

Worksheet: Array Size in VBA



Level 1: Understanding Basics

An array is a collection of values of the same data type. It allows you to store multiple values under a single variable name. The size of an array refers to the number of elements it can hold.

In VBA, you declare an array using the following syntax:

vba

Dim myArray(5) As Integer ' Declares an integer array with 6 elements (0 to 5)

Level 2: Declaring Arrays

Fixed-Size Array:

vba

Dim numbers(4) As Double ' Array with 5 elements (0 to 4)

Dynamic Array:

vba

Dim dynamicArray() As String ' Declare without specifying size
ReDim dynamicArray(9) ' Resize to hold 10 elements (0 to 9)

Level 3: Array Bounds

Arrays in VBA are zero-indexed, meaning the first element is accessed using index 0. The last index is the size of the array minus one.

Worksheet: Array Size in VBA



vba

Dim values(2) As Integer ' Array with 3 elements (0 to 2)

values(0) = 10 ' First element

values(2) = 20 ' Third element

Level 4: Multi-Dimensional Arrays

Arrays can have multiple dimensions, like a table. For a 2D array, you specify rows and columns.

vba

Dim matrix(2, 3) As Integer ' 3x4 matrix (0 to 2 rows, 0 to 3 columns)

matrix(1, 2) = 5 ' Accessing a specific element

Level 5: Dynamic Array Resizing

Dynamic arrays can be resized using the ReDim statement. However, this erases existing data. To preserve data, use a temporary array.

vba

Dim dynamic() As Integer

ReDim dynamic(2) ' Size 3

ReDim Preserve dynamic(4) ' Resize and keep existing values

Level 6: Array Functions

LBound and UBound: Get lower and upper bounds of an array.

vba

Dim myArray(5 To 10) As Integer

Debug.Print LBound(myArray) ' Outputs 5

Worksheet: Array Size in VBA



Debug.Print UBound(myArray) ' Outputs 10

Array Length: Calculate the number of elements in an array.

```
vba  
Dim nums(1 To 5) As Integer  
Dim length As Integer  
length = UBound(nums) - LBound(nums) + 1 ' Calculate length
```

Level 7: Array Iteration

Loop through array elements using a For loop or a For Each loop.

```
vba  
Dim values(4) As String  
For i = LBound(values) To UBound(values)  
    values(i) = "Element " & i  
Next i  
  
For Each element In values  
    Debug.Print element  
Next element
```

Level 8: Arrays as Function Parameters

Pass arrays to functions for processing.

```
vba  
Function SumArray(nums() As Integer) As Integer  
    Dim total As Integer  
    For i = LBound(nums) To UBound(nums)  
        total = total + nums(i)  
    Next i  
    SumArray = total  
End Function
```

Worksheet: Array Size in VBA



Level 9: Array Sorting (Example)

vba

```
Sub BubbleSort(arr() As Integer)
  Dim n As Integer
  n = UBound(arr) - LBound(arr) + 1
  Dim i As Integer, j As Integer
  For i = 0 To n - 2
    For j = 0 To n - i - 2
      If arr(j) > arr(j + 1) Then
        Dim temp As Integer
        temp = arr(j)
        arr(j) = arr(j + 1)
        arr(j + 1) = temp
      End If
    Next j
  Next i
End Sub
```

Remember, mastering arrays is essential for efficient data handling and manipulation in VBA. Practice and experimentation are key to becoming proficient in working with arrays at varying levels of complexity.