**встроенные константы visual basic.**

**Константы, объявление, ключевое слово Const, встроенные константы, vbCrLf**

*Константы* — еще один контейнер для хранения данных, но, в отличие от переменных, они не изменяются в ходе выполнения VBA-программы. Для чего нужны константы:

* код становится лучше читаемым/убираются потенциальные ошибки;
* чтобы изменить какое-либо значение, которое много раз используется в программе (например, уровень налога) — это можно сделать один раз.

В VBA константы определяются при помощи ключевого слова Const:

*Const COMP\_NAME As String = “Microsoft”*

При попытке в теле процедуры изменить значение константы будет выдано сообщение об ошибке.

Константы очень удобны при работе с группами именованных элементов (дни недели, месяцы, цвета, клавиши, типы окон и т.п.). Они позволяют использовать в коде программы легко читаемые обозначения вместо труднозапоминаемых числовых кодов. Например, строки

*UserForm1.BackColor = vbGreen*

и

*UserForm1.BackColor = 65280*

функционально одинаковы, но в чем смысл первой строки, догадаться гораздо легче.

В VBA встроено множество служебных констант: календарных, для работы с файлами, цветами, формами, типами дисков и т.п. Просмотреть их можно через справочную систему VBA: Microsoft Visual Basic Documentation -> Visual Basic Reference -> Constants. Про одну из констант (она находится в разделе Miscellaneous) следует сказать особо: константа vbCrLf позволяет произвести переход на новую строку. Например:

*MsgBox ("Первая строка" + vbCrLf + "Вторая строка")*

Множество наборов констант встроено в объектные модели, которые мы будем рассматривать в последних разделах этого курса.

***Проверка условий в VBA, оператор If...Then... Else, вложенные конструкции If***

*Операторы условного перехода* — одни из самых важных и часто используемых элементов в языках программирования. Общий принцип их работы прост: проверяется соответствие каким-то условиям (истинность или ложность каких-либо выражений) и в зависимости от этого выполнение программы направляется по одной или другой ветви. В VBA предусмотрено два оператора условного перехода: If… Then... Else и Select Case.

Оператор *If… Then*... Else  — самый популярный у программистов. Полный его синтаксис выглядит так:

*If Условие Then
Команды1*

*[ElseIf Условия N Then
Команды N]*

*[Else
Команды2]*

*End If*

При этом:

* *Условие* — выражение, которое проверяется на истинность. Если оно истинно, то выполняются Команды1, если ложно — Команды2;
* *УсловияN* — дополнительные условия, которые также можно проверить. В случае, если они выполняются (выражение УсловияN истинно), то выполняются КомандыN.

Оператор If…Then... Else применяется:

* когда нужно проверить на соответствие одному условию и в случае соответствия сделать какое-то действие:

*If nTemperature < 10 Then*

*MsgBox "Одеть куртку"*

*End If*

* когда нужно сделать то, же что и в предыдущем примере, а в случае несоответствия выполнить другое действие:

*If nTemperature < 10 Then*

*MsgBox "Одеть куртку"*

*Else*

*MsgBox "Одеть ветровку"*

*End If*

* когда нужно проверить на соответствие нескольким условиям (обратите внимание на использование логических операторов):

*If (nTemperature < 10) And (bRain = True) Then*

*MsgBox "Одеть куртку и взять зонтик"*

*End If*

* когда в случае, если первая проверка вернула False, нужно проверить на соответствие еще нескольким условиям (в этом случае удобно использовать ElseIf):

*If (bIGoInCar = True) Then*

*MsgBox "Одеться для машины"*

*ElseIf nTemperature < 10 Then*

*MsgBox "Одеть куртку"*

*Else*

*MsgBox "Можно идти в рубашке"*

*End If*

В этом примере, поскольку bIGoInCar — переменная типа Boolean и сама по себе принимает значения True или False, первая строка может выглядеть так:

*If bIGoInCar Then …*

Некоторые замечания по использованию If…Then... Else:

* ключевое слово Then должно находиться в одной строке с If и условием. Если вы перенесете его на следующую строку, будет выдано сообщение об ошибке;
* если разместить команду, которую нужно выполнить при истинности проверяемого условия, на одной строке с If и Then, то End If можно не писать:

*If nTemperature < 10 Then MsgBox "Одеть куртку"*

* если же вы используете несколько команд или конструкции Else/ElseIf, то End If в конце нужно писать обязательно — иначе возникнет синтаксическая ошибка.
* для выражения If…Then настоятельно рекомендуется использовать отступы для выделения блоков команд. Иначе читать код будет трудно.
* операторы If…Then можно вкладывать друг в друга:

*If MyVar = 5 Then*

*MsgBox “MyVar = 5”*

*If MyVar = 10 Then*

*MsgBox “MyVar = 10”*

*End If*

*End If*

***Проверка условий в VBA, оператор Select Case***

Оператор *Select Case* идеально подходят для проверки одного и того же значения, которое нужно много раз сравнить с разными выражениями. Синтаксис его очень прост:

*Select Case sDayOfWeek*

*Case "Понедельник"*

*MsgBox "Салат из шпината"*

*Case "Вторник"*

*MsgBox "Салат из морской капусты"*

*…*

*Case Else*

*MsgBox "На этот день у нас ничего не предусмотрено"*

*End Select*

Некоторые замечания по поводу Select Case:

* строка Case "Понедельник" на самом деле означает Case sDayOfWeek = "Понедельник", просто такое равенство подразумевается по умолчанию. Но вам ничего не мешает использовать другой оператор сравнения или целый набор таких операторов:

*Case 0 To 5, 15, Is > 55*

*MsgBox "Напомнить о прививках"*

* Слово Is при этом можно пропустить — компилятор VBA добавит это ключевое слово за Вас. Несколько критериев в Case объединяются аналогично операторы OR — то есть выполнение пойдет по этой ветви, если тестируемое значение будет удовлетворять хотя бы одному из критериев. Критерии для сравнения разделяются запятыми.
* при использовании диапазона (0 To 5) включаются и границы диапазона (в данном случае 0 и 5).

 ***Оператор GoTo в VBA, ситуации применения GoTo***

Оператор GoTo — это оператор безусловного перехода, когда ход выполнения программы без проверки каких-либо условий перепрыгивает на метку в коде. Пример применения GoTo может выглядеть так:

*GoTo EngineNotStarted*

*…*

*EngineNotStarted :*

*MsgBox "Едем на метро"*

*…*

EngineNotStarted: — это метка, для нее используется имя (выбираемое по правилам назначения имен для переменных), которое оканчивается на двоеточие.

Иногда использование GoTo очень удобно — например, когда нам нужно добиваться от пользователя ввода правильного значения неизвестное число раз. Однако использование GoTo категорически не рекомендуется, потому что код становится трудночитаемым. Чаще всего GoTo можно заменить на конструкцию Do While или на вызов функции из самой себя.

**Задание для самостоятельной работы 3.2: Работа с операторами условного перехода**

**Подготовка:**

* Создайте новую книгу Excel и сохраните ее как C:\LabCondConstructions.xls.
* Откройте редактор Visual Basic в Excel и создайте в этой книге новый стандартный модуль Module1.
* Введите в этом стандартном модуле следующий код:

*Public Sub IfThenSub()*

*Dim nResult As Integer*

*nResult = MsgBox("Нажмите кнопку", vbYesNo, "Окно сообщения")*

*ThisWorkbook.Worksheets(1).Range("A1").Value = "Вы нажали кнопку: " & nResult*

*ThisWorkbook.Worksheets(1).Range("A1").Columns.AutoFit*

*End Sub*

* Запустите этот код на выполнение и убедитесь, что он выполняется без ошибок. Этот код должен вставлять в ячейку A1 первого листа вашей книги текстовое значение вида "Вы нажали кнопку: 6" в зависимости от того, какая кнопка была нажата в окне сообщения.

**Задание 1:**

Измените код этой процедуры таким образом, чтобы вместо чисел в ячейку прописывалось строковое значение нажатой кнопки (например, "Вы нажали кнопку: Повтор"). Используйте при этом синтаксическую конструкцию IF…THEN…ELSE.

**Задание 2:**

Замените в вашей процедуре строку

*nResult = MsgBox ("Нажмите кнопку", vbYesNo , "Окно сообщения")*

на

*nResult = MsgBox ("Нажмите кнопку", vbAbortRetryIgnore , "Окно сообщения")*

Измените вашу процедуру таким образом, чтобы она вставляла в ячейку A1 значения "Отменить", "Повторить" или "Пропустить" в зависимости от того, какая кнопка была нажата в окне сообщения. Используйте при этом синтаксическую конструкцию Select…Case.

