

Tutorial 3 – Básico do VBA do Excel

Este tutorial contém a 3ª lição sobre a série de Tutoriais do VBA Básico do Excel. Ele cobre tópicos na criação e gerenciamento de array e entendimento das estruturas de laço e de decisão do VBA. Iniciantes na programação VBA serão encorajados a percorrerem de cabo a rabo as lições anteriores desta série se eles ainda não fizeram isto. Este documento contém informação sobre os seguintes tópicos.

- [Criando e Administrando Array](#)

- [Declarar um Array com a Declaração Dim](#)
 - [Redimensionando um Array com a Declaração Redim](#)
 - [Administrando Array Dinâmicos](#)
 - [Criar um Array Multidimensional](#)
 - [Encontrar o Tamanho de um Array](#)

- [Estruturas de Decisão - IF e Select Case](#)

- [IF ... Then](#)

- [IF ... Then ... Else](#)

- [IF ... Then ... Elseif](#)

- [Select Case](#)

- [Estruturas Loop](#)

- [For ... Next](#)

- [For ... Next Loop com Step](#)

- [Do While ... Loop](#)

- [Do Until ... Loop](#)

- [Do ... Loop While](#)

- [Do ... Loop Until](#)

Microsoft Support site ou a seção Ajuda (Help) do VBA do Excel no seu computador contém exemplos compreensivos da maioria das coisas cobertas neste tutorial. Para mais informação, por favor refira-se a elas.

Criando e Administrando Array [Microsoft Support](#)

Declarando um Array com a Declaração Dim

Um array é um conjunto de elementos indexados seqüencialmente tendo o mesmo tipo de dado intrínseco. Cada elemento de um array tem um único número índice identificador. Mudanças feitas num elemento de um array não afetam os outros elementos.

Antes de assinalar valores para uma array, o array precisa ser criado. Você pode declarar o array

usando a declaração **Dim**.

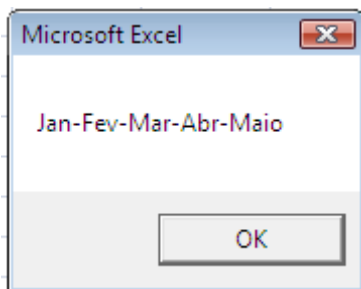
Por exemplo, para declarar um array unidimensional com 5 elementos, digite o seguinte:

```
Dim Arr(4)
```

Os índices dos elementos do array começam do 0 a menos que a **Option Base 1** seja especificada na área pública (área externa ao procedimento sub). Se Option Base 1 for especificada, o índice iniciará a partir de 1.

O exemplo seguinte atribui valores ao array e mostra todos os valores numa caixa de mensagem :

```
Option Base 1
Sub atribuirArray()
    Dim Arr(5)
    Arr(1) = "Jan"
    Arr(2) = "Fev"
    Arr(3) = "Mar"
    Arr(4) = "Abr"
    Arr(5) = "Maio"
    MsgBox Arr(1) & "-" & Arr(2) & "-" & Arr(3) & "-" &
Arr(4) & "-" & Arr(5)
End Sub
```



* O número dentro do array, i.e. Arr(1), é o índice. Um (1) é o índice do primeiro elemento no array.

Redimensionando um Array com a Declaração Redim

A declaração **ReDim** é usada para dimensionar ou redimensionar um array dinâmico que já foi formalmente declarado.

Por exemplo, se você já declarou um array com um número de elementos de valor 5 e decidiu mudar o número do elementos para 6, você pode fazer o seguinte para redimensionar o array:

```
Redim Arr(6)
```

Nós o incorporamos no nosso último exemplo:

```
Option Base 1
Sub atribuirArray( )
    'Dim Arr(5)
    Redim Arr(6)

    Arr(1) = "Jan"
    Arr(2) = "Fev"
    Arr(3) = "Mar"
    Arr(4) = "Abr"
    Arr(5) = "Mai"
```

```

        Arr(6) = "Jun"

        MsgBox Arr(1) & "-" & Arr(2) & "-" & Arr(3) & "-" &
Arr(4) & "-" & Arr(5)
    End Sub

```

Note que a declaração **Dim Arr(5)** está entre aspas, pois deixar sua declaração original na sub causará um erro de compilação.

