

**E**  
**X**



**Universidad Pedagógica  
de Durango**

**C O L E C C I Ó N**

**CUADERNOS INFORMÁTICOS**

oooooooooooooooooooooooooooo  
oooooooooooooooooooooooooooo  
Número 3

**COORDINACIÓN DE DIFUSIÓN  
Y EXTENSIÓN UNIVERSITARIA**

**C**  
**E**  
**L**

**X P**

**2003**



Universidad Pedagógica  
de Durango

COLECCIÓN  
CUADERNOS INFORMÁTICOS

EXCEL

XP

2003

Martínez-Leiva ..... Número 3

COORDINACIÓN DE DIFUSIÓN  
Y EXTENSIÓN UNIVERSITARIA



## PRÓLOGO

Los procesos de educación se han visto influenciados en su desarrollo histórico por algunos acontecimientos que han sido producto de la capacidad creadora del hombre; así ha sucedido con la computación y la Internet, que han significado un salto cualitativo en la edificación del conocimiento y en el desarrollo de modelos de enseñanza.

Esta era se caracteriza por la cancelación de fronteras y límites en las distancias y en el tiempo, acceso a la información para la toma de decisiones, asimismo, ofrece un acceso inmediato a fuentes enciclopédicas de conocimiento, que anteriormente eran patrimonio de unos cuantos.

Uno de los grandes desafíos que se nos presentan es generar una cultura del manejo de la computación y de Internet, que contribuya a facilitar el acceso al conocimiento y a mejorar los niveles de preparación de los individuos. En este ámbito, a los docentes les corresponde un papel preponderante.

Una contribución importante a estos procesos de formación, lo constituyen los Cuadernos Informáticos, cuyos autores son el Mtro. Luis Manuel Martínez Hernández y la Mtra. Elizabeth Leyva Arellano, que en esta tercera entrega, nos proporcionan información presentada en forma didáctica y de fácil comprensión, para adentrarnos o profundizar en el uso de las herramientas de Excel.

## UNIVERSIDAD PEDAGÓGICA DE DURANGO

### DIRECTORIO

**Miguel Gerardo Ruvalcaba Álvarez**

Director General

**Alejandra Mendez Zúñiga**

Secretaria Académica

**Martín Arredondo Guerrero**

Coordinador de Docencia

**Jesús Flores García**

Coordinador de Investigación y Posgrado

**Jorge Gustavo Olvera Sierra**

Coordinador de Difusión y Extensión Universitaria

**Paula Elvira Ceceñas Torrero**

Coordinadora de Servicios de Apoyo Académico

### Serie

### Cuadernos Informáticos

Cuaderno Informático  
Excel XP 2003

Publicación Número 3.  
2da. Reimpresión

### Revisión y Corrección de Estilo

Paula Elvira Ceceñas Torrero  
Jesús Carrillo Álvarez

### Autores

Luis Manuel Martínez Hernández  
María Elizabeth Leyva Arellano

### Diseño

Luis Manuel Martínez Hernández

### Oficinas

Av. 16 de Septiembre #132  
Col. Silvestre Dorador  
C.P. 34070. Durango, Dgo.  
<http://www.upd.edu.mx>  
email: [webmaster@www.upd.edu.mx](mailto:webmaster@www.upd.edu.mx)  
Tel. y Fax: (618)128-6015 y 128-4407

Nota: Excel, Word, PowerPoint y Windows son marcas registradas de Microsoft, así como las demás marcas mencionadas en esta publicación son propiedad de sus respectivas Compañías.



## PRÓLOGO A LA SEGUNDA EDICIÓN

Existe hoy en la actualidad una influencia creciente de las Nuevas Tecnologías de la Información y Comunicación (NTIC) sobre las sociedades del mundo, siendo esta la clave para un progreso acelerado y mejora de la vida.

Pero la tecnología no alcanza a solucionar los problemas con profundidad, ya que dependen de contextos tan amplios, como la naturaleza, la sociedad y la vida de cada individuo. Sin embargo, la educación pretende realizar una contribución realista a los problemas a partir de la fragmentación o división del conocimiento en campos específicos de especialización.

Para llegar a comprender cómo la visión especializada de los problemas y dificultades llegó a invadir la totalidad de nuestras vidas, hay que entender el significado de fragmentación y como opera en la práctica.

La especialización fue el paso fundamental en el desarrollo de la civilización, así pues, para estudiar cualquier materia se comienza por el acto natural de abstracción, para poder así centrarse en ciertos rasgos de interés.

Para ser capaz de prestar atención a algo es necesario abstraer o aislar primero sus características principales de toda la infinita complejidad de su entorno. En la actualidad, el uso cada vez mas generalizado de las hojas de calculo en la vida cotidiana, ha hecho que las escuelas implementen programas remediales para enseñar este tipo de herramientas que se emplean en cualquier actividad de la vida diaria.

Siguiendo esta carrera sin fin en lo que es el uso de programas de cómputo, ponemos a su alcance la nueva versión del LIBRO EXCEL XP 2003, el cual le permitirá obtener un mayor provecho de esta herramienta.

La computación al igual que todas las cosas, está constantemente en proceso de evolución y cambio. En este proceso, los avances que se hacen en el área de las NTIC pueden tener importantes consecuencias para el establecimiento de teorías y conceptos en otros campos.

Esperando continuar con este tipo de actualización y cambio, le presentamos a usted el nuevo volumen del libro de EXCEL XP 2003, mismo que le ayudará en su vida diaria, así como en la investigación científica, ya que no podemos poner fin a esta avalancha de cambios de nuevos conocimientos.

Con este libro se termina la actualización de los contenidos de la colección de los Cuadernos Informáticos a las versiones más actuales y en los casos necesarios se han cambiado los contenidos, así como se han agregado nuevos contenidos, todo ello con la finalidad de mantenernos en la preferencia de nuestros lectores.

Esperamos seguir contando con su apoyo a este tipo de obras para poder continuar llevando la difusión de la cultura a todos los rincones de nuestro planeta.

## INDICE

<b>Introducción.....</b>	<b>1</b>
<b>Unidad 1 .....</b>	<b>3</b>
<b>Aspectos Generales .....</b>	<b>3</b>
Barra de menús .....	6
Barra de herramientas .....	7
Barras de reglas .....	7
Zona de edición .....	7
Barras de desplazamiento .....	7
Barra de estado .....	8
Menús contextuales .....	8
<b>Mover barras de herramientas .....</b>	<b>8</b>
Cambio de tamaño .....	9
Posición original .....	11
<b>Mostrar barras de herramientas .....</b>	<b>11</b>
<b>Ocultar barras de herramientas .....</b>	<b>13</b>
<b>Conceptos básicos .....</b>	<b>14</b>
Desplazamiento por hoja .....	15
<b>Selección de un rango de celdas .....</b>	<b>15</b>
Escribir, modificar y borrar texto .....	16
Modificar y borrar el texto .....	17
Copiar una hoja .....	21
Mover una hoja .....	22
Eliminar una hoja .....	22
<b>Tipos de datos .....</b>	<b>22</b>
Valores numéricos.....	23
Valores de texto .....	23
Fechas y horas .....	23
Copiar y mover celdas .....	24
Dar nombre a las celdas.....	24
Guardar el trabajo .....	25



<b>Unidad 2</b>	<b>29</b>
Formato de celdas	29
Auformato	31
Alineación de datos	31
Tipos y fuentes de letra	32
Bordes, rellenos y color de letra	33
Formato de números	33
Listas	34
Creación de Formulario	35
Ordenar una lista de datos	37
Validación de datos	38
Funciones especiales de búsqueda	40
Filtrado de datos	41
 <b>Unidad 3</b>	 <b>43</b>
Copiar, mover y seriación de datos	43
Creación de series	44
Copiar y mover celdas	47
Pegado especial	47
Insertar y eliminar filas y columnas	49
Buscar y reemplazar datos	50
Selección de celdas no adyacentes	51
Llenar datos en un rango	51
Borrar todos los datos de la hoja	52
Inmovilizar paneles	52
División de ventanas	53
 <b>Unidad 4</b>	 <b>55</b>
La sintaxis de la fórmula	55
Prioridad de las fórmulas	56
Mensajes de error	56
Referencias	59

<b>Unidad 5</b>	<b>61</b>
<b>Funciones</b>	<b>61</b>
Sintaxis de una función	61
La función autosuma	61
La función PROMEDIO	63
<b>El Asistente de Funciones</b>	<b>63</b>
Sugerir una función	66
<b>Funciones Anidadas</b>	<b>66</b>
 <b>Unidad 6</b>	 <b>67</b>
Crear y modificar una gráfica	71
 <b>Unidad 7</b>	 <b>79</b>
Constantes matriciales	81
<b>Vínculos y referencias en Excel</b>	<b>82</b>
Auditoria de hojas	84
<b>Protección de hojas</b>	<b>85</b>
Insertar comentarios	87
Subtotales	89
<b>Tablas Dinámicas</b>	<b>91</b>
<b>Búsqueda de objetivos</b>	<b>95</b>
<b>Tablas de datos de una y dos variables</b>	<b>98</b>
<b>Escenarios</b>	<b>99</b>
<b>Solver</b>	<b>102</b>
Especificando el objetivo	105
Especificando las celdas de cambio (Variables de cambio)	107
Definiendo restricciones	107
Especificando una restricción de números enteros	110
Guardando y reutilizando los parámetros de solver	110
Asignado los valores de solver a un escenario	111
Otras opciones de solver	112
La opción Adoptar Modelo Lineal	112
La importancia de usar valores iniciales apropiados	113
La opción de mostrar resultado de las iteraciones	113
Generando Informes	113
El informe de Sensibilidad	114

El informe de Sensibilidad para un Modelo Lineal .....	114
El informe de Respuesta .....	114
El Informe de Límites .....	115
Cuando el Solver es incapaz de resolver .....	115

## **Unidad 8 ..... 121**

<b>Acceso a datos del exterior</b> .....	<b>121</b>
<b>Creación de una consulta</b> .....	<b>122</b>
<b>Microsoft Query</b> .....	<b>124</b>
Devolver datos a Excel .....	125

## **Unidad 9 ..... 127**

<b>Impresión en Excel</b> .....	<b>127</b>
Configuración de la Hoja .....	127
Imprimir .....	132
Corrección ortográfica .....	134

## **Unidad 10 ..... 137**

<b>Macros</b> .....	<b>137</b>
Macro .....	137
Objetos propiedades y métodos .....	137
Objeto .....	137
Propiedades .....	137
Métodos .....	138
Conjuntos .....	138
Objetos de objetos .....	138
Programación Orientada a objetos o Progr. basada en objetos .....	139
Editor de Visual Basic .....	139
Insertar módulo .....	141
Insertar procedimiento .....	141
True y False .....	144
Variables .....	144
La función Inputbox .....	145
La sentencia Opción explicit .....	146
Conversión de tipo de datos .....	148
Funciones de conversión de tipos .....	149
Variables de Objetos .....	150



<b>Estructuras condicionales</b>	<b>151</b>
Estructuras if	151
Estructuras if anidadas	154
Operadores lógicos	155
Estructura select case	157
Estructuras repetitivas (ciclos)	166
<b>Procedimientos y funciones</b>	<b>176</b>
Definición de procedimiento	177
Variables Locales y Globales	180
Funciones	184
La cláusula Private	187
Importar y exportar módulos	190
<b>La grabadora de macros</b>	<b>194</b>
Insertar Funciones de Microsoft Excel desde Visual Basic	203
<b>Detección de errores y depuración de programas</b>	<b>205</b>
Herramientas de depuración	206
El modo Interrupción	209
<b>Controles de formulario en la hoja de cálculo</b>	<b>224</b>
Mostrar la barra de herramientas para cuadros de control	225
Cuadros combinados (ComboBox)	229
Segunda Lista	231
 <b>Unidad 11</b>	 <b>245</b>
<b>Funciones más comunes en Excel</b>	<b>245</b>
Buscarv	245
Si	247
Funciones Estadísticas	249
<b>Matrices</b>	<b>258</b>
Mdeterm	261
Trasponer	262
Minversa	263
Mmult	264
Función =Si()	265
Función =Pago()	270
Utilización de botones de control	273
 <b>Configuración y Atajos para Excel</b>	 <b>281</b>



## INTRODUCCIÓN

Nuestro mundo es cambiante, está lleno de paradigmas, los nuevos inventos y descubrimientos se han hecho por la ruptura de estos paradigmas, qué pensó la gente de la época de galileo, qué pensaron los grandes industriales suizos quienes eran los que controlaban el mercado de los relojes cuando les presentaron los nuevos relojes electrónicos sin pernos, qué pasaría con la humanidad si nos quedamos estancados y no tratamos de avanzar en nuestros conocimientos; es por ello que continuando y avanzando con nuestra tarea como Universidad presentamos el tercer libro de la serie de cinco, **Microsoft Excel**, que trata de cubrir el hueco que la mayoría de nosotros tenemos en lo que respecta al conocimiento y manejo de las hojas de cálculo.

Este libro se ha hecho con la finalidad de tener un material para los alumnos de cualquier diplomado, especialidad o carrera universitaria que necesite utilizar una hoja de cálculo como herramienta de apoyo para sus trabajos escolares o en la oficina.

Este libro es un auxiliar del maestro y nunca suple a una computadora, por lo que lo más recomendable es que se utilice o se lea cuando está frente a la computadora y pueda estar practicando lo que está aprendiendo, de esta manera el aprendizaje es más significativo.

El programa Excel siempre ha sido visto como una hoja en la que podemos hacer cuadros, sumas y restas y lo excepcional de este programa es que se pueden hacer gráficas, pero no es sólo una hoja tabular avanzada, es en sí misma un lenguaje de programación avanzada, el Visual Basic, el cual sólo está limitado por la imaginación y la habilidad del que lo esté utilizando; se pueden resolver ecuaciones lineales, hacer estadísticas, buscar escenarios, buscar objetivos, etc.

Esta obra se presenta en varias unidades, las cuales tienen prácticas, que ayudan a reforzar la teoría, y para que el alumno la pueda poner en práctica para un mejor aprendizaje significativo y en determinado momento no necesita más que una computadora para continuar su aprendizaje.

Dentro de la misma práctica, se van vertiendo los conceptos necesarios de la hoja de cálculo, mismos que van acompañados de la imagen que se ve en la computadora para una mejor comprensión del alumno, si es que no tienen en el momento de leerlo una computadora, ya que un icono o una gráfica es más fácil de recordar que una receta de varios pasos.

El libro está estructurado en once unidades y dos compendios; en la primera unidad se trata de dar un panorama general del Excel, así como las partes que componen esta aplicación; en la segunda unidad se da una visión de la unidad básica del Excel que es la celda, los formatos de la celda, los tipos de fuentes y sus atributos, así como la creación de formularios; en la tercera unidad se muestra cómo copiar, mover e insertar datos, filas y columnas, así como la búsqueda de datos y el autollenado de celdas; en la unidad cuatro se



estudia lo relativo a las fórmulas, su sintaxis, prioridad y mensaje de error; la unidad cinco nos muestra las funciones, cómo se crean, cómo se utilizan y cómo utilizar el asistente de funciones; la unidad seis nos muestra las capacidades del Excel para hacer gráficas y su facilidad para utilizarlas; la unidad siete es muy importante aunque un poco complicada, pues nos enseña lo relativo a las matrices y cómo se utilizan para hacer tablas dinámicas, buscar funciones objetivas, cómo utilizar el solver y cómo utilizar escenarios, entre otros temas de mucha importancia que se encuentran en esa unidad; la unidad ocho nos enseña cómo consultar datos del exterior mediante la utilización del Query; la unidad nueve explica cómo imprimir y sus diferentes opciones; la unidad diez es muy importante a la vez complicada para muchos usuarios, pero también es muy poderoso el conocimiento y manejo de esta unidad ya que nos introduce en el aprendizaje del lenguaje Visual Basic para Aplicaciones, por medio del cual el Excel deja de ser una simple hoja de cálculo y se convierte en un medio ambiente de programación con su propio lenguaje, ésta es la unidad más pesada, pues nos muestra el panorama de lo poderoso que es un lenguaje de programación en donde los límites de la hoja de cálculo es sólo mi imaginación y habilidad para resolver un problema de la vida real; la unidad once nos muestra algunas funciones de uso más común, como son las funciones estadísticas, entre otras.

Además se presentan dos apartados muy interesantes, uno para los usuarios de Excel y otro para los usuarios de Word, en donde encontrará trucos, atajos y configuraciones que le harán más fácil el uso de estas herramientas.

Con esta obra esperamos que cualquier persona que la lea y la ponga en práctica se convierta en un usuario experto en el uso de esta hoja de cálculo, y el experto encuentre información que le sirva para agilizar su trabajo con esta herramienta.

Mucha suerte y que la lectura de esta obra le sea muy placentera, como lo ha sido para nosotros.

Luis Manuel Martínez Hernández

Ma. Elizabeth Leyva Arellano



E X C E L   X P   2 0 0 3



[www.upd.edu.mx](http://www.upd.edu.mx)

CURSO DE

# Excel XP 2003

---

## INTRODUCCIÓN

Nuestro mundo es cambiante, está lleno de paradigmas, los nuevos inventos y descubrimientos se han hecho por la ruptura de estos paradigmas, qué pensó la gente de la época de Galileo, qué pensaron los grandes industriales suizos quienes eran que controlaban el mercado de los relojes cuando les presentaron los nuevos relojes electrónicos sin pernos, qué pasaría con la humanidad si nos quedamos estancados y no tratamos de avanzar en nuestros conocimientos; es por ello que continuando y avanzando con nuestra tarea como Universidad presentamos el tercer libro de la serie de cinco, **Microsoft Excel**, que trata de cubrir el hueco que la mayoría de nosotros tenemos en lo que respecta al conocimiento y manejo de las hojas de cálculo.

