PROGRAMACIÓN CON VBA: MACROS

EXCEL 2010

Manual de Referencia para usuarios

Salomón Ccance
CCANCE WEBSITE

PROGRAMACIÓN CON VBA: MACROS

Para comenzar a programar en VBA tenemos que entrar al editor de Visual Basic desde el menú **Herramientas**, seleccione **Macro** y, a continuación, haga clic en **Editor de Visual Basic**.

Utilizar instrucciones condicionales para tomar decisiones

Las instrucciones condicionales evalúan si una condición es **True** o **False** y a continuación especifican las instrucciones a ejecutar en función del resultado. Normalmente, una condición es una expresión que utiliza un operador de comparación para comparar un valor o variable con otro.

Elegir la instrucción condicional a utilizar

- **If...Then...Else**: Salto a una instrucción cuando una condición es **True** o **False**
- **Select Case**: Selección de la instrucción a ejecutar en función de un conjunto de condiciones

Utilizar bucles para repetir código

Empleando bucles es posible ejecutar un grupo de instrucciones de forma repetida. Algunos bucles repiten las instrucciones hasta que una condición es **False**, otros las repiten hasta que la condición es **True**. Hay también bucles que repiten un conjunto de instrucciones un número determinado de veces o una vez para cada objeto de una colección.

Elegir el bucle a utilizar

- **Do...Loop**: Seguir en el bucle mientras o hasta una condición sea **True**.
- **For...Next**: Utilizar un contador para ejecutar las instrucciones un número determinado de veces.
- **For Each...Next**: Repetición del grupo de instrucciones para cada uno de los objetos de una colección.

Ejecutar varias instrucciones sobre el mismo objeto

Normalmente, en Visual Basic, debe especificarse un objeto antes de poder ejecutar uno de sus métodos o cambiar una de sus propiedades. Se puede usar la instrucción **With** para especificar un objeto una sola vez para una serie completa de instrucciones.

- With: Ejecutar una serie de instrucciones sobre el mismo objeto

Instrucción If...Then...Else

Se puede usar la instrucción **If...Then...Else** para ejecutar una instrucción o bloque de instrucciones determinadas, dependiendo del valor de una condición. Las instrucciones **If...Then...Else** se pueden anidar en tantos niveles como sea necesario. Sin embargo, para hacer más legible el código es

aconsejable utilizar una instrucción **Select Case** en vez de recurrir a múltiples niveles de instrucciones **If...Then...Else** anidadas.

Ejecutar una sola instrucción cuando una condición es True

Para ejecutar una sola instrucción cuando una condición es **True**, se puede usar la sintaxis de línea única de la instrucción **If...Then...Else**. El siguiente ejemplo muestra la sintaxis de línea única, en la que se omite el uso de la palabra clave **Else**:

```
Sub FijarFecha()
    miFecha = #13/2/95#
    If miFecha < Now Then miFecha = Now
End Sub
```

Para ejecutar más de una línea de código, es preciso utilizar la sintaxis de múltiples líneas. Esta sintaxis incluye la instrucción **End If**, tal y como muestra el siguiente ejemplo:

```
Sub AvisoUsuario(valor as Long)
    If valor = 0 Then
        Aviso.ForeColor = "Red"
        Aviso.Font.Bold = True
        Aviso.Font.Italic = True
    End If
End Sub
```

Ejecutar unas instrucciones determinadas si una condición es True y ejecutar otras si es False

Use una instrucción **If...Then...Else** para definir dos bloques de instrucciones ejecutables: un bloque que se ejecutará cuando la condición es **True** y el otro que se ejecutará si la condición es **False**.

```
Sub AvisoUsuario(valor as Long)
    If valor = 0 Then
        Aviso.ForeColor = vbRed
        Aviso.Font.Bold = True
        Aviso.Font.Italic = True
    Else
        Aviso.Forecolor = vbBlack
        Aviso.Font.Bold = False
        Aviso.Font.Italic = False
    End If
End Sub
```

Comprobar una segunda condición si la primera condición es False

Se pueden añadir instrucciones **ElseIf** a una instrucción **If...Then...Else** para comprobar una segunda condición si la primera es **False**. Por ejemplo, el siguiente procedimiento función calcula una bonificación salarial dependiendo de la clasificación del trabajador. La instrucción que sigue a la instrucción **Else** sólo se ejecuta cuando las condiciones de todas las restantes instrucciones **If** y **ElseIf** son **False**.


```
Function Bonificación(rendimiento, salario)
    If rendimiento = 1 Then
        Bonificación = salario * 0.1
    ElseIf rendimiento = 2 Then
        Bonificación= salario * 0.09
```

```
    ElseIf rendimiento = 3 Then
        Bonificación = salario * 0.07
    Else
        Bonificación = 0
    End If
End Function
```

Instrucción Select Case

La instrucción **Select Case** se utiliza como alternativa a las instrucciones **ElseIf** en instrucciones **If...Then...Else** cuando se compara una expresión con varios valores diferentes. Mientras que las instrucciones **If...Then...Else** pueden comparar una expresión distinta para cada instrucción **ElseIf**, la instrucción **Select Case** compara únicamente la expresión que evalúa al comienzo de la estructura de control.

