Технология программирования

Блок 3. Сложные структуры данных и методы их обработки

Тема 1. Массивы и записи. Многомерные массивы

Цели изучения темы:

- ознакомить студентов с понятием массива, многомерных массивов и их особенностями.

Задачи изучения темы:

- сформировать навыки работы с массивами.

В результате изучения данной темы Вы будете
знать:

- основные принципы построения структуры и приемы работы с инструментальными средствами, поддерживающим и создание программного обеспечения;
- основные правила работы с массивами;
- особенности работы с многомерными массивами;

уметь:

- программировать задачи сиспользованием методов проектирования и инструментальных средств разработки программного обеспечения;
- использовать основные модели информационных технологий и способы их применения для решения задач;

владеть:

- навыками разработки новых методов и средств проектирования программного обеспечения, навыками и приемами работы с инструментальными средствами;
- основными и дополнительными командами для работы с массивами.

Учебные вопросы темы:

1. Массивы.
2. Многомерные массивы.
3. Дополнительные команды работы с массивами.

Основные термины и понятия, которые Вам предстоит изучить:

массив, размерность, нумерация элементов, многомерные массивы.

Вопрос 1. Массивы

Массивом называется упорядоченная последовательность величин, обозначаемая одним именем. Упорядоченность заключается в том, что элементы массива располагаются в последовательных ячейках памяти. Чтобы получить доступ к нужному элементу массива нужно указать имя массива и индекс этого элемента. Имя массива образуется также как имя переменой. Различают одномерные и многомерные массивы. Одномерный массив – это список переменных, двумерный массив – таблица, имеющая строки и столбцы. Элементы одномерного массива снабжаются одним индексом, заключенным в круглые скобки. Он определяет порядковый номер элемента в массиве.

У каждого массива 5 основных характеристик: имя, размерность, число элементов, номер первого элемента и тип элементов.

- **Имя** – правила именования массивов аналогичны правилам именования переменных.
- **Размерность** – одномерные массивы напоминают одну строку таблицы, каждая ячейка которой содержит какие-то данные. Многомерные массивы имеют больше измерений. Их можно сравнивать с таблицами, имеющими множество строк и столбцов и с наборами таблиц.
- **Нумерация элементов** подчиняется следующим правилам:
  - По умолчанию нумерация элементов массива начинается с 0. Первый по счету элемент получит индекс 0, второй - 1 и т.д.
  - В объявлении отдельного массива можно явно указать индекс его первого и последнего элемента, разделив их ключевым словом To.
  - Если вы хотите, чтобы индексы всех массивов начинались с 1, добавьте в раздел объявлений модуля (вне процедур, функций и обработчиков событий) команду **Option Base 1**.
- **Тип** - подчиняется тем же правилам, которые мы ранее рассмотрели для переменных. Если пользователь не указал его явно, массив получит тип по умолчанию - Variant. Это требует больше системных ресурсов, но позволяет обрабатывать значения различных типов.

Не всегда количество элементов и размерность массива известны до начала работы программы. VBA умеет работать с динамическими массивами, параметры которых можно менять в ходе выполнения программы.

Для объявления массивов используют оператор **Dim**. Объявить массив можно двумя способами. Первый заключается в указании общего количества элементов. Например, так:

```
Dim MyArrayA(30) As Single
```

Объявленный массив MyArrayA содержит 31 элемент (с индексами от 0 до 30) типа Single. Поскольку нумерация явно не задана, элементы получат индексы по обычным правилам.

Можно объявить массив и другим способом:

```
Dim MyArrayB(1 To 25)
```

Массив MyArrayB содержит 25 элементов. Границы нумерации заданы явно - первый элемент получит индекс 1, второй - 2 и т.д. Тип не указан - в массиве можно хранить любые данные.

Работа с элементами массива отличается от работы с переменными использованием индексов для работы с различными элементами массива.

Пример:

1. Объявить одномерный массив на 3 элемента.
2. Внести в первый элемент число 5 в программе, во второй – запросив значение с помощью окна ввода.
3. Вычислить в третьем элементе массива произведение значений, хранящихся в первом и втором элементах.
4. Вывести полученное значение в окне сообщения.

```
Dim A(2) As Integer
A(0) = 5
A(1) = InputBox("Введите значение второго
элемента")
A(2) = A(0) * A(1)
MsgBox A(2)
```

В конце работы программы, если на вопрос о вводе числа мы введем число 2, массив **A** будет иметь такой вид:

Индекс	**0**	**1**	**2**
Значение	5	2	10

Пример: Сгенерировать массив из 5 целых значений

```
Dim mas(5) As Integer
For i% = 0 To 4
    mas(i) = Int((10 * Rnd) + 1)
Next i
```

Обратите ваше внимание, что в этом примере используется неявное объявление при работе с циклами в VBA. **i%** — означает неявное объявление переменной **i** в формате integer. Такая конструкция по сути

заменяет следующую**:** **Dim i As integer**. Это используется для сокращения кода и для удобства написания и чтения. В старых версиях VBA необходимо указывать знак формата после каждого использования неявной переменной. В более поздних версиях достаточно всего один раз

Вопрос 2. Многомерные массивы

Многомерные массивы имеют несколько измерений. Количество измерений ограничено 60.

