
cofide® CAPACITACIÓN EMPRESARIAL
LE DAMOS LA MÁS CORDIAL
bienvenida al curso:
Automatización de Procesos
con Macros en Excel
Ing. Rodolfo González M.
www.cofide.mx

Objetivo

Aprender a automatizar tareas repetitivas a fin de aumentar la productividad al trabajar con Excel, conocer qué son las macros, su uso, finalidad y cómo se realizan a fin de lograr automatizar procedimientos repetitivos en Excel. Aprender la forma de utilizar el editor de Visual Basic para desarrollar y probar las macros, conocer la estructura del lenguaje de programación, tipos de datos, operadores y funciones; desarrollarán macros para automatizar procedimientos.

Temario

Introducción a las macros

Macros de grabación

Editor de Visual Basic

Diseño de rutinas

Manejo de variables

Instrucciones de control

Macros de movimiento

Introducción a las macros

MACRO

Una macro consiste en una serie de comandos y funciones que se almacenan en un módulo de Visual Basic y que está disponible siempre que sea necesario ejecutar la tarea. Una macro se graba igual que se graba música en un casete; a continuación, se ejecuta la macro para que repita los comandos.

Antes de grabar o escribir una macro, planifique los pasos y los comandos que desea que ejecute la macro. Si se comete algún error mientras se graba la macro, también se grabarán las correcciones que se realicen. Cada vez que se grabe una macro, ésta se almacenará en un nuevo módulo adjunto a un libro.

Con el Editor de Visual Basic, se pueden modificar macros, copiar macros de un módulo en otro, copiar macros entre diferentes libros, cambiar de nombre a los módulos que almacenan las macros o cambiar de nombre a las macros.

FORMAS DE HACER UNA MACRO EN EXCEL

Se puede hacer una Macro de dos maneras:

Grabación de Macros

Se refiere a la grabación de todos los pasos que se van realizando, al mismo tiempo que Excel genera código de los pasos ejecutados.

Creación de Macros

Se refiere a la codificación de una macro, se escribe el programa mediante el uso del editor de Visual Basic.

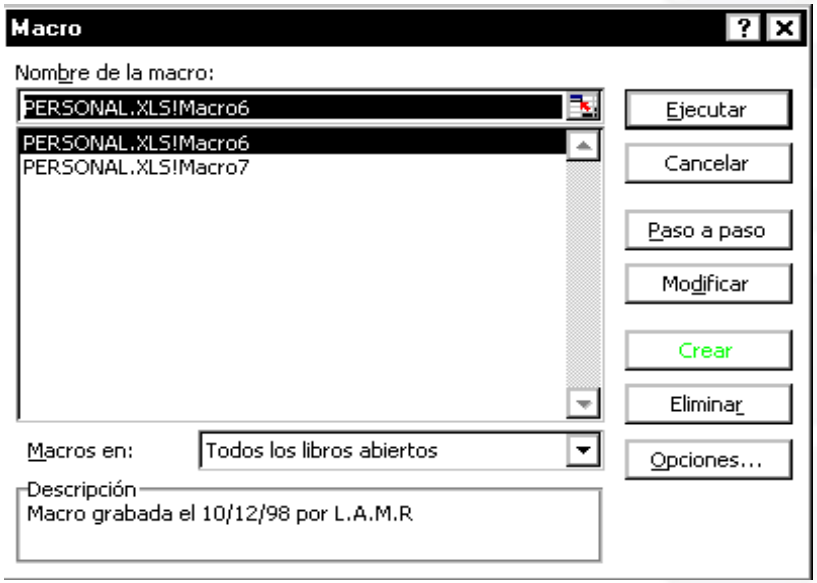
EJECUCIÓN DE UNA MACRO

Una vez grabada, una macro puede ejecutarse en Microsoft Excel o en el Editor de Visual Basic. Normalmente, se ejecutará la macro en Microsoft Excel; sin embargo, puede ejecutarse desde el Editor de Visual Basic, mientras se realiza la macro. Para interrumpir la macro antes de que finalice las acciones que se han grabado, presione ESC