En el siguiente ejemplo, la instrucción **Select Case** evalúa el argumento rendimiento que se pasa al procedimiento. Observe que cada instrucción **Case** puede contener más de un valor, una gama de valores, o una combinación de valores y operadores de comparación. La instrucción opcional **Case Else** se ejecuta si la instrucción **Select Case** no encuentra ninguna igualdad con los valores de la instrucciones **Case**.

```
Function Bonificación(rendimiento, salario)
    Select Case rendimiento
        Case 1
            Bonificación = salario * 0.1
        Case 2, 3
            Bonificación = salario * 0.09
        Case 4 To 6
            Bonificación = salario * 0.07
        Case Is > 8
            Bonificación = 100
        Case Else
            Bonificación = 0
    End Select
End Function
```

Instrucción Do...Loop

Se pueden usar instrucciones **Do...Loop** para ejecutar un bloque de instrucciones un número indefinido de veces. Las instrucciones se repiten mientras una condición sea **True** o hasta que llegue a ser **True**.

Repetir instrucciones mientras una condición es True

Hay dos formas de utilizar la palabra clave **While** para comprobar el estado de una condición en una instrucción **Do...Loop**. Se puede comprobar la condición antes de entrar en el bucle, o después de que el bucle se haya ejecutado al menos una vez.

En el siguiente procedimiento ComPrimeroWhile, la condición se comprueba antes de entrar en el bucle. Si miNum vale 9 en vez de 20, las instrucciones contenidas en el bucle no se ejecutarán nunca. En el

procedimiento ComFinalWhile, las instrucciones contenidas en el bucle sólo se ejecutarán una vez antes de que la condición llegue a ser **False**.

```
Sub ComPrimeroWhile()
    contador = 0
    miNum = 20
    Do While miNum > 10
        miNum = miNum - 1
        contador = contador + 1
    Loop
    MsgBox "El bucle se ha repetido " & contador & " veces."
End Sub

Sub ComFinalWhile()
    contador = 0
    miNum = 9
    Do
        miNum = miNum - 1
        contador = contador + 1
    Loop While miNum > 10
    MsgBox "El bucle se ha repetido " & contador & " veces."
End Sub
```

Repetir instrucciones hasta que una condición llegue a ser True

Hay dos formas de utilizar la palabra clave **Until** para comprobar el estado de una condición en una instrucción **Do...Loop**. Se puede comprobar la condición antes de entrar en el bucle (como muestra el procedimiento ComPrimeroUntil) o se pueden comprobar después de que el bucle se haya ejecutado al menos una vez (como muestra el procedimiento ComFinalUntil). El bucle sigue ejecutándose mientras la condición siga siendo **False**.

```
Sub ComPrimeroUntil()
    contador = 0
    miNum = 20
    Do Until miNum = 10
        miNum = miNum - 1
        contador = contador + 1
    Loop
    MsgBox "El bucle se ha repetido " & contador & " veces."
End Sub

Sub ComFinalUntil()
    contador = 0
    miNum = 1
    Do
        miNum = miNum + 1
        contador = contador + 1
    Loop Until miNum = 10
    MsgBox "El bucle se ha repetido " & counter & " veces."
End Sub
```


Instrucción de salida de Do...Loop desde dentro del bucle

Es posible salir de **Do...Loop** usando la instrucción **Exit Do**. Por ejemplo, para salir de un bucle sin fin, se puede usar la instrucción **Exit Do** en el bloque de instrucciones **True** de una instrucción **If...Then...Else** o **Select Case**. Si la condición es **False**, el bucle seguirá ejecutándose normalmente.

En el siguiente ejemplo, se asigna a miNum un valor que crea un bucle sin fin. La instrucción **If...Then...Else** comprueba esa condición y ejecuta entonces la salida, evitando así el bucle sin fin.

```
Sub EjemploSalida()
    contador = 0
    miNum = 9
    Do Until miNum = 10
        miNum = miNum - 1
        contador = contador + 1
        If miNum < 10 Then Exit Do
    Loop
    MsgBox "El bucle se ha repetido " & contador & " veces."
End Sub
```

Nota Para detener la ejecución de un bucle sin fin, presione la tecla ESC o CTRL+PAUSE.

Instrucción For...Next

Las instrucciones **For...Next** se pueden utilizar para repetir un bloque de instrucciones un número determinado de veces. Los bucles **For** usan una variable contador cuyo valor se aumenta o disminuye cada vez que se ejecuta el bucle.