**ЯЗЫК ПРОГРАММИРОВАНИЯ VISUAL BASIC. ИСПОЛЬЗОВАНИЕ СТАНДАРТНЫХ ДИАЛОГОВЫХ ОКОН ВВОДА И ВЫВОДА ИНФОРМАЦИИ**

Стандартное диалоговое окно InputBox используется для ввода небольших фрагментов текста. Окно InputBox состоит из четырех элементов:

* строка заголовка;
* приглашение к вводу;
* поле ввода со значением, предлагаемым по умолчанию;
* две кнопки (ОК и Cancel).

Функция InputBox выводит на экран диалоговое окно, содержащее сообщение и поле ввода, устанавливает режим ожидания ввода текста пользователем или нажатия кнопки, а затем возвращает значение типа String, содержащее текст, введенный в окне.

Функция имеет следующий синтаксис:

InputBox(приглашение[, заголовок] [, умолчание] [, Хпоз][Yпоз][,файл справки, контекст]).

Приглашение — строковое выражение, отображаемое как сообщение в диалоговом окне. Приглашение служит подсказкой пользователю, какую информацию он должен ввести в специальное поле ввода, находящееся в окне.

Заголовок — строковое выражение, отображаемое в заголовке диалогового окна. Если заголовок опущен, в строку заголовка помещается имя приложения.

Умолчание — строковое выражение, отображаемое в поле ввода и используемое по умолчанию, если пользователь не введет другой строки. Если этот параметр опущен, поле ввода изображается пустым.

Хпоз — числовое выражение, задающее расстояние по горизонтали между левой границей диалогового окна и левым краем экрана (в твипах). Если этот параметр опущен, то диалоговое окно выравнивается по центру экрана по горизонтали.

Yпоз — числовое выражение, задающее расстояние по вертикали между верхней границей диалогового окна и краем экрана. Если параметр опущен, то диалоговое окно помещается по вертикали примерно на 1/3 высоты экрана.

Файл справки — строковое выражение, определяющее имя файла справки, содержащего справочные сведения о данном диалоговом окне. Если этот аргумент указан, необходимо также указать аргумент «контекст».

Контекст — числовое выражение, определяющее номер соответствующего раздела справочной системы. Если этот аргумент указан, то необходимо указывать аргумент «файл справки».

Возвращаемым значением данной функции является информация, вводимая пользователем. Visual Basic автоматически присваивает этой информации тип String. Если вместо ввода информации пользователь просто нажимает кнопку ОК или клавишу Enter, функция возвращает пустую строку (""). Функцию InputBox с двумя и большим числом аргументов можно использовать только в выражении. Наличие запятых, соответствующих отсутствующим аргументам, обязательно.

Для вывода различных сообщений используется стандартное диалоговое окно MsgBox. Вид этого окна может быть различным, не оно всегда включает следующие компоненты:

* текст сообщения;
* заголовок;
* пиктограмма;
* набор кнопок.

MsgBox можно вызывать как процедуру и как функцию. Процедура выводит на экран заданное пользователем текстовое сообщение. Синтаксис вызова процедуры следующий:

MsgBox текст[, опция] [, заголовок] [, файл справки, контекст].

Функция выводит на экран диалоговое окно, содержащее сообщение, устанавливает режим ожидания нажатия кнопки пользователем, а затем возвращает значение типа Integer, указывающее, какая кнопка была нажата. Синтаксис вызова функции следующий:

переменная = MsgBox(TeKcr[, опция] [, заголовок] [, файл справки, контекст]).

Текст — строковое выражение, отображаемое как сообщение в диалоговом окне (максимальная длина 1024 символа)

Опция — дополнительная информация. Эта опция представляет собой числовое значение, составленное как сумма значений, которые указывают число и тип отображаемых кнопок, тип используемого значка, основную кнопку. Таким образом, опция = Button + Icon + Default. Каждому из параметров соответствует числовое значение, узнать которое можно, вызвав справочную систему Visual Basic. Редактор программного кода позволяет сформировать значение опции, выбирая нужные значения из предлагаемого списка.

Заголовок — строковое выражение, отображаемое в строке заголовка диалогового окна. Если этот аргумент опущен, то в строку заголовка помещается имя приложения.