Представить многомерные массивы можно так:

- одномерный – строка записей
- двухмерный – лист
- трехмерный – книга
- четырехмерный - книжная полка
- пятимерный - книжный шкаф
- шестимерный – библиотека
- семимерный – несколько библиотек

Чаще всего применяются двумерные массивы (матрицы). Матрицу можно представить в виде обычной таблицы с несколькими строками и столбцами.

Элементы двумерного массива снабжаются двумя индексами, заключенными в круглые скобки и разделенными запятой. Первый индекс номер строки, второй номер столбца, на пересечении которых расположен элемент в таблице (матрице). Например: B(2,3)=6. Все используемые массивы должны быть описаны до их использования в программе

Для того чтобы объявить двумерный массив, нужно воспользоваться командой Dim с указанием размерности каждого из измерений. Остальные правила объявления таких массивов и работы с ними аналогичны таковым для одномерных массивов. Например, мы можем указать лишь размеры измерения массива:

```
Dim MyArrayA(10, 1) As Single
```

Массив MyArrayA содержит 11 строк и 2 столбца типа Single.

Можно в явном виде задать границы размерностей:

```
Dim MyArrayB(1 To 25, 1 To 5)
```

Массив MyArrayB содержит 25 строк и 5 столбцов. Границы нумерации заданы явно. Тип не указан - в массиве можно хранить любые данные.

Если тип массива не задан, то во многих случаях это будет замедлять выполнение вашей программы. Хотя это и является удобным.

Пример программы, которая объявляет двумерный массив 5x2 и предлагает ввести в него фамилии и номера телефонов сотрудников.

```
Dim MyArray(1 To 5, 1 To 2)
For i = 1 To 5
 MyArray(i, 1) = InputBox("Введите фамилию №" & i)
 MyArray(i, 2) = InputBox("Введите Телефон №" & i)
Next i
```

Заполненный массив MyArray

Индекс	1	2
1	Иванов	898989898
2	Петров	343434343
3	Сидоров	565656565
4	Александров	121111212
5	Маринин	545454544

Эта программа очень похожа на те, которые мы писали для работы с одномерными массивами. В цикле, тело которого повторяется 5 раз, мы поочередно запрашиваем фамилию и номер телефона.

Один цикл неудобно использовать для работы с массивами больших размерностей. Нетрудно представить себе, какой громоздкой получится решение задачи копирования одной матрицы 100x100 в другую такую же.

Помогает решать подобные задачи механизм вложенных циклов.

Например, для заполнения массива 10x10 случайными целыми числами от 1 до 10 можно написать такую программу

```
Dim MyArray(1 To 10, 1 To 10)
For i = 1 To 10
    For j = 1 To 10
        MyArray(i, j) = Int(Rnd(1) * 10)
    Next j
Next i
```

Внешний цикл (i) выполняется один раз, после чего внутренний (j) - десять раз. За один проход внешнего цикла внутренний выполняет десять - заполняется первая строка массива (с индексами от 1,1 до 1,10) и т.д.

Могут сложиться обстоятельства, при которых точно неизвестно, сколько элементов потребуется в массиве. В VBA имеется возможность при помощи оператора *ReDim* переопределять размерность массива, а во время объявления не указывать его размерность.

ReDim [Preserve] *varname*(*subscripts*) [As *Type*] [, *varname*(*subscripts*) [As *Type*]]

varname – обязательный элемент – имя существующего массива;
subscripts – обязательный элемент – размерность существующего массива;
Type – любой тип VBA. Необходимо использовать отдельный оператор *As Type* для каждого массива, который определяется;
Preserve – необязательный элемент. Его использование приводит к тому, что данные, уже имеющиеся в массиве, сохраняются после изменения его размерности.

Dim Array_DBL() As Single – объявляет динамический массив
ReDim Array_DBL(2, 9) – делает массив двумерным
ReDim Array_DBL(3, 7) – изменяет размер двумерного массива
ReDim Preserve Array_DBL(1 To 3, 1 To 5) – изменяет последний размер массива, сохраняя содержимое

Обратите внимание! Можно изменять только последнее измерение многомерного массива, когда используется ключевое слово *Preserve*.

