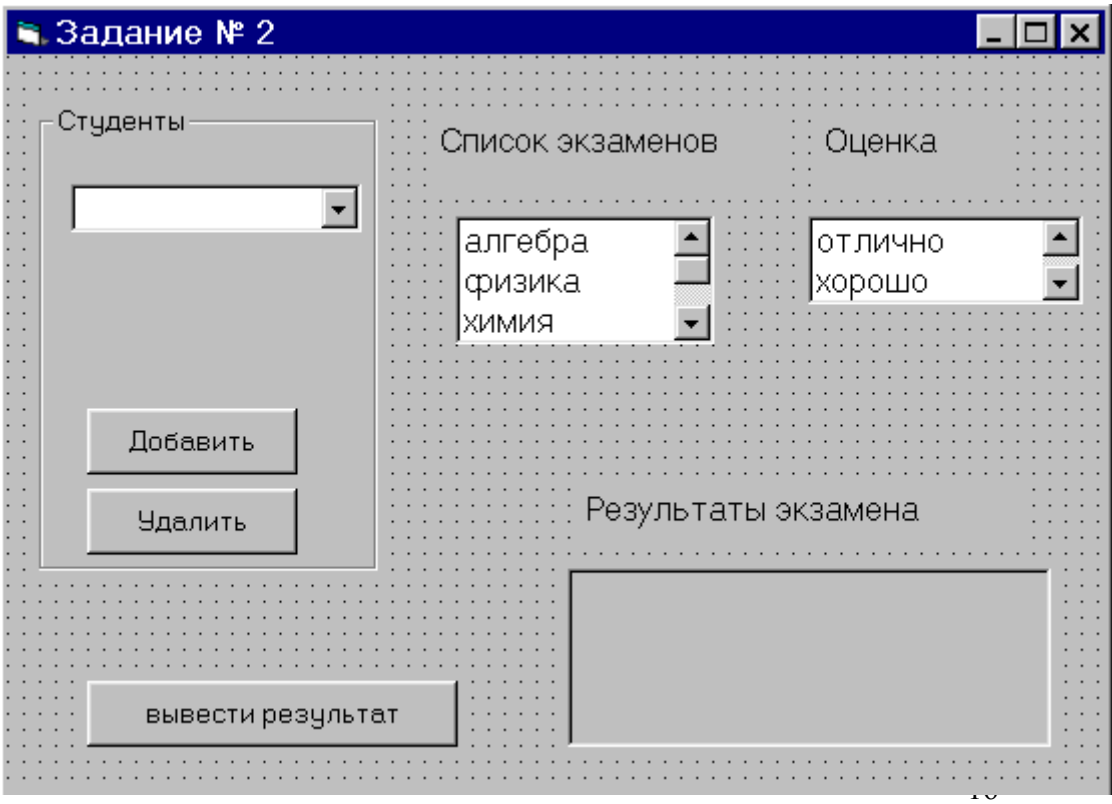
Задание № 1

Разработать приложение, которое в зависимости от выбранного переключателя в группе «Метод платежа» отображает в окне приложения только одну из следующих групп переключателей:«Наличными » или «Кредитная карта»



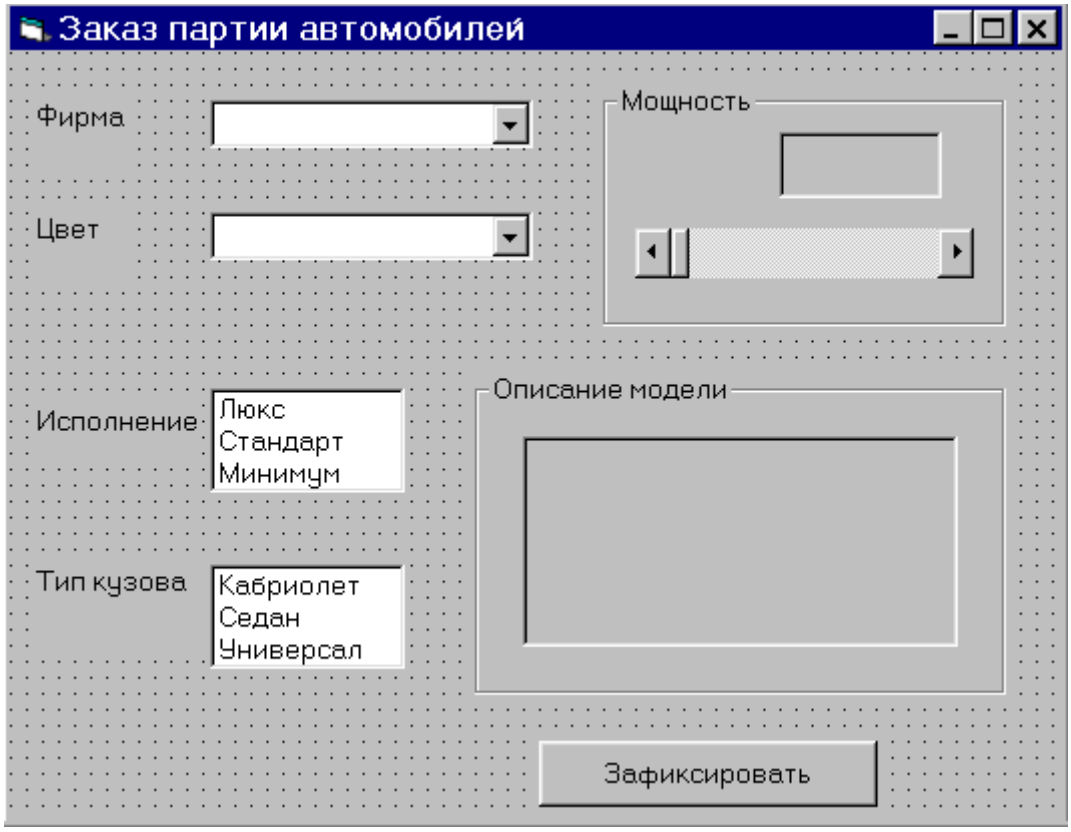
Задание № 2

Разработать приложение, позволяющее добавлять и удалять фамилии студентов в список, а также выводить в метку выбранные значения из трех списков (при щелчке по командной кнопке). Значения для списков «Студенты», «Список экзаменов», «Оценка» сформировать на стадии проектирования



