Программирование на языке VBA

VBA - Visual Basic for Applications

это объектно-ориентированный язык программирования

Основные понятия

- объект
- семейство
- метод
- класс
- свойство
- событие
- объектная модель

Объектные модели Microsoft Office 2002

- библиотека объектов Microsoft Access (Microsoft Access 10.0 Object Library);
- библиотека объектов доступа к данным (Microsoft DAO 3.6 Object Library);
- библиотека объектов ActiveX (Microsoft ActiveX Data Objects 2.6);
- расширения ADO для поддержки DDL и защиты информации (Microsoft ADO Extensions 2.6 for DDL and Security);
- библиотека объектов Microsoft Jet и репликации (Microsoft Jet and Replication Objects 2.6).

Кроме этого, в приложениях Access обычно используются:

- библиотека объектов Visual Basic (Microsoft Visual Basic for Applications);
- библиотека объектов Microsoft Office (Microsoft Office 10.0 Object Library). Две эти библиотеки являются общими для всех приложений Microsoft Office.

Процедуры VBA

- Sub <имяПроцедуры> (<аргумент1>, <аргумент2>, ...)
- <операторVisualBasic1>
  <операторVisualBasic2>

- End Sub

Функции VBA

```
Function <имяФункции>
  (<аргумент1>, <аргумент2>, ... )
  <операторVisualBasicl>
   <операторVisualBasic2>

  <имяФункции> =
   <возвращаемоеЗначение>
End Function
```

Примеры вызова процедуры под именем CrossRC с передачей ей двух аргументов (константы и выражения):

```
CrossRC 7, i + 2
```

ИЛИ

```
Call CrossRC(7, i + 2)
```

Пример вызова двух функций — Left и Mid, и использования возвращаемого ими значения в выражении:

```
yStr = Left(y, 1) & Mid(y, 2, 1)
```

Допускается два различных способа передачи переменных процедуре или функции: *по ссылке* и *по значению.*

Пример:

```
Sub Main()

a = 10

b = 20

c = 30

Call Examplel(a, b, c)

Call MsgBox(a)
Call MsgBox(b)
Call MsgBox(c)
End Sub

Sub Example1(x, ByVal y, ByRef z)

x = x + 1

y = y + 1

z = z + 1
```

Программа может состоять (и обычно состоит) из многих процедур и функций, которые могут располагаться в одном или нескольких *модулях.* Модули группируются в *проекты,* при этом в одном проекте могут мирно сосуществовать несколько различных программ, использующих общие модули или процедуры.

Каждая из процедур, находящихся в одном модуле, должна иметь уникальное имя, однако в проекте может содержаться несколько различных модулей.

- Если в проекте содержится несколько различных процедур с одним и тем же именем, необходимо для уточнения имени использовать при вызове процедуры следующий синтаксис:

  <имяМодуля>.<имяПроцедуры>

- Если при этом имя модуля состоит из нескольких слов, следует заключить это имя в квадратные скобки. Например, если модуль называется "Графические процедуры", а процедура — "Крестик", вызов может выглядеть следующим образом:

  [Графические процедуры].Крестик

- Допускается также использование процедур, расположенных и в других проектах. При этом может потребоваться еще один уровень уточнения имени:

  <имяПроекта>.<имяМодуля>.<имяПроцедуры>

Переменные, константы и типы данных

Объявление переменных
Dim <имяПеременной> [As<типДанных>]

Пример:
Dim i As Integer, j As Integer
Dim x As Double

Типы данных

Array Массив переменных, для ссылки на конкретный элемент массива используется индекс.

Boolean Требуемая память: зависит от размеров массива Принимает одно из двух логических значений: True или False. Требуемая память: 2 байта Byte Число без знака от 0 до 255 Требуемая память: 1 байт

Currency Используется для произведения денежных вычислений с фиксированным количеством знаков после десятичной запятой, в тех случаях, когда важно избежать возможных ошибок округления. Диапазон возможных значений: от -922 337 203 685 477,5808 до 922 337 203 685 477,5807. Требуемая память: 8 байтов. Символ определения типа по умолчанию: @

Date Используется для хранения дат. Диапазон возможных значений: от 1 января 0100 г. до 31 декабря 9999 г. Требуемая память: 8 байтов