Файл справки — строковое выражение, определяющее имя файла справки, содержащего справочные сведения о данном диалоговом окне. Если этот аргумент указан, необходимо также указать аргумент «контекст».

Контекст — числовое выражение, определяющее номер соответствующего раздела справочной системы. Если этот аргумент указан, то необходимо указывать аргумент «файл справки».

Функцию MsgBox с двумя и более аргументами можно использовать только в выражении (присваивать переменной). Наличие запятых, соответствующих отсутствующим аргументам, является обязательным.

**Функции или Операторы ввода (InputBox) и вывода (MsgBox)**

31/05/12
Автор: [Владимир Ткаченко](http://www.lessons-tva.info/author.html)
Источник: <http://www.lessons-tva.info/>

В VBA ввод и вывод информации (для взаимодействия с пользователем) можно осуществлять в диалоговых окнах. Диалоговое окно ввода значений реализуется встроенной функцией **InputBox.** В окне ввода, реализованное функцией InputBox, отображается поле для ввода значения переменной, в которое пользователь должен ввести определенное значение. Далее пользователь должен нажать кнопку ОК.

Функция**InputBox()** имеет следующий синтаксис:

Имя\_Переменной = InputBox( Prompt, [Title], [Default], [XPos], [YPos], [HelpFile], [Context])

Где аргументы: Prompt или Сообщение - обязательный аргумент, который задает в диалоговом окне информационное сообщение. Все остальные аргументы являются необязательными. Title задает заголовок окна. На рис 1 приведен модуль, в котором применена функция InputBox.


Рис. 1.

После выполнения модуля 2 появляется окно сообщения "Ввод значений переменных" (рис.2), в котором нужно ввести число и нажать кнопку ОК. В окне диалога (Рис.2), реализованном функцией InputBox (рис 1), отображаются: Заголовок окна - Ввод значений переменных; Сообщение - Введите число; Кнопки (по умолчанию) - ОК и Cancel; Поле предназначенное для ввода значений переменной.


Рис. 2.

Для вывода информации применяются диалоговые окна сообщений, реализуемые оператором **MsgBox** или функцией **MsgBox()**. MsgBox может использоваться как оператор. Оператор MsgBox осуществляет вывод информации в диалоговом окне и устанавливает режим ожидания нажатия кнопки пользователем.

Оператор MsgBox имеет следующий синтаксис:
MsgBox Prompt, [Buttons], [Title], [HelpFile], [Context]

Где аргументы: Prompt или Сообщение - обязательный аргумент, задающий в окне выводимое информационное сообщение. Все остальные аргументы являются необязательными. Buttons - Кнопки, которые можно использовать в диалоговом окне вывода сообщений. В окне сообщений могут применяться различные кнопки (ОК, Отмена и т.д.). Если не указывать, какие кнопки необходимо отображать в окне сообщений, то по умолчанию отображается кнопка ОК. Кроме того, в диалоговых окнах вывода сообщений можно использовать различные значки (vbQuestion - значок вопросительного знака, vbExclamation - значок восклицательного знака и т.д.).

Модуль, в котором MsgBox используется как оператор, приведен на рис. 3 (оператор MsgBox "3", vbOKCancel, "Вывод значений").


Рис. 3

При запуске модуля 4 на исполнение отображается окно сообщений "Вывод значений" (рис. 4), в котором необходимо нажать кнопку ОК. В окне (Рис.4), реализованном оператором MsgBox (Рис. 3), отображаются: Заголовок окна - Вывод значений; Сообщение - 3; Кнопки - ОК и Отмена.


Рис. 4

Например, для вычисления функции типа y = 5 x2 + 7 x + 9, можно использовать функцию InputBox и оператор MsgBox (рис. 5)


Рис. 5

После выполнения модуля 5 отображается окно ввода


Рис. 6

После ввода числа, например 789, и щелчка на кнопке ОК, появляется окно сообщения, в котором отображается результат вычисления функции у = 5 x2 + 7 x + 9.


Рис. 7

MsgBox можно использовать в качестве функции. Функция MsgBox() имеет следующий синтаксис: MsgBox (Prompt, [Buttons], [Title], [HelpFile], [Context]). В этом случае в окне диалога используют несколько различных кнопок. При нажатии кнопки в окне диалога функция MsgBox() возвращает значение типа Integer, которое зависит от того, какая из кнопок была нажата в диалоговом окне вывода сообщений.