Пример. Программа просит пользователя ввести количество сотрудников, которое сохраняет в переменной ArraySize, а потом создает массив, одна из размерностей которого равняется ArraySize.

Чтобы воспользоваться динамическим массивом, сначала нужно объявить пустой массив, например, командой Dim MyArray(), а потом задать размерность массива командой ReDim

```
Dim MyArray()
      ArraySize = InputBox("Введите количество
сотрудников")
      ReDim MyArray(1 To ArraySize, 1 To 2)
```

В итоге, если на вопрос программы о количестве сотрудников мы ввели число 15, будет создан двумерный массив размерностью 15x2.

Если в программе возникла ситуация, когда последней размерности объявленного и заполненного массива не хватает для хранения данных, вы можете увеличить его командой ReDim с ключевым словом Preserve. Благодаря ему данные, внесенные ранее в массив, будут сохранены. Например, для добавления двух дополнительных столбцов в динамический массив. нужно использовать такую команду:

```
ReDim Preserve MyArray(1 To ArraySize, 1 To 4)
```

Вопрос 3. Дополнительные команды работы с массивами

Для работы с массивами вы можете использовать еще некоторые команды.

Array (Список аргументов)- позволяет быстро заполнять массив. Например, массив MyArray заполняется числами 1, 2, 6, 9 и 19, после чего первый элемент массива выводится в окне сообщения.

```
Dim MyArray
MyArray = Array(1, 2, 6, 9, 19)
MsgBox MyArray(0)
```

IsArray (Имя переменной) - возвращает True если переменная является массивом. Например, мы объявляем две переменные - одну из них как массив, вторую - как обычную переменную. Далее мы используем оператор ***IsArray*** для проверки того, является ли переменная массивом. После чего программа выводит соответствующее сообщение. Здесь мы использовали оператор сравнения ***If***.

```
Dim MyArray(10)
Dim MyArr
If IsArray(MyArray) Then _
     MsgBox ("Переменная MyArray - массив") _
Else MsgBox ("Переменная MyArray - не массив")
If IsArray(MyArr) Then _
     MsgBox ("Переменная MyArr - массив") _
Else MsgBox ("Переменна MyArr - не массив")
```

LBound (Имя Массива, Размерность) - возвращает нижнюю границу для указанной размерности массива.

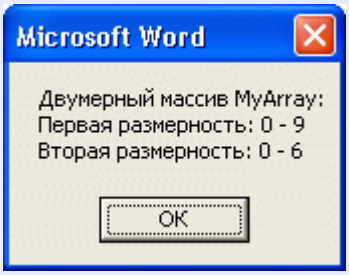
UBound (Имя Массива, Размерность) - возвращает верхнюю границу для указанной размерности массива.

Рассмотрим пример. Создадим динамический двумерный массив, размерности которого заданы с помощью генератора случайных чисел. После этого с помощью операторов ***LBound*** и ***UBound*** узнаем размерности массива и выведем их в окнах сообщений. Далее – используем двойной цикл для заполнения массива случайными числами.

```
Dim MyArray()
    ReDim MyArray(Int(Rnd * 5 + 5), Int(Rnd * 5 +
5))
    MsgBox ("Двумерный массив MyArray:" + Chr(13)
+ _
    "Первая размерность:" + _
    Str(LBound(MyArray, 1)) + " -" + _
    Str(UBound(MyArray, 1)) + Chr(13) + _
    "Вторая размерность:" + _
    Str(LBound(MyArray, 2)) + " -" + _
    Str(UBound(MyArray, 2)))
    For i = LBound(MyArray, 1) To UBound(MyArray,
1)
        For j = LBound(MyArray, 2) To
UBound(MyArray, 2)
```

```
            MyArray(i, j) = Int(Rnd * 100)
        Next j
    Next i
```

В нашем случае команда ***LBound*** для обеих размерностей массива возвращает **0** так как по умолчанию нумерация элементов массива начинается с **0**. А вот ***Ubound*** возвращает границу каждой из размерностей, которая установлена случайным образом с помощью оператора *ReDim*. На рисунке. вы можете видеть окно сообщения с информацией о границах массива.



Erase Имя_массива - очистить массив. Элементы обычных массивов, содержащих числовые данные, обнуляются. Если мы применим команду ***Erase*** к массиву строк - каждый его элемент будет хранить строку нулевой длины (""). Применяя команду ***Erase*** к динамическому массиву, мы очищаем память, выделенную этому массиву командой *ReDim*. Причем, для повторного использования динамического массива, придется снова устанавливать его размерности. Если команда ***Erase*** применяется к объектному массиву, в каждый его элемент записывается специальное значение *Nothing*, которое означает пустую ссылку на объект.

Функция ***Split(Имя Массива, Разделитель)*** позволяет нам превратить символьную строку в массив. Эти данные (строка) присваиваются заранее объявленному строковому (As String) одномерному динамическому массиву. Размерность устанавливается автоматически в зависимости от количества подстрок.

```
Dim MyArray()
Dim MyStr As String
MyStr = "IF/VLOOKUP/SUM/COUNT/ISNUMBER/MID"
MyArray = Split (MyStr, "/")
MsgBox MyArray (0)   ' возвращает IF
MsgBox MyArray (1)   ' возвращает VLOOKUP
MsgBox MyArray (2)   ' возвращает SUM
MsgBox MyArray (3)   ' возвращает COUNT
MsgBox MyArray (4)   ' возвращает ISNUMBER
MsgBox MyArray (5)   ' возвращает MID
```

Следующие три команды вернут в массив одни и те же значения

```
MyArray = Array("IF", "VLOOKUP", "SUM", "COUNT", "
ISNUMBER", "MID")
```

```
MyArray = Split("IF,VLOOKUP,SUM,COUNT,ISNUMBER,MID
", ",")
MyArray = Split("IF VLOOKUP SUM COUNT ISNUMBER MID
", " ")
```

Обратной к ***Split*** является функция ***Join.***

Join(Имя Массива, [Разделитель]) служит для слияния всех элементов заданного массива в одну строку со вставкой между ними необязательного разделителя. Т.е. Возвращает строку, созданную путем объединения множества подстрок, содержащихся в массиве с разделителем между ними**.**

[Разделитель] Необязательный аргумент - символ, используемый для разграничения подстрок в возвращаемой строке. Если параметр опущен, применяется символ пробела (" "). В том случае, когда аргумент является строкой нулевой длины (""), производится конкатенация всех элементов списка без добавления разделителей:

```
Dim MyArray(1 To 3)
Dim Val As String
MyArray = Array("Новая", "функция", "VBA")
MsgBox Join (MyArray, "_") 'возвращает
"Новая_функция_VBA"
```

Для перебора элементов массива так же есть способ не заботиться об определении нижней и верхней границ, если алгоритм не требует от нас знания текущего индекса массива. Для этого используется команда цикла ***For Each… Next***

For Each element In group

[statements]

[Exit For]

[statements]

Next [element]

element – обязательный атрибут в операторе ***For Each***, необязательный атрибут в операторе ***Next*** – переменная, используемая для циклического прохода элементов группы (диапазон, массив, коллекция), которая предварительно должна быть объявлена с соответствующим типом данных.

group – обязательный атрибут. Группа элементов (диапазон, массив, коллекция), по каждому элементу которой последовательно проходит цикл ***For Each... Next***

statements – необязательный атрибут. Операторы вашего кода. (*Если не использовать в цикле свой код, смысл применения цикла теряется.*)

Exit For – необязательный атрибут. Оператор выхода из цикла до его окончания.

Пример: Присвоим массиву список наименований животных и в цикле ***For Each... Next*** запишем их в переменную **a**. Информационное окно MsgBox выведет список наименований животных из переменной **a**.

```
Dim element As Variant, a As String, group As
Variant
group = Array("бегемот", "слон", "кенгуру",
"тигр", "мышь")
a = "Массив содержит следующие значения:" &
Chr(13)
  For Each element In group
    a = a & Chr(9) & element
  Next
MsgBox a
```

Массив содержит следующие значения:

бегемот
слон
кенгуру
тигр
мышь

Повторим те же действия, но всем элементам массива в цикле ***For Each... Next*** присвоим значение «Попугай». Информационное окно MsgBox выведет список наименований животных, состоящий только из попугаев, что доказывает возможность редактирования значений элементов массива в цикле ***For Each... Next***.

```
Dim element As Variant, a As String, group As
Variant
group = Array("бегемот", "слон", "кенгуру",
"тигр", "мышь")
a = "Массив содержит следующие значения:" &
Chr(13)
  For Each element In group
    element = "Попугай"
    a = a & Chr(9) & element
  Next
MsgBox a
```

Массив содержит следующие значения:

Попугай
Попугай
Попугай
Попугай
Попугай

Контрольные вопросы:

1. Что такое массив?
2. Как получить доступ к элементу массива?
3. Какой цикл удобнее использовать для перебора элементов массива?
4. Каковы особенности нумерации элементов массива в VBA?
5. Какая команда переопределяет размер массива?
6. Какие функции определяют верхние и нижние границы массива?
7. Какая команда помогает быстро заполнить массив значениями?
8. Для чего служат команды split и join?
9. Какую команду используют для перебора массива, когда неизвестно количество элементов?