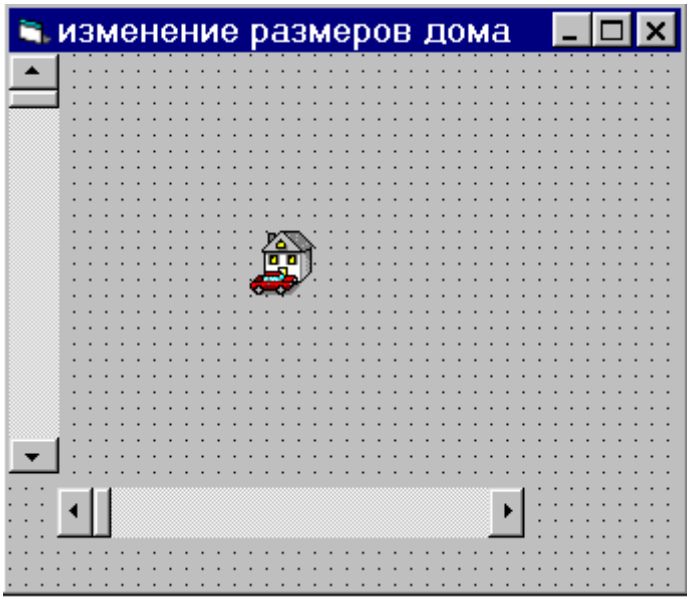
Задание № 3

Разработать приложение, позволяющее с помощью списков «Фирма», «Цвет», «Исполнение», «Тип кузова» и полосы прокрутки дать описание модели автомобиля. Выбранные значения должны отображаться в метке по щелчку по кнопке «Зафиксировать» . Значения списков формируются на стадии проектирования.



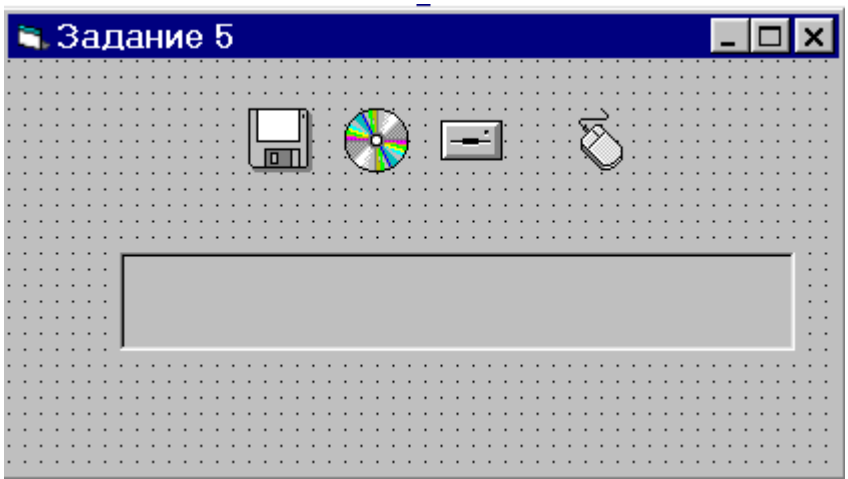
Задание № 4

Разработать приложение, которое с помощью полос прокруток позволяет менять высоту и ширину изображения.



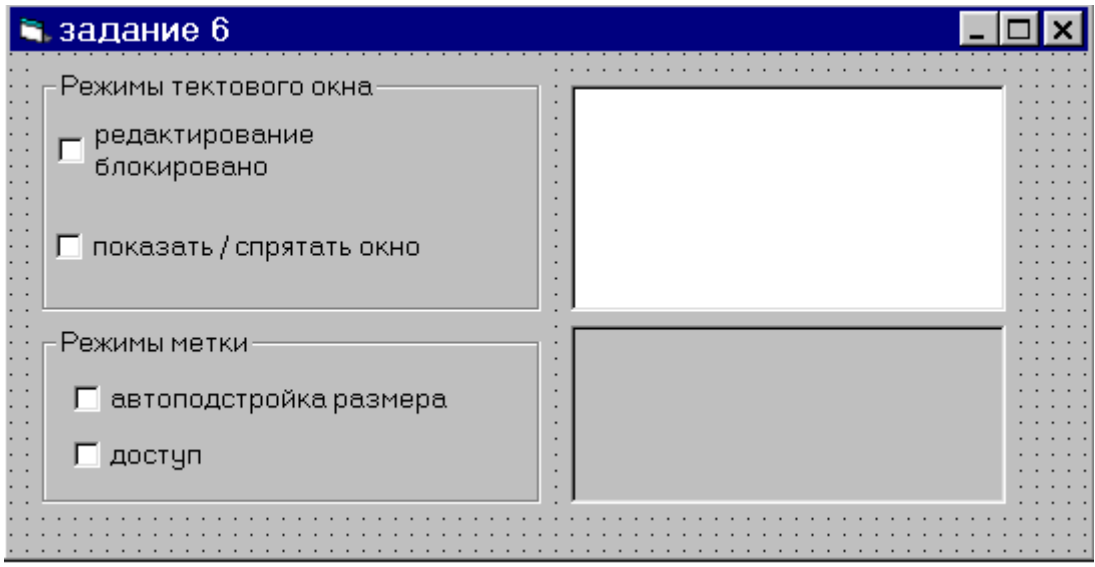
Задание № 5

Разработать приложение, позволяющее при щелчке по одному из изображений выводить сообщение о его назначении.



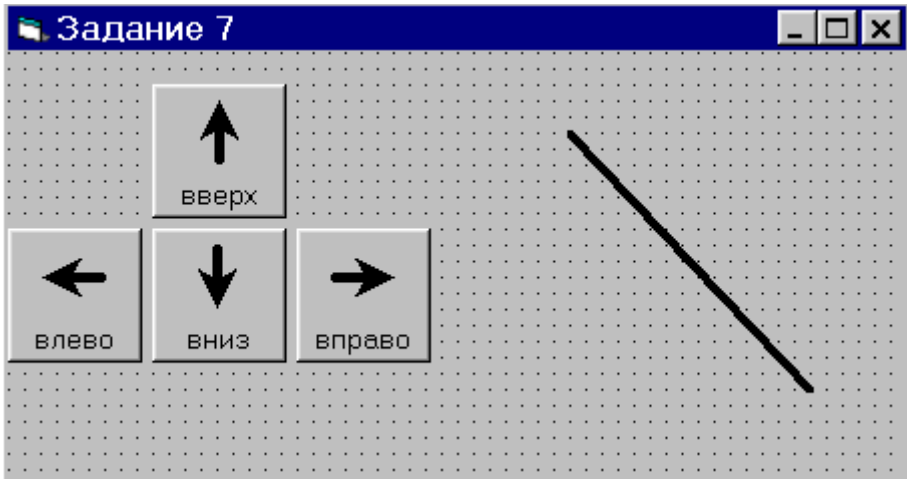
Задание № 6

Разработать приложение, позволяющее менять режимы работы текстового окна и метки с помощью групп соответствующих флажков.



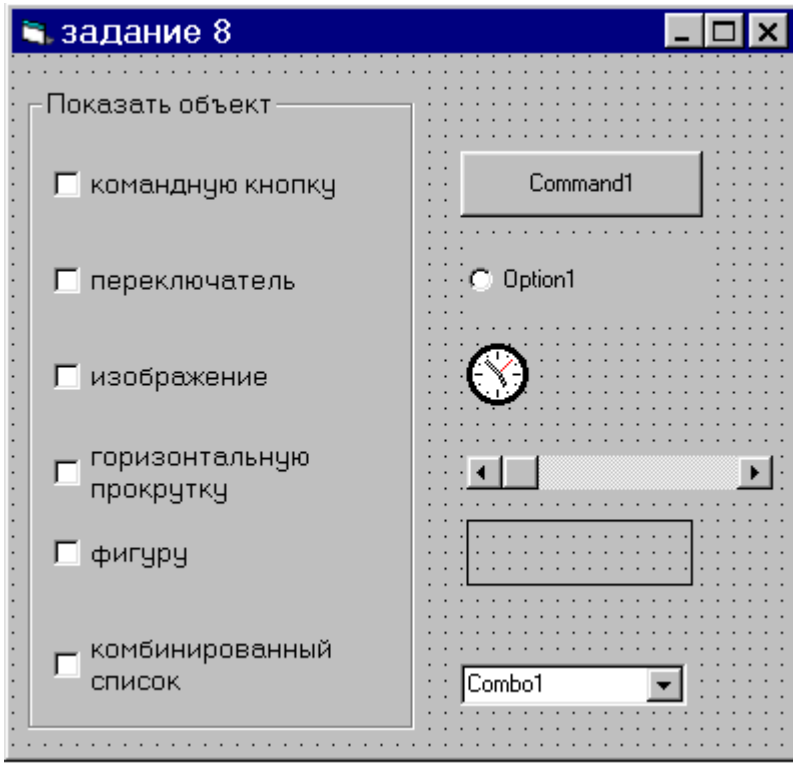
Задание № 7

Разработать приложение, позволяющее с помощью командных кнопок перемещать линию в выбранном направлении.



Задание № 8

Разработать приложение, позволяющее с помощью группы флажков отображать или не отображать элементы управления в окне приложения..



Лабораторная работа №2.

Типы данных. Операторы присваивания. Процедуры. Написание простейшей программы на VB.

Для работы в VB необходимы в основном 2-а окна:
a) Окно программы (содержит текст программы, запускается командой View\Code)
b)Окно отладки (содержит полученные в результате выполнения программы результаты, запускается командой View\Immediate window).

Основное наполнение модуля.

Самое начало модуля называется общей областью, в которой располагаются общие описания, например, типа данных, используемого по умолчанию. Типы данных для VB:
1) Byte – Массивы данного типа служит для хранения двоичных данных, например, изображений. Использование данного типа предохраняет двоичные данные во время преобразования формата.
2) Boolean – для хранения логических (булевых) значений. По умолчанию значением булевской переменной является False - ложь.

3) Currency - для хранения чисел с дробной частью до четырех цифр и целой частью до 15 цифр, то есть данных с фиксированной десятичной точкой, удобных для денежных вычислений.
4) Double - для хранения чисел с плавающей десятичной точкой.
5) Integer- для хранения малых целых числовых значений (от -32768 до +32767).
6) Long – для хранения больших целых числовых значений (от -2 147483648 до +2 147483647).
7) Object – поскольку VB является объектно-ориентированным языком, в нем можно манипулировать различными объектами, адреса расположения которых в памяти (указатели) имеют этот тип.
8) String – по умолчанию данные строкового типа имеют переменную длину и могут удлиняться или укорачиваться. Однако такие строки занимают на 10байт памяти больше, поэтому можно объявить строки фиксированной длины, указав количество символов. Если количество символов будет меньше объявленного, то свободные места заполняются пробелами, при попытке занесения большего количества символов лишние отбрасываются.
9) Variant – может быть использован для хранения данных всех базовых типов без выполнения преобразования (приведения) типов. Применение данного типа позволяет выполнять операции, не обращая внимание на тип данных, которые они содержат. Удобен для объявления переменных, тип которых заранее неизвестен.
10) Date – используется для хранения даты и времени.
Для объявления типов данных используются операторы описания.

Операторы описания

Объявление переменной производится операторами Dim, Private за которым следует имя переменной и необязательная часть с ключевым словом As, после которого задается тип переменной, например Dim name [As type]. Оператор Dim используется только вне модуля, в его общей части и делает описываемую переменную доступной из всех процедур всех модулей проекта. Оператор Private служит для объявления переменной уровня модуля, доступной только процедурам данного модуля.

Если необходимо объявить константу, то используется оператор Const (напр. Const a=5).
Для присвоения же конкретного числового значения переменной используются оператор присваивания.

Оператор присваивания

Инструкция Let Присваивает значение выражения переменной или свойству:
[Let] имяПеременной = выражение

Явное использование ключевого слова Let зависит от вкуса пользователя, обычно это слово опускают, записывая, например, a=5.
Значение выражения может быть присвоено переменной, только если оно имеет совместимый с этой переменной тип данных. Невозможно присвоить строковое выражение числовой переменной или числовое выражение строковой переменной. Такая попытка приведет к ошибке во время компиляции.

Переменным типа Variant могут присваиваться как строковые, так и числовые выражения. Однако обратное не всегда верно.

Присвоение выражения с одним из числовых типов переменной с другим числовым типом данных преобразует значение выражения в тип данных результирующей переменной. (т.е. если a - integer, то если let a=2,33, то будем иметь целое значение 2).

После объявления типов данных вводятся процедуры - наборы описаний и инструкций, сгруппированных для выполнения. Существует три типа процедур:

1) процедура Sub - набор команд, с помощью которого можно решить определенную задачу. При ее запуске выполняются команды процедуры, а затем управление передается в приложение пакета MS Office или процедуру, которая вызвала данную процедуру. Закрывается процедура командой End Sub.
2) процедура Function (функция) также представляет собой набор команд, который решает определенную задачу. Различие заключается в том, что такие процедуры обязательно возвращают значение, тип которого можно описать при создании функции.
3) процедура Property используется для ссылки на свойство объекта. Данный тип процедур применяется для установки или получения значения пользовательских свойств форм и модулей.
Рассмотрим, теперь, простейшую программу (наберём и откомпилируем её).

Пример программы:

```
Dim b As Integer
Dim c As Integer
Dim p As Integer
Const a = 4
Private Sub Command1_click()
b = 5
c = 4
p = a + b + c
Debug.Print a, b, c, p
End Sub
```

Эта программа делает расчёт периметра треугольника со сторонами 4, 5, 4.
Команда Private Sub Command1_click() - служит для определения переменной уровня модуля и процедуры Sub и используется практически в любой программе VBA.
Команда Debug.Print a, b, c, p - отображает значения всех указанных величин в окне “Окно отладки (Immediate)”.
Команда End Sub - закрывает процедуру.

Задания

1. Даны два числа a и b. Получить их сумму, разность и произведение.

2. Даны действительные числа x и y. Получить (|x| – |y|) / (1 + |x·y|).

3. Дана длина ребра куба. Найти площадь грани, площадь полной поверхности и объем этого куба.

4. Даны два действительных положительных числа. Найти среднее арифметическое и среднее геометрическое этих чисел.

5. Даны катеты прямоугольного треугольника. Найти его гипотенузу и площадь.

6. Определить периметр правильного n-угольника, описанного около окружности радиуса r.

7. Даны x, y, z. Вычислить a, b, если

а) $a = \frac{\sqrt{|x-1|} - \sqrt{|y|}}{1 + x^3/2 + x^3/4}, b = x(arctg(z) + e^{-(x+3)})$;

б) $a = \frac{3 + e^{x-1}}{1 + x^2|y - tg(z)|}, b = 1 + |y - x| + \frac{(y-x)^2}{2} + \frac{|y-x|^3}{3}$;

в) $a=(1+y)\frac{x+y/(x^2+4)}{e^{x-2}+1/(x^2+4)}, b=\frac{1+\cos(y-2)}{x^4/2+\sin^2 x}$;

г) $a=y+\frac{x}{y^2+\frac{x^2}{y+x^3/3}}, b=\left(1+tg^2\frac{x}{2}\right)$;

д) $a=\frac{2\cos(x-\pi/6)}{1/2+\sin^2 y}, b=1+\frac{z^2}{3+z^2/5}$;

е) $a=\frac{1+\sin^2(x+y)}{2+\left|x-2x/(1+x^2y^2)\right|}+x, b=\cos^2\left(arctg\frac{1}{x}\right)$;

ж) $a=\ln\left|y-\sqrt{|x|}\left(x-\frac{y}{z+x^2/4}\right)\right|, b=x-\frac{x^2}{3!}+\frac{x^3}{4!}$.

8. Дана сторона равностороннего треугольника. Найти площадь этого треугольника.

9. Известна длина окружности. Найти площадь круга, ограниченного этой окружностью.

10. Найти площадь кольца, внутренний радиус которого равен 20, а внешний – заданному числу r (r>20).

Лабораторная работа №3

Функции MsgBox и InputBox. Условный оператор If

MsgBox (“сообщение”, [кнопки, заголовок]) - эта функция отображает диалоговое окно, содержащее сообщение длиной до 1024 символов, в которое с помощью операции конкатенации можно включить значение переменных, а также (необязательно) кнопки для реакции на отображения окна (по умолчанию только кнопка ОК).

При задании сложного диалогового окна, при помощи функции MsgBox используются следующие константы:

1) Для задания внешнего вида окна сообщения:
vbCritical, vbQuestion, vbExclamation, vbInformation.
2) Для задания кнопок в окне сообщения:
vbOkCancel, vbAbortRetryIgnore, vbYesNOCancel, vbYesNO, vbRetryCancel.
3) Для задания дальнейших действий после нажатия на соответствующую кнопку:
vbOk, vbCancel, vbAbort, vbRetry, vbIgnore, vbYes, vbNO.

Программа 1:

```
Private Sub Решение_click()
y = MsgBox("Закрыть окно", vbQuestion + vbYesNoCancel, "Сообщение Windows")
End Sub
```

InputBox (“сообщение”[, заголовок] [, значение по умолчанию] [, координата x] [, координата y]) - функция, применяемая для ввода значений переменных в программу. Эта функция отображает диалоговое окно, содержащее окно ввода, кнопки ОК и Отмена, сообщение (подсказку для ввода) и (необязательно) заголовок окна, значение, вводимое по умолчанию, координаты окна по горизонтали и вертикали.

Пример:

ввод стороны фигуры можно задать командой

a = InputBox ("первая сторона")
Условный оператор If - это оператор позволяющий задавать выполнение тех или иных действий в зависимости от заданных условий. Основными составляющими для этого служат:
1) if (если)
2) then (тогда)
3) else (иначе)
Пример:
Выражение - если a>1 то b= a+1 иначе b=a-1 будет иметь вид
If a>1 then b= a+1 else b=a-1

Программа 2:

Компьютер должен перемножить два числа. Если их произведение больше 2000, то компьютер должен напечатать текст “Произведение большое”, иначе “Произведение маленькое” и чему равно само произведение

```
Dim a, b, y As Integer
Private Sub Command1_click()
a = InputBox("первое число")
b = InputBox("второе число")
y = a*b
If y>2000 Then Debug.Print “Произведение большое” Else Debug.Print “Произведение маленькое”
Debug.Print y
End Sub
```

Оператор If может применяться для переменных любого типа (в т.ч. и строкового).

Программа 3:

Вводится слово. Если это слово - колхозник то получаем слово - фермер, иначе - неизвестный.

```
Dim Slovo As String
Private Sub Command1_Click()
Slovo = InputBox(“Введите слово”)
If Slovo = “колхозник” Then Slovo = “фермер” Else Slovo = “неизвестный”
Debug.Print Slovo
End Sub
```

Рассмотрим программу, которая включает в себя сложную функцию MsgBox и оператор If.

Программа 4:

Вводятся два произвольных числа. Затем задаётся вопрос “Вы уверены что хотите их перемножить?” и варианты ответов: “да”, “нет”. Если ответ “да” - то числа перемножаются, иначе - складываются.

```
Dim a, b, d As Double
Private Sub Command1_click()
a = InputBox("первое число")
b = InputBox("второе число")
y = MsgBox("Вы уверены что хотите их перемножить ? ", vbCritical + vbYesNo, "Вопрос")
If y = VbYes Then d=a*b else d=a+b
Debug.Print d
End Sub
```

Правила записи однострочного оператора If. Многострочный оператор If. Логические операции. Оператор варианта Select Case.

Правила записи однострочного оператора If

Синтаксическая схема 1 (если условие одно и оператор тоже один)
If условие then оператор else оператор
Пример: If a>1 then b=a+1 else b=a+2
Синтаксическая схема 2 (если условие одно, а операторов - несколько)
If условие then оператор1 : оператор2: .... else оператор3 : оператор4...
Пример: If a>1 then b=a+1 : c=a-1 else b=a+2 : c=a+4
Правила записи многострочного оператора If
Синтаксическая схема 2 (если условие одно, а операторов - несколько)
If условие Then
оператор1
оператор2
....
Else
оператор3
....
End If

Программа 1:

В компьютер вводится целое число a
1. Если a<0 то компьютер должен сказать: “число отрицательное”
2. Если a=0 то: “вы ввели нуль”
3. Если a>100 то компьютер должен сказать: “число большое”
4. В остальных случаях компьютер ничего не должен говорить, а только вычислить и напечатать квадрат этого числа.

```
Dim a As Long.
Private Sub Command1_click()
a = InputBox("Введите число")
If a<0 Then
MsgBox (“Число отрицательно”)
ElseIf a=0 Then
MsgBox (“Вы ввели нуль”)
ElseIf a>100 Then
MsgBox (“Число большое”)
Else
Debug.Print a^2
End If
MsgBox(“До свиданья”)
End Sub
```

Здесь оператор ElseIf - переводится как “иначе если” и тоже является условным оператором.

Логические операции

Основным логическим оператором является оператор And, представляющий собой связующее звено для каких-либо условий. Рассмотрим его применение к конкретной задаче

Программа 2:

Необходимо из представленной группы людей выбрать человека чей рост<180 см а глаза - голубые.

```
Dim Tsvet As String          ‘Цвет
Dim Rost As Integer          ‘Рост
Private Sub Command1_click()
Tsvet= InputBox("Введите цвет глаз")
Rost = InputBox("Введите рост в сантиметрах")
y = a*b
If Tsvet=“Голубой” And Rost<180 Then Debug.Print “Кандидат найден” Else Debug.Print “Неудача”
End Sub
```

Ещё одним логическим оператором является оператор Or - “или”

Оператор Варианта Select Case

Select Case в переводе на русский означает “выбери вариант”

Рассмотрим применение этого оператора на примере задачи

Программа 3:

Компьютер спрашивает студента какую отметку тот получил по предмету и реагирует на неё подходящим текстом.

```
Dim Otmetka As Long.
Private Sub Command1_click()
Otmetka = InputBox("Какую оценку ты получил")
Select Case Otmetka
Case 1,2
Debug.Print “Кошмар”
Case 3
Debug.Print “Неважно”
Case 4
Debug.Print “Неплохо”
Case 5
Debug.Print “Почти гений”
Case Else
Debug.Print “Таких отметок не бывает”
End Select
End Sub
```

Замечание: если бы нам нужно было бы в качестве условия задать например ряд чисел от 50 до 100, то это можно было бы сделать при помощи оператора To:

```
Case 50 To 100.
```

Задания

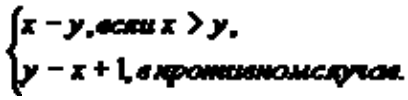
1. Даны действительные числа x, y. Получить:
   а) max(x,y);
   б) min(x,y);
   в) max(x,y), min(x,y).
2. Даны действительные числа x, y, z. Получить:
   а) max(x,y,z);
   б) min(x,y,z);
   в) max(x,y,z), min(x,y,z).

3. Даны действительные числа x, y, z. Вычислить:

а) max2(x+y+z,xyz);
б) min(x+y+z/2,xyz)+1;
в) min(x+5y+z,2x-y+z) + max(x+y+z,xy+z).

4. Даны действительные числа a, b, c. Удвоить эти числа, если a > b > c, и заменить их абсолютными значениями, если это не так.

5. Даны действительные числа x, y. Вычислить z:



6. Даны два действительных числа. Вывести первое число, если оно больше второго, и оба числа, если это не так.

7. Даны два действительных числа. Заменить первое число нулем, если оно меньше или равно второму, и оставить без изменения в противном случае.

8. Даны действительные числа x, y (x ¹ y). Меньшее из этих двух чисел заменить их полусуммой, а большее – их удвоенным произведением.

9. Даны действительные числа x, y, z. Выяснить, существует ли треугольник с длинами сторон x, y, z.

10. Даны действительные числа a, b, c (a ¹ 0). Выяснить, имеет ли уравнение $ax^2 + bx + c = 0$ действительные корни. Если действительные корни имеются, то найти их. В противном случае ответом должно служить сообщение, что действительных корней нет.

Лабораторная работа №4

Случайные величины. Форматирование результата. Команда Format. Цикл. Оператор перехода Goto. Метки.

Случайные величины

Для задания случайных величин используется оператор Rnd. Например команда p=Rnd задаёт число p как ряд каких-либо случайных чисел. Для задания более широкого диапазона случайных чисел перед оператором Rnd добавляется оператор Randomize.

Программа 1.

Необходимо задать произвольный ряд случайных чисел

```
Dim p As Double
Private Sub Command1_click()
Randomize
p = Rnd
Debug.Print p
End Sub
```

Теперь, рассмотрим случай, когда нам необходимо задать случайные числа из конкретного диапазона.

Программа 2.

Необходимо задать целое число p из диапазона от 200 до 210

```
Dim p As Double
Private Sub Command1_click()
```

```
t = 200 + 11 * Rnd
p = Int(t)
Debug.Print t, p
End Sub
```
Здесь Int - оператор округления t до целого числа.
Форматирование результата. Команда Format.
В результате выполнения программы часто в качестве результаты получаются числа с очень большим количеством цифр после запятой. Для ограничения этого количества используется команда Format. Например, нам необходимо вывести как результат какое-либо число p с точностью до трёх знаков после запятой. Тогда вместо команды Debug.Print p мы записываем Debug.Print Format (p, "0.000").

Цикл. Оператор перехода Goto. Метки.

Цикл - многократно выполняемая последовательность каких-либо команд.
Допустим, мы хотим чтобы компьютер повторил выполнение следующего фрагмента: “Это тело цикла.....” десять раз

Программа 3.

```
Dim i As Integer
Private Sub command1_click()
i = 0
m1:  i = i + 1
      If  i < 10 Then
      Debug.Print "Это"
      Debug.Print "тело"
      Debug.Print "цикла"
      Debug.Print "....."
      GoTo m1
      Else
      GoTo m2
      End If
m2: End Sub
```
Здесь i - счётчик, ограничивающий количество выполненных фрагментов,
m1, m2 - метки, задающие выполнение каких-либо операций.
Goto - оператор безусловного перехода, переводится “идти к”.
Если в процессе выполнения цикла происходит зацикливание то необходимо нажать сочетание клавиш “Ctrl + break”.

Цикл. Оператор цикла Do и его разновидности. Оператор цикла While ... Wend.

Оператор Do

Этот оператор имеет три вида записи :
1) Do Loop While (Loop - “петля, возврат назад”, While - “пока”)
Синтаксис:
```
Do
операторы
операторы
.....
Loop While условие продолжения работы цикла
```

Программа 1

Необходимо напечатать числа 3, 5, 7, 9.

```
Dim i As Integer
Private Sub command1_click()
Debug.Print "Начало счёта"
i = 3
Do
    Debug.Print i
    i = i + 2
    Loop While i <= 9
    Debug.Print "Конец счёта"
    End Sub
```

2) Do Loop Until (Until - “до тех пор пока”)

Синтаксис:

```
Do
операторы
операторы
.....
```

Loop While условие завершения работы цикла

Т.е. в программе 1 команда Loop While i <= 9 заменится командой Loop While i > 9.

3) Do Until Loop

Синтаксис:

```
Do Until условие продолжения работы цикла
операторы
операторы
.....
Loop
```

Тогда фрагмент цикла из программы 1 будет иметь вид

```
i = 3
Do Until i > 9
            Debug.Print i
    i = i + 2
    Loop
```

Программа 2

Компьютер предлагает человеку ввести слово, после чего распечатывает это слово снабдив его восклицательным знаком. Затем опять предлагает ввести слово и так до тех пор пока не будет введено слово “Хватит”. Распечатав его с восклицательным знаком, компьютер отвечает “Хватит, так хватит” и заканчивает работу.

Применим второй вариант оператора Do

```
Dim Slovo As String
Private Sub Command1_Click()
Do
Slovo = InputBox("Введите слово")
Debug.Print Slovo; "!"
Loop Until Slovo = "Хватит"
Debug.Print "Хватит так хватит"
End Sub
```

Оператор While ... Wend

Данный оператор широко использовался в ранних версиях Basic. Его использование в Visual Basic 6.0. возможно, но целесообразней использовать операторы Do и For.

Синтаксис:
While условие продолжения работы цикла
операторы
операторы
.....
Wend

Задание 1

Дополнить программу 2 так, чтобы перед распечаткой каждого слова стоял его порядковый номер.

Задание 2

Для ... вычислять и печатать и до тех пор, пока .

Дополнение:

Использовать оператор Do.

Цикл. Оператор цикла For и виды его записи.

Оператор For

При программировании на Visual Basic возникает много задач, для решения которых цикл нужно выполнить определённое количество раз. В этом случае удобно использовать оператор цикла For.
Для этого оператора используется обычно два вида записи:
1) Когда шаг равен единице
   For i=1 To n
   операторы
   Next i
Здесь For - для, To - до, n - какое-либо число, Next i - увеличение i на единицу и возврат.

Программа 1

Необходимо напечатать слово “Привет” 20 раз.

```
Dim i As Integer
Private Sub command1_click()
For i = 1 To 20
Debug.Print “Привет”
Next i
End Sub
```

2) Когда шаг - произвольный
   For i=1 To n Step k
   операторы
   Next i
Здесь k - произвольный шаг.

Программа 2

Необходимо напечатать квадраты чисел от нуля до единицы с шагом 0,01

```
Dim a As Double
Private Sub command1_click()
```

```
For a = 0 To 1 Step 0.01
Debug.Print a, a*a
Next a
End Sub
```

Счётчики и сумматоры. Вложенные циклы

Сумматор - это переменная величина в которой подсчитывается сумма чего-либо. Отличие сумматора от счётчика в том, что счётчик всегда увеличивается на единицу, а сумматор на какое либо число.

Пример 1

В компьютер вводится N чисел. Вычислить и напечатать их сумму.

```
Dim s, a As Double
Dim i, N As Integer
Private Sub Summa_click()
N = InputBox("Сколько чисел будем складывать ?")
s = 0
For i = 1 To N
a = InputBox("Введите очередное число")
s = s + a
Next i
Debug.Print "Сумма равна:", s
End Sub
```

Вложенные циклы - под этим термином подразумевается конструкция “цикл внутри цикла”.

Пример 2

Напечатать таблицу умножения от 2 до 9.

```
Dim a, b, p As Double
Private Sub Tabl_click()
For a = 2 To 9
For b = 2 To 9
p = a * b
Debug.Print a; "*"; b; "="; p
Next b
Debug.Print
Next a
End Sub
```

Задания

1. Дано натуральное n. Вычислить:

а) $2n$;

б) $n!$;

в) $\frac{1}{\sin 1}+\frac{1}{\sin 1+\sin 2}+\ldots+\frac{1}{\sin 1+\ldots+\sin n}$

г) $\left(1+\frac{1}{1^2}\right)\cdot\left(1+\frac{1}{2^2}\right)\cdot\ldots\cdot\left(1+\frac{1}{n^2}\right)$;

д) $\frac{\cos 1}{\sin 1}\cdot\frac{\cos 1-\cos 2}{\sin 1-\sin 2}+\ldots+\frac{\cos 1+\cos n}{\sin 1+\ldots+\sin n}$

2. Даны действительное число a, натуральное число n. Вычислить:

а) a^n;

б) $a\times (a + 1) \times \ldots \times (a + n - 1)$;

в) $\frac{1}{a} - \frac{1}{a\cdot(a+1)} + \ldots + \frac{1}{a\cdot(a-1)\cdot\ldots\cdot(a+n)}$

г) $a\times (a - n) \times (a - 2n) \times\ldots\times (a - n^2)$;

3. Вычислить:

$(1 + \sin 0{,}1) + (1 + \sin 0{,}2) + \ldots + (1 + \sin 10)$.

4. Дано действительное a. Найти:

а) среди чисел $1, 1+\frac{1}{2}, 1+\frac{1}{2}+\frac{1}{3}, \ldots$ первое, большее a;

б) такое наименьшее n, что $1+\frac{1}{2}+\ldots+\frac{1}{n} > a$.

5. Даны натуральное n, действительное x. Вычислить:

а) $\sin x + \sin^2 x + \ldots + \sin^n x$;

б) $\sin x + \sin x^2 + \ldots + \sin x^n$;

в) $\sin x + \sin \sin x + \ldots + \sin \sin \sin x + \sin \sin \ldots \sin x$.

6. Дано натуральное число n.

а) Сколько цифр в числе n?

б) Чему равна сумма его цифр?

в) Найти первую цифру числа n?

7. Даны натуральные числа n, m. Получить сумму m последних цифр числа n.

8. Дано натуральное число n. Выяснить, входит ли цифра 3 в запись числа n^2.

9. Даны натуральное число n, действительное число x. Вычислить:

а) $\left(\frac{x}{2}\right)^n$; б) $\frac{(x^n)^n}{2^n}$; в) $\frac{(x^2)^n}{2^{2n}}$;

10. Дано целое число m >. Получить наибольшее целое k, при котором $4^k < m$.

Лабораторная работа №5

Массивы. Переменные с индексами. Одномерные массивы переменных величин.

Массивы - одно из главных средств хранения в памяти компьютера больших объёмов информации. В основе массивов лежит понятие индекса.

Переменные с индексами

В математике широко применяются так называемые индексированные переменные. Эти индексированные переменные в Visual Basic, например, обозначаются как:

X(1) X(2) B(8) Y(i) Y(i-6) Z(i, j) Z(i+1, j)

Зачем математикам нужны индексированные переменные? Их удобно применять хотя бы при операциях над числовыми рядами. Числовой ряд - это просто несколько чисел, выстроенных по порядку одно за другим. Чисел в ряду может быть много и даже бесконечно много.

Возьмем, например, бесконечный ряд чисел Фибоначчи: 11235 8 13 21 34... Попробуйте догадаться, по какому закону образуются эти числа. Если не догадались, то я подскажу: каждое из чисел, начиная с третьего, является суммой двух предыдущих. А теперь попробуем записать это утверждение с помощью языка математики. Для этого обозначим каждое из чисел Фибоначчи индексированной переменной таким образом:

первое число Фибоначчи обозначим так: f(1);

второе число Фибоначчи обозначим так: f(2) и т. д. Тогда можно записать, что f(1)=1 f(2)=1 f(3)=2 f(4)=3 f(5)=5 f(6)=8... Очевидно, что

f(3)=f(1)+f(2), .

f(4)=f(2)+f(3), .

f(5)=f(3)+f(4) и т.д.

Как математически одной формулой записать тот факт, что каждое из чисел является суммой двух предыдущих? Математики в индексном виде записывают это так:

Для пояснения подставим вместо i любое число, например 6. Тогда получится:

f(6)=f(6-2)+f(6-1) или f(6)=f(4)+f(5).

Вот еще примеры, когда математики предпочитают использовать индексы. Пусть мы на протяжении года каждый день раз в сутки измеряли Температуру за окном. Тогда вполне естественно обозначить через t(l) температуру первого дня года, t(2) - второго, ..., t(365) - последнего. Пусть 35 спортсменов прыгали в высоту. Тогда через h(l) можно обозначить высоту, взятую первым прыгуном, h(2) - вторым и т. д.

Одномерные массивы переменных величин

Одна из типичных задач программирования формулируется примерно так. Имеется большое количество данных, например тех же температур или высот. С этими данными компьютер должен что-нибудь сделать, например вычислить среднегодовую температуру, количество морозных дней, максимально взятую высоту и т. п. Раньше мы уже вычисляли подобные вещи, и при этом данные вводили в компьютер с клавиатуры одно за другим в одну и ту же ячейку памяти. Однако программистская практика показывает, что удобно, а часто и необходимо иметь данные в оперативной памяти сразу все, а не по очереди. Тогда для задачи про температуру нам понадобится 365 ячеек. Эти 365 ячеек мы и назовем массивом. Итак, массивом можно назвать ряд ячеек памяти, отведенных для хранения значений индексированной переменной.

Рассмотрим на простом примере, как Visual Basic управляется с массивами.

Предположим, в зоопарке живут 3 удава. Известна длина каждого удава в сантиметрах (500, 400 и 600). Какая длина получится у трех удавов, вытянутых в линию?

Обозначим длину первого удава - dlina(1), второго - dlina(2), третьего - dlina(3). Прикажем Visual Basic отвести под эту индексированную переменную массив ячеек в памяти:

Dim dlina (1 To 3) As Integer

Здесь 1 - нижняя граница индекса, 3 - верхняя граница индекса.

Слово To обозначает до. В целом эту строку можно перевести так: Отвести в памяти под переменную dlina ряд ячеек типа Integer, пронумерованных от 1 до 3. Мы видим, что значением переменной величины является не одно число, а целый ряд чисел. Вот программа полностью:

```
Dim dlina(1 To 3) As Integer
Dim summa As Integer
Private Sub Command1_Click() dlina(1) = 500 dlina(2) = 400
dlina(3) = 600
summa = dlina(1) + dlina(2) + dlina(3)
Debug.Print summa
End Sub
```

Задания

Использование одномерных массивов

1. Даны натуральное число n, массив A[n]. Вычислить:

а) $a_1 + \ldots + a_n$; б) $a_1 \times \ldots \times a_n$; в) $\sin |a_1 + \ldots + a_n|$; г) $|a_1| \times |a_2| \times \ldots \times |a_n|$;
д) $a_1 + \ldots + a_n$ и $a_1 \times a_2 \times \ldots \times a_n$; е) $a_1, a_1+a_2, \ldots, a_1+a_2+ \ldots + a_n$;
ж) $a_1 \times a_1, a_1 \times a_2, a_1 \times a_3, \ldots, a_1 \times a_n$; з) $|a_1|, |a_1 + a_2|, |a_1 + \ldots + a_n|$;
и) $-a_1, a_2, -a_3, \ldots, (-1)^n \times a_n$.

2. Дано натуральное число n. Получить последовательность $b_1, \ldots, b_n$, где при i = 1, 2, …, n значение b_i равно:

а) i ; б) i^2 ; в) i! ; г) 2^{i+1} ; д) $2^i + 3^{i+1}$; е) $(-1)^i \times 3^i$; ж) 1+1/2+…1/i; з) 2^i /i!.

3. Вычислить значения многочлена $x^3 - 9x^2 + 3{,}2x - 7{,}5$ для x = 0, 1, …, 5.

4. Даны натуральное число n, действительные числа a, b (a¹b). Получить $r_1, r_2, \ldots, r_n$, где $r_i = a + ih$, h = (b - a) / n.

5. Получить таблицу температур по Цельсию от 0 до 100 градусов и их эквивалентов по шкале Фаренгейта, используя для перевода формулу

$$t_F = \frac{9}{5} t_C + 32$$

.

6. Вычислить значения функции $y = 5x^3 - 3x^2 + 5$ для значений x, изменяющихся -3 … 1, с шагом 0,1.

7. Даны натуральные числа i, n, действительные числа $a_1, \ldots, a_n$ (i £ n). Найти среднее арифметическое всех чисел $a_1, \ldots, a_n$, кроме a_i.

8. Даны действительные числа $a_1, a_2, \ldots$ Известно, что $a_1>0$ и что среди $a_2, a_3, \ldots$ есть хотя бы одно отрицательное число. Пусть $a_1, \ldots, a_n$ – члены данной последовательности, предшествующие первому отрицательному члену (n заранее неизвестно). Получить:

а) $a_1 + a_2 + \ldots + a_n$; б) среднее арифметическое $a_1, \ldots, a_n$;
в) $a_1, a_1a_2, a_1a_2a_3, \ldots, a_1a_2 \ldots a_n$; г) $a_1a_2 + a_2a_3 + \ldots + a_{n-1}a_n + a_na_1$;
д) $(-1)^n a_n$; е) $n + a_n$;
ж) $|a_1 - a_n|$.

9. Даны натуральное число n, действительные числа $a_1, \ldots, a_n$. Получить числа $b_1, \ldots, b_n$, которые связаны с $a_1, \ldots, a_n$ следующим образом:

$b_1 = a_1$, $b_n = a_n$, $b_i = (a_{i+1} - a_i)/3$, i=2, …, n-1.

10. Даны натуральные числа x, $y_1, \ldots, y_{100}$. ($y_1 < y_2 \ldots < y_{100}$, $y_1 < x < y_{100}$). Найти натуральное k, при котором $y_{k-1} < x$ £ y_k.

Использование двумерных массивов

1. Даны целые числа a_1, a_2, a_3. Получить целочисленную матрицу B[3x3], для которой $b_{ij} = a_i - 3a_j$.

2. Даны действительные числа $a_1, \ldots, a_{10}, b_1, \ldots, b_{20}$. Получить действительную матрицу C[20x10], для которой $c_{ij} = a_j / (1 + |b_i|)$.

3. Получить A[10x12] – целочисленную матрицу, для которой

$$a_{ij} = i + 2j.$$

4. Даны натуральное число n, действительная матрица размера nx9. Найти среднее арифметическое:

а) каждого из столбцов;
б) каждого из столбцов, имеющих четные номера.

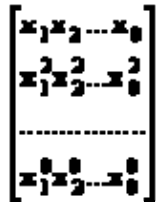
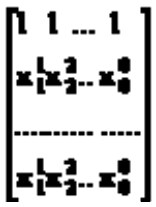
5. Дано натуральное число n. Выяснить, сколько положительных элементов содержит матрица A[nxn], если

а) $a_{ij} = \sin (i + j/2)$;
б) $a_{ij} = \cos (i^2 + n)$;
в) $a_{ij} = \sin (i^2 – j^2/n)$.

6. Дана действительная матрица размера n x m, в которой не все элементы равны нулю. Получить новую матрицу путем деления всех элементов данной матрицы на ее наибольший по модулю элемент.

7. Дана действительная квадратная матрица порядка 12. Заменить нулями все ее элементы, расположенные на главной диагонали и выше нее.

8. Даны действительные числа $x_1, \ldots, x_8$. Получить действительную квадратную матрицу порядка 8:

а)  б)  в) 

9. Дана действительная матрица размера nxm. Определить числа $b_1, \ldots, b_n$, равные соответственно:

а) суммам элементов строк;
б) произведениям элементов строк;
в) наименьшим значениям элементов строк;
г) значениям средних арифметических элементов строк;
д) разностям наибольших и наименьших значений элементов строк.

10. Даны натуральное число n, действительная матрица A[nxn]. Получить последовательность элементов главной диагонали $a_{11}, a_{22}, \ldots, a_{nn}$.

Лабораторная работа №6

Процедуры. Взаимодействие процедур в программе.

В изложенном ранее материале мы в каждой программе всегда использовали только одну процедуру Private Sub Название_Click (). Теперь рассмотрим случай использования в одной программе нескольких процедур. Возьмём задачу на поиск max, которую мы рассматривали на прошлом занятии:
Найти максимальное из вводимых в компьютер чисел.
Первоначально программа выглядела так:

```
Dim max, Ch As Double
Dim i, n As Integer
Private Sub VMax_click()
max = InputBox ("Введите предполагаемый максимум")
n = InputBox ("Введите количество рассматриваемых чисел")
For i = 1 To n
Ch = InputBox ("Введите число", i)
If Ch > max Then max = Ch
Next i
Debug.Print "Максимум равен "; max
End Sub
```

Теперь разобьём программу на последовательность процедур:

```
Dim max, Ch As Double
Dim i, n As Integer
Private Sub VMax_click()
:VVOD: USL: PECH:
End Sub
Private Sub VVOD()
max = InputBox("Введите предполагаемый максимум")
n = InputBox("Введите количество рассматриваемых чисел")
End Sub
Private Sub USL()
For i = 1 To n
Ch = InputBox("Введите число", i)
If Ch > max Then max = Ch
Next i
End Sub
Private Sub PECH()
Debug.Print "Максимум равен "; max
End Sub
```

Вышеизложенная методика, используемая для программ содержащих большое количество операций, позволяет оптимизировать размер программы и упростить процесс её написания.

Задание

Разбить следующие программы на последовательность процедур:

1)
```
Dim a, b, d As Double
  Private Sub Command1_click()
  a = InputBox("первое число")
  b = InputBox("второе число")
  y = MsgBox("Вы уверены что хотите их перемножить ? ", vbCritical + vbYesNo,
  "Вопрос")
  If y = VbYes Then d=a*b else d=a+b
  Debug.Print d
  End Sub
```

2)
```
Dim Slovo As String
  Private Sub Command1_Click()
  Do
  Slovo = InputBox("Введите слово")
  Debug.Print Slovo; "!"
  Loop Until Slovo = "Хватит"
  Debug.Print "Хватит так хватит"
  End Sub
```

Задания

Построение процедур и функций

1. Даны действительные числа s, t. Получить f(t,-3s,3.22) + f(5.2,t,s-t),

где f (a,b,c) = $\dfrac{2a - b - \sin c}{5 + |c}$

2. Даны действительные числа s, t. Получить g(1.2,s) + g(t,s) - g(2s-1,st),

где $g(a,b) = \frac{a^3 + b^3}{a^3 + 2ab + 3b^3 + 4}$

3. Даны действительные числа a, h, натуральное число n. Вычислить

$$f(a) + 2f(a + h) + \ldots + 2f(a + (n-1)h) + f(a + nh),$$

где $f(x) = (x^2 + 1)\cos^2 x$.

4. Даны действительные числа a, b, c. Получить

$$\frac{\max(a, a+b) + \max(a, b+c)}{1 + \max(a+bc, \delta 23)}$$

5. Даны действительные числа a, b. Получить

$$u = \min(a, b),\ v = \min(ab, a + b),\ x = \min(u^2 + v, 3.14)$$

6. Даны действительные числа s, t. Получить

$$h(s, t) + \max(h^2(s-t, st), h^3(s-t, s+t)) + h(1,1),$$

где

$h(a,b) = \frac{a}{1+b^2} + \frac{b}{1+a^2} - (a-b)^2$.

7. Даны действительные числа $x_1, y_1, x_2, y_2, \ldots, x_{10}, y_{10}$. Найти периметр десятиугольника, вершины которого имеют соответственно координаты $(x_1, y_1), (x_2, y_2), \ldots, (x_{10}, y_{10})$. (Определить функцию вычисления расстояния между двумя точками, заданными своими координатами).

8. Даны натуральное число n, действительные числа $x_1, y_1, x_2, y_2, \ldots, x_n, y_n$. Найти площадь n-угольника, вершины которого при некотором последовательном обходе имеют координаты $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$. (Определить функцию вычисления площади треугольника по координатам его вершин.)

9. Составить функцию, позволяющую определить позицию самого правого вхождения заданного символа в исходную строку. Если строка не содержит символа, результатом работы процедуры должна быть -1.

10. Составить процедуру, изменяющую в исходной строке символов все единицы нулями и все нули единицами. Замена должна выполняться, начиная с заданной позиции строки.

Список использованной литературы

1. Росс Нельсон, Running Visual Basic 3 for Windows, пер. с англ. – М.: Издательский отдел «Русская Редакция» ТОО «Channel Trading Ltd.», 1995
2. Кауэлл Дж.Visual Basic 4.0: просто - о самом существенном, Пер. с англ. - М. :ИНФРА-М, 1998.
3. Волченков Н.Г. Учимся программировать: Visual Basic 5, - М.:»Диалог-МИФИ», 1998
4. Волченков Н.Г, Программирование на Visual Basic 6 в 3-х ч., М.:ИНФРА-М, 2000
5. Visual Basic 6.0, Руководство для профессионалов, пер. с англ.- СПб.:БХВ – Санкт-Петербург, 1999