Manage Dynamic Array

Uma palavra de cautela no uso da Declaração **Redim** para redimensionar um array – redimensionar o array pode apagar os elementos nele. No exemplo seguinte, todos os valores atribuídos anteriormente para redimensionar o array são apagados. Somente o valor atribuído ao array após o redimensionamento permanece.

```

Option Base 1
Sub atribuirArray( )
    Redim Arr(5)

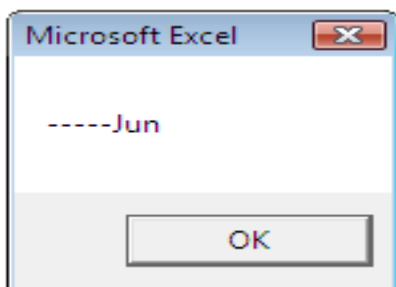
    Arr(1) = "Jan"
    Arr(2) = "Fev"
    Arr(3) = "Mar"
    Arr(4) = "Abr"
    Arr(5) = "Mai"

    Redim Arr(6)

    Arr(6) = "Jun"

    MsgBox Arr(1) & "-" & Arr(2) & "-" & Arr(3) & "-" &
Arr(4) & "-" & Arr(5) & "-" & Arr(6)
End Sub

```



Trocando o **Redim Arr(6)** com **Redim Preserve Arr(6)**, todos os valores permanecerão. Por exemplo:

```

Option Base 1
Sub atribuirArray( )
    Redim Arr(5)

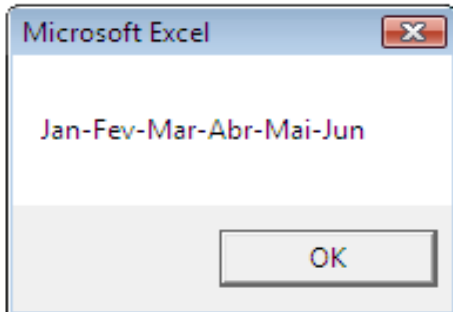
    Arr(1) = "Jan"
    Arr(2) = "Fev"
    Arr(3) = "Mar"
    Arr(4) = "Abr"
    Arr(5) = "Mai"

    Redim Preserve Arr(6)

```

```
Arr(6) = "Jun"
```

```
Msgbox Arr(1) & "-" & Arr(2) & "-" & Arr(3) & "-" &  
Arr(4) & "-" & Arr(5) & "-" & Arr(6)  
End Sub
```



Criar Array Multidimensional

Um array pode também armazenar dados multidimensionais. Para simplificar nosso tutorial, um exemplo de um array bidimensional é usado. Assuma que você tenha dados das vendas anuais de uma loja local na seguinte tabela e você quer armazenar os dados num array bidimensional:

	<u>Ano 2007</u>	<u>Ano 2008</u>
Venda de CD	1.000	1.500
Vendas de DVD	1.200	2.000

Primeiro criamos o array como segue:

```
Dim Arr(2,2)
```

Daí, então, atribuímos os valores para o array. Nós tratamos primeiro a dimensão como o ano e a segunda dimensão como a venda de produto:

```
arr(1,1) = 1000  
arr(1,2) = 1200  
arr(2,1) = 1500  
arr(2,2) = 2000
```

Mostramos agora os valores do array com uma caixa de mensagem:

```
Msgbox "Venda de CD em 2007 é " & arr(1,1) & vbCrLf & "Sale of  
CD in 2008 is " _  
      & arr(2,1) & vbCrLf & "Venda de DVD em  
2007 é " & arr(1,2) & vbCrLf _  
      & "Venda de DVD em 2008 é " & arr(2,2)
```

