24.04.2020

ОБЪЕДИНЕНИЕ «ПРОГРАММИРОВАНИЕ»,

2-й год обучения

Тема:
«Работа с массивами в Visual Basic»

Цели:

1. Ввести понятие массива, его размерности. Познакомиться с фиксированными и динамическими массивами, способами их описания
2. Научить обучающихся использовать массивы в программном коде.
3. Развивать у учащихся способность к алгоритмическому мышлению
4. Развивать умение анализировать взаимосвязь между различными объектами проекта.
5. Формировать устойчивое внимание

Теоретические сведения:

1. Описание массива

Массив – это группа переменных одного типа, объединенных одним именем. Массив можно использовать для хранения записей небольшой базы данных. При использовании массивов резко упрощается обработка однотипных данных.

Как и другие переменные, массивы описываются с помощью инструкций **Dim**, **Static**, **Private** или **Public**. Разница между скалярными переменными (т.е. не массивами) и массивами состоит в том, что для последних надо указывать размер массива. Массив с заданным размером называется массивом фиксированного размера. Массив с переменным размером называется динамическим.

Начало индексации массива с 0 или 1 определяется параметрами инструкции **Option Base**. Если не указано **Option Base 1**, нижняя граница индексов массива равняется нулю.

2. Описание массива фиксированного размера

Для объявления массива используется следующий синтаксис:

Dim | Public | Private | ArryName (Subscript) As DataType

или Dim | Public | Private | ArrayName (count 1 TO count2) As DataTyp

где ArryName – имя массива:

Subscript – номер последнего элемента в массиве;

count 1 и count 2 – индексы первого и последнего элементов массива.

Как и при описании других переменных, если тип данных при описании массива не задается, подразумевается, что элементы массива имеют тип **Variant**.

Максимальные размеры массива варьируются в зависимости от имеющейся операционной системы и доступной памяти. Использование массивов,

превышающих по размеру объем доступной системной оперативной памяти, замедляет работу программы, поскольку при этом данные должны читаться с диска и записываться на диск.

3. Описание динамического массива

Если массив описан как динамический, можно изменять его размер во время работы программы. Для описания динамического массива используются инструкции **Static**, **Dim**, **Private**, или **Public** с пустыми скобками, как показано в следующем примере.

```
Dim sngArray() As Single
```

Примечание. Можно воспользоваться инструкцией **ReDim** для неявного описания массива внутри процедуры. При этом надо точно задавать имя массива. В случае опечатки, даже если в модуле есть инструкция **Option Explicit**, будет создан второй массив.

В процедуре внутри области определения массива используется инструкция **ReDim** для изменения числа размерностей, определения числа элементов и задания верхних и нижних границ индексов для каждой размерности. Инструкцию **ReDim** можно применять для изменения динамического массива столько раз, сколько потребуется. Однако при каждом применении данные, содержащиеся в массиве, теряются. Инструкция **ReDim Preserve** увеличивает размер массива, сохраняя при этом его содержимое. В следующем примере показывается, как можно увеличить массив varArray на 10 элементов без уничтожения текущих значений элементов массива.

```
ReDim Preserve varArray(UBound(varArray) + 10)
```

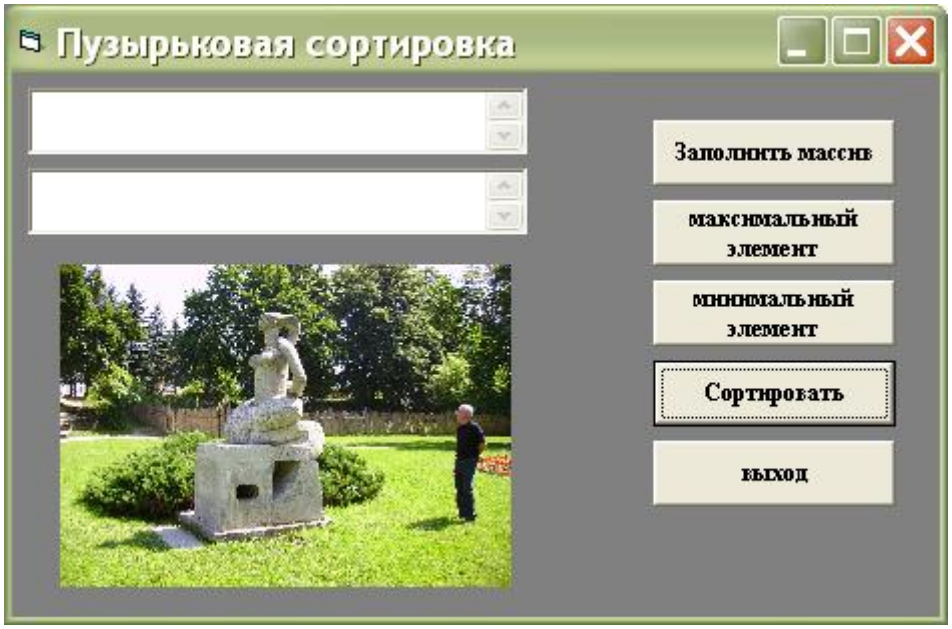
Примечание. Использование ключевого слова **Preserve** вместе с динамическим массивом позволяет изменить только верхнюю границу последней размерности массива, однако изменение числа размерностей невозможно.

Практическое задание:

I. **Создание проекта.**
   a. **Создать проект**, в котором формируется массив из десяти случайных целых чисел, находится его максимальный и минимальный элементы, а затем производится сортировка элементов массива.

Технология работы:

На форму устанавливаются два объекта TextBox (txtDim, txtSort) для отображения исходного и отсортированного массива и пять управляющих кнопок для заполнения и сортировки массива, нахождения максимального и минимального элементов, а также завершения работы (CmdDim, CmdSort, CmdMax, CmdMin, CmdExit). Элемент Image1 использован для дизайна.



2. Создание программного кода:

```
Dim bytA(1 To 10), bytMin, bytI, bytJ, bytK, bytR, bytN As Byte

Private Sub cmdDim_Click()
Randomize
txtDim.Text = ""
For bytI = 1 To 10
bytA(bytI) = Int(Rnd * 10)
txtDim.Text = txtDim.Text + Str(bytA(bytI))
Next bytI
End Sub

Private Sub CmdMax_Click()
bytMax = bytA(1)
bytN = 1
For bytI = 2 To 10
If bytA(bytI) > bytMax Then bytMax = bytA(bytI): bytN = bytI
Next bytI
MsgBox "Максимальный элемент " & bytMax & "  его индекс " & bytN
End Sub

Private Sub CmdMin_Click()
bytMax = bytA(1)
bytN = 1
For bytI = 2 To 10
If bytA(bytI) < bytMax Then bytMax = bytA(bytI): bytN = bytI
```

```
Next bytI
MsgBox "Минимальный элемент " & bytMax &  "  его индекс " & bytN
End Sub

Private Sub cmdSort_Click()
txtSort.Text = ""
For bytI = 1 To 10
For bytJ = bytI + 1 To 10
If bytA(bytI) > bytA(bytJ) Then bytR = bytA(bytI): bytA(bytI) = bytA(bytJ):
bytA(bytJ) = bytR
Next bytJ
txtSort.Text = txtSort.Text + Str(bytA(bytI))
Next bytI
End Sub

Private Sub Command1_Click()
End
End Sub
```

Конец программы.