1. Abra el libro que contiene la macro.

2. Seleccione Macro en el *Menú* Herramientas o pestaña de programador y, a continuación, haga clic en Macros.

3. En el cuadro Nombre de la macro, escriba el nombre de la macro que desea ejecutar.



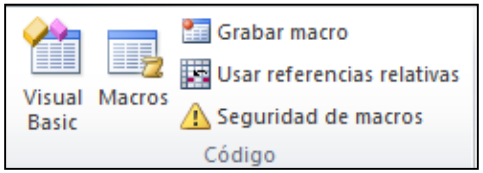
Macros de grabación

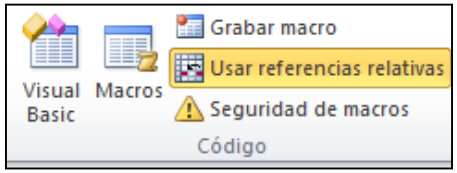
Macros de Grabación: Herramientas

•Seleccione Usar referencias relativas en la carpeta de Programador

Si se seleccionan celdas mientras se está ejecutando una macro, ésta seleccionará las mismas celdas independientemente de la celda que se haya seleccionado en primer lugar, ya que graba referencias absolutas de celda. Si desea tener una macro para seleccionar celdas independientemente de la posición que tenga la celda activa cuando se ejecute la macro, configure el grabador de macros para que grabe referencias relativas de celda. En la barra de herramientas Detener grabación, haga clic en Referencia.

Microsoft Excel continuará grabando macros con referencias relativas hasta que termine la sesión con Microsoft Excel o hasta que haga clic otra vez en Referencias relativas.

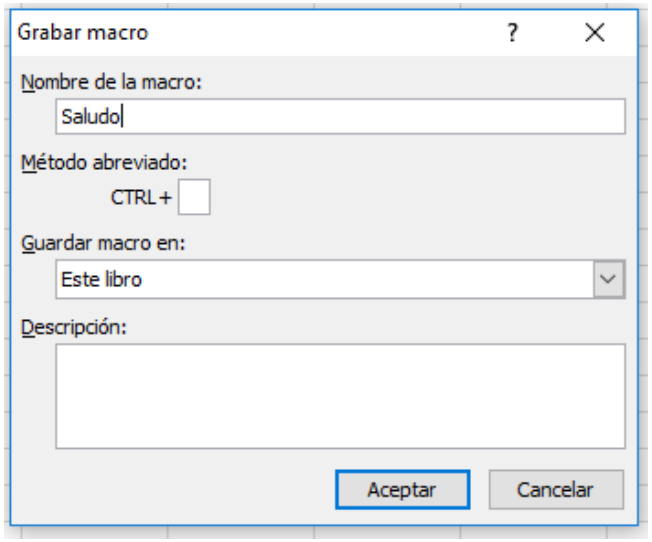




Macros de Grabación: Herramientas

•Seleccione Grabar Macro en la carpeta de Programador

En el cuadro Nombre de la macro, escriba un nombre para la macro.



Macros de Grabación

•El primer carácter del nombre de la macro debe ser una letra. Los demás caracteres pueden ser letras, números o caracteres de subrayado. No se permiten espacios en un nombre de macro; puede utilizarse un carácter de subrayado como separador de palabras.

•Para ejecutar la macro presionando un método abreviado, escriba una letra en el cuadro Tecla de método abreviado. Puede utilizarse CONTROL + letra (para letras minúsculas) o CONTROL + MAYÚS + letra (para letras mayúsculas), donde letra es cualquier tecla del teclado. La tecla de método abreviado que se utilice no puede ser ni un número ni un carácter especial. La tecla de método abreviado suplantará a cualquier tecla de método abreviado predeterminada en Microsoft Excel mientras esté abierto el libro que contiene la macro.

•En el cuadro Guardar macro en, haga clic en la ubicación en que desea almacenar la macro. Si desea que la macro esté disponible siempre que se utilice Microsoft Excel, almacene la macro en el libro de macros personales en la carpeta INICIAR. Para incluir una descripción de la macro, escriba la descripción en el cuadro Descripción.

•Haga clic en Aceptar.

