Arrays in Visual Basic 6

An array is a collection of simple variables of the same type to which the computer can efficiently assign a list of values. Array variables have the same kinds of names as simple variables. An array is a consecutive group of memory locations that all have the same name and the same type. To refer to a particular location or element in the array, we specify the array name and the array element position number. The Individual elements of an array are identified using an index. Arrays have upper and lower bounds and the elements have to lie within those bounds. Each index number in an array is allocated individual memory space and therefore users must evade declaring arrays of larger size than required. We can declare an array of any of the basic data types including variant, user-defined types and object variables. The individual elements of an array are all of the same data type.

1 Declaring arrays: Arrays may be declared as Public (in a code module), module or local. Module arrays are declared in the general declarations using keyword Dim or Private. Local arrays are declared in a procedure using Dim. Array must be declared explicitly with keyword "As".
There are two types of arrays in Visual Basic namely:

- **1-1 Fixed-Size Array:** The size of array always remains the same-size doesn't change during the program execution. When an upper bound is specified in the declaration, a Fixed-array is created. The upper limit should always be within the range of long data type.

One Dimension Array:

Declaring a fixed-array, if array-Name is the name of an array variable and N is a whole number, then the statement

Dim *ArrayName* (N) As *Var Type*

Where, the Dim statement is said to dimension the array and (N) is the range of the array. The array holds either all string values or all numeric values, depending on whether ***Var Type*** is string or one of the numeric type names.

For example:

Dim Num (5) As Integer

In the above illustration, num is the name of the array, and the number 6 included in the parentheses is the upper limit of the array. The above declaration creates an array with 6 elements, with index numbers running from 0 to 5.

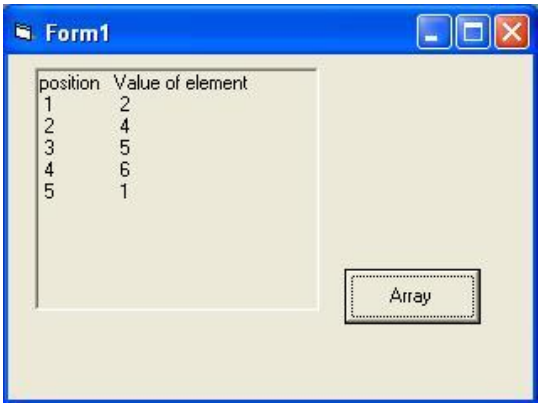
The numbers inside the parentheses of the individual variables are called **subscripts**, and each individual variable is called a **subscripted variable** or **element**. The elements of an array are assigned successive memory locations. The following figure shows the memory location for the array Num(5)

	Num (0)	Num (1)	Num (2)	Num (3)	Num (4)	Num (5)
Num (5)	1	3	-10	5	3	2

Example 1: Write a code program to read of one dimensional array A(5). Print the value and position of each element.

Solution:

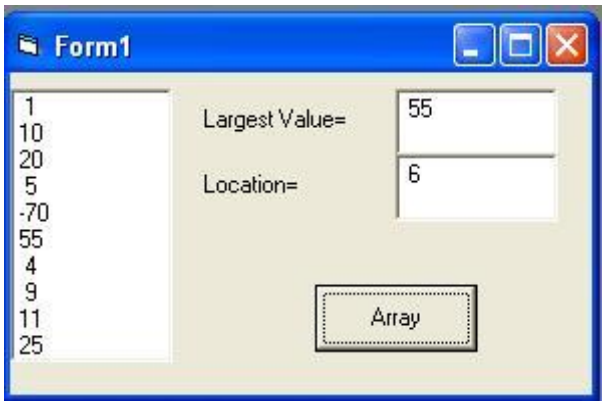
```
Dim A(5) as
single
Picture1.cls
Picture1.Print "position"; Space (3); "Value of
element" For I=1 To 5
A( I )=
Val(InputBox(“”))
Next I
For I= 1 to 5
Picture1.Print I; Space (11);
A(I) Next
```



Example 2: Suppose A is a one dimension array with 10 elements is entered into listbox. Write a program segment to find the location J such that A (J) contains the largest value in A. Display the Largest value and the location into textboxes.

Solution:

```
Dim A(10) as single
For I=1 To 10
A(I)=Val(list1.list(I-1))
Next
Max=A(1) : P=1
For J=1 to 10
If A(J)> Max Then
Max=A(J) : P= J
End If
Next
Text1.text= Str(Max)
Text2.text= Str(P)
End Sub
```



Two Dimensional Arrays:

Arrays can have multiple dimensions. A common use of multidimensional arrays is to represent tables of values consisting of information arranged in rows and columns. To identify a particular table element, we must specify two indexes: The first (by convention) identifies the element's row and the second (by convention) identifies the element's column.

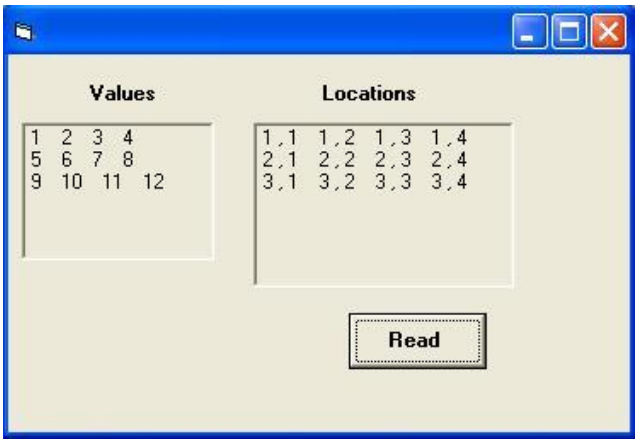
Tables or arrays that require two indexes to identify a particular element are called two dimensional arrays. The following statement declares a two-dimensional array (3 by 3) within a procedure.

Dim Avg (3, 3) as Single

Avg (Row, Col.)	**Avg (0,0)**	**Avg (0,1)**	**Avg (0,2)**	**Avg (0,3)**
	Avg (1,0)	**Avg (1,1)**	**Avg (1,2)**	**Avg (1,3)**
	Avg (2,0)	**Avg (2,1)**	**Avg (2,2)**	**Avg (2,3)**
	Avg (3,0)	**Avg (3,1)**	**Avg (3,2)**	**Avg (3,3)**

Avg (3, 3)	**2**	**6**	**1**	**0**
	3	**1**	**6**	**-3**
	7	**3**	**1**	**5**
	5	**4**	**-2.5**	**9**

Example3: Write a code program to read of two dimensional array A(3,4) on a row by row. Print the value and position of each element.



Solution:

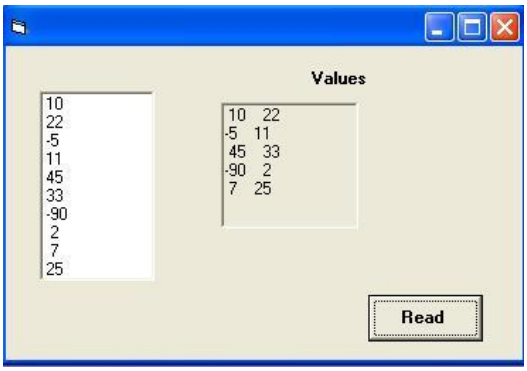
```
Dim A(3,4) As Single
For I=1 To 3                    (Rows)
For J= 1 To 4                   (Columns)
A(I,J) =Val(InputBox(“”))
Next J
Next I
For I=1 To 3
For J= 1 To 4
Picture1.Print A(I, J) ; Space(2) ;
Picture2.Print I ; " , " ; J ; Space(2) ;
Next J
Picture1.Print
Picture2.Print
Next I
```

Example 4: Re-write a code in example above on a column by column

Example 5: Write a code program to create a two dimensional array N (5X2) into List Box on row by row. Print the values of array N.

Solution:

```
Dim N(5,2) As Single
K=0
For I = 1 To 5
For J=1 To 2
N(I,J)= Val (List1.List (K))
K=K+1
Next J, I
```



```
For I=1 To 5
For J= 1 To 2
Picture1.Print N(I, J) ; Space(2) ;
Next J : Picture1.Print : Next I
```

Exercise 1: Write a code program to read the elements of the array T(5,3) on a row by row. Calculate the SUM of elements in each row and stored in column 4. Print a new array T(5,4) and the sum of all individual row sums, the cumulative sum for all rows.

Exercise 2: Suppose N is a (5x2) matrix array is entered into ListBox on row by row. Write a program segment to find the location I and J such that N (I,J) contains the largest value in N. Print the values of array N. Display the Largest value and the location into textboxes.