

Arrays:

1. Introduction:

1.1 Definition

An array is a consecutive group of memory locations that all have the same name and the same type. Arrays could be named like any other variable.

To refer to a particular location or element in the array, we specify the array name and the element position number (i.e. the index of that element). An index is a value that refers to a particular array element.

Comparing the array with a simple variable:

Simple variable is used to store a single value. On the other hand, an array variable is used to represent many values of the same type with one variable name.

The following figure shows a six-element Integer array named numbers.

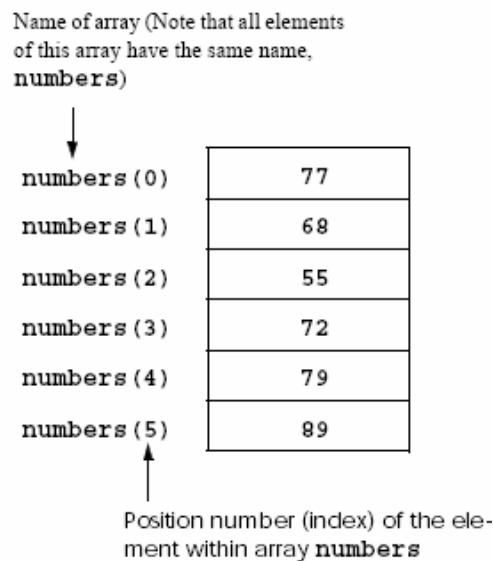


Figure1: A six-element array.

Any element of this array can be referred by giving the array named followed by the element index in parentheses, ().

The first element of the array is referred to as **numbers (0)**, the second element of the array is referred to as **numbers (1)**, and so on. In general the i^{th} element of the array **numbers** is referred to as **numbers (i -1)**.

Declaring Arrays:

To declare an array you need to specify the name of the array, the type for all the elements of the array, and the number of elements.

The syntax of array declaration is:

```
Dim arrayname ( 1 To n) As dataType
```

For example:

```
Dim month(1 To 12) As String
```

```
Dim score(1 To 30) As Single
```

```
Dim students(1 To 30) As String
```

When declaring an array as:

```
Dim numbers(1 To 5) As Integer
```

The value **5** is defined as the upper bound (i.e., the highest valid index) of the array **numbers**. The lower bound is defined of the array **numbers** defaults to **1**.

When declaring several arrays using a single line declaration:

```
Dim b (1 To 99) As Integer, x (1 To 26) As Long, s (1 To 14) As String
```

Initializing Arrays:

The individual numeric array can be initialized to zero by default. These are the ways to initialize an array (i.e., filling the array):

1. Assignment statement:

```
Dim numbers(1 To 5) As Integer
numbers(1) = 11
numbers(2) = 22
numbers(3) = 33
numbers(4) = 44
numbers(5) = 55
```

2. Repetition Statements:

Example 1:

```
Private Sub Command1_Click()
Dim numbers(1 To 5) As Integer
For x = 1 To 5
    numbers(x) = x + 1
    Print numbers(x);
Next x
End Sub
```

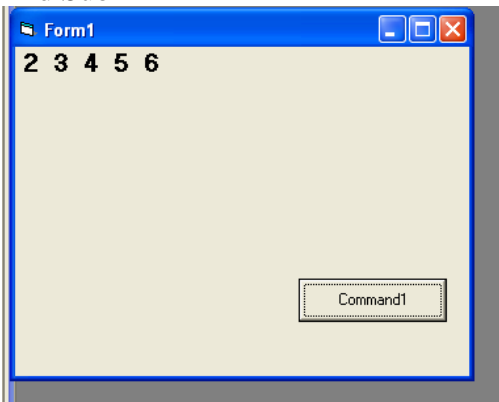


Figure 2: The Output of Example 1.

Example 2:

This program initializes the array to even elements:

```
Private Sub Command1_Click()
Dim x(1 To 10) As Integer
Dim i As Integer
Print "index" & Space$(3) & "value"
For i = 1 To 10
    x(i) = 2 + 2 * i
Next i
For i = 1 To 10
    Print Space$(3) & i & Space$(7) & x(i)
Next i
End Sub
```

Output:

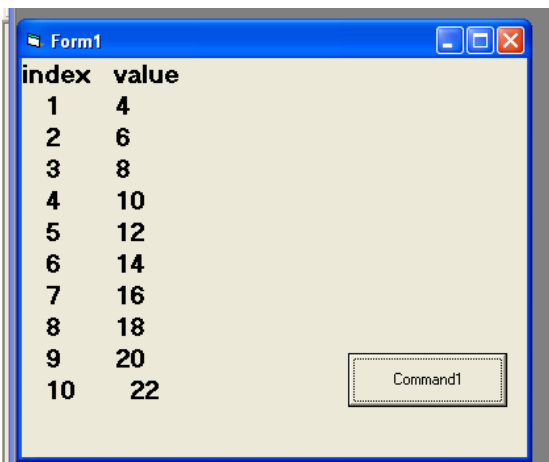


Figure 3: The Output of Example 2.

Notes: this program introduces functions **LBound** and **UBound**. Function **LBound** returns the lower bound (i.e., the lowest numbered index value) of the array and the function **UBound** returns the upper bound (i.e., the highest numbered index value) of the array.