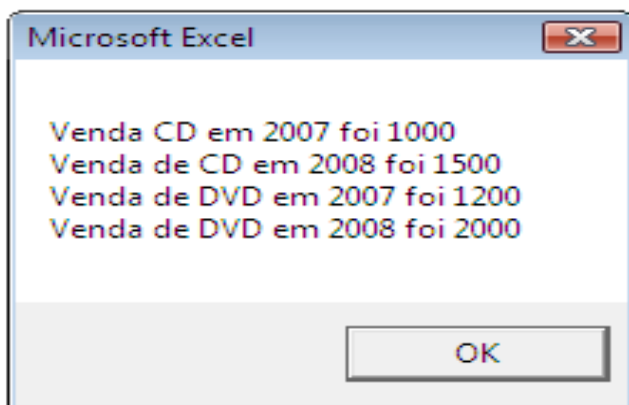
O procedimento completo é como segue:

```
Option Base 1  
Sub multDimArray( )  
    Dim Arr(2,2)  
  
    arr(1,1) = 1000  
    arr(1,2) = 1200  
    arr(2,1) = 1500  
    arr(2,2) = 2000
```

```

Msgbox "Venda CD em 2007 foi " & arr(1,1) & vbCrLf
& "Venda de CD em 2008 foi " _
      & arr(2,1) & vbCrLf & "Venda de DVD em
2007 foi " & arr(1,2) & vbCrLf _
      & "Venda de DVD em 2008 foi " &
arr(2,2)
End Sub

```



* vbCrLf significa no VB Carriage Return Line Feed. Ele coloca um retorno e uma nova linha como mostrado na caixa de mensagem acima. O sublinhado "_" no final da primeira linha da caixa de mensagem significa "continuar na próxima linha"

Encontrar o Tamanho de um Array

O maior subscrito disponível para a dimensão indicada de um array pode ser obtida usando a função **Ubound**. No nosso exemplo de array unidimensional, Ubound(arr) é 5.

No exemplo de array bidimensional acima, existiam duas figuras de limite superior – ambos são 2. **Ubound** retorna os seguintes valores para um array com estas dimensões*:

```
Dim A(1 To 100, 0 To 3, -3 To 4)
```

<u>Declaração</u>	<u>Valor de Retorno</u>
Ubound(A, 1)	100
Ubound(A, 2)	3
Ubound(A, 3)	4

* Exemplo tirado da seção Ajuda do VBA Excel.

A função **Ubound** é usada com a função **Lbound** para determinar o tamanho de um array. Use uma função **Lbound** para encontrar o limite inferior da dimensão de um array.

<u>Declaração</u>	<u>Valor de Retorno</u>
Lbound(A, 1)	1
Lbound(A, 2)	0
Lbound(A, 3)	-3

Para obter o tamanho de um array, use a seguinte formula:

Ubound(Arr) - Lbound(Arr) + 1

Por exemplo:

```

Ubound(A,1) - Lbound(A,1) + 1
= 100 - 1 + 1
= 100

```

```
Ubound(A,2) - LBound(A,2) + 1
= 3 - 0 + 1
= 4
```

```
Ubound(A,3) - LBound(A,3) + 1
= 4 - (-3) + 1
= 8
```

Para mais informações sobre arrays verifique [Microsoft Support](#)

Estruturas de Decisão - IF e Select Case

Declaração IF ... Then

A **IF ... Then** é uma condição simples e roda uma única declaração ou um bloco de declarações.

Exemplo, a declaração seguinte configura a variável Status para "Adulto" se a declaração for verdadeira:

```
If Idade >= 18 Then Status = "Adulto"
```

Você também pode usar um bloco de múltiplas linhas na declaração **If** como segue:

```
If Idade >= 18 Then
    Status = "Adulto"
    Vota = "Sim"
End If
```

Note que no caso do bloco de múltiplas linhas, a declaração **End If** é necessária, onde o caso linha única não.

IF ... Then ... Else

A declaração **If ... Then ... Else** é usada para definir dois blocos de condições – verdadeiro e falso.

Exemplo:

```
If Idade >=22 Then
    Bebe = "Sim"
Else
    Bebe = "Não"
End If
```

Novamente, note que a declaração **End If** é necessária neste caso também pois existe mais do que um bloco de declarações.

IF ... Then ... Elseif

O **IF ... Then ... Elseif** é usado para testar condições adicionais sem usar novas declarações **If ... Then**.

Por Exemplo:

```

If Age >= 18 and Age < 22 Then
    MsgBox "Você pode votar"
Elseif Age >=22 and Age < 62 Then
    MsgBox "Você pode beber e votar"
Elseif Age >=62 Then
    MsgBox "Você está eleito para aplicar no seu Social Security Benefit"
Else
    MsgBox "Você não pode beber ou votar"
End If

```

Note que a última condição sob Else é, implicitamente, Idade < 18.

Select Case

A declaração **Select Case** é uma alternativa à declaração Elseif. Este método é mais eficiente e legível em codificação que a declaração **If ... Then ... Elseif**.

Exemplo:

```

Select Case Grau
Case Is >= 90
    Conceito = "A"
Case Is >= 80
    Conceito = "B"
Case Is >= 70
    Conceito = "C"
Case Is >= 60
    Conceito = "D"
Case Else
    Conceito = "Aborrecido"
End Select

```

Estruturas de Laço

For ... Next

Use o laço **For ... Next** se o número de laços já é definido e conhecido. Um laço **For ... Next** usa uma variável contadora que aumenta ou diminui de valor durante cada iteração do laço. Esta estrutura de laço é usada na maioria das vezes nos nossos exemplos.

Aqui está um exemplo do laço **For ... Next**:

```

For i = 1 to 10
    Cells(i, 1) = i
Next i

```

	A	
1	1	
2	2	
3	3	
4	4	
5	5	
6	6	
7	7	
8	8	
9	9	
10	10	
11		
12		

Neste exemplo, i é a variável contadora de 1 até 10. O processo de looping enviará valor à primeira coluna da activesheet e imprime i (o qual vai de 1 a 10) para a linha 1 até a 10 daquela coluna.

Note que a variável contadora, por default, aumenta por um incremento de 1.

Laço For ... Next Com Step

Você pode usar uma palavra chave **Step** para especificar um incremento diferente para a variável contadora.

Por exemplo:

```
For i = 1 to 10 Step 2
  Cells(i, 1) = i
Next i
```

Este processo de laço imprimirá valores com um incremento de 2 na linha 1, 3, 5, 7 e 9 na coluna um.

	A	
1	1	
2		
3	3	
4		
5	5	
6		
7	7	
8		
9	9	
10		

Você pode também decrementar no laço atribuindo um valor negativo após a palavra **Step**.

Por exemplo:

```
For i = 10 to 1 Step -2
  Cells(i, 1) = i
Next i
```

Este processo do laço imprimirá valores com um incremento de -2 começando do 10 na linha 10,

8, 6, 4 e 2 na coluna um.

	A
1	
2	2
3	
4	4
5	
6	6
7	
8	8
9	
10	10

Do While ... Loop

Você pode usar o **Do While ... Loop** para testar uma condição no início do laço. Ela rodará o laço tantas vezes quanto a condição for verdadeira e pára quando a condição tornar-se falsa. Por exemplo:

```
i = 1
Do While i <= 10
  Cells(i, 1) = i
  i = i + 1
Loop
```

Este processo de laço conduzirá ao mesmo resultado que o exemplo das estruturas **For ... Next**.

Uma coisa para se tomar cuidado é que algumas vezes o laço poderá ser um laço infinito. E ele acontece quando a condição nunca se tornar falsa. Em tal caso, você pode parar o laço pressionando **[ESC]** ou **[CTRL] + [BREAK]**.

Do Until ... Loop

Você pode testar a condição no início do laço e daí então rodar o laço até a condição teste tornar-se verdadeira.

Exemplo:

```
i = 1
Do Until i = 11
  Cells(i, 1) = i
  i = i + 1
Loop
```

Este processo de laço conduz o mesmo resultado que no exemplo de estruturas **For ... Next**.

Do ... Loop While

Quando você quiser garantir que o laço rodará no mínimo uma vez, você pode colocar o teste no final do laço. O laço parará quando a condição tornar-se falsa. (compare esta estrutura de laço ao laço Do ... While.)

Por Exemplo:

```
i = 1
Do
  Cells(i, 1) = i
  i = i + 1
Loop While i < 11
```

Este processo de laço conduz ao mesmo resultado que no exemplo das estruturas **For ... Next** exemplo.

Do ... Loop Until

Esta estrutura de laço, como o **Do ... Loop While**, garante que o laço rodará no mínimo uma vez, você pode colocar o teste no final do laço. O laço pára quando a condição tornar-se verdadeira. (compare esta estrutura de laço com o Laço **Do ... Until**.)

Por Exemplo:

```
i = 1
Do
  Cells(i, 1) = i
  i = i + 1
Loop Until i = 11
```

Este processo de laço conduz ao mesmo resultado que no exemplo das estruturas **For ... Next**.