Double Числовые значения с плавающей точкой двойной точности. Диапазон возможных значений для отрицательных чисел: от -1 ,797693 13486232E308 до -4,94065645841 247E-324. Диапазон возможных значений для положительных чисел: от 4,94065645841 247E-324 до 1, 7976931 3486232E308. Требуемая память: 8 байтов. Символ определения типа по умолчанию: #

Типы данных

Integer Короткие целые числовые значения. Диапазон возможных значений: от -32 768 до 32 767. Требуемая память: 2 байта. Символ определения типа по умолчанию: %

Long Длинные целые числовые значения.
Диапазон возможных значений: от -2 147 483 648 до 2 147 483 647.
Требуемая память: 4 байта. Символ определения типа по умолчанию: &

Object Используется только для хранения ссылок на объекты.
Требуемая память: 4 байта

Single Числовые значения с плавающей точкой обычной точности.
Диапазон возможных значений для отрицательных чисел: от -3.402823E38 до -1 ,401 298E-45.
Диапазон возможных значений для положительных чисел: от 1 ,401 298E-45 до 3.402823E38.
Требуемая память: 4 байта. Символ определения типа по умолчанию: !

Типы данных

String

Используется для хранения строковых значений. Длина строки: от 0 до 64 Кбайтов. Требуемая память: 1 байт на символ. Символ определения типа по умолчанию: $

Variant

Может использоваться для хранения различных типов данных: даты/времени, чисел с плавающей точкой, целых чисел, строк, объектов.
Требуемая память: 16 байтов, плюс 1 байт на каждый символ строковых значений.
Символ определения типа по умолчанию: отсутствует

Для определения типа данных аргументов процедуры или функции используется описание типа данных непосредственно в заглавной строке процедуры или функции. Например:

```
Sub SplitStr(str1 As String,
str2 As String, str3 As String)
```

Определение типа данных возвращаемого функцией значения завершает заглавную строку функции, например:

```
Function FindSplitSpace
(strl As String) As Integer
```

Константы

Const <имяКонстанты>
[As <типДанных>] = <выражение>

где <выражение> — эго любое значение или формула, возвращающая значение, которое должно использоваться в качестве константы.

Например, следующий оператор определяет целую константу maxLen:

Const maxLen% = 30

Предопределенные константы

Встроенные константы, относящиеся к объектам Access, начинаются с префикса ac, относящиеся к объектам Excel — с префикса xl, относящиеся к объектам Word — с префикса wd, а относящиеся к объектам VBA — с префикса vb.

Например, в команде

DoCmd.OpenForm "Orders",

acNormal, , stLinkCriteria

используется встроенная константа Access acNormal.

Ссылки на объекты

Dim <имяПеременкой> As Object

Set <имяПеременной> =
  <ссылкаНаОбъект>

Пример:

```
Dim MyBase As Database
Set MyBase = CurrentDb( )
```

Массивы

Dim <имяМассива> (<размер1>, <размер2>, ...)
As <типДанных>
где <размер1>, <размер2> и т.д. задают размеры массива — количество индексов и максимально допустимое значение для каждого. конкретного индекса. При этом индексирование элементов массива по умолчанию начинается с нуля.

Пример:

```
Dim Array1 (9) As Integer

Dim Array2 (4, 9) As Variant
```

При объявлении массива можно указать не только верхнюю границу индекса, но и его нижнюю границу.

Dim <имяМассива> (<мин1> To <макс1>, ...)
As <типДанных>

Например:
Dim Temperature (-14 To 0)
As Single

Синтаксис объявления и определения размеров динамического массива:

Dim <имяМассива> ( ) As <типДанных>
ReDim <имяМассива> (<размер1>,<размер2>,...)

Область действия переменных и процедур

- уровень процедуры;
- уровень модуля;
- уровень проекта.

```
Public Al As String Private A2 As Integer
Dim A3 As Single
Sub Prod ()
Dim A4 As Integer
Static A5 As Integer
Al = "Текстовая строка 1"
A2= 2
A3 = 3.14
A4 = A4 + 4
A5 = A5 + 5
MsgBox A4
MsgBox A5
End Sub

Sub Proc2 () Procl
MsgBox Al
MsgBox A2
MsgBox A3
MsgBox A4
MsgBox A5
Procl End Sub
```