Macros de Grabación: Creación

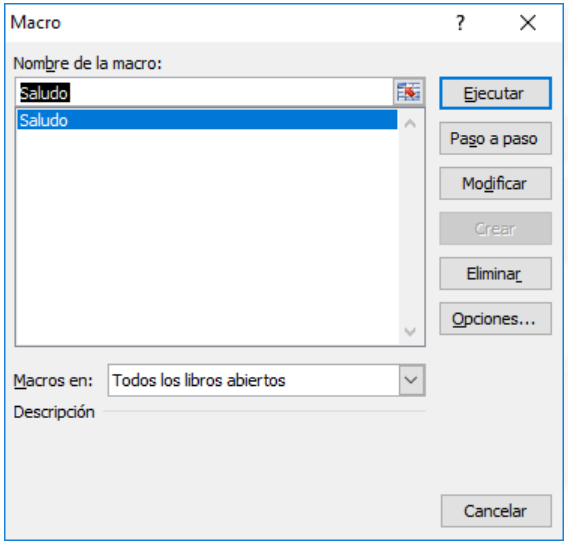
1. Defina el escenario de arranque y término de la macro
2. Trate de usar métodos abreviados del teclado si le es posible

3. Defina los pasos a realizar en la macro
4. Practique la ejecución y determine si todos los pasos son correctos

5. Active le grabadora de Macros
6. Defina nombre para la macro
7. Inicie la grabación
8. Ejecute los pasos previamente practicados
9. Termine la grabación
10. Guarde el archivo
11. Ponga a prueba su trabajo !

Macros de Grabación: Ejecución

Una vez grabada, una macro puede ejecutarse en Microsoft Excel o en el Editor de Visual Basic. Normalmente, se ejecutará la macro en Microsoft Excel; sin embargo, puede ejecutarse desde el Editor de Visual Basic, mientras se realiza la macro. Para interrumpir la macro antes de que finalice las acciones que se han grabado, presione ESC.



1. Guarde el libro que contiene la macro.
2. Seleccione Macros de la pestaña de programador
3. Seleccione el nombre de la macro, y de ejecutar

Editor de Visual Basic

Para poder mostrar el entorno de Visual Basic:

1. Seleccione Macro en el menú Herramientas o desarrollador y, a continuación, haga clic en Editor de Visual Basic.

2. Aparece la ventana del Editor el cual consta de las siguientes partes:

Barra de Título

Es la barra horizontal situada en la parte superior de la pantalla, contiene el nombre de la aplicación.

Barra de Menús

Proporciona las herramientas necesarias para desarrollar, probar y archivar la aplicación.

Barra de Herramientas:

Contiene botones que tienen accesos directos a algunos elementos de menú utilizados con frecuencia.

Puede hacer clic una sola vez en un botón de la barra de herramientas para realizar la acción representada por el botón. Puede seleccionar la opción Información sobre herramientas de la ficha General del cuadro de diálogo Opciones si desea mostrar información sobre los botones de la barra de herramientas

Guardar <nombre de documento principal >

Guarda el documento principal, incluidos el proyecto y todos sus componentes: formularios y módulos

Cortar

Quita el control o texto seleccionado y lo coloca en el Portapapeles.

Copiar

Copia el control o texto seleccionado en el Portapapeles.

Pegar

Inserta el contenido del Portapapeles en la ubicación actual del cursor.

Buscar

Abre el cuadro de diálogo Buscar y busca el texto especificado en el cuadro Buscar.

Deshacer

Deshace la última acción de edición.

Rehacer

Restaura las últimas acciones descartadas de edición de texto si no se han realizado otras acciones desde la última operación de Deshacer.

Ejecutar Sub/UserForm o Ejecutar macro

Ejecuta el procedimiento actual si el cursor está en un procedimiento, ejecuta el UserForm si un UserForm está activo actualmente o ejecuta una macro si no está activa la ventana Código ni un UserForm.

Interrumpir

Detiene la ejecución de un programa y cambia al modo de interrupción.

Restablecer <proyecto>

Borra las variables de nivel de módulo de la pila de ejecución y restablece el proyecto

Modo de diseño

Activa y desactiva el modo de diseño.