Este libro se ha hecho con la finalidad de tener un material para los alumnos de cualquier diplomado, especialidad o carrera universitaria que necesite utilizar una hoja de cálculo como herramienta de apoyo para sus trabajos escolares o en la oficina.

Este libro es un auxiliar del maestro y nunca suple a una computadora, por lo que lo más recomendable es que se utilice o se lea cuando está frente a la computadora y pueda estar practicando lo que está aprendiendo, de esta manera el aprendizaje es más significativo.

El programa Excel siempre ha sido visto como una hoja en la que podemos hacer cuadros, sumas y restas y lo excepcional de este programa es que se pueden hacer gráficas, pero no es sólo una hoja tabular avanzada, es en sí misma un lenguaje de programación avanzada, el Visual Basic, el cual sólo está limitado por la imaginación y la habilidad del que lo esté utilizando; se pueden resolver ecuaciones lineales, hacer estadísticas, buscar escenarios, buscar objetivos, etc.

Esta obra se presenta en varias unidades, las cuales tienen prácticas, que ayudan a reforzar la teoría, y para que el alumno la pueda poner en práctica para un mejor aprendizaje significativo y en determinado momento no necesitar más que una computadora para continuar su aprendizaje.

Dentro de la misma práctica, se van vertiendo los conceptos necesarios de la hoja de cálculo, mismos que van acompañados de la imagen que se ve en la computadora para una mejor comprensión del alumno, si es que no tienen en el momento de leerlo una computadora, ya que un icono o una gráfica es más fácil de recordar que una receta de varios pasos.

El libro está estructurado en once unidades y un compendio; en la primera unidad se trata de dar un panorama general del Excel, así como las partes que componen esta aplicación; en la segunda unidad se da una visión de la unidad básica del Excel que es la celda, los formatos de la celda, los tipos de fuentes y sus atributos, así como la creación de formularios; en la tercera unidad se muestra cómo copiar, mover e insertar datos, filas y columnas, así como la búsqueda de datos y el autollenado de celdas; en la unidad cuatro se



estudia lo relativo a las fórmulas, su sintaxis, prioridad y mensaje de error; la unidad cinco nos muestra las funciones, cómo se crean, cómo se utilizan y cómo utilizar el asistente de funciones; la unidad seis nos muestra las capacidades del Excel para hacer gráficas y su facilidad para utilizarlas; la unidad siete es muy importante aunque un poco complicada, pues nos enseña lo relativo a las matrices y cómo se utilizan para hacer tablas dinámicas, buscar funciones objetivas, cómo utilizar el solver y cómo utilizar escenarios, entre otros temas de mucha importancia que se encuentran en esa unidad; la unidad ocho nos enseña cómo consultar datos del exterior mediante la utilización del Query; la unidad nueve explica cómo imprimir y sus diferentes opciones; la unidad diez es muy importante a la vez complicada para muchos usuarios, pero también es muy poderoso el conocimiento y manejo de esta unidad ya que nos introduce en el aprendizaje del lenguaje Visual Basic para Aplicaciones, por medio del cual el Excel deja de ser una simple hoja de cálculo y se convierte en un medio ambiente de programación con su propio lenguaje, ésta es la unidad más pesada, pues nos muestra el panorama de lo poderoso que es un lenguaje de programación en donde los límites de la hoja de cálculo es sólo mi imaginación y habilidad para resolver un problema de la vida real; la unidad once nos muestra algunas funciones de uso más común, como son las funciones estadísticas, entre otras.

Además se presenta un apartado muy interesante para los usuarios de Excel, en donde encontrará trucos, atajos y configuraciones que le harán más fácil el uso de estas herramientas.

Con esta obra esperamos que cualquier persona que la lea y la ponga en práctica se convierta en un usuario experto en el uso de esta hoja de cálculo, y el experto encuentre información que le sirva para agilizar su trabajo con esta herramienta.

Mucha suerte y que la lectura de esta obra le sea muy placentera, como lo ha sido para nosotros.

Luis Manuel Martínez Hernández

Ma. Elizabeth Leyva Arellano



## UNIDAD 1

---

### Aspectos Generales

Todas las empresas públicas y privadas se están adaptando al avance tecnológico que representa la informática. Esto va a llevar a que, en muy poco tiempo, para personas como comerciantes, abogados, diseñadores, médicos, profesores, entre otros, sea requisito indispensable tener conocimientos de informática para acceder a un puesto de trabajo.

Excel es una hoja de cálculo integrado en el paquete Microsoft Office, el cual incluye 4 herramientas en su edición estándar (Word, Excel, PowerPoint y Outlook) y 5 en su versión profesional (Word, Excel, PowerPoint, Outlook y Access). Una hoja de cálculo es un programa de computadora que nos permite crear tablas, bases de datos y gráficos con texto o imágenes, y programas que no requieran la potencia de una base de datos.

Aprender a manejar Excel no es difícil, simplemente es necesario dedicarle unas cuantas horas de aprendizaje (no demasiadas) y sobre todo crear muchas tablas y bases de datos. Si ha decidido aprender a manejar Excel es muy recomendable que practique todo lo que aprenda, ya que en el aprendizaje de cualquier programa de computadora es la práctica la parte esencial para el dominio de esta herramienta.

Las hojas de cálculo son una de las herramientas informáticas más utilizadas hoy día en una empresa u oficina, para realizar trabajos de clase o para trabajar en casa. Una de las hojas de cálculo más utilizadas hoy día es Excel, pero podemos encontrar otros procesadores de texto igualmente buenos como por ejemplo Quattro, Lotus 123, Star Office, etc.

Para aprender a utilizar una hoja de cálculo es muy recomendable que tenga conocimientos de Windows 95/98, ya que es el entorno donde van a desarrollarse todos los procesos, asimismo, es conveniente tener conocimientos de informática básica (qué es un disco duro, por ejemplo) aunque este último punto no es tan necesario como el primero. Una buena referencia es el libro 1 de la colección de Cuadernos Informáticos, el cual lleva por título PC y WINDOWS.

El programa de Microsoft Excel se distribuye normalmente con el paquete Microsoft Office, como se mencionó anteriormente, este paquete está compuesto por varias aplicaciones. Las más destacadas son las siguientes:

- Microsoft Word: Procesador de Textos. El Procesador de palabras es una aplicación que nos permite crear documentos escritos, cartas, apuntes, carteles, etc. y además podemos aplicar formatos de columnas, tipos de letra distintos, tablas, insertar imágenes, cuadros de texto, etc. Dando a los textos un aspecto profesional.

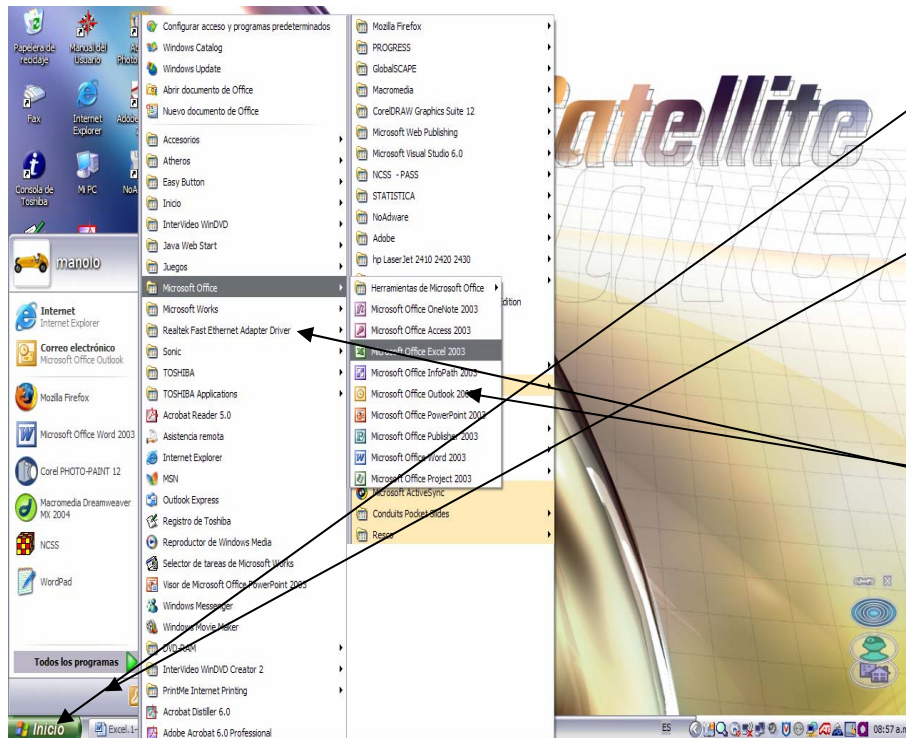


- Microsoft Excel: Hoja de Cálculo. La hoja de cálculo es muy utilizada hoy día en comercios, oficinas, etc. Nos permite manejar bases de datos y realizar operaciones y cálculos de una forma muy sencilla. Además podemos crear gráficas.
- Microsoft PowerPoint: es un programa para crear presentaciones. Por ejemplo la presentación de una tesis, del Plan Nacional de Desarrollo, las ventas de una empresa, etc.
- Microsoft Access: Gestor de Base de Datos. Es otra aplicación muy utilizada hoy día, ya que nos permite manejar bases de datos. Ejemplos de bases de datos son productos de un comercio, datos de clientes o proveedores, etc.

El orden lógico para aprender los programas pasa por conocer primero Windows 95 o Windows 98 ya que es lo que se denomina Sistema Operativo. Windows es el conjunto de programas básicos para que el ordenador funcione. Si un ordenador no tiene Windows (u otro sistema operativo) instalado, no funciona.

Por lo tanto, podemos decir que Windows es el entorno en el que vamos a trabajar con el ordenador y, si no conocemos este entorno, difícilmente podremos sacar partido a una aplicación (por ejemplo Excel). Por lo que es muy recomendable aprender a manejar Windows antes que cualquier otra cosa.

Para comenzar a trabajar con Excel primero debes asegurarte de que está instalado en tu ordenador. Si no estás seguro de tenerlo ya instalado, realiza los siguientes pasos:



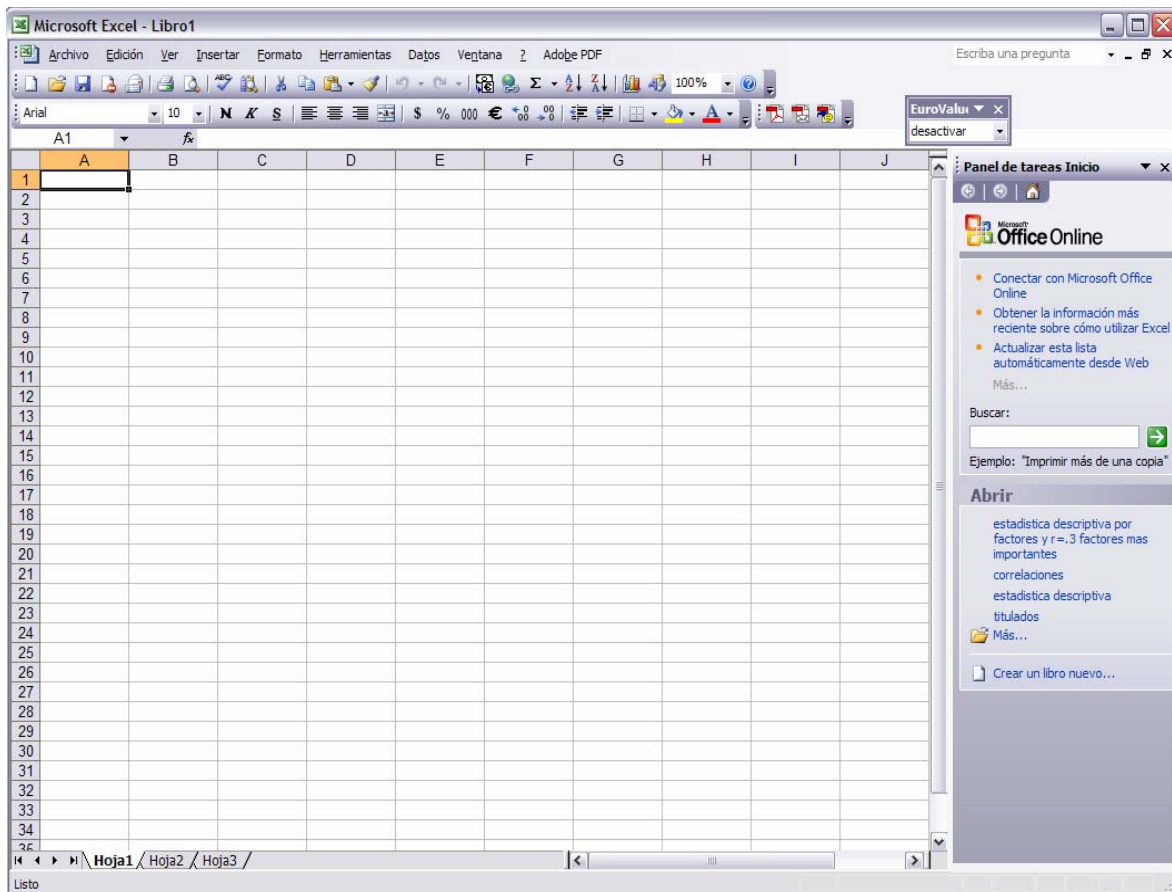
- Pulsa el botón “Inicio” (en la parte inferior izquierda de la pantalla).
- Selecciona la opción “Programas”
- Dentro de la opción “Programas” debe aparecer la opción “Microsoft Excel” (lleva una X de color azul delante).
- También es posible que “Microsoft Excel” se encuentre en otro sitio (Fíjate que existen muchas opciones dentro de “Inicio”: Accesorios). Puedes probar a buscarlo.

· Otra posibilidad es que tengas un acceso directo en el escritorio (un icono). Si lo encuentras sólo tienes que hacer un doble clic sobre dicho icono.

Si son demasiado complicadas para ti las opciones anteriores o no entiendes lo que es un “icono” o un “doble clic” quizás deberías aprender algo de Windows 95/98 antes de continuar.

Una de las cosas importantes en el momento de empezar a trabajar con un nuevo programa es familiarizarnos con las diferentes partes de éste. En esta primera lección vamos a hacer una pequeña enumeración de estas partes y poco a poco iremos profundizando más en sus características y propiedades.

Ahora que ya iniciamos Excel, observa cómo en unos segundos se ha puesto en funcionamiento **Excel**. En tu computadora aparecerá una pantalla parecida a ésta.



A continuación se explicarán las diferentes partes de que podemos ver en esta pantalla.



## Barra de título

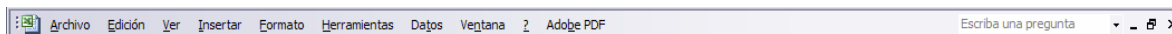


En esta pantalla nos aparece el icono de **Excel 2003**, el nombre del programa que estamos utilizando **Microsoft Excel**, el nombre del documento que estamos editando, en este caso **Libro1** y los tres botones típicos de **Windows** (de ahora en adelante llamaremos solo Windows a cualquiera de las versiones de Windows, ya sea 95, 98, Milenium, NT, 2000 o XP).

El nombre del documento, si no ha sido guardado en ningún momento será **Libro1**, **Libro2**, etc. mientras que si ya hemos guardado alguna vez aparecerá su nombre.

Lo primero que podemos ver es la barra de título, a la izquierda de esta barra se encuentra el icono de Excel y después el nombre de Microsoft Excel, a continuación el nombre del archivo, que automáticamente Excel le pone el nombre de libro1; a la derecha de esta barra se encuentran los botones de minimizar, maximizar y el de cerrar el programa.

## Barra de menús



Con esta barra podemos acceder a los menús desplegables de **Excel**. Con estos menús podemos llevar a cabo todas las opciones de las que dispone **Excel**. Conforme las vayamos utilizando iremos viendo dónde se encuentran y cómo están ordenadas.

Observa que en la parte derecha de esta barra aparecen los botones: minimizar, restaurar y cerrar; los cuales hacen referencia al documento que tenemos activo en pantalla. Mientras que los mismos botones que aparecen en la **Barra de título** hacen referencia a todo **Excel** y a su contenido.

## Barras de herramientas



Por defecto, al iniciar **Excel** por primera vez nos aparecen dos **barras de herramientas** como las que aparecen en la imagen superior. La primera de las barras es la barra de **formato**, mientras que la inferior es la barra **estándar** (pueden aparecer en diferente orden).

Hay muchas más **barras de herramientas** que podemos visualizar y con las que podemos realizar funciones determinadas. A lo largo del curso podremos ver cómo trabajar con ellas.



Sitúa el puntero del ratón sobre alguno de los botones que forman parte de las **barras de herramientas** y espera unos segundos.

Observa cómo te aparece un pequeño cuadrado de texto que te explica para qué sirve dicho botón (Observa la imagen de la derecha). Dedica un rato a moverte por los diferentes botones mirando esta pequeña ayuda.



## Barras de reglas



Las **barras de estado**, nos informan en qué celda nos encontramos, así como del texto o fórmulas que se hayan tecleado en la celda.

Estas barras se pueden mostrar u ocultar a voluntad. Más adelante veremos cómo hacerlo.

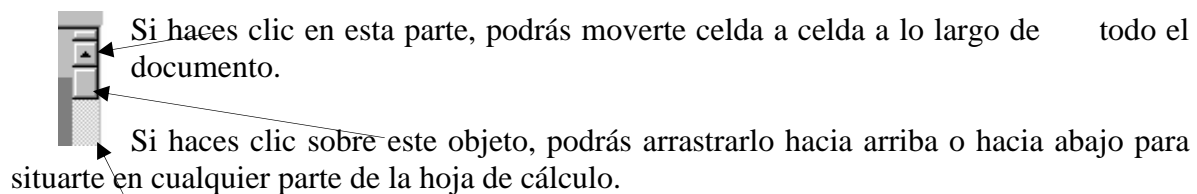
## Zona de edición

Es la zona central de la pantalla en la que editaremos nuestro documento.

## Barras de desplazamiento

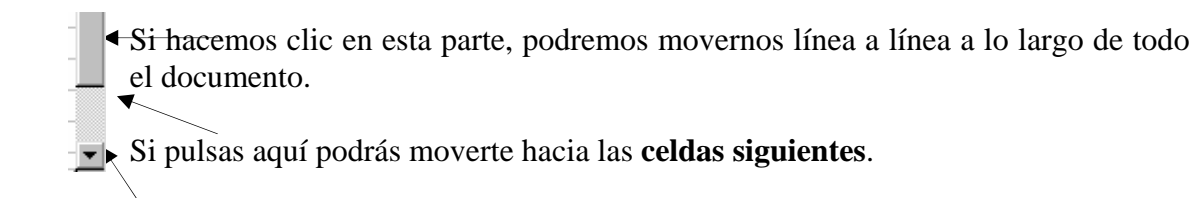
Estas barras nos permiten movernos por todo el documento, pudiendo hacerlo de diferentes formas.

Observa la parte superior de la **barra de desplazamiento** vertical.



Si haces clic en esta otra parte de la hoja de cálculo, irás saltando pantalla a pantalla por nuestro hoja.

Ahora vamos a observar la parte inferior de **nuestra barra de desplazamiento** vertical.



Si pulsas aquí podrás moverte hacia la **celda siguiente**.

Estos dos últimos botones tienen la misma función que si pulsamos las teclas [RePág] y [AvPág] retrocede página y avance página.

Con la **barra de desplazamiento** horizontal podrás moverte de una forma parecida, pero a lo ancho de nuestro documento.

Más adelante realizaremos prácticas en las que nos moveremos de un lugar a otro de nuestro documento.

## Barra de estado



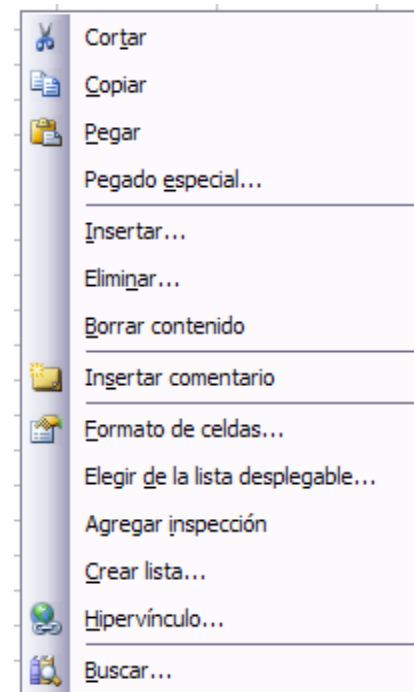
La **barra de estado** nos muestra información sobre la hoja de cálculo que tenemos activo. Nosotros podemos ver si están activadas las mayúsculas, el teclado numérico, y lo más importante, si es que está lista la hoja de cálculo para recibir información o se encuentra ejecutando un programa.

## Menús contextuales

Como en la gran mayoría de programas de Windows, aparecen **menús contextuales** que nos facilitan la selección de diferentes opciones.

Cada zona de la pantalla de **Excel** tiene un **menú contextual** diferente, el cual nos puede facilitar el trabajo. Las opciones que aparecen en dichos menús podemos encontrarlas en otros lugares, pero puede ser que tengan un acceso más complicado.

Si pulsamos el botón derecho del ratón sobre **cualquier celda** nos aparecerá un **menú contextual** como el de la imagen.



## Mover barras de herramientas

En este apartado vamos a ver cómo podemos empezar a personalizar **Excel** para adaptarlo a nuestro gusto.

Aprenderemos cómo mover las diferentes **barras de herramientas**, cómo cambiar su tamaño y su posición.

## Cambio de tamaño:

1. Sitúa el puntero del ratón entre dos botones de forma que ninguno de ellos quede seleccionado.

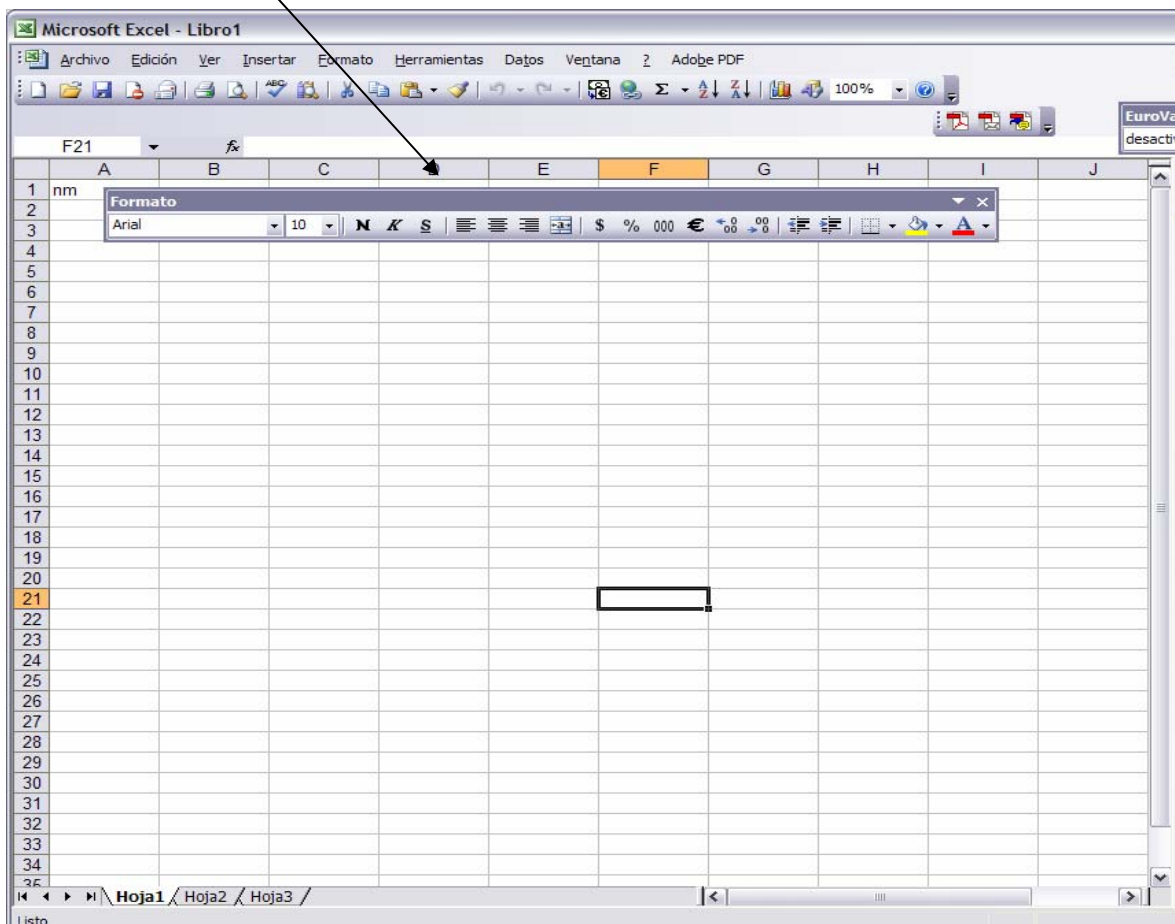
Si te cuesta, puedes situar el puntero sobre las pequeñas barras que separan los botones en las barras de herramientas o en uno de los extremos.

2. Pulsa el botón izquierdo del ratón, no lo sueltes.

Observa cómo aparece una zona punteada alrededor de dicha barra.

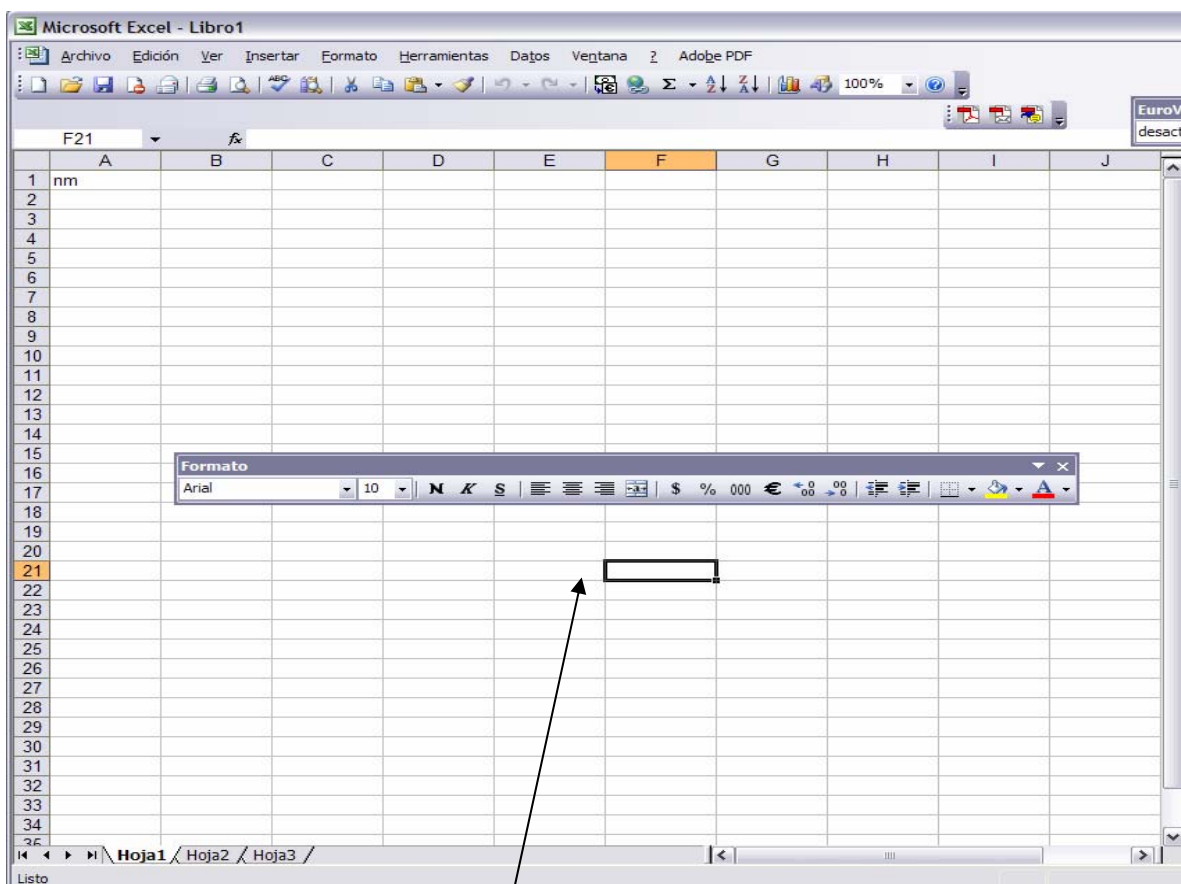
3. Mueve el ratón hacia el centro de la pantalla sin soltar el botón.

Al mover el ratón te aparece la barra de herramientas que se puede mover alrededor de la pantalla como el de la imagen siguiente, en la versión del 97 aparece un cuadro con contorno.



4. Suelta el botón del ratón.



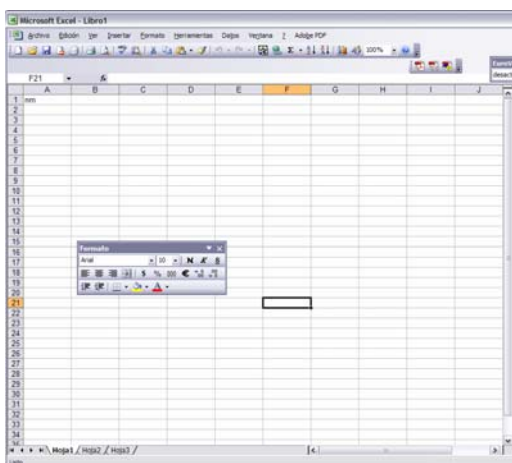


Ahora aparecerá la barra de herramientas que has seleccionado en el centro de la pantalla. Ahora dicha barra pasa a ser **flotante**, antes estaba **anclada** al borde superior de la pantalla.

Nosotros podemos cambiar el tamaño de esta barra de herramientas. Eso sí, cambiaremos el tamaño pero en todo momento visualizaremos el mismo número de botones.

5. Colócate sobre uno de los bordes de esta barra flotante.

Espera que el ratón se convierta en una doble flecha.



6. Pulsa el botón izquierdo del ratón y sin soltarlo muévete hasta que veas que el tamaño de la barra de herramientas cambia.

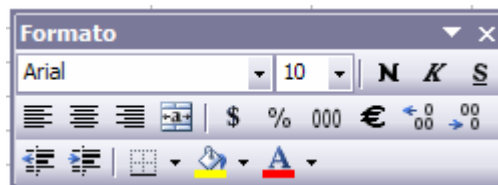
7. Suelta el botón del ratón.

Fíjate cómo la posición de los botones que componen esta barra ha cambiado, pero no ha quedado ninguno de ellos oculto.

## Cambio de posición

Cuando tenemos una barra de herramientas flotante, podemos cambiar su posición en la pantalla. La barra de herramientas flotante es como una pequeña ventana que siempre se mantiene visible. Nosotros podemos variar su posición mediante la **barra de título**.

1. Pulsa un clic sobre la **barra de título** de la **barra de herramienta** flotante. No sueltes el botón del ratón.



2. Mueve la ventana hasta otra posición de la pantalla.

Observa cómo el tamaño de dicha barra de herramientas se conserva, lo único que cambia es su posición en la pantalla.

## Posición original

Para volver a colocar nuestra barra flotante en el lugar de origen, lo único que deberemos hacer es arrastrar esta ventana hasta la zona donde estaba en un principio (Por debajo de la barra de menús).

1. Arrastra la ventana hasta situarla debajo de la barra de menús.

Observa cómo la **barra de herramientas** vuelve a tener el tamaño original y vuelve a estar anclada.

Nosotros también podemos hacer que esta barra de herramientas **flotante** esté anclada en otro lugar de la pantalla. Las zonas en las que puede estar anclada una **barra de herramientas** son cualquiera de los laterales de la ventana de **Excel**.

2. Arrastra cualquiera de las barras de herramientas hacia la derecha de la ventana principal de Excel.

Observa cómo su tamaño ha cambiado y toma una posición vertical.

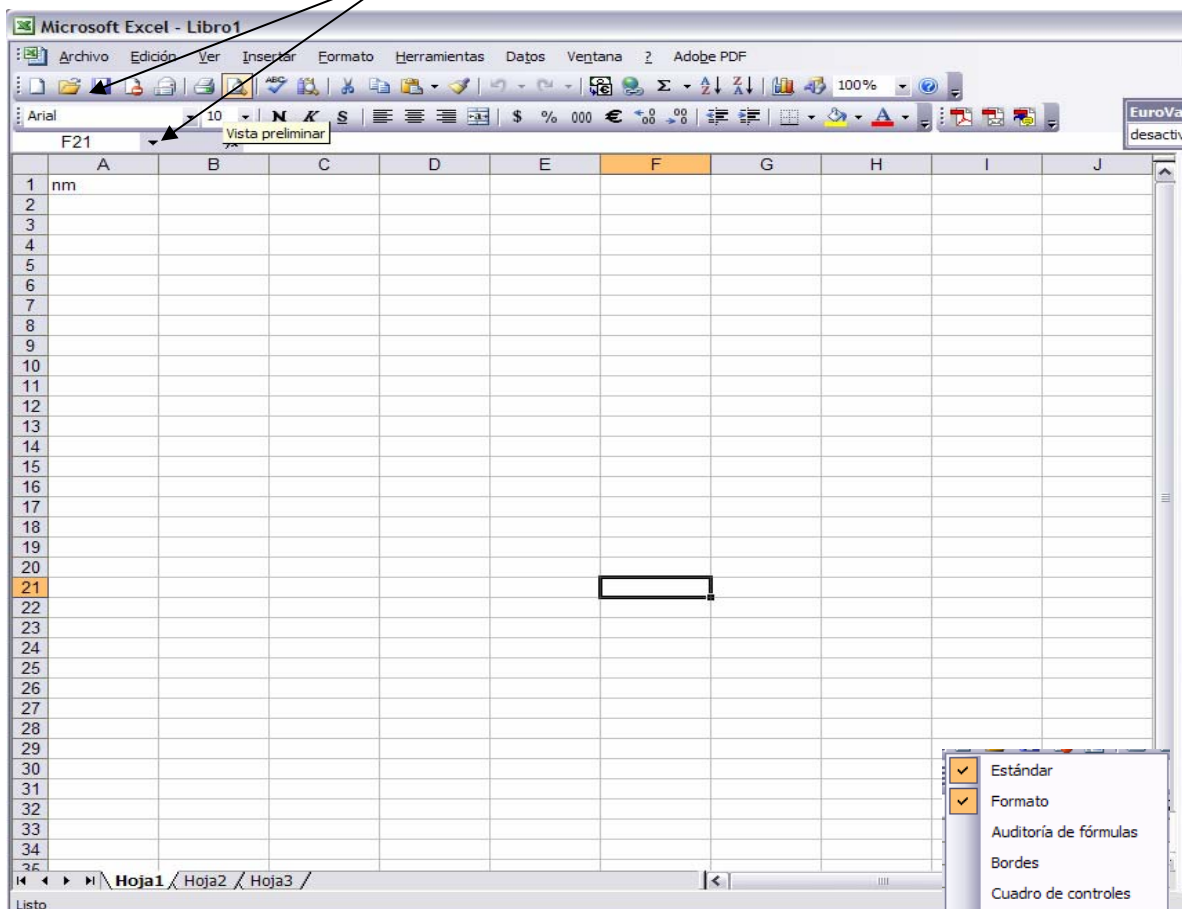
## Mostrar barras de herramientas

Vamos a ver cómo podemos ocultar o mostrar **barras de herramientas** en las cuales tendremos diferentes botones con los que trabajar. La explicación de para qué sirve cada barra de herramientas y sus respectivos botones lo iremos viendo a lo largo de todo el curso.




Tenemos varias maneras para mostrar nuevas barras de herramientas.

1. Haz clic con el botón derecho sobre cualquiera de las barras de herramientas que tenemos en la pantalla.



Seguidamente te aparecerá un menú contextual como el de la imagen de la derecha.

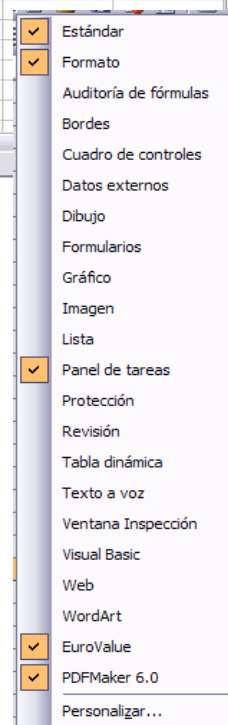
En este menú contextual te aparecen los nombres de todas las barras de herramientas que hay disponibles en **Excel**.

Observa que las dos primeras muestran una marca como ésta. 

Esto quiere decir que estas barras de herramientas están visibles.

2. Haz un clic sobre la barra de herramientas **Dibujo**.

Acto seguido ha aparecido dicha barra de herramientas en pantalla.



Según la barra de herramientas que escojas verás que puede aparecer **anclada** o **flotante**.

Para que aparezca una barra de herramientas en pantalla también podemos ir a la opción **Barra de Herramientas** del menú **Ver**.

3. Accede a la opción **Barra de Herramientas** del menú **Ver**.
4. Selecciona la barra de herramientas **Tablas y bordes**.

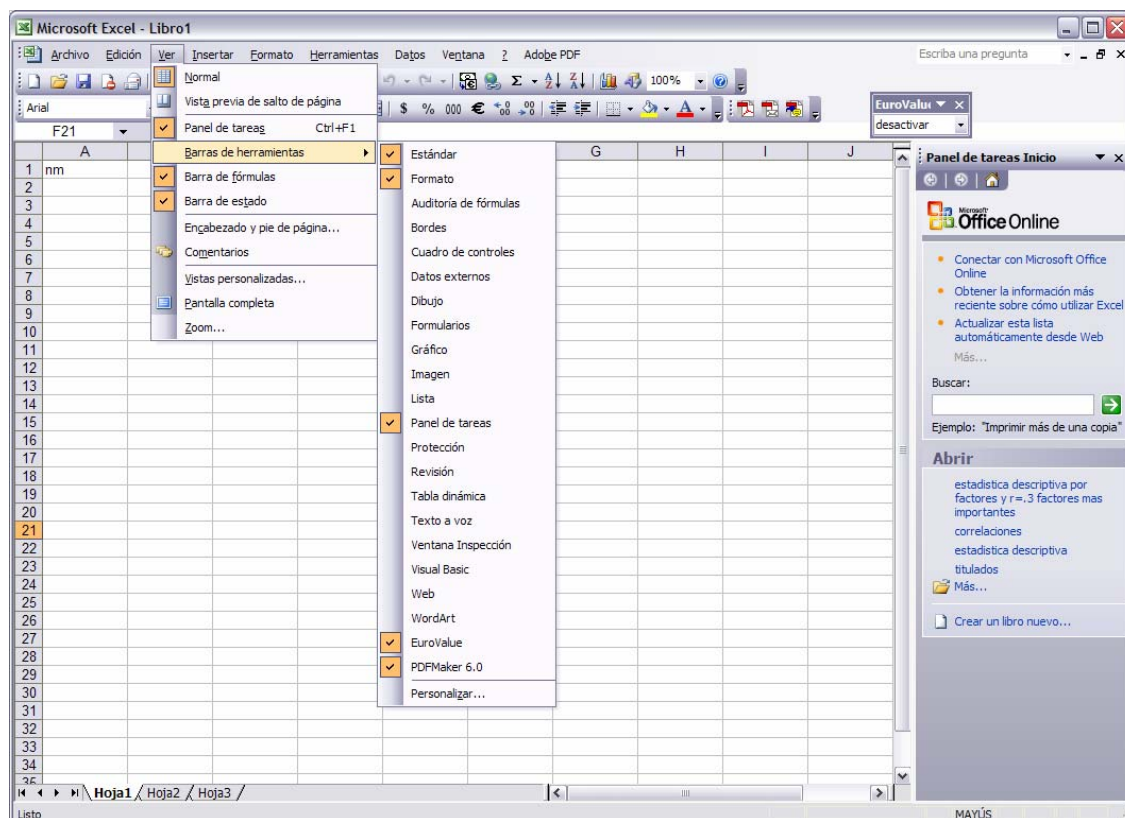
Recuerda que puedes modificar la posición de estas barras a tu gusto.

## Ocultar barras de herramientas

Vamos a ver cómo podemos ocultar alguna de las barras de herramientas que tenemos en pantalla.

Si la barra que deseamos ocultar está anclada, podemos hacerlo de varias maneras: podemos utilizar el menú contextual, la barra de menús o hacer que esta barra pase a ser flotante y entonces utilizar el botón **cerrar** que aparece en la barra de título.

1. Muestra el **menú contextual** de las **barras de herramientas**.



2. Haz un clic sobre el nombre de la barra **Tablas y bordes**. (Observa cómo ha desaparecido dicha barra.)
3. Si la barra de herramientas **dibujo** no es flotante, haz que lo sea.
4. Haz clic sobre el botón **cerrar** de dicha barra.



Observa cómo también la barra de herramientas ha desaparecido.

Para ocultar una barra de herramientas también se puede hacer utilizando la opción **Barra de Herramientas** del menú **Ver**.

Bien, ahora ya conocemos muchas de las diferentes partes de la pantalla de **Excel**, como visualizar y ocultar barras de herramientas. Sabiendo esto ya podemos ir personalizando un poco **Excel** y adaptándolo para que nos sea mucho más fácil el trabajo con él.

## Conceptos básicos

Antes de comenzar con nuestra primera hoja, vamos a ver algunos conceptos fundamentales de Excel:

**Hoja:** se denomina así a la zona donde estamos trabajando. Cada hoja tiene un nombre identificativo que podemos cambiar. Los nombres de las hojas se pueden observar en la zona inferior de la pantalla. Estos nombres se pueden cambiar.

**Celda:** cuadro individual que forma parte de la hoja. En las celdas introduciremos los datos.

**Columna:** se nombran de la **A** a la **Z** y están dispuestas en vertical. Después de la columna **Z**, nos encontramos con la columna **AA, AB, AC...** y así hasta la **AZ**. Seguidamente, comenzaría la **BA, BB..** y así hasta la última columna que es la **IV** (en total el número máximo de columnas es de 255 por hoja)

**Fila:** dispuestas en horizontal, se numeran desde la **1** hasta la **16.384** que es la última.

**Libro de trabajo:** conjunto de hojas. Un libro puede tener varias hojas. Al grabarlo, se crea un fichero con la extensión **XLS** con todas las hojas que tuviese el libro.

**Rango:** grupo de celdas adyacentes, es decir, que se tocan. Un rango de celdas por ejemplo que va desde la **A1** hasta la **A5** se reflejaría con el siguiente nombre: **A1:A5**

El nombre de un rango siempre hará referencia a la primera y a la última celda seleccionadas.

Observa en la siguiente página algunos ejemplos de rangos:



	A
1	
2	
3	
4	
5	

Rango A1:A5

	A	B	C
1			
2			
3			
4			

Rango B1:C4

## Desplazamiento por la hoja

De momento vamos a echar un vistazo a la forma de trabajar con Excel. Por ello, no te preocupes si de momento no entiendes algunos de los conceptos que veremos a continuación.

Para desplazarte a través de las celdas de Excel puedes utilizar alguno de estos métodos:

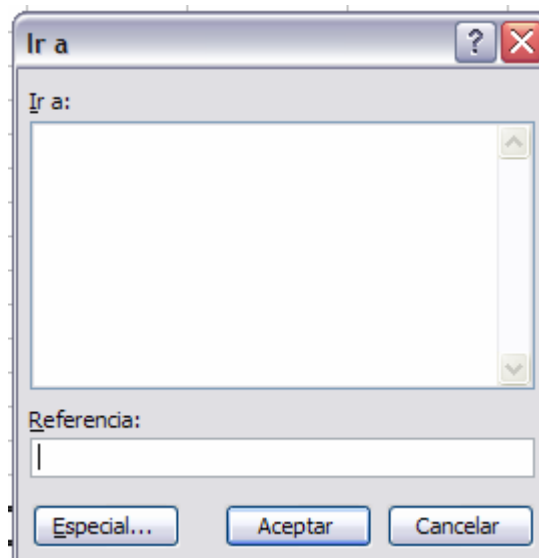
- Con las **teclas de movimiento** de cursor del teclado
- Con **un clic** en una celda específica
- Con la función **Ir a....** del menú **Edición** (o bien la tecla **F5**).

*Abre el menú **Edición** y escoge la opción **Ir a....***

*En la casilla **Referencia**, escribe por ejemplo **G230** y acepta.*

*Ahora el cursor ha saltado a la celda **G230**. Para volver a la celda inicial **A1** existe una combinación común en muchos programas de Windows:*

*Pulsa la combinación **Ctrl+Inicio***



## Selección de un rango de celdas

Para seleccionar celdas simplemente debemos situar el cursor en medio de una celda, pulsar el botón izquierdo del ratón y, sin soltarlo, “arrastrar” hacia alguna dirección. Es exactamente igual que cuando seleccionas un texto en cualquier aplicación Windows.

Otra forma de seleccionar celdas es manteniendo oprimida la tecla **[Shift]+[→]**, es decir, mantener oprimida la tecla Shift y sin dejar de oprimirla, oprimir la tecla de cursos a la derecha o arriba o abajo o a la derecha, según se requiera.

Para ejemplificar lo anterior, siga los siguientes pasos:


Selecciona un grupo de celdas(rango A1:B6)

	A	B
1		
2		
3		
4		
5		
6		

Para quitar la selección tan sólo debemos pulsar **un clic** en cualquier otra celda o bien pulsar **una tecla de desplazamiento** del teclado.

Prueba tú mismo a seleccionar varios tipos de rangos.

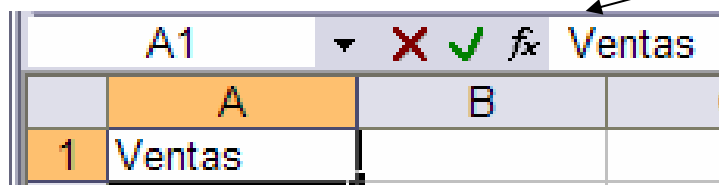
## Escribir, modificar y borrar texto

La escritura del texto en Excel es sumamente sencilla. Tan sólo hemos de situarnos en la celda requerida y escribir directamente el texto. Una vez escrito, podemos aceptarlo pulsando la tecla Intro o bien en la barra de fórmulas pulsar el botón **Introducir** 

Sitúate en la celda **A1** y escribe el siguiente texto:

**Ventas**


Observa que al comenzar a escribir, aparece automáticamente el texto en la **barra de fórmulas** así como los botones se ven activados.



Ahora podemos aceptar el texto de las siguientes formas:

- Pulsando **Intro**
- Pulsando alguna tecla de desplazamiento del teclado
- Pulsando el botón **Introducir** de la barra de fórmulas

Si queremos cancelar la entrada de datos podemos:

- Pulsar el botón **Cancelar** de la barra de herramientas 
- Pulsar la tecla **Esc**

*Acepta la entrada*

## Modificar y borrar el texto

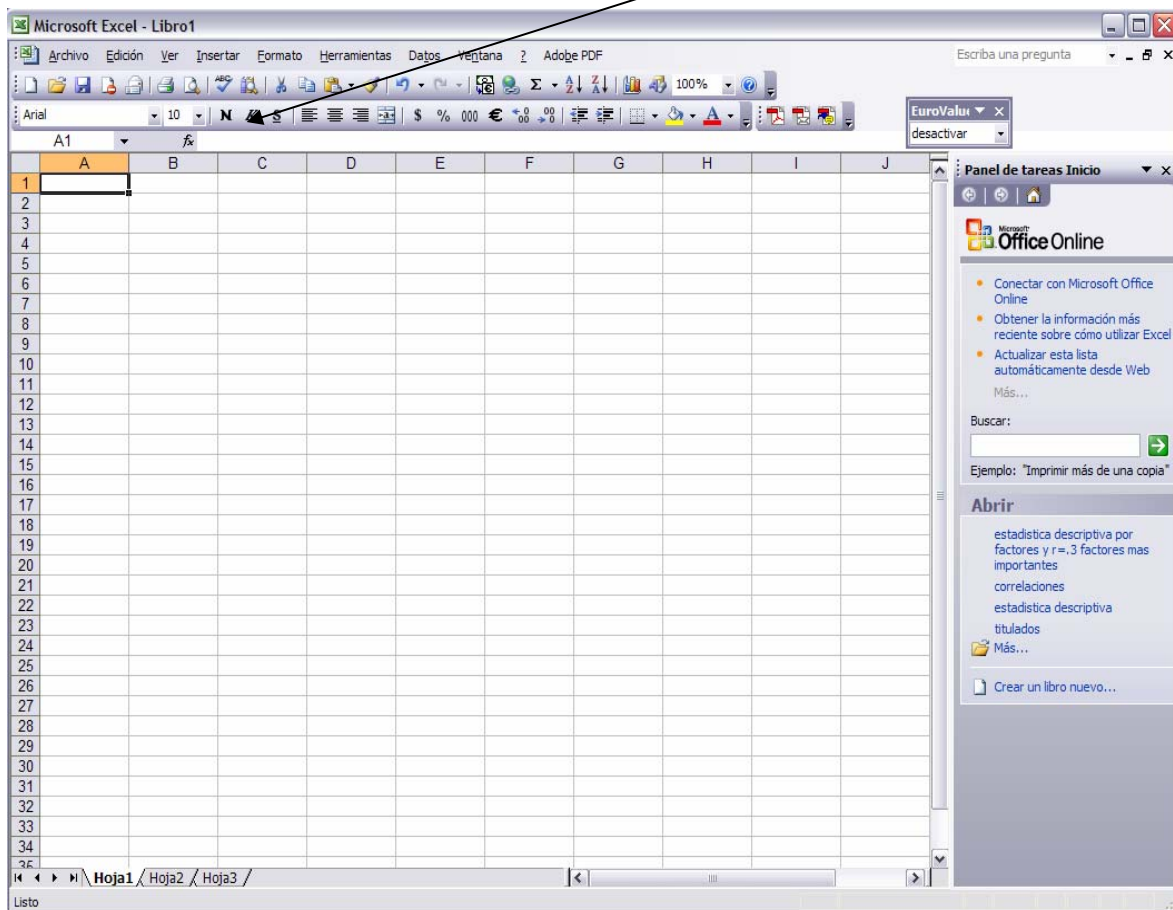
---

Para modificar el texto de una celda existen varias formas, la primera es:

Situar el cursor en la celda y **escribir directamente** el nuevo texto.

Otra forma es:

Situar el cursor en la celda y pulsar **click** en la barra de fórmulas.



**La primera forma es:**

Pulsar **doble clic** en la celda del texto

Y la última y que es la más usual para muchas personas es:

Situar el cursor en la celda y pulsar la tecla **F2**



Para borrar el texto de una celda tenemos 3 formas, la primera es más rápida y por lo mismo más utilizada es:

Situar el cursor en la celda y pulsar la tecla **Supr**

**La segunda es:**

**Ir a Edición – Borrar**

Y la última es:

Pulsar el **botón derecho** y escoger la opción Eliminar.

Para poner en práctica las formas de modificar y borrar, teclea el siguiente texto en la hoja de cálculo.

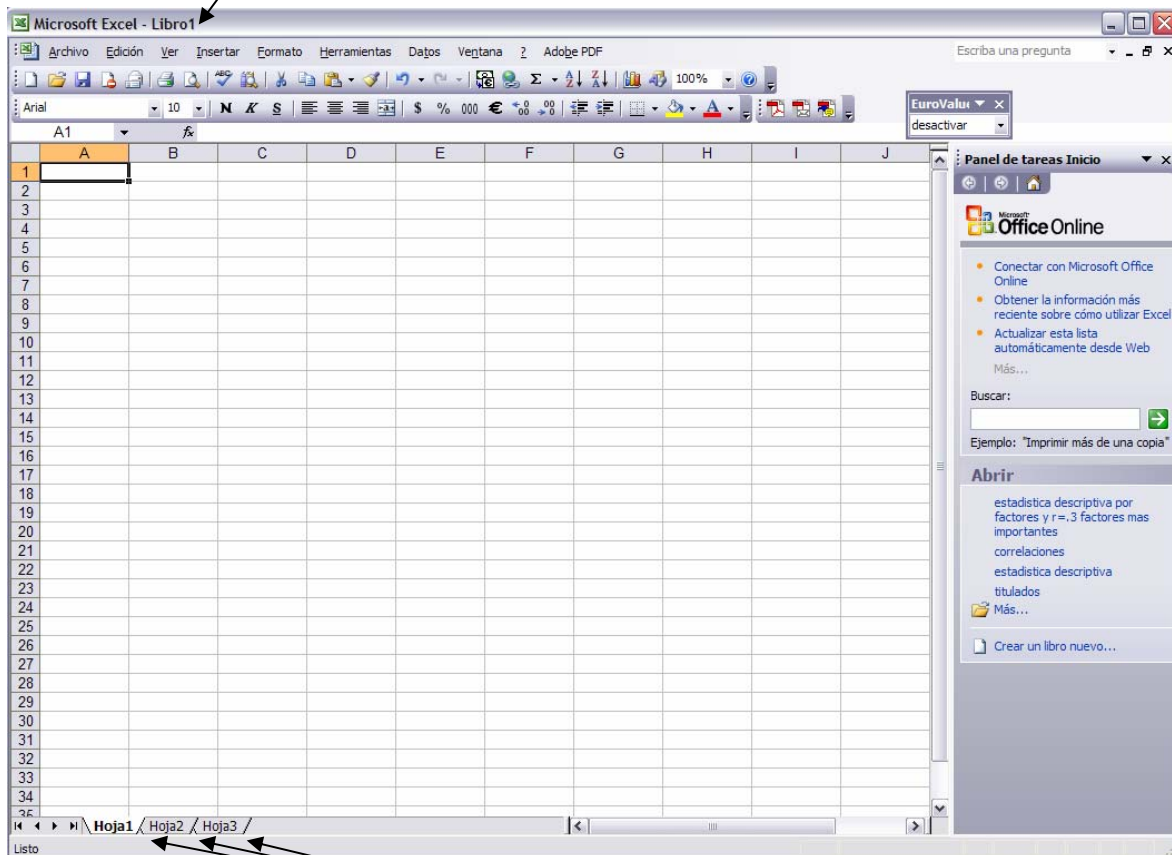
*Escribe la siguiente lista de datos:*

	A	B
1	Ventas	
2	Ingresos varios	
3	TOTAL	
4		
5	Compras	
6	Gastos varios	
7	TOTAL	

Observa que el contenido de las celdas A2 y A6 sobrepasan el ancho de la columna. No te preocupes por ello. En estas primeras prácticas seguramente verás alguna opción algo avanzada y que no entenderás demasiado. No te preocupes por ello, pues ahora sólo se trata de familiarizarse con el modo de trabajo de Excel.



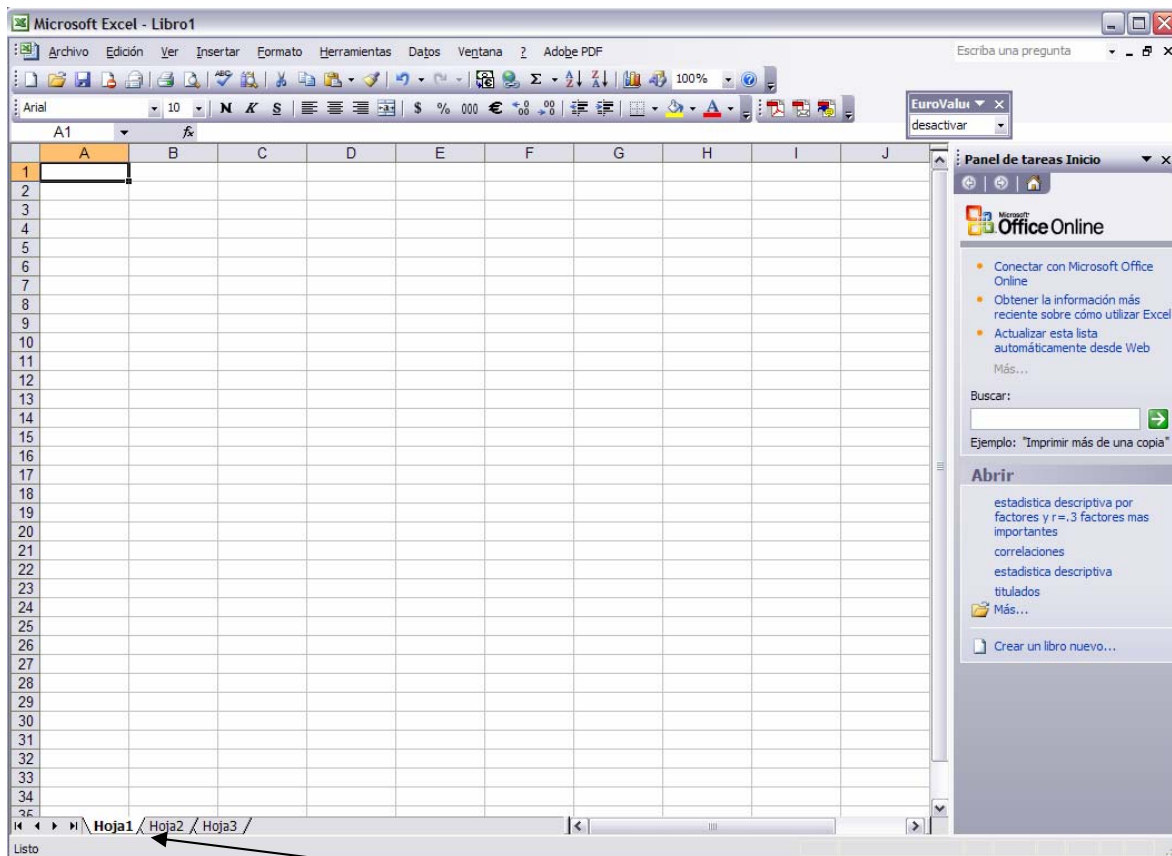
La forma de guardar la información en Excel es mediante un archivo, el cual Microsoft le llama libro de trabajo y como anteriormente vimos, el nombre que automáticamente se le da es el de **Libro1.xls**, si es que no existe este nombre, de otra forma se le asignan en vez de un “1” un número “2”, y el nombre por lo tanto quedaría como **Libro2.xls**, y así sucesivamente hasta que encuentre un nombre de hoja de cálculo que no exista en el directorio de trabajo, el cual es por lo general “**Mis Documentos**”.



Un libro de trabajo consta de varias hojas. Inicialmente, Excel 2003 nos permite trabajar con tres hojas a las cuales nombra Hoja1, Hoja2 y Hoja 3, estas etiquetas las podemos observar en la parte inferior de la hoja en la que estamos trabajando. No obstante, podemos insertar hojas, copiarlas, moverlas, borrarlas, seleccionarlas.

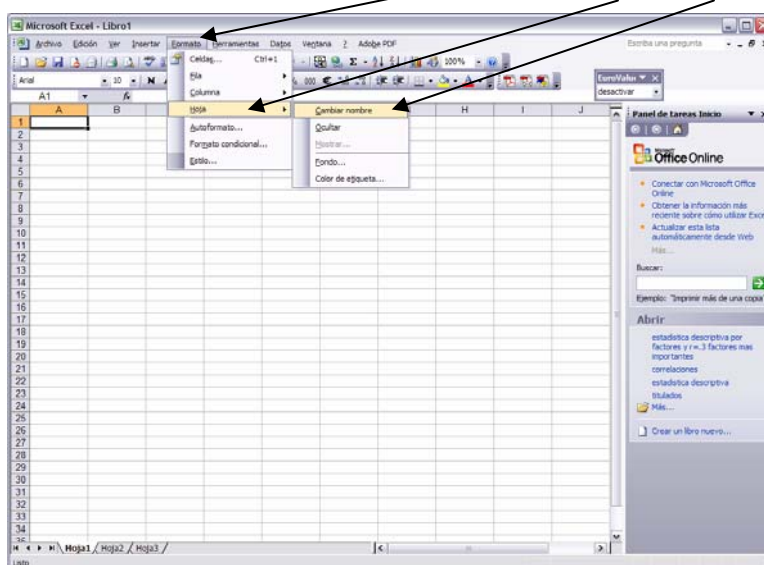
Cuando guardamos nuestra información en el libro de Excel, es importante nombrar nuestras hojas con nombres diferentes, por ejemplo si trabajamos con las ventas de un negocio, podremos necesitar que en cada hoja estén las ventas de un mes, por ejemplo en vez de llamarse Hoja1 se le podría llamar Enero, a la Hoja2 Febrero y así sucesivamente. Para que le cambiemos el nombre a las hojas, podemos seguir los siguientes pasos:





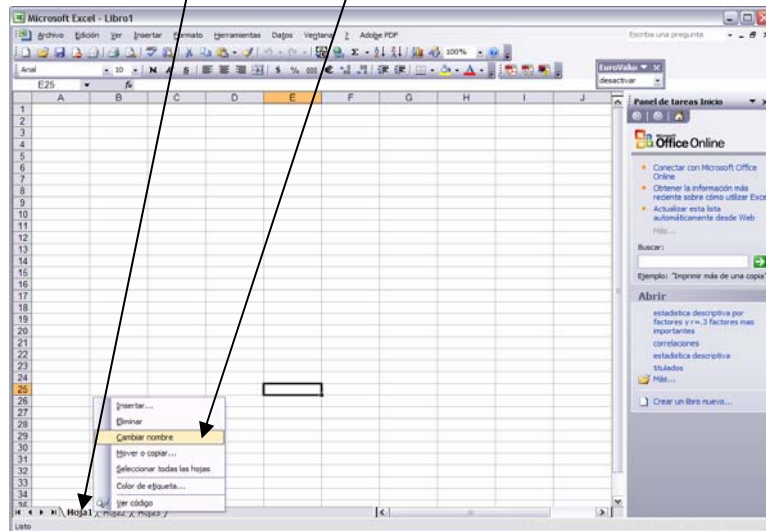
*Haz doble clic sobre la etiqueta de la Hoja1 y cuando esté seleccionada, es decir, se encuentre el fondo en negro y las letras en blanco, y escribe directamente: **Enero** (Intro para finalizar).*

Otro sistema para cambiar el nombre será desde **Formato – Hoja – Cambiar nombre**.



### La tercera forma es:

colocar el cursor en la pestaña del nombre “**Hoja1**” y hacer clic en el botón derecho del ratón y a continuación aparecerá un menú contextual, seguidamente, seleccionamos la opción de “**Cambiar Nombre**” y a continuación cambiamos el nombre.



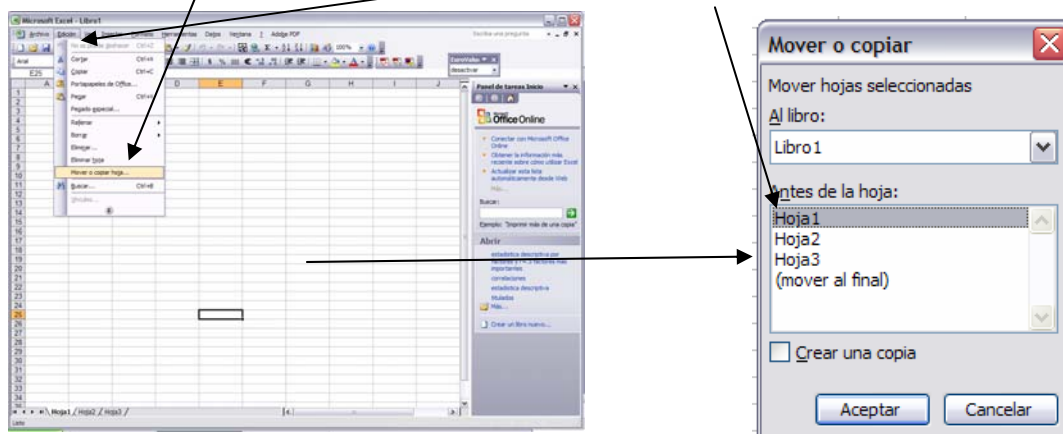
### Copiar una hoja

Utilizando el ratón, posiciona el cursor en donde se encuentra el nombre de la hoja que deseas copiar.

Manteniendo oprimida la tecla de **CONTROL [Ctrl]** (y sin dejar de oprimirla), haz clic en el botón izquierdo del ratón y sin dejar de presionar el botón izquierdo del ratón (ni la tecla control), mueve el ratón a la derecha y deja de presionar el botón del ratón y la tecla control (a este proceso se le llama arrastrar y soltar), La hoja mostrará un 2 entre paréntesis:



Otra forma es seleccionando desde la barra de herramientas la opción de **Edición – Mover o copiar hoja** donde veremos un cuadro de diálogo en el que se selecciona el libro de trabajo y el lugar donde queremos colocar la hoja.



## Mover una hoja

Arrastra directamente (sin mantener la tecla de **CONTROL** pulsada), la hoja **Listado** hacia otra posición.

## Insertar una hoja

Selecciona con un clic la hoja **Listado**

Abre el menú **Insertar** y escoge la opción **Hoja**

La hoja nueva se inserta adaptando una numeración correlativa:

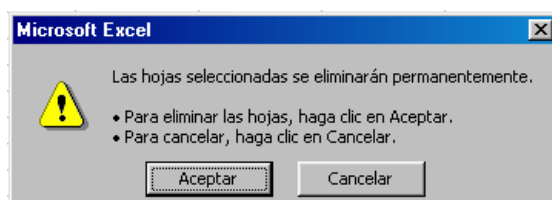


También podríamos insertarla con el botón derecho del Mouse.

## Eliminar una hoja

Selecciona cualquier hoja y pulsa el botón derecho del ratón. Escoge la opción **Eliminar**. Acepta el mensaje de Excel.

A continuación aparecerá un cuadro de diálogo como el siguiente:



Tenga cuidado al eliminar las hojas, pues no se pueden volver a recuperar a menos de que no haya guardado los cambios en el archivo.

También podríamos eliminarla desde la opción **Edición - Eliminar hoja**.

## Tipos de datos

Ya hemos visto cómo escribir datos en las celdas. Vamos a ver ahora qué tipo de datos acepta Excel:

- **Valores constantes.** Todo tipo de datos que escribamos directamente, ya sea texto o números. Este valor no cambia a no ser que lo modifiquemos o borremos.
- **Fórmulas.** Un valor especial que hace referencia a celdas, nombres, datos, etc, y que producen un resultado. Las fórmulas comienzan siempre con el signo de igual (=). Si modificamos el valor de una celda a la que la fórmula esté haciendo referencia, el resultado de la fórmula varía automáticamente.

## Valores numéricos

Excel posee para los valores numéricos el formato general, es decir, que podemos escribir un número como 200000 sin separadores de miles (el cero) y Excel lo dejará tal y como lo hemos escrito. También podemos utilizar signos como:

- El **punto (.)** para expresar decimales.
- La **coma (,)** para separar los millares.
- El signo **menos (-)** para indicar cantidades negativas. Éstas se pueden indicar también entre paréntesis.
- El signo del **porcentaje (%)**

Otras consideraciones importantes a la hora de introducir valores numéricos son:

- Las fracciones debemos introducirlas de forma especial, ya que por ejemplo 4/3 Excel lo tomará como una fecha y colocará en la celda el cuatro de marzo (4-mar). En su lugar introduciremos 1 1/3
- Si el valor no cabe en la celda, se visualizarán los signos #####. Debemos cambiar el ancho de la columna (como veremos más adelante) para poder visualizar todas las cifras.
- Si deseamos introducir un número y que Excel lo tome como un texto, debemos anteponer al número el signo del apóstrofe ('). Ejemplo: '1,996 *Ventas anuales*.

## Valores de texto

Un texto es cualquier conjunto de caracteres que Excel no considera como un número. Podemos introducir directamente los caracteres en la celda.

- Un texto puede invadir la celda y celdas de su derecha, y éste se visualizará por completo siempre que las celdas estén vacías. Si no lo están, el texto será recortado en la celda.
- Los textos pueden ajustarse (centrados, alineados, retornos automáticos, etc.)

## Fechas y horas

Las fechas se almacenan internamente como números de serie que Excel cuenta desde el día 1 de Enero de 1990 y que transforma en caracteres legibles en pantalla. El usuario puede introducir las fechas de varias formas: 23/07/98, 23-Marzo-98, 23-mar-1998, etc.

Las horas pueden representarse en formatos de 12 ó 24 horas. Por ejemplo: 2:10 pm, 14:10.

Para introducir la fecha actual existe un método rápido, el cual es **[Ctrl.]+[Mayús.]+[;]** (recuerde, presionar la tecla control y sin dejar de oprimirla, presionar la tecla de mayúsculas y sin dejar de presionar las dos teclas anteriores, presionar la tecla de “;”).

Para introducir la hora actual existe también un método más rápido, el cual es [Ctrl.]+[Mayús.]+[:]

## Copiar y mover celdas

Para copiar o mover celdas podemos recurrir a las conocidas opciones de **Copiar**, **Cortar** y **Pegar** o bien utilizar el sistema de arrastrado.

Para que esto quede más claro, vamos a hacer el siguiente ejercicio:

*Escribe un texto corto en cualquier celda*

*Accede a la opción **Edición – Copiar** o bien al botón **Copiar***



Observa que en la celda aparecen puntos parpadeantes.

*Pulsa un clic en cualquier otra celda.*

*Accede a **Edición – Pegar** o pulsa el botón **Pegar***



Observa que la zona parpadeante continúa activa.

*Pulsa la tecla **Esc**.*

Si en vez de la opción **Copiar** hubiésemos utilizado la opción **Cortar**, el dato de la celda origen hubiese desaparecido, es decir, se hubiera movido.

Otro método para copiar es el siguiente:

*Sitúa el puntero del ratón en un borde de la celda a copiar, pulsa la tecla de **CONTROL** y sin soltarla, presiona la tecla izquierda del ratón, arrastra la celda a otra posición. Suelta después.*

Con este método, si no pulsamos la tecla de control, la celda se movería. Asimismo, podemos copiar o mover un rango de celdas seleccionado con los mismos métodos.

## Dar nombres a las celdas

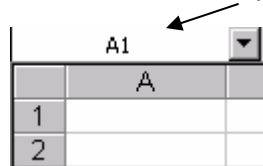
Es posible asignar un nombre a una celda o a un rango, esto nos permite:

- Desplazarnos a esa celda o rango más fácilmente
- Utilizar los nombres en una fórmula
- Identificar mejor los rangos (por ejemplo: **Ventas**)

Supongamos que queremos hacer un reporte de ventas de autos para los doce meses del año, para ello en vez de tener una columna que se llama **A1**, le cambiaremos el nombre

y le pondremos **Ventas**, para dar un nombre a una celda haremos lo siguiente:

*Sitúa el cursor en la celda **A1** y pulsa un clic en la casilla de los **Nombres de celdas**:*



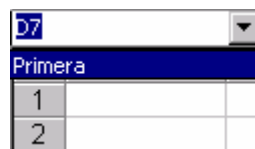
*A continuación escribe por ejemplo: **Primera** y pulsa **Intro**.*



La celda ha recibido un nombre.

*Sitúa el cursor en cualquier otra celda.*

*Abre la lista de nombres y escoge **Primera***



El cursor salta a la celda con ese nombre; en nuestro caso, a la celda **A1**.

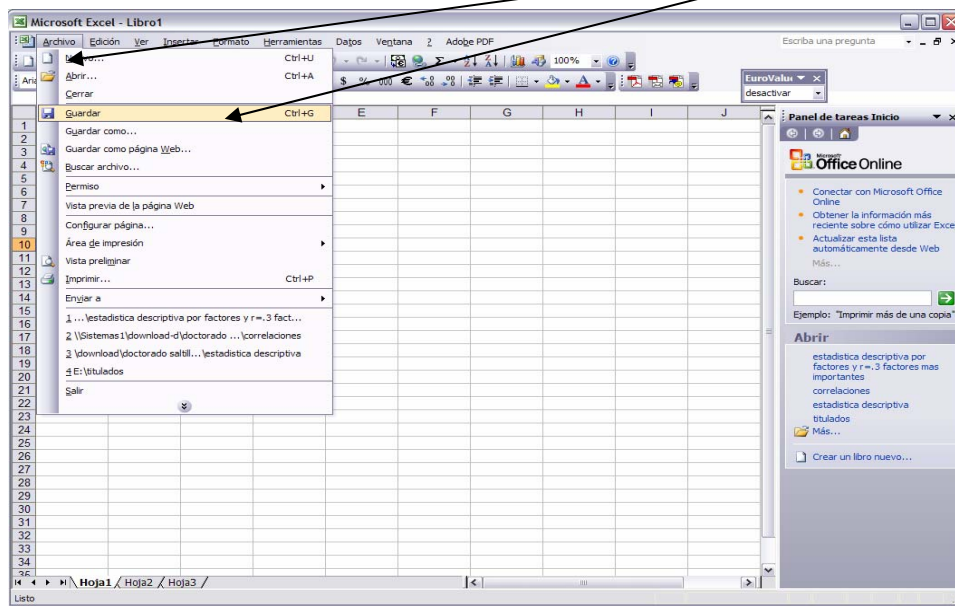
Asimismo, si seleccionamos un rango entero de celdas, podemos también asignarle un nombre y utilizarlo para desplazarnos a él.

## Guardar el trabajo

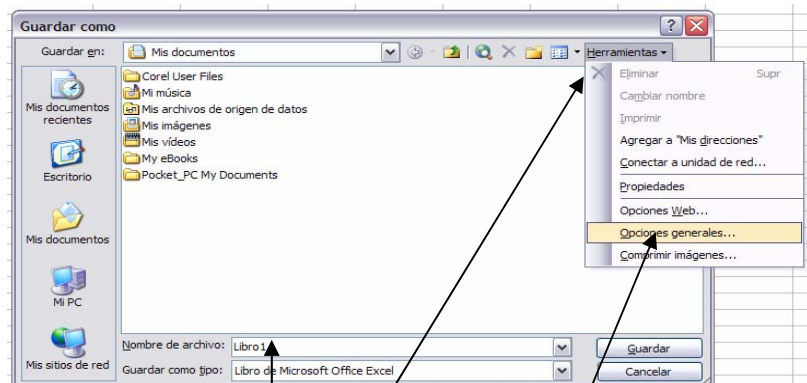
Una acción que se debe realizar al menos cada 5 minutos, es guardar la información, pues muchas veces, cuando estamos trabajando, se nos olvida guardar la información y a veces pasan 8 horas desde que comenzamos a trabajar y a veces se pierde todo el trabajo, pero Excel tiene una configuración, para que se autoguarde cada 10 minutos, pero en algunos equipos esta configuración ha sido cambiada por los usuarios, lo que impide que se tenga esta protección y no se autoguarda. Esta función se activa cuando se apaga accidentalmente la computadora.

Visto lo anterior, y ya que hayamos terminado las hojas con las que trabajemos, debemos guardar la información en disco. Existen dos métodos para guardar el archivo, el primero es:

Seleccionar de la barra de menús Archivo – Guardar.



Aparecerá una cuadro de diálogo como el siguiente:

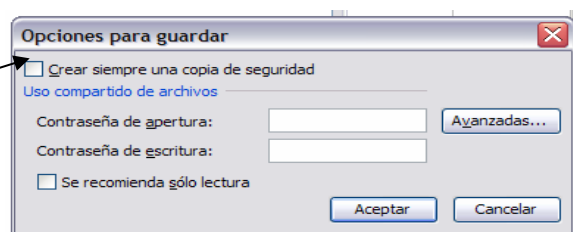


Dé el nombre que desee al documento

Presionar en el botón de Herramientas – Opciones generales

Al presionar en este botón aparecerá una ventana como la que se muestra a continuación:

Si seleccionamos la opción de crear siempre copia de seguridad, esto permitirá que tenga un respaldo de lo que tenía guardado inmediatamente antes de que guardara la última vez, esto es útil en el caso de que borre información que no quería borrar, por ejemplo, celdas, columnas, hojas, etc., lo que permite abrir la copia de respaldo y guardarla con otro nombre.





Tenemos dos cajas de texto, a las cuales podemos poner una contraseña, ésta nos permitirá que no todas las personas lean nuestra información si la contraseña es de protección, y que no pueda guardar los cambios o modificaciones que le haga el archivo en caso de que también haya puesto una contraseña contra escritura. Las contraseñas no tienen que ser iguales, lo que se recomienda es que sean diferentes y que tengan al menos seis caracteres.

Si se selecciona la casilla de verificación “**Se recomienda sólo lectura** ” aparecerá un cuadro de diálogo al abrir el archivo o al guardarlo recordándonos esta selección

Haga clic en el botón de guardar.

Cuando guardamos un libro, se están guardando todas las hojas con las que estemos trabajando en aquel momento a menos de que estuviéramos trabajando en la versión de Excel 4.0 o anterior, pues para guardar varias hojas necesitábamos guardarla como un Workbook. Excel guarda sus archivos en formato XLS aunque podemos guardarlo en otros formatos de hojas de cálculo.

*La segunda forma de guardar un archivo es*



*pulsar el botón **Guardar***

Aparecerá el cuadro de diálogo que vimos anteriormente y se continúa con los pasos que se hicieron.

Para las prácticas del curso te recomendamos que crees una carpeta especial para guardar los archivos.

Es importante crear siempre una copia de seguridad de los archivos que se consideren importantes. Nunca te fíes de tener sólo una copia del archivo o archivos, aunque sea en el disco duro.

También es importante señalar que si has utilizado en tu hoja alguna característica que no existía en versiones anteriores de Excel, se perderán si el libro se abre con alguna de las versiones más antiguas (Excel 7, Excel 5...)





## UNIDAD 2

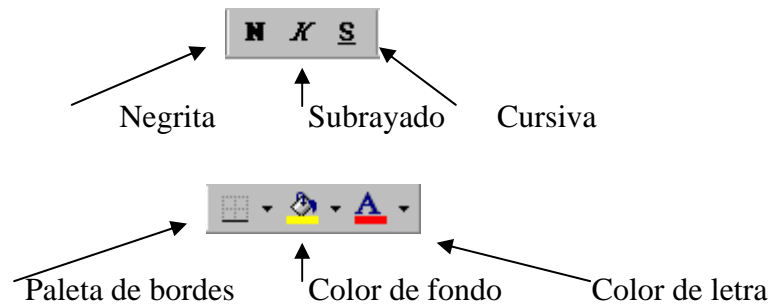
En esta unidad veremos cómo utilizar algunos de los íconos de la barra de herramientas estándar, esto nos permitirá darle un aspecto más profesional a nuestros trabajos.

### Formato de celdas

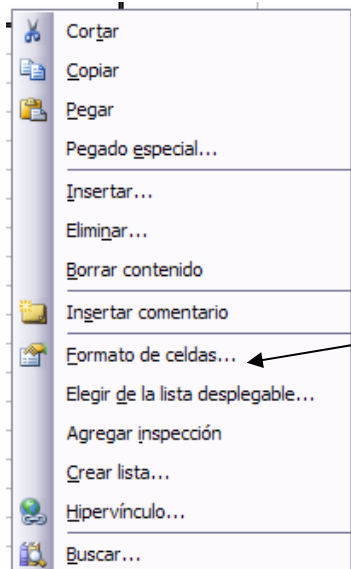
La hojas que hemos ido creando hasta el momento nos han servido para comenzar a introducirnos en el modo de trabajo de Excel, pero su aspecto estético deja bastante que desear.

Podemos dar a nuestra hoja un aspecto bastante más llamativo y hasta fácil de manejar utilizando diferentes formatos de letras, colores, fondos, etc.

Existen varios botones en la barra de herramientas que permiten cambiar algunas de las características mencionadas:



No obstante, existe un menú bastante completo desde donde podemos escoger o hasta modificar alguna característica del formato de las celdas, para acceder a este menú podemos hacerlo de dos formas, la primera es la siguiente:



Sitúate en la celda o rangos de celdas seleccionadas

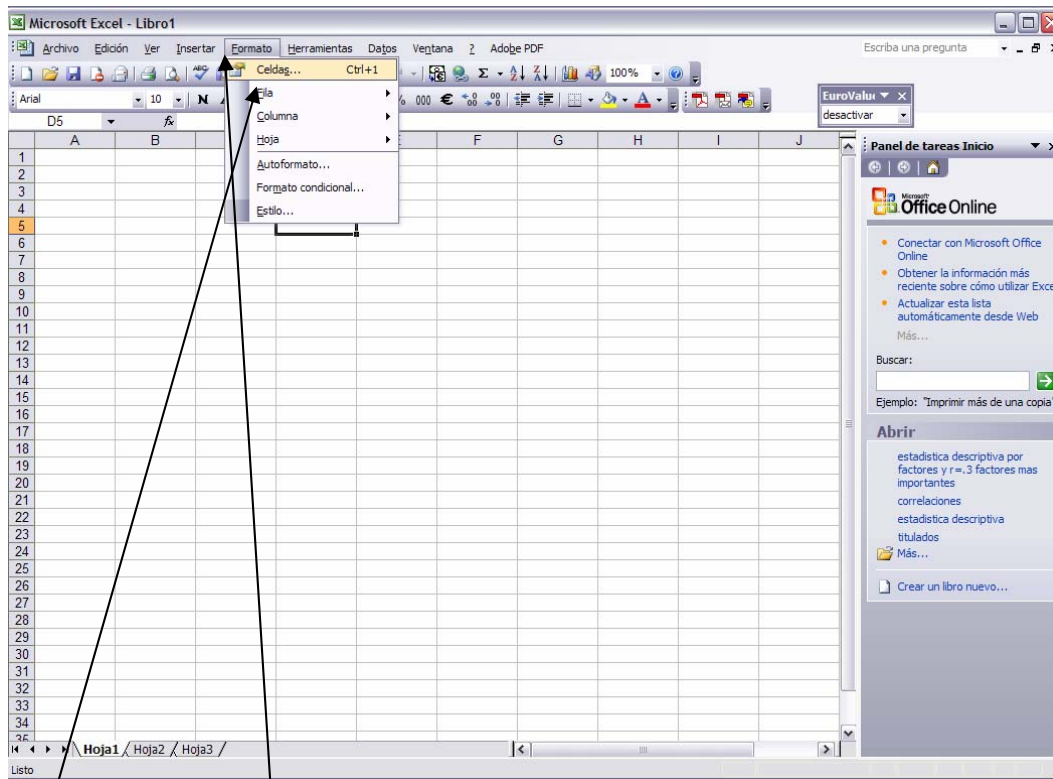
Haz clic en el botón derecho del ratón

Cuando aparezca el menú contextual

Selecciona la opción de **formato de celdas**



La segunda forma de acceder a el menú de formato de celdas, es el siguiente:

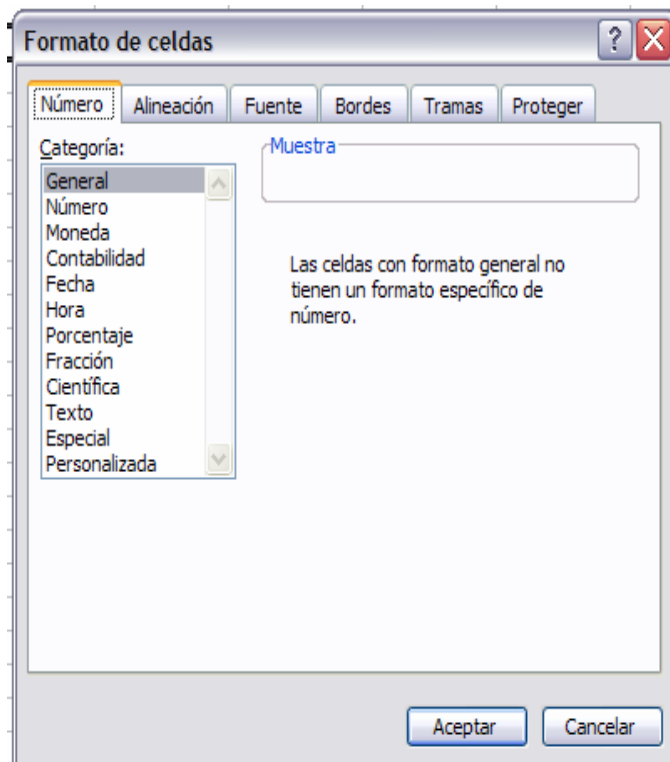


Accede a la opción **Formato - Celdas**:

A continuación aparecerá un menú como el siguiente:

Aquí podemos escoger los formatos para los números, alineación, tipo de letra, colores, etc.

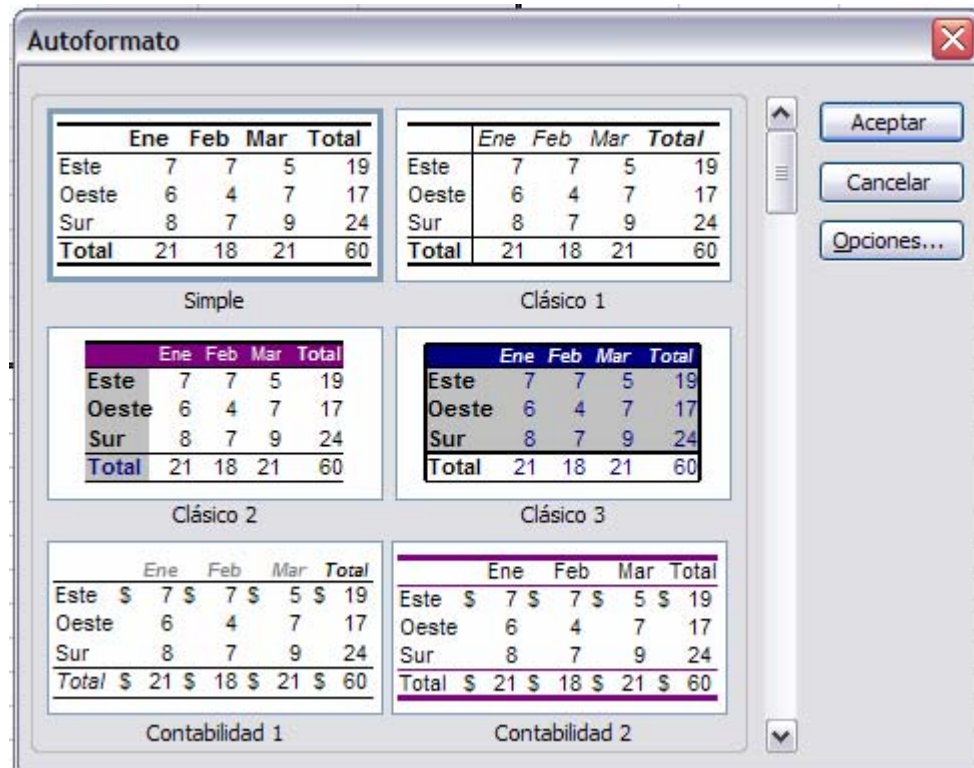
Para colocar un formato a un grupo de celdas, éstas deberían estar seleccionadas previamente. Cuando colocamos un formato cualquiera, por ejemplo formato Moneda, Bordes exteriores y color de letra azul, al salir del cuadro de diálogo podemos escribir y los datos aparecerán ya con el formato escogido. Para dejar esto más



claro, presiona sobre las pestañas de **Alineación**, **Fuente**, **Bordes**, **Tramas** y **Proteger** para familiarizarte con su contenido. Finalmente, sal del cuadro de diálogo.

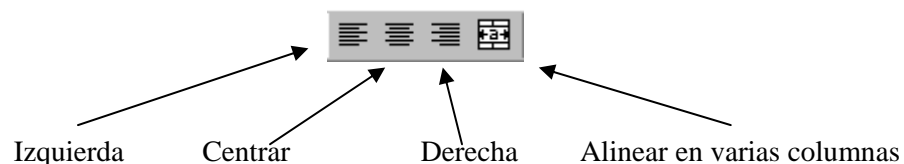
## Autoformato

Otra forma de establecer un formato para las celdas de una hoja que ya contenga datos, es con la opción llamada **Autoformato**. Esta opción nos lleva a un menú desde donde podemos elegir entre varios modelos preestablecidos.



## Alineación de los datos

Para alinear los datos de una celda tenemos los botones de la barra de herramientas:



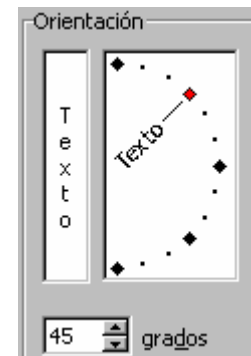
Observa el efecto de cada una de ellas en una celda:



A	B	C	D
Alineación izquierda	Ventas		
Alineación centrada	Ventas		
Alineación derecha	Ventas		
Alineación en varias columnas	Ventas		

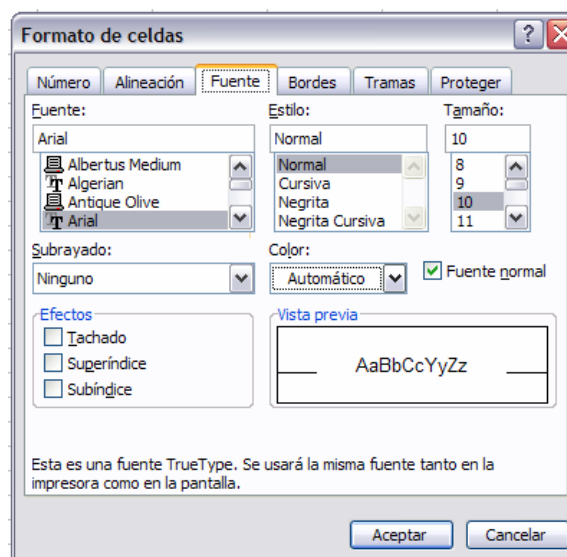
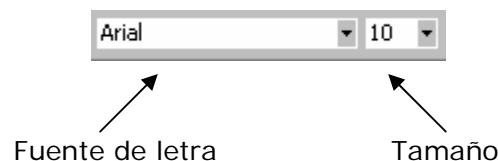
En la cuarta fila, para centrar en varias columnas hemos seleccionado previamente el rango B4:D4 y después hemos pulsado en el botón **Combinar y centrar**. Este último ejemplo se utiliza sobre todo para ajustar el texto cuando éste sobrepasa la anchura de una columna.

Desde el cuadro que hemos visto anteriormente (**Formato – Celdas**) podemos también utilizar un tipo de alineación más específica.



## Tipos y fuentes de letra

Aparte de los botones típicos de la barra de herramientas (negrita, subrayado, cursiva, fuente...) ya hemos visto que existe un cuadro de diálogo bastante más completo desde el cual podemos cambiar el aspecto de una celda o rango.



## Bordes, rellenos y color de letra

Podemos establecer bordes para las celdas o rangos. Es importante no confundir los bordes desde las opciones que vamos a ver, los bordes de referencia que vemos normalmente en Excel. Estos últimos son por defecto de color gris y nos sirven para tener la referencia de las celdas. Podemos incluso ocultarlas o elegir a la hora de la impresión entre imprimirlas o no. En cambio, los bordes añadidos son por defecto negros y forman parte de los datos de la hoja a la hora de visualizarlos o imprimirlos.

Paleta desplegable de bordes



Cuadro de diálogo de bordes



Podemos seleccionar el lado a marcar con un borde, el grosor, el color, si es horizontal, vertical o diagonal, etc.

Paletas de relleno de color y color de letra

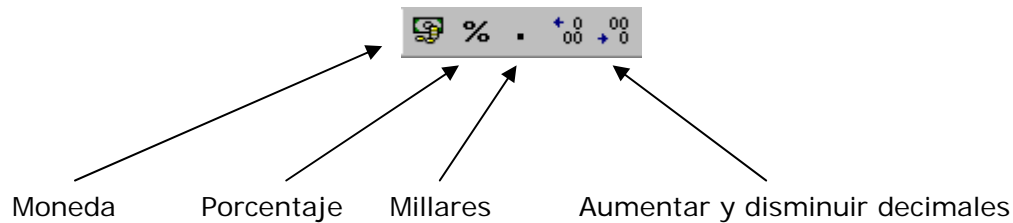


## Formato de los números

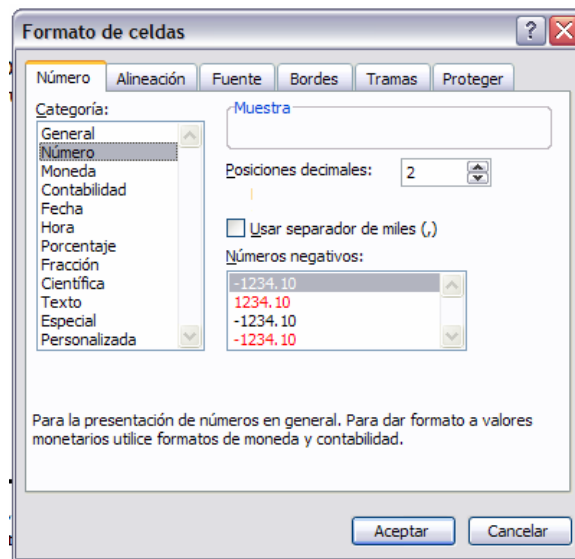
Cuando introducimos números en una hoja de cálculo, el formato de las celdas es el **General**, es decir, números sin ceros, separadores de miles, ni formato monetario, y alineados a la derecha. Podemos darle a las celdas numéricas formatos más descriptivos

y hasta más complejos y personalizados.

Para ello, tenemos una pequeña barra de herramientas con los formatos más utilizados, y también el cuadro de diálogo **Formato –Celdas**.



Desde el menú de **Formato – Celdas** también podemos cambiar el formato de los números accediendo a un completo menú con numerosos formatos preestablecidos.




## Listas

Las listas es una de las opciones que más se utiliza en Excel. Permiten almacenar datos en forma de columnas a modo de base de datos para posteriormente realizar cálculos, consultar datos, realizar subtotales, etc.

Normalmente, una lista contiene las cabeceras de los datos en la primera fila. Estas cabeceras son los títulos de los campos. Un campo es un dato individual con un nombre propio. Observa el siguiente gráfico:

CA



	A	B	C	D
1	<b>Apellidos</b>	<b>Nombre</b>	<b>Ciudad</b>	<b>Carrera</b>
2	Martínez Hernández	Luis Manuel	Durango	Matemáticas
3	Treviño Maese	Oscar	Monterrey	Matemáticas
4	Barbosa Trinidad	Jose Juan	Durango	Informática
5	Gutierrez Polo	Giscela	Guadalajara	Arquitecta
6	Drew Carreón	Verónica	Durango	Ingeniero

Algunas normas a tener en cuenta en la creación de listas son:

- El tamaño máximo de una lista es el mismo que la hoja completa.
- Dejar un espacio por encima y por debajo, de manera que la lista quede aislada del resto de la hoja.
- Los títulos de los campos deben situarse en la primera fila.
- No hay que dejar espacios en blanco al principio del nombre de un campo porque afectará a operaciones posteriores.
- Se recomienda asignar formatos distintos a las cabeceras de columna y a los datos.
- De ser posible, dejar una única lista en la hoja.

Las operaciones más comunes con listas son:

- Utilizarla para imprimir listados.
- Ordenarla por un campo en concreto.
- Crear una ficha llamada “formulario” para trabajar con la lista.
- Filtrar datos, es decir, obtener datos de la lista según unas condiciones específicas.
- Crear informes de resumen de sub-totales de datos.

## Creación de un formulario

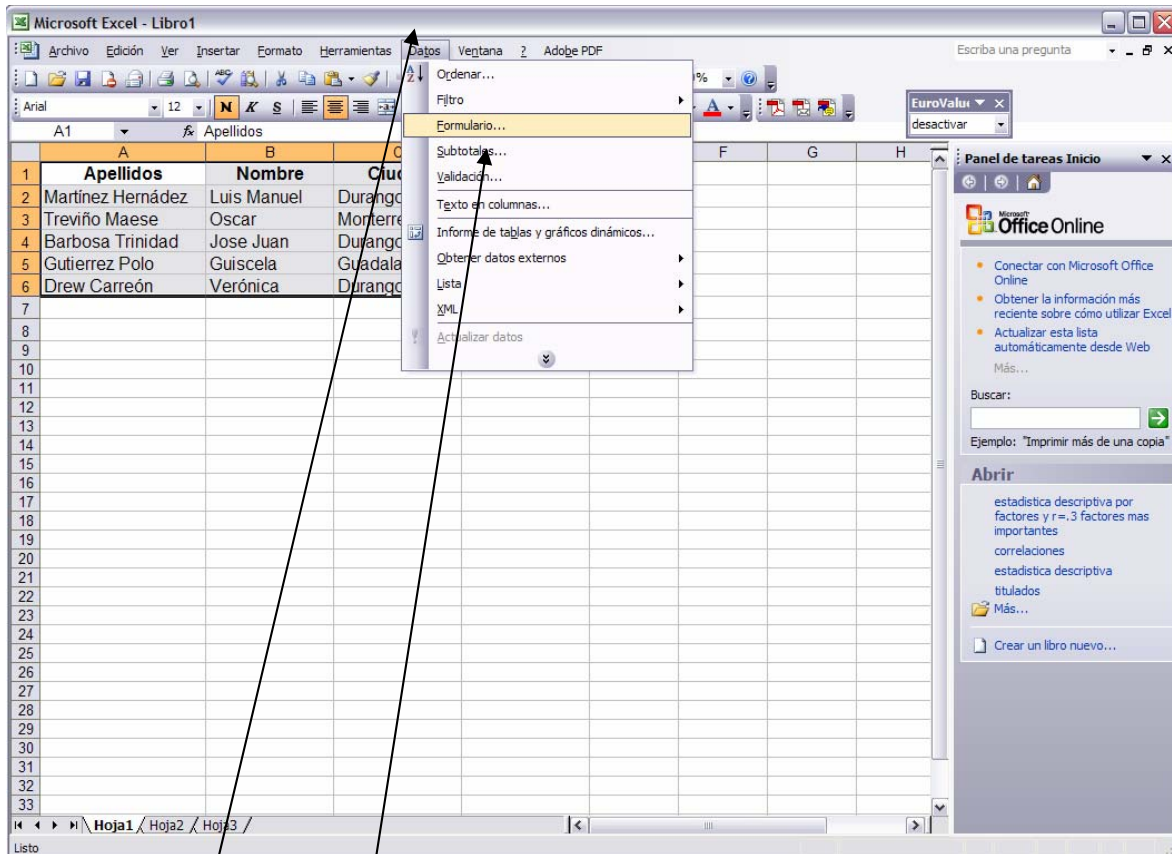
Vamos a ver cómo se crea una ficha de formulario, para ello, sigamos los siguientes pasos:

*Teclea los datos de la hoja de cálculo que se encuentra en el principio de esta página.*





Selecciona todo el rango de datos (A1:B6).



Accede a Datos – Formulario. Te aparecerá automáticamente la ventana:

The 'Formulario' window is open, showing the first record of the data. The fields are: Apellidos: Martínez Hernández, Nombre: Luis Manuel, Ciudad: Durango, Carrera: Matemáticas. The window also includes buttons for 'Nuevo', 'Eliminar', 'Restaurar', 'Buscar anterior', 'Buscar siguiente', 'Criterios', and 'Cerrar'.

La forma de utilizar esta ficha es sumamente sencilla:

- Para desplazarte por los registros debes pulsar las flechas de la lista.

- Observa en la parte superior derecha: muestra el número de registro (fila) donde estamos situados.
- Para crear uno nuevo, puedes pulsar el botón **Nuevo**.
- Para filtrar datos, debes pulsar el botón **Criterios**.

Para encontrar una persona específica, utilizamos el botón de criterios, y presionamos enter, para dejar este punto más claro, realice los siguientes pasos:

1. *Pulsa el botón **Criterios**.*
2. *Pulsa clic en el campo **apellido** y escribe: **Treviño**.*
3. *Pulsa el botón **enter**.*
4. *Observa que ha aparecido el cuarto registro (2 de 5).*

## Ordenar una lista de datos

Puede ocurrir que en ciertos momentos nos interese una misma lista impresa y ordenada por diferentes campos (fechas, nombres, precios, etc). Para ordenar una lista, Excel dispone de dos opciones:

Ordenación rápida: Excel ordena rápidamente a través de un campo mediante el botón **Orden ascendente** situado en la barra de herramientas.

Ordenación por prioridades de campo: Excel permite ordenar también por varios campos. Supongamos que queremos ordenar nuestra lista ciudad, para ello sigue los siguientes pasos:

*Sitúa el cursor en cualquier celda de la columna **C** (columna de **Ciudad**).*

*Pulsa el botón **Orden ascendente** de la barra de herramientas.*



Observa que excepto las cabeceras de columna, el resto de los datos se ha ordenado alfabéticamente por el campo **Ciudad**.

Podemos efectuar la misma ordenación pero en forma descendente a través del botón



*Accede a **Datos – Ordenar** y te aparecerá un cuadro de diálogo:*

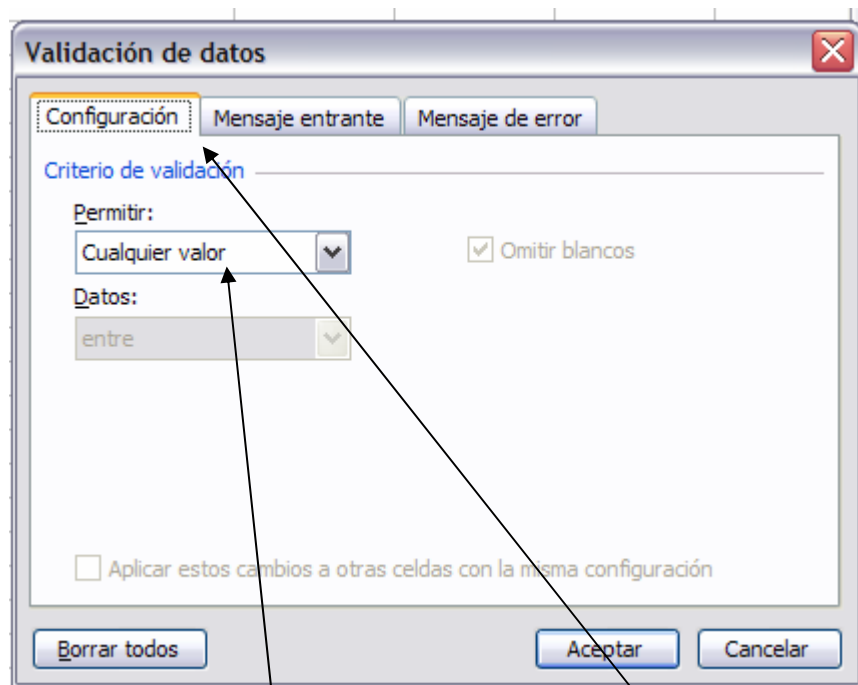
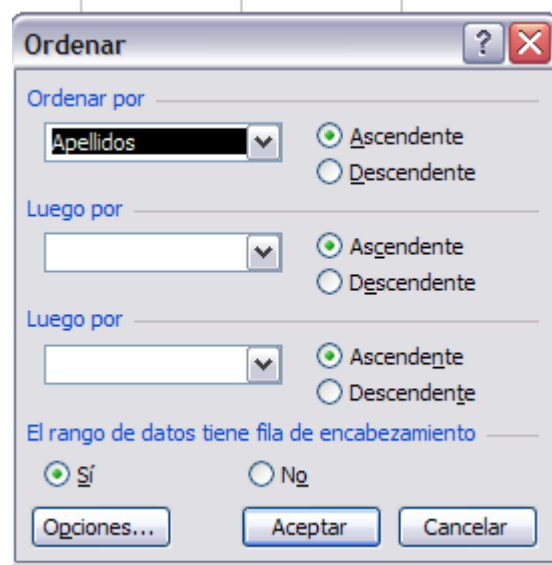
Desde aquí podemos establecer los tres criterios de ordenación que comentábamos anteriormente. En la imagen se aprecia que por prioridad, queremos la ordenación a través del campo **Ciudad**, pero podemos ordenar la lista por **Apellido** y después por **Carrera**, para ello tendríamos que llenar los otros dos criterios.

### Validación de los datos

Imagina que existe una celda que tendrá siempre un dato elegido de entre una lista. En vez de escribir manualmente ese dato, podemos crear una lista desplegable, restringir entradas, limitar el número de caracteres de la celda, mostrar mensajes de ayuda, etc., como se muestra a continuación:

Sitúa el cursor en la celda **D2**.

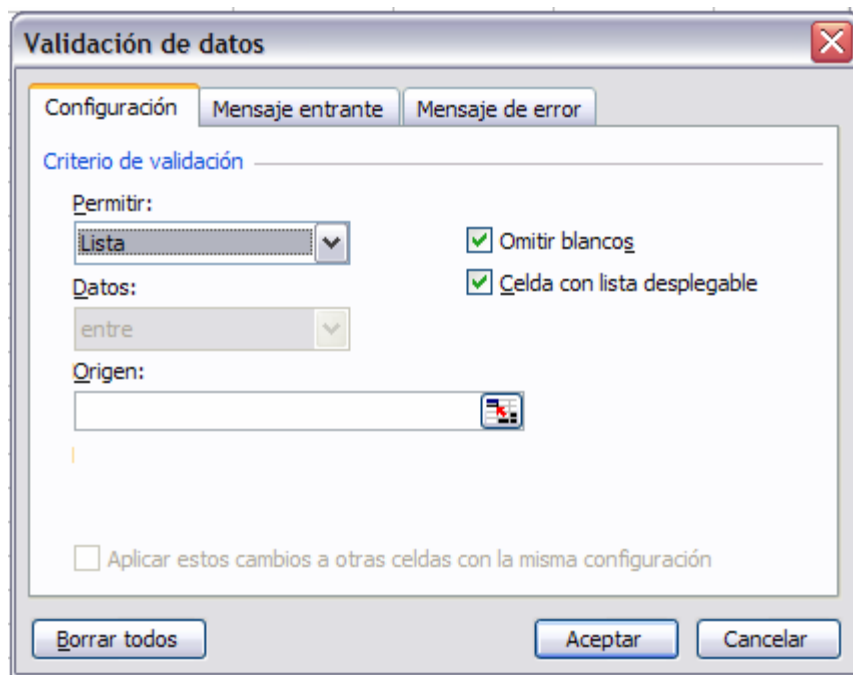
Accede a **Datos – Validación** y te aparecerá el siguiente cuadro:



En la lista desplegable **Permitir** de la ficha de **Configuración** podemos elegir qué valores serán válidos para la celda activa. Desplégala y observa las distintas posibilidades de la misma.

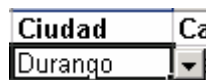


Elige finalmente la opción **Lista**. Te aparecerá una casilla de texto; escribe lo siguiente (separado por punto y coma):

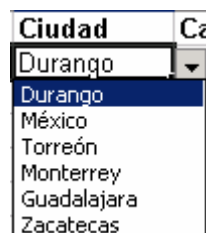


**Durango,México,Torreón,Monterrey,Guadalajara,Zacatecas**

Acepta el cuadro de diálogo.



Observarás que ha aparecido una flecha típica de las listas desplegables, si das un clic en el botón izquierdo del ratón aparecerá la siguiente lista desplegable:



Para aplicar la misma validación a las demás celdas debemos de:

Con el cursor situado en la celda de la lista que acabamos de crear, pulsa la combinación de teclas **Ctrl + C**(copiar al portapapeles).

Selecciona el resto del rango (D2:D6)

Pulsa **Ctrl + V** (pegar del portapapeles)

Pulsa **Esc** para finalizar la selección.

## Funciones especiales de búsqueda

Vamos a ver algunas funciones interesantes que podemos aplicar a las listas de datos.

**BUSCARV:** compara el valor de la búsqueda con la primera columna de la lista y nos devuelve un valor asociado en la misma fila.

**BUSCARH:** compara el valor de la búsqueda con la primera fila de la lista y nos devuelve un valor asociado en la misma columna.

**COINCIDIR:** compara el valor de búsqueda con el contenido de cierta columna que se le pasa como parámetro y devuelve el índice del registro de la lista.

**INDICE:** a partir del índice de la lista, nos proporciona el dato del campo o columna que se le pasa como parámetro.

Veamos algunos ejemplos en la práctica:

*Añade las siguientes celdas a la hoja:*

	A	B	C	D
1	<b>Apellidos</b>	<b>Nombre</b>	<b>Ciudad</b>	<b>Carrera</b>
2	Martínez Hernández	Luis Manuel	Durango	Matemáticas
3	Treviño Maese	Oscar	Monterrey	Matemáticas
4	Barbosa Trinidad	José Juan	Durango	Informática
5	Gutierrez Polo	Guiscela	Guadalajara	Arquitectura
6	Drew Carreón	Verónica	Durango	Ingeniero
7				
8	<b>Apellidos</b>	<b>Nombre</b>	<b>Ciudad</b>	<b>Carrera</b>

Sitúa el cursor en **A9** y escribe: **Treviño Maese** (puedes usar las opciones de Copiar y Pegar).

Sitúa el cursor en la celda **B9**.

Escribe la siguiente fórmula:

**=BUSCARV(A9;A2:D6;2)**

Pulsa la tecla **Intro**.

Observa que ha aparecido el nombre de la lista que corresponde con los apellidos escritos. Esta fórmula busca un valor (A9) en un rango de celdas (A2:D6) y nos devuelve el valor que encuentra, dos posición a su derecha (contándose ella), es decir,



el nombre.

Es una función que trabaja perfectamente para localizar datos en una lista extensa y devolvernos un dato concreto de la misma fila.

Si ahora pruebas a escribir otros apellidos que existan en la lista, comprobarás que la fórmula funciona y se actualiza.

Si la lista no está ordenada alfabéticamente, hay que añadir el parámetro **FALSO** en esta función. Por ejemplo: **=BUSCARV(A9;A2:D6;2;FALSO)** porque de lo contrario, no funcionaría correctamente.

## Filtrado de datos

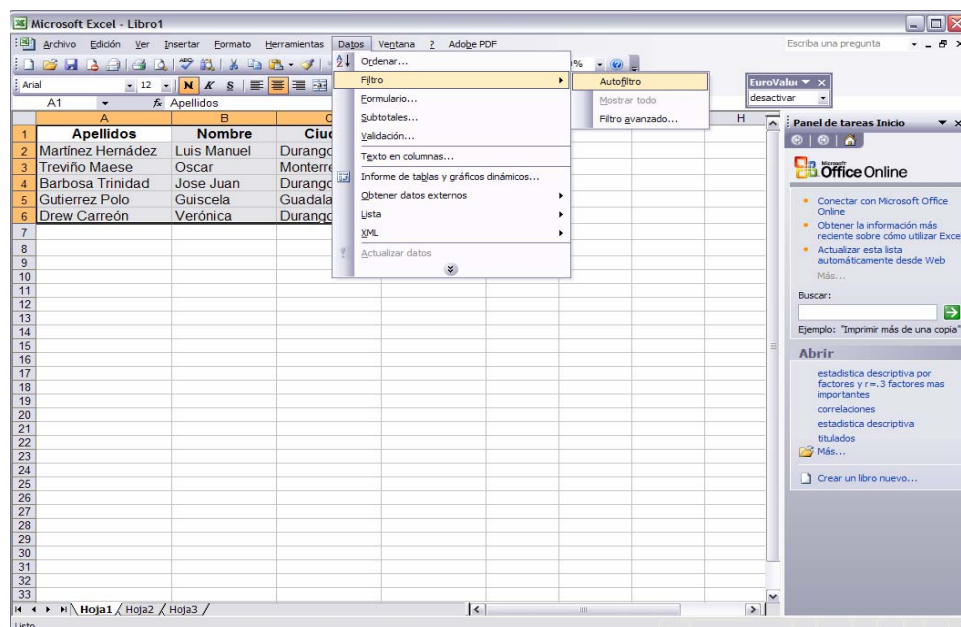
Otra posibilidad para trabajar con listas son los llamados **Filtros**. Éstos actúan en forma de lista desplegable y nos permiten filtrar o elegir datos según unas condiciones específicas, para ello sigamos los siguientes pasos:

*Selecciona el rango de la lista A1:D6*

*Accede a **Datos – Filtro – Autofiltro** y pulsa un clic en cualquier parte de la lista para quitar la selección.*

Observa que han aparecido las típicas flechas correspondientes a las listas desplegables comunes en Windows.

*Abre la lista correspondiente al campo **Ciudad** y selecciona **Monterrey***



*A continuación aparecerán unas listas desplegables en donde se encuentran los títulos*

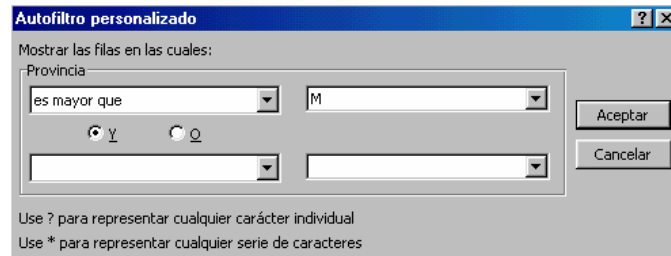
de las celdas como se puede apreciar:

	A	B	C	D
1	Apellidos	Nombre	Ciudad	Carrera
2	Martínez Hernández	Luis Manuel	Durango	Matemáticas

Ahora, vuelve a abrir la misma lista y selecciona la opción **Todas**.

Abre la lista del campo **Ciudad** y elige la opción **Personalizar...**

Prepara el cuadro de diálogo de la siguiente forma:



Observa que han aparecido las ciudades cuya inicial comience a partir de la letra **M**.

Vuelve a mostrar **todas** las ciudades.

En campos numéricos, podríamos por ejemplo ejecutar una consulta que nos muestra los valores más altos, valores a partir de un número determinado, etc.

## UNIDAD 3

En esta Unidad profundizaremos en el estudio de los tipos de datos así como la realización de nuevos ejemplos y ejercicios.

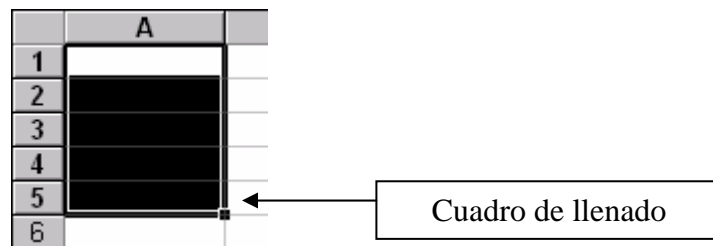
### Copiar, mover y seriación de datos

Ya vimos en la lección anterior cómo podemos copiar celdas con las típicas opciones de Copiar – Cortar y Pegar. Veamos cómo copiar celdas de otra forma.

Cuando el cursor está situado en una celda o estamos seleccionando un rango, el puntero del ratón puede adquirir varias formas según donde esté situado. Por ejemplo, si lo situamos (sin pulsar clic) sobre la selección, el puntero del ratón adquiere una forma de cruz blanca. Esta forma significa que estamos en modo selección normal.

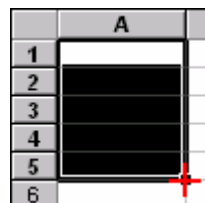


Cuando situas el puntero del ratón en la esquina inferior derecha de la celda o de la selección (sobre un punto negro) el puntero del ratón adquiere una forma de cruz negra. Esta forma indica que estamos en modo copiar o modo llenado. Si arrastramos la celda pulsando el botón izquierdo, realizaremos un *llenado* de celdas



Si situamos el puntero del ratón sobre el cuadro de llenado, éste adquirirá una forma de cruz negra. En la siguiente ilustración mostramos esta forma para que se destaque mejor:

Si arrastramos hacia abajo o hacia un lado, el contenido de las celdas se copiará:



Observa los siguientes pasos:





Situamos el puntero sobre el cuadro de llenado

	A
1	Carne
2	
3	
4	
5	
6	
7	

Arrastramos hacia abajo

	A	B
1	Carne	
2		
3		
4		
5		
6		Carne
7		

Soltamos el ratón y el contenido de la celda se copia

	A
1	Carne
2	Carne
3	Carne
4	Carne
5	Carne
6	

Para quitar la selección en negro, simplemente pulsaremos un clic fuera de la misma, en cualquier celda de la hoja.

## Creación de series

Excel permite crear series de datos a partir del valor inicial de la primera celda o celdas. Simplemente tenemos que utilizar el cuadro de llenado y Excel creará una serie automática.

*Copia los siguientes datos:*

	A	B	C	D
1	Enero	1	Lunes	Tuerca 1
2				

*Selecciona el rango de A1:D1*



Recuerda que para seleccionar el rango posiciona el cursor en la celda A1 y haz clic en el botón izquierdo, pero sin dejar de presionarlo y luego arrastra el ratón hasta la celda D1, después suelta el botón izquierdo del ratón; la segunda forma es posicionarse en la celda A1, presiona la tecla Shift y mantenla presionada, después presiona la tecla de control a la derecha 3 veces y ya tiene seleccionadas las celdas.

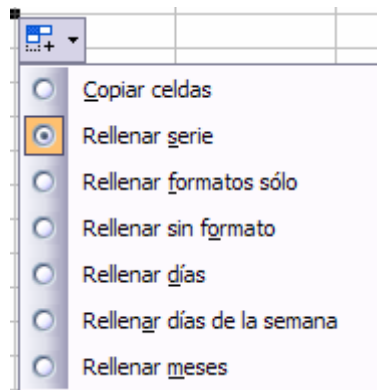
Arrastra el cuadro de llenado unas cuantas celdas hacia abajo:

Suelta el botón del ratón:

Observa cómo Excel ha creado una serie automática de los datos que hemos preparado. De esta forma podemos ahorrarnos tiempo y trabajo en formar listas de datos numeradas, meses, días, etc.

	A	B	C	D
1	Enero		1 Lunes	Tuerca 1
2	Febrero		2 Martes	Tuerca 2
3	Marzo		3 Miércoles	Tuerca 3
4	Abril		4 Jueves	Tuerca 4

Si no nos interesa que realice una serie automática sino que simplemente copie los mismos valores que las celdas iniciales, arrastraremos el cuadro de llenado pulsando al mismo tiempo la tecla de **Control**.

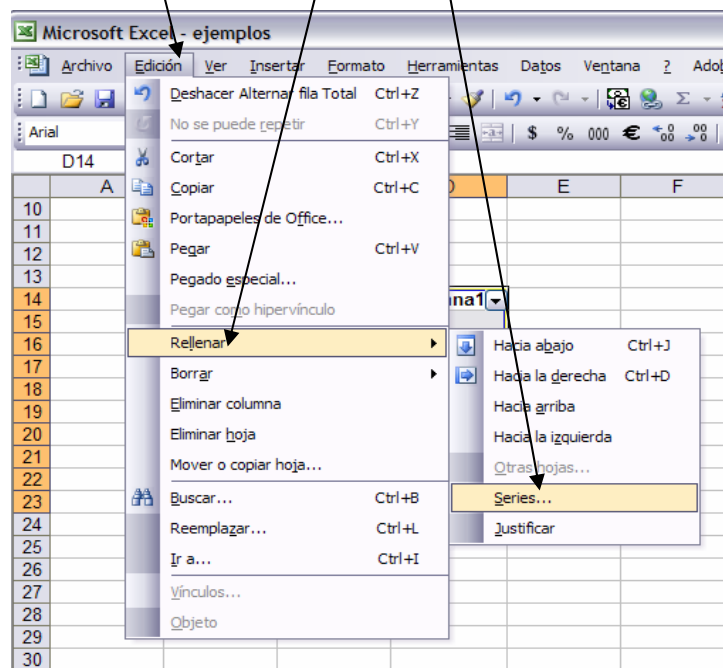


Desde este menú existen opciