ЛАБОРАТОРНАЯ РАБОТА № 3

Тема: Переменные. Операторы: перехода по условию, выбора, логические. Математические функции.

Переменные в Visual Basic

Переменная – это именованный участок памяти, в котором хранится значение, которое может быть изменено программой. Каждая переменная имеет *своё имя*. Оно может достигать 255 символов в длину, начинается всегда с буквы латинского алфавита, за которой могут следовать другие буквы, цифры и знак подчёркивания. Регистр символов значения не имеет. Нельзя использовать в качестве имен переменных ключевые слова *Visual Basic*, названия объектов или свойств. Каждая переменная имеет определённый тип. Кроме того, программист может определить и свой тип. В табл. 2 приведены основные типы данных в *Visual Basic*.

Таблица 2

Основные типы данных в Visual Basic

Тип данных	Размер
Byte (Байт)	1 байт
Integer (Целое)	2 байта
Long (Длинное целое)	4 байта
Single (Одинарной точности с плавающей десятичной точкой)	4 байта
Double (Двойной точности с плавающей десятичной точкой)	8 байт
Currency (Денежные единицы)	8 байт
String (Строка)	1 байт на символ
Boolean (Логический)	2 байта
Date (Дата)	8 байт
Variant (Вариант)	16 байт плюс 1 байт на каждый символ строковых значений

Объявление переменной

В *Visual Basic* используется *явное* и *неявное* объявление переменной.

Явное объявление осуществляется оператором ***Dim*** (от dimension – размерность). Синтаксис этого оператора следующий:

Dim *ИмяПеременной* ***As*** *ТипПеременной*

Например: *Dim a As* Long; *Dim* b *As* Byte; *Dim* myStr *As* String; *Dim* x *As* Boolean.

Если не указывать тип переменной, то она будет объявлена как *Variant*, которая может хранить данные любого размера или формата. Такие типы переменных допускают значительную гибкость использования.

Вы можете объявить переменную и без использования оператора *Dim*: этот процесс называется *неявным объявлением*. В этом случае вы просто используете переменную, не выделяя специально под нее память с помощью оператора *Dim*. Неявное объявление более удобно в том смысле, что вам не нужно тратить время на ввод оператора *Dim* для переменной, но с точки зрения управления ресурсами программы, это не всегда хорошо, поскольку даёт меньше возможностей контролировать распределение памяти для переменных.

После объявления переменной ей присваивается значение по умолчанию: *для строки это – " "* (*пустая строка*); *для чисел* – 0; *для Boolean – False*.

Значение переменной можно изменять с помощью *оператора присваивания* (=). Слева от знака равенства должна находиться переменная, а справа – присваиваемое значение или выражение (формула). Например: a = 3456; b = 25; a = a + b; myStr = "VB"; x = True.

Присвоение – это действие, заключающееся в том, что значение правой части записывается в ячейку памяти, отведенную для хранения значения переменной, находящейся слева от знака равенства в *операторе присваивания*.

Область видимости переменной

В *Visual Basic* есть три вида областей видимости, характеризующих доступность переменных:

- ✓ *локальная*;
- ✓ *контейнера*;
- ✓ *глобальная*.

Локальные переменные. Локальными являются переменные, объявленные внутри процедуры или функции. Они доступны только внутри этой процедуры или функции.

Переменные контейнера. Переменные контейнера определяются в главной секции (*General*) и доступны только внутри соответствующего контейнера, т.е. формы, или модуля.

Глобальные переменные. Глобальные переменные определяются в главной секции (*General*) модуля. При этом вместо оператора *Dim* используется зарезервированное слово *Public*. Глобальные переменные доступны во всех модулях и формах проекта.

Ввод и вывод данных с помощью стандартных функций

Ввод данных. Входные данные можно ввести при помощи функции ввода *InputBox*. Синтаксис функции:

Pm = ***InputBox***(prompt, title) ,

где Pm – возвращаемое значение функции; prompt – приглашение к вводу; title – строка заголовка.

Функция *InputBox* вызывает диалоговое окно на экране, в котором вы можете ввести текст и присвоить его переменной.

Упражнение 1. Ввод данных с помощью функции *InputBox*.

1. Создайте на форме проекта одно поле *Label*1 и две командные кнопки *Command*1 и *Command*2.
2. Измените свойство *Caption* кнопок *Command*1 и *Command*2 на следующие значения: **Поле ввода** и **Выход** соответственно.
3. Дважды щелкните на командной кнопке **Поле ввода**. В окне *Code* будет отображена процедура Command1_Click. Введите следующие операторы для описания двух переменных и вызова функции *InputBox*.

```
Dim Prompt, FullName
Prompt = "Введите, пожалуйста, свое имя"
FullName = InputBox(Prompt):   Label1.Caption = FullName
```

4. Дважды щелкните на командной кнопке **Выход**. В окне *Code* будет отображена процедура Command2_Click. Введите оператор End для выхода из программы.
5. Запустите программу на выполнение.
6. Сохраните проект под именем MyInputBox.

Вывод данных. Для вывода различных сообщений в *Visual Basic* имеется окно, которое отображается функцией вывода *MsgBox*. Синтаксис функции:

Pm = ***MsgBox*** prompt [,type] [,title] ,

где prompt – текст сообщения; type – номер кнопки (от 1 до 5); title – строка заголовка.

Переменной Pm присваивается результат, возвращаемый функцией, который указывает, какая из кнопок диалогового окна была выбрана. Если вам требуется лишь отобразить сообщение, то вы можете обойтись без оператора присвоения (=), переменной Pm и аргумента type.

Упражнение 2.

В предыдущую программу MyInputBox (упражнение 1) добавьте функцию *MsgBox*, чтобы отобразить имя пользователя, введенное в диалоговом окне InputBox.

Отобразим сообщение с помощью функции M*sgBox*. Для этого в форме созданной программы MyInputBox, в процедуре Command1_Click удалите оператор Label1.Caption = FullName и введите:

MsgBox (FullName), , "Введенные данные"

Этот оператор осуществит вызов функции *MsgBox*, отобразит содержимое переменной FullName в диалоговом окне и поместит строку "Введенные данные" в заголовок окна. Аргумент type и переменная Pm не используются.

Примечание. Если переменная Pm не используется, в скобки заключается только первый аргумент.

Работа с основными математическими операторами Visual Basic

Формула представляет собой оператор, содержащий числа, переменные, операции и ключевые слова или же комбинацию этих элементов, и создающий новое значение. В табл. 3 приведены математические символы, используемые в операторах *Visual Basic*.

Таблица 3

Математические символы, используемые в операторах *Visual Basic*

Операция Математическое действие

1	**2**
+	Сложение
–	Вычитание
*	Умножение
/	Деление
\	Целая часть от деления
Mod	Остаток от деления
^	Возведение в степень
&	Слияние (конкатенация) строк

Математические функции Visual Basic

Математические функции возвращают программе значение. В табл. 4 приведены некоторые математические функции, где аргумент **n** соответствует числу, переменной или выражению, которое используется в качестве входного значения функции.

Таблица 4

Математические функции

Функция	Действие
Abs(n)	Возвращает абсолютное значение **n**.

Atn(n)	Возвращает арктангенс **n**, в радианах.
Cos(n)	Возвращает косинус угла **n**. Угол n выражен в радианах .
Exp(n)	Возвращает константу **E** в степени **n**.
Rnd(n)	Генерирует случайное число между 0 и 1
Sgn(n)	Возвращает -1, если **n** меньше 0; 0, если **n** равно 0; и +1, если **n** больше 0
Sin(n)	Возвращает синус угла **n**. Угол **n** выражен в радианах.
Sqr(n)	Возвращает квадратный корень **n**.
Str(n)	Преобразует числовое значение в строку.
Tan(n)	Возвращает тангенс угла **n**. Угол **n** выражен в радианах.
Val(n)	Преобразует строку в числовое значение.

Упражнение 3. Разработайте проект вычисления арифметического выражения $\frac{\sin x - 2.4x}{4.8}$, где $x = 1.8$.

1. Создайте на форме проекта две командные кнопки *Command*1 и *Command*2.

2. Измените свойство *Caption* кнопок *Command*1 и *Command*2 на следующие значения: **Вычислить** и **Выход** соответственно.

3. Дважды щелкните на командной кнопке **Вычислить**. В окне *Code* будет отображена процедура Command1_Click. Введите следующие операторы:

```
Dim X As Single, Y As Single
X=1.8
Y= (Sin(X) – 2.4*X)/48
MsgBox "Y = " & Y
```

4. Дважды щелкните на командной кнопке **Выход**. В окне *Code* будет отображена процедура Command2_Click. Введите оператор End для выхода из программы.

5. Запустите программу на выполнение.

Использование выражений с условиями

Операторы в программе выполняются в той последовательности, в которой они записаны. Однако достаточно часто требуется изменить порядок выполнения операторов в зависимости от выполнения (или невыполнения) определенного условия. В *Visual Basic*, как и во всех языках программирования, существуют управляющие конструкции, предназначенные для управления порядком выполнения команд.

Различают три типа управляющих операторов, позволяющих программировать разветвляющиеся алгоритмы:

1. ***If*** ... ***Then*** ...
2. ***If*** ... ***Then*** ... ***Else*** ...
3. ***Select*** ... ***Case*** ...

Основанием для принятия решений в управляющих конструкциях являются логические (условные) выражения.

Логические выражения – это такие выражения, которые возвращают одно из двух значений: *True* (*Истина*) или *False* (*Ложь*). Логические выражения содержат *логические отношения* (*операции сравнения*): = (равно), > (больше), < (меньше), <> (не равно), >=

(больше или равно), <= (меньше или равно). Например, условное выражение Price < 100 имеет значение *True* (*Истина*), если переменная Price содержит значение, меньшее 100, или имеет значение *False* (*Ложь*), если переменная Price содержит значение большее или равное 100.

Логические отношения могут быть связаны *логическими операциями*:

AND(*И*) реализует конъюнкцию логических значений – возвращает значение *True* (*Истина*), если все участвующие в операции выражения имеют значение *True*. В остальных случаях возвращается значение *False* (*Ложь*);

OR(*ИЛИ*) реализует дизъюнкцию логических значений – возвращает значение *True*, если хотя бы одно из участвующих в операции выражений имеет значение *True*. В случае, когда все выражения имеют значение *False*, возвращается значение *False*;

XOR(*Исключающее ИЛИ*) — возвращает значение *True* (*Истина*), если только одно из участвующих в операции выражений имеет значение *True*. В остальных случаях возвращается значение *False*;

NOT(*НЕ*) реализует инверсию логических значений – операция отрицания. Возвращает обратное для значения выражения значение, то есть если выражение равно *True*, то возвращается *False* и наоборот, если значение выражения равно *False*, то возвращается значение *True*.

Конструкция с условием *If* ... ***Then*** позволяет анализировать логические значения выражений и выполнять определенные действия в зависимости от результата. В простейшем случае структура с условием *If* ...*Then* записывается в виде одной строки:

If условие ***Then*** оператор

Конструкция с условием *If* ...***Then*** ... ***Else*** ... может содержать несколько логических выражений. Такой блок операторов может состоять из нескольких строк и содержать ключевые слова: *ElseIf*, *Else*, *End If*. Синтаксис структуры следующий:

```
If условие1 Then
    операторы, выполняемые если условие1 истинно
ElseIf условие2 Then
   операторы, выполняемые если условие2 истинно
   [Здесь могут находиться дополнительные блоки ElseIf и
    операторы]
Else
   операторы, выполняемые если ни одно из условий не
   является истинным
End If
```

Упражнение 4. Разработайте проект вычисления функции, используя структуру *If* ...*Then* ... *Else*:

$$Y=\begin{cases}\frac{1}{x}+x^3+\sqrt{x^4}, & \text{при} \quad x<-1;\\ x^4-x^3+x^2, & \text{при} \quad -1\le x\le 1;\\ \sqrt{x^2+1}+1, & \text{при} \quad x>1.\end{cases}$$

1. Создайте на форме проекта две командные кнопки *Command*1 и *Command*2.

2. Измените свойство *Caption* кнопок *Command*1 и *Command*2 на следующие значения: **Вычислить** и **Выход** соответственно.

3. Дважды щелкните на командной кнопке **Вычислить**. В окне *Code* будет отображена процедура Command1_Click. Введите следующие операторы:

```
Dim X As Single, Y As Single
X = InputBox("Введите значение  X", "Ввод  X")
If  X < –1 Then
 Y = 1 / X + X ^ 3 + Sqr(X ^ 4)
ElseIf  X > 1 Then
   Y = Sqr(X ^ 2 + 1) + 1
Else
   Y = X ^ 4 - X ^ 3 + X ^ 2
End If
MsgBox "Y = " & Y
```

4. Дважды щелкните на командной кнопке **Выход**. В окне *Code* будет отображена процедура Command2_Click. Введите оператор End для выхода из программы.

5. Запустите программу на выполнение.

Конструкция *Select* ... ***Case*** ... позволяет обрабатывать в программе несколько условий и аналогична блоку конструкций *If ...Then ... Else*. Эта конструкция состоит из анализируемого выражения и набора операторов *Case* на каждое возможное значение выражения. Работает эта конструкция следующим образом. Сначала *Visual Basic* вычисляет значение заданного в конструкции выражения. Затем полученное значение сравнивается со значениями, задаваемыми в операторах *Case* конструкции. Если найдено искомое значение, выполняются команды, приписанные данному оператору *Case*. После завершения выполнения конструкций управление будет передано конструкции, следующей за ключевым словом *End Select*. Синтаксис конструкции *Select Case* следующий:

Select Case сравниваемоеЗначение
Case значение1
Операторы, выполняемые при условии совпадения значения1 *и* сравниваемогоЗначения
Case значение2
Операторы, выполняемые при условии совпадения значения2 *и* сравниваемогоЗначения
. . .
Case Else
Операторы, выполняемые при условии ***несовпадения*** *со* значением1 *и* значением2
End Select

В приведенном ниже примере демонстрируется использование структуры *Select Case* для вывода программой сообщения о возрасте человека. Если значение переменной Age совпадает со значением в одном из операторов *Case*, в поле метки отображается соответствующее сообщение.

```
Select Case Age
  Case 14:   Labell .Caption = "Вам 14 лет."
  Case 16:    Labell .Caption = " Вам 16 лет."
  Case 18:    Label1 .Caption = " Вам 18 лет."
```

```
Case 21:    Labell.Caption = " Вам 21 год."
Case 60:    Label1.Caption = " Вам 60 лет."
End Select
```

Использование операций сравнения в структуре *Select … Case*. Структура *Select … Case*, как и структура *If … Then,* допускает использование операций сравнения. *Visual Basic* позволяет использовать операции сравнения, чтобы иметь возможность проверять несколько критериев в структуре *Select … Case*. В качестве операции сравнения в *Visual Basic* используются операторы: =, <>, >, <, >=, и <=. Чтобы использовать операции сравнения, нужно включить в выражение ключевое слово *Is* или ключевое слово *To.*

Ключевое слово *Is* используется для сравнения проверяемой переменной со значением выражения, следующего после слова *Is*. Ключевое слово ***To*** задает диапазон значений. В следующей структуре используются ключевые слова *Is*, *To* для проверки переменной Age и отображения одного из трёх сообщений:

```
Select Case Age
Case Is < 13:      Label1 .Caption = "Радуйтесь детству!"
Case 13 To 19:     Labell.Caption = "Радуйтесь юности!"
Case Else:         Label1.Caption = "Прекрасный возраст."
End Select
```

Упражнение 5. Разработайте проект вычисления функций (используя структуру *Select … Case*):

При n=1 $Y = x^3 - 7x^2$, x=1.2

При n=2 $Y = 3x^2 + 7$, x=4

При n=3 Y =x + 1.25 , x=1.5

1. Создайте на форме проекта две командные кнопки *Command*1 и *Command*2.

2. Измените свойство *Caption* кнопок *Command*1 и *Command*2 на следующие значения: **Вычислить** и **Выход** соответственно.

3. Дважды щелкните на командной кнопке **Вычислить**. В окне *Code* будет отображена процедура Command1_Click. Введите следующие операторы:

```
Dim X As Single, Y As Single
n = InputBox("Введите значение n", "Ввод n")
Select Case n
Case 1: X = 1.2: Y = X ^ 3 - 7 * X ^ 2
Case 2: X = 4: Y = 3 * X ^ 2 + 7
Case 3: X = 1.5: Y = X + 1.25
End Select
MsgBox "Y = " & Y
```

4. Дважды щелкните на командной кнопке **Выход**. В окне *Code* будет отображена процедура Command2_Click. Введите оператор End для выхода из программы.

5. Запустите программу на выполнение.

Упражнение 6. Создайте программу аналогичную калькулятору.

1. Создайте на форме следующие объекты: четыре элемента *Label*; два элемента *TextBox*; элемент *Frame*; четыре элемента *OptionButton*; две командные кнопки. Расположите их, как показано на рис. 10.

2. Для командной кнопки **Вычислить** введите программный код:

```
Dim First As Single, Second As Single  ' объявление переменных
First = Val(Text1.Text)  ' запись в перемен. First значения из Text1
Second = Val(Text2.Text) ' запись в перемен. Second знач. из Text2
   If Option1.Value = True Then         ' если выбрано сложение
      Label4.Caption = First + Second
   ElseIf Option2.Value = True Then   ' если выбрано вычитание
       Label4.Caption = First - Second
   ElseIf Option3.Value = True Then    ' если выбрано умножение
      Label4.Caption = First * Second
   ElseIf Option4.Value=True Then      ' если выбрано деление
      Label4.Caption = First / Second
   Else
      MsgBox "Оператор не выбран"
   End If
```

3. Для командной кнопки **Выход** введите единственный оператор End.
4. Запустите программу на выполнение.

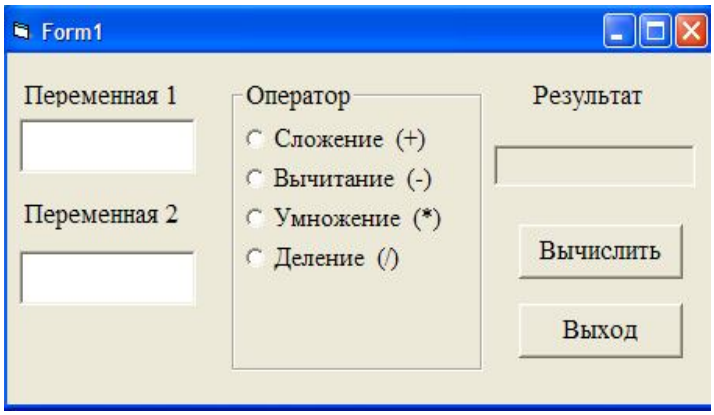

Рис. 10. Интерфейс программы из упражнения 6

Упражнение 7. Дополните программу, из упражнения 6, следующими операторами:

\	Целая часть от деления
Mod	Остаток от деления
^	Возведение в степень
&	Слияние (конкатенация) строк

Упражнение 8. Используя конструкцию *Select ... Case* , создайте программу, в которой по заданному номеру месяца, будет выводиться соответствующее время года.

Контрольные вопросы:

1. Дайте определение переменной.
2. Функция *InputBox*, ее назначение, синтаксис.
3. Функция *MsgBox*, ее назначение, синтаксис.
4. Что такое логическое выражение, операции сравнения?
5. Перечислите логические операции и их результаты.
6. Объясните структуру *If* **...** *Then* **...** и *If* **...** *Then* **...** *Else* **...** .
7. Структура *Select … Case* и её использование.