El siguiente procedimiento hace que el equipo emita un sonido 50 veces. La instrucción **For** determina la variable contador x y sus valores inicial y final. La instrucción **Next** incrementa el valor de la variable contador en 1.

```
Sub Bips()
    For x = 1 To 50
        Beep
    Next x
End Sub
```

Mediante la palabra clave **Step**, se puede aumentar o disminuir la variable contador en el valor que se desee. En el siguiente ejemplo, la variable contador j se incrementa en 2 cada vez que se repite la ejecución del bucle. Cuando el bucle deja de ejecutarse, total representa la suma de 2, 4, 6, 8 y 10.

```
Sub DosTotal()
    For j = 2 To 10 Step 2
        total = total + j
    Next j
    MsgBox "El total es " & total
End Sub
```

Para disminuir la variable contador utilice un valor negativo en **Step**. Para disminuir la variable contador es preciso especificar un valor final que sea menor que el valor inicial. En el siguiente ejemplo, la variable contador miNum se disminuye en 2 cada vez que se repite el bucle. Cuando termina la ejecución del bucle, total representa la suma de 16, 14, 12, 10, 8, 6, 4 y 2.

```
Sub NuevoTotal()
    For miNum = 16 To 2 Step -2
        total = total + miNum
    Next miNum
    MsgBox "El total es " & total
End Sub
```

Nota No es necesario incluir el nombre de la variable contador después de la instrucción **Next**. En los ejemplos anteriores, el nombre de la variable contador se ha incluido para facilitar la lectura del código.

Se puede abandonar una instrucción **For...Next** antes de que el contador alcance su valor final, para ello se utiliza la instrucción **Exit For**. Por ejemplo, si se produce un error se puede usar la instrucción **Exit For** en el bloque de instrucciones **True** de una instrucción **If...Then...Else** o **Select Case** que detecte específicamente ese error. Si el error no se produce, la instrucción **If…Then…Else** es **False** y el bucle continuará ejecutándose normalmente.

Instrucción For Each...Next

Las instrucciones For Each...Next repiten un bloque de instrucciones para cada uno de los objetos de una colección o para cada elemento de una matriz. Visual Basic asigna valor automáticamente a una variable cada vez que se ejecuta el bucle. Por ejemplo, el siguiente procedimiento cierra todos los formularios excepto el que contiene al procedimiento que se está ejecutando.

```
Sub CierraFormul()
    For Each frm In Application.Forms
        If frm.Caption <> Screen.ActiveForm.Caption Then frm.Close
    Next
End Sub
```

El siguiente código recorre todos los elementos de una matriz e introduce en cada uno de ellos el valor de la variable índice I.

```
Dim PruebaMatriz(10) As Integer, I As Variant
For Each I In PruebaMatriz
    PruebaMatriz(I) = I
Next I
```

Recorrer un conjunto de celdas

Se puede usar el bucle **For Each...Next** para recorrer las celdas pertenecientes a un rango determinado. El siguiente procedimiento recorre las celdas del rango A1:D10 de la Página1 y convierte cualquier valor absoluto menor de 0,01 en 0 (cero).

```
Sub RedondeoACero()
    For Each miObjeto in miColeccion
        If Abs(miObjeto.Value) < 0.01 Then miObjeto.Value = 0
    Next
End Sub
```

Salir de un bucle For Each...Next antes de que finalice

Se puede salir de un bucle **For Each...Next** mediante la instrucción **Exit For**. Por ejemplo, cuando se produce un error se puede usar la instrucción **Exit For** en el bloque de instrucciones **True** de una instrucción **If...Then...Else** o **Select Case** que detecte específicamente el error. Si el error no se produce, la instrucción **If…Then…Else** es **False** y el bucle se seguirá ejecutando normalmente.

El siguiente ejemplo detecta la primera celda del rango A1:B5 que no contiene un número. Si se encuentra una celda en esas condiciones, se presenta un mensaje en pantalla y **Exit For** abandona el bucle.

```
Sub BuscaNumeros()
    For Each miObjeto In MiColeccion
        If IsNumeric(miObjeto.Value) = False Then
            MsgBox "El objeto contiene un valor no numérico."
            Exit For
        End If
    Next c
End Sub
```

Instrucción With

La instrucción **With** permite especificar una vez un objeto o tipo definido por el usuario en una serie entera de instrucciones. Las instrucciones **With** aceleran la ejecución de los procedimientos y ayudan a evitar el tener que escribir repetidas veces las mismas palabras.

El siguiente ejemplo introduce en un rango de celdas el número 30, aplica a esas celdas un formato en negrita y hace que su color de fondo sea el amarillo.


```
Sub RangoFormato()
    With Worksheets("Hoja1").Range("A1:C10")
        .Value = 30
        .Font.Bold = True
        .Interior.Color = RGB(255, 255, 0)
```

```
    End With
End Sub
```

Las instrucciones **With** se pueden anidar para aumentar su eficiencia. El siguiente ejemplo inserta una formula en la celda A1 y selecciona a continuación el tipo de letra.

```
Sub MiEntrada()
    With Workbooks("Libro1").Worksheets("Hoja1").Cells(1, 1)
        .Formula = "=SQRT(50)"
        With .Font
            .Name = "Arial"
            .Bold = True
            .Size = 8
        End With
    End With
End Sub
```