Explorador de proyectos

Abre el Explorador de proyectos que muestra una lista jerárquica de los proyectos abiertos actualmente y su contenido.

Ventana de Propiedades

Abre la ventana de Propiedades para que puedan verse las propiedades del control seleccionado.

Examinador de objetos

Muestra el Examinador de objetos, que presenta una lista de bibliotecas de objetos, biblioteca de tipos, clases, métodos, propiedades, eventos y constantes que se pueden utilizar en código, así como los módulos y procedimientos definidos para el proyecto.

Cuadro de herramientas

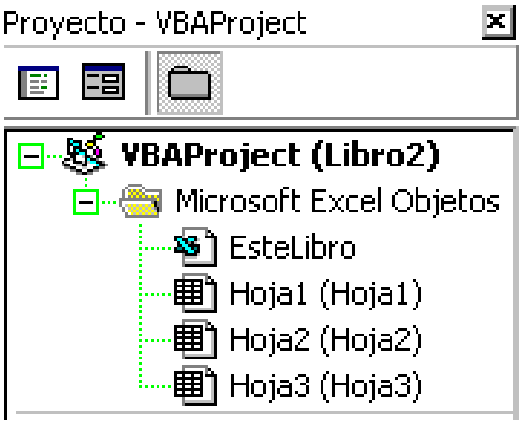
Muestra u oculta el cuadro de herramientas que contiene todos los controles y los objetos insertables (Como un gráfico de Microsoft Excel) disponibles para la aplicación. Sólo está disponible cuando está activo un UserForm.

Explorador de Proyectos

El Explorador de proyectos muestra una lista jerárquica de los proyectos y todos los elementos contenidos o que hace referencia en cada proyecto.

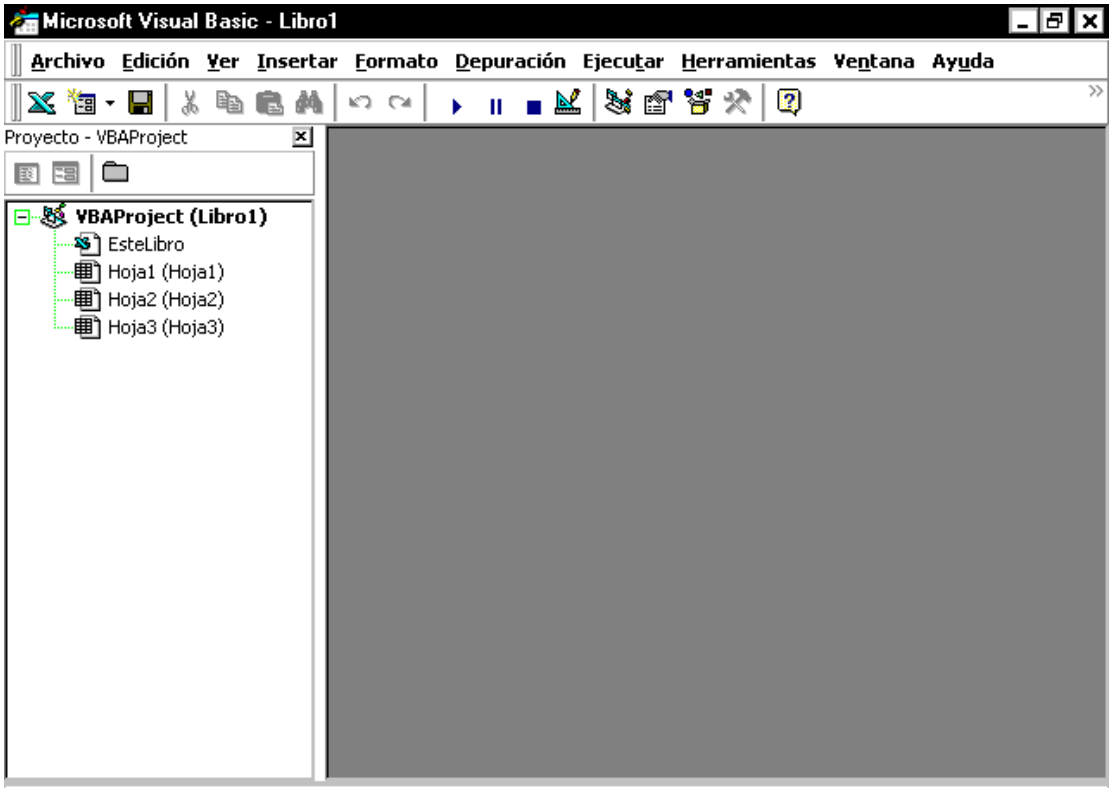
1. En el menú Ver, elija Explorador de proyectos (CTRL+R) o utilice el cuadro de herramientas abreviado:

ELEMENTOS DE LA  ***VENTANA***



La ventana del formulario inicial

Ocupa la mayor parte del centro de la pantalla. En ella es donde se personaliza la ventana que verán los usuarios. La documentación de Visual Basic utiliza el término form (formulario) para una ventana personalizable.



Diseño de rutinas

PROCEDIMIENTOS

Definición

Son una secuencia con nombre de instrucciones que se ejecutan como una unidad. Por ejemplo, Function, Property y Sub son todos tipos de procedimientos. Un nombre de procedimiento siempre se define a nivel de módulo. Todo el código ejecutable debe estar contenido en un procedimiento. Los procedimientos no se pueden anidar dentro de otros procedimientos.

Procedimiento Sub

Un procedimiento Sub es una serie de instrucciones Visual Basic, encerradas entre un par de instrucciones Sub y End Sub, que realizan acciones específicas pero no devuelven ningún valor. Un procedimiento Sub puede aceptar argumentos, como constantes, variables o expresiones que le pasa el procedimiento que ha efectuado la llamada. Si un procedimiento Sub no tiene argumentos, la instrucción Sub debe incluir un par de paréntesis vacío.

Sintaxis

```
Sub

Sub nombre [(lista_argumentos)]

    [instrucciones]

    [Exit Sub]

    [instrucciones]

End Sub
```

Ejemplo de la Instrucción Sub

```
Sub saludo()
    MsgBox ("bienvenido al curso de macros")
End Sub
```

INPUTBOX

Muestra un mensaje en un cuadro de diálogo, espera que el usuario escriba un texto o haga clic en un botón y devuelve un tipo String con el contenido del cuadro de texto.

imputbox(prompt)

Sintaxis

InputBox(prompt[, title][, default][, xpos][, ypos][, helpfile, context])

La sintaxis de la función InputBox consta de estos argumentos con nombre:

prompt

Expresión de cadena que se muestra como mensaje en el cuadro de diálogo. La longitud máxima de prompt es de aproximadamente 1024 caracteres, según el ancho de los caracteres utilizados. Si prompt consta de más de una línea, puede separarlos utilizando un carácter de retorno de carro (Chr(13)), un carácter de avance de línea (Chr(10)) o una combinación de los caracteres de retorno de carro-avance de línea (Chr(13) y Chr(10)) entre cada línea y la siguiente.

Title (Opcional).

Expresión de cadena que se muestra en la barra de título del cuadro de diálogo. Si omite title, en la barra de título se coloca el nombre de la aplicación.

Default(Opcional).

Expresión de cadena que se muestra en el cuadro de texto como respuesta predeterminada cuando no se suministra una cadena. Si omite default, se muestra el cuadro de texto vacío.

xpos (Opcional).

MsgBox

Muestra un mensaje en un cuadro de diálogo, espera a que el usuario haga clic en un botón y devuelve un tipo Integer correspondiente al botón elegido por el usuario.

Sintaxis

Respuesta=msgbox(“pregunta”)

Sintaxis

MsgBox(prompt[, buttons][, title][, helpfile, context])

La sintaxis de la función MsgBox consta de estos argumentos con nombre:

prompt	Expresión de cadena que representa el prompt en el cuadro de diálogo. La longitud máxima de prompt es de aproximadamente 1024 caracteres, según el ancho de los caracteres utilizados. Si prompt consta de más de una línea, puede separarlos utilizando un carácter de retorno de carro (Chr(13)) o un carácter de avance de línea (Chr(10)), o una combinación de caracteres de retorno de carro-avance de línea (Chr(13) y Chr(10)) entre cada línea y la siguiente.

buttons	Opcional. Expresión numérica que corresponde a la suma de los valores que especifican el número y el tipo de los botones que se pretenden mostrar, el estilo de icono que se va a utilizar, la identidad del botón predeterminado y la modalidad del cuadro de mensajes. Si se omite este argumento, el valor predeterminado para buttons es 0..
title	Opcional. Expresión de cadena que se muestra en la barra de título del cuadro de diálogo. Si se omite title, en la barra de título se coloca el nombre de la aplicación.
helpfile	Opcional. Expresión de cadena que identifica el archivo de Ayuda que se utiliza para proporcionar ayuda interactiva en el cuadro de diálogo. Si se especifica helpfile, también se debe especificar context.
context	Opcional. Expresión numérica que es igual al número de contexto de Ayuda asignado por el autor al tema de Ayuda correspondiente. Si se especifica context, también se debe especificar helpfile.

El argumento buttons tiene estos valores:

Constante	Valor	Descripción
vbOKOnly	0	Muestra solamente el botón Aceptar.
VbOKCancel	1	Muestra los botones Aceptar y Cancelar.
VbAbortRetryIgnore	2	Muestra los botones Anular, Reintentar e Ignorar.
VbYesNoCancel	3	Muestra los botones Sí, No y Cancelar.
VbYesNo	4	Muestra los botones Sí y No.

Estas constantes las especifica Visual Basic for Applications. Por tanto, el nombre de las mismas puede utilizarse en cualquier lugar del código en vez de sus valores reales.

Valores devueltos

Constante	Valor	Descripción
vbOK	1	Aceptar
vbCancel	2	Cancelar
vbAbort	3	Anular
vbRetry	4	Reintentar
vbIgnore	5	Ignorar
vbYes	6	Sí
vbNo	7	No

MOVIMIENTO DE LA CELDA ACTIVE

ACTIVECELL.OFFSET(FILA,COLUMNA).SELECT

```
Sub mover()

    ActiveCell.Offset(0, 4).Select

End Sub
```

ESCRIBIR EN LA CELDA ACTIVE

ACTIVECELL.VALUE = VALOR

```
Sub mover()
    ActiveCell.Offset(0, 4).Select
    ActiveCell.Value = "Curso"
End Sub
```

LEER EN LA CELDA ACTIVE

VARIABLE ALMACÉN = ACTIVECELL.VALUE

```
Sub copiar()

    dato = ActiveCell.Value

    ActiveCell.Offset(0, 4).Select

    ActiveCell.Value = dato

End Sub
```

Manejo de variables

DECLARAR VARIABLES

Definición de variable

Un lugar de almacenamiento con nombre que puede contener cierto tipo de datos que puede ser modificado durante la ejecución del programa. Cada variable tiene un nombre único que la identifica dentro de su nivel de ámbito. Puede especificar un tipo de datos o no.

Los nombres de variable deben comenzar con un carácter alfabético, deben ser únicos dentro del mismo ámbito, no deben contener más de 255 caracteres y no pueden contener un punto o carácter de declaración de tipo.

Declaración de Variables

Para declarar variables, se utiliza normalmente una instrucción Dim. La instrucción de declaración puede incluirse en un procedimiento para crear una variable de nivel de procedimiento. O puede colocarse al principio de un módulo, en la sección Declarations, para crear una variable de nivel de módulo.

Las variables se pueden declarar como uno de los siguientes tipos de datos: Boolean, Byte, Integer, Long, Currency, Single, Double, Date, String (para cadenas de longitud variable), String * longitud (para cadenas de longitud fija), Object, o Variant. Si no se especifica el tipo de datos, el tipo de datos Variant es el predefinido. También es posible crear un tipo definido por el usuario empleando la instrucción Type.

Tipos de Datos

Característica de una variable que determina qué tipo de datos puede tener. Los tipos de datos incluyen Byte, Boolean, Integer, Long, Currency, Single, Double, Date, String, Object, Variant (predeterminado) y tipos definidos por el usuario, así como tipos específicos de objetos

String (Texto)

Tipo de datos que consiste en una secuencia de caracteres que representa a los caracteres por sí mismos en vez de sus valores numéricos. Un tipo String puede incluir letras, números, espacios en blanco y signos de puntuación. El tipo de datos String puede almacenar cadenas de longitud fija en un intervalo de 0 a aproximadamente 63000 caracteres y cadenas dinámicas en un intervalo de longitud de 0 a aproximadamente 2 mil millones de caracteres. El carácter de declaración de tipo, es el signo de dólar ($) que representa el tipo String en Visual Basic.

Boolean

Las variables tipo Boolean se almacenan como números de 16 bits (2 bytes), pero sólo pueden ser True o False. Las variables tipo Boolean se presentan como True o False.

(Cuando se utiliza Print) o #TRUE# o #FALSE# (cuando se utiliza Write #). Utilice las palabras clave True y False para asignar uno de los dos estados a las variables tipo Boolean.

Cuando se convierten a tipo Boolean otros tipos numéricos, 0 se convierte en False, y el resto de los valores se convierten en True. Cuando los valores tipo Boolean se convierten a otros tipos de datos numéricos, False se convierta en 0 y True se convierte en -1.

Byte

Las variables tipo Byte se almacenan como números de 8 bits (1 byte) sencillos sin signo con un intervalo de valores entre 0 y 225.
El tipo de dato Byte es útil para almacenar datos binarios.

Variant

El tipo de datos Variant se especifica automáticamente si no se especifica otro tipo de datos al declarar una constante, variable, o argumento. Las variables declaradas como del tipo de datos Variant pueden contener valores numéricos, cadenas de texto, fecha, hora o Booleans y pueden convertir los valores que contienen de forma automática. Los valores numéricos Variant ocupan 16 bytes de memoria (lo que sólo es significativo en procedimientos grandes o módulos complejos) y son más lentos a la hora de su acceso que las variables de tipo explícito de los restantes tipos. Es muy raro utilizar el tipo de datos Variant para una constante. Los valores de cadena Variant necesitan 22 bytes de memoria.

Currency

Las variables tipo Currency se almacenan como números de 64 bits (8 bytes) en un formato de número entero a escala de 10.000 para dar un número de punto fijo con 15 dígitos a la izquierda del signo decimal y 4 dígitos a la derecha. Esta representación proporciona un intervalo de -
922.337.203.685.477,5808 a 922.337.203.685.477,5807. El carácter de declaración de tipo para Currency es el signo @.

El tipo de datos Currency es útil para cálculos monetarios y para cálculos de punto fijo, en los cuales la precisión es especialmente importante.

Date

Las variables tipo Date, se almacenan como números IEEE de signo flotante de 64 bits (8 bytes) que van del 1 de enero del 100 al 31 de diciembre de 9999 y horarios de 0:00:00 a 23:59:59. Cualquier valor reconocible de fecha literal se puede asignar a las variables tipo Date. Los literales de fechas se deben poner entre caracteres de signo de número (#). Por ejemplo, #1 Enero, 1993# o #1 Ene 93#.

Las variables tipo Date, presentan fechas de acuerdo al formato de fecha corto reconocido por su sistema. La hora se presenta de acuerdo al formato de hora reconocido por su sistema (12 ó 24 horas).

Cuando se convierten a tipo Date otros tipos de datos numéricos, los valores a la izquierda del signo decimal representan la información de fecha, mientras que los valores a la derecha del signo decimal representan la hora. Medianoche es 0 y mediodía es 0,5. Los números enteros negativos representan fechas anteriores al 30 de diciembre de 1899.

Double

Las variables tipo Double (coma flotante y precisión doble), se almacenan como números IEEE de coma flotante de 64 bits (8 bytes) con valores de -1,797693134862 32E308 a -4,940656458412 47E-324 para valores negativos y de 4,940656458412 47E-324 a 1,797693134862 32E308 para valores positivos. El carácter de declaración de tipo para Double es el signo de número (#).

Integer

Las variables tipo Integer, se almacenan como números de 16 bits (2 bytes) con valores que van de -32.768 a 32.767. El carácter de declaración de tipo para el tipo Integer es el signo de porcentaje (%).

Las variables tipo Integer también se pueden utilizar para representar valores enumerados. Un valor enumerado puede contener un conjunto finito de números enteros únicos, cada uno de los cuales tiene un significado especial en el contexto en el que se utiliza. Los valores enumerados proporcionan una forma cómoda de seleccionar entre un número conocido de opciones. Por ejemplo, cuando se pregunta al usuario que elija un color de una lista, se podría tener 0 = negro, 1 = blanco y así sucesivamente. Es una buena práctica de programación definir constantes, utilizando la instrucción Const para cada valor enumerado.

Long

Las variables tipo Long (entero largo) se almacenan como números con signo de 32 bits (4 bytes) con un valor comprendido entre -2.147.483.648 y 2.147.483.647. El carácter de declaración de tipo para Long es el signo &.

Object

Las variables tipo Object se almacenan como direcciones de 32 bits (4 bytes) que hacen referencia a objetos. Al utilizar la instrucción Set, una variable declarada con tipo Object puede tener asignado cualquier referencia a un objeto.

Single

Las variables tipo Single (coma flotante y precisión simple) se almacenan como números IEEE de coma flotante de 32 bits (4 bytes) con valores que van de -3,402823E38 a -1,401298E-45 para valores negativos y de 1,401298E-45 a 3,402823E38 para valores positivos. El carácter de declaración de tipo para Single es el signo de exclamación (!).

Si no pudiste desarrollar el programa del capítulo o tuviste algún problema, solicítame el algoritmo !

hugorglez@hotmail.com, Rodolfo González

Instrucciones de control

Condiciones

UTILIZAR INSTRUCCIONES 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.

Sintaxis

Puede utilizar la siguiente sintaxis en formato de bloque:

```
If condición Then
    [instrucciones]
[ElseIf condición-n Then
    [instrucciones_elseif] ...
[Else
    [instrucciones_else]]
End If
```

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 califica()
   If (ActiveCell.Value > 6) Then
         ActiveCell.Offset(0, 1).Select
      ActiveCell.Value = "Aprobó"
   End If
End Sub
```

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

```
Sub determina_sexo()
   If (ActiveCell.Value = "h") Then
      ActiveCell.Offset(0, 1).Select
      ActiveCell.Value = "hombre"
   Else
      ActiveCell.Offset(0, 1).Select
      ActiveCell.Value = "Mujer"
   End If
End Sub
```

Comprobar una segunda condición si la primera condición es False

```
Sub dia()
    Dim num_dia As Integer
    num_dia = ActiveCell.Value
    If (num_dia = 1) Then
        ActiveCell.Offset(0, 1).Select
      ActiveCell.Value = "Lunes"
    ElseIf (num_dia = 2) Then
      ActiveCell.Offset(0, 1).Select
      ActiveCell.Value = "Martes"
    ElseIf (num_dia = 3) Then
      ActiveCell.Offset(0, 1).Select
      ActiveCell.Value = "Miercoles"
    ElseIf ...
          ...
        ...
    Else
        ActiveCell.Offset(0, 1).Select
      ActiveCell.Value = "Fin de semana"
    End If
End Sub
```

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.

sintaxis

Do While (condición)

[instrucciones]

[instrucciones]

Loop

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 Contar()
        contador = 0
        Do While contador < 10
                contador = contador + 1
            MsgBox  ( contador )
        Loop
End Sub
```

Macros de movimiento

Funciones:

Reg valor redondeado=Round (#, No de decimales)

Reg valor en mayúsculas= Ucase (texto)

Archivo de trabajo:

Seleccione la liga para descargar el archivo de Excel que se usa en el video

Descargar:

https://1drv.ms/x/s!AvugG6a26m5qmD9gZeRoCPYTG3tO?e=W44EQu

Si no pudiste desarrollar el programa del capítulo o tuviste algún problema, solicítame el algoritmo !

hugorglez@hotmail.com, Rodolfo González

Automatización de Procesos con Macros en Excel

.....Gracias

Ing. Rodolfo González M.

hugorglez@hotmail.com