3. Using Random number Function:

Example 3:

```
Private Sub Command1_Click()
Dim x(1 To 10) As Integer
Dim i As Integer
Print "index" & Space$(3) & "value"
For i = 1 To 10
x(i) = 1 + Int(Rnd() * 20)
Next i
For i = 1 To 10
Print Space$(3) & i & Space$(7) & x(i)
Next i
End Sub
```

Output:

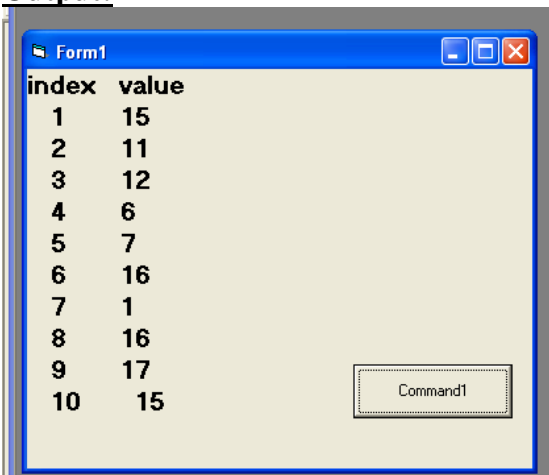


Figure 4: The Output of Example 3.

Example 4:

```
Private Sub Command1_Click()
Dim score(1 To 10) As Single
Dim count As Integer
Dim sum As Integer
sum = 0
Print "index" & Space$(3) & "value"
For count = 1 To 10
score(count) = 1 + Int(Rnd() * 100)
```

```

Print Space$(3) & count & Space$(7) & score(count)
Next count
For count = 1 To 10
    sum = sum + score(count)
Next count
Print "The summation of the element values are: "; sum
End Sub

```

Output:

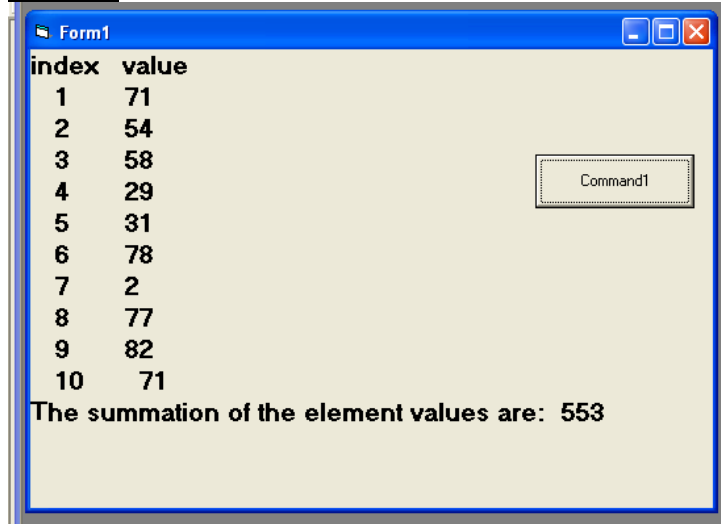


Figure 5: The Output of Example 4.

Array of Characters:

Example 5:

```

Dim teamName(1 To 4) As String
Dim n As Integer
teamName(1) = "ali"
teamName(2) = "ahmad"
teamName(3) = "nour"
teamName(4) = "mohammed"

```

Swapping two elements:

Example 6:

```

Private Sub command1_click()
Dim x(1 To 5) As Integer, y%
For i = 1 To 5
x(i) = i * 2 + 2
Print x(i);
Next i
Print
y = x(4)
x(4) = x(5)
x(5) = y
Print x(4); x(5)
End Sub

```

Output:

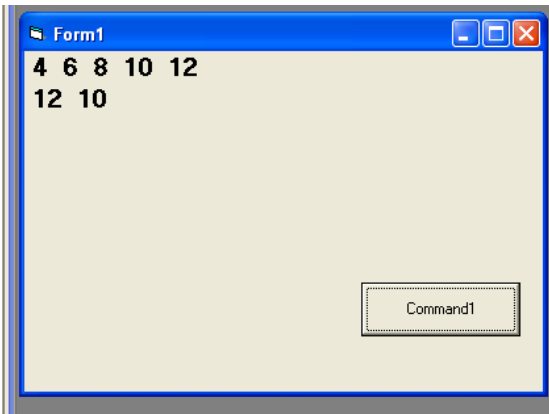


Figure 6: The Output of Example 5.

Filling Array elements with random numbers using Rnd() function:

Example 7:

```
Private Sub command1_click()  
Dim myArray(1 To 6) As Integer  
Dim i As Integer  
For i = 1 To 6  
myArray(i) = Int(10 * Rnd())  
Print myArray(i);  
Next i  
End Sub
```

Output:

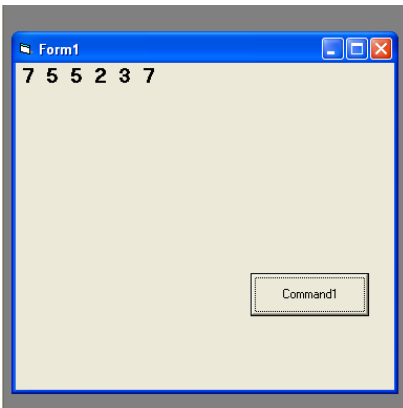


Figure 6: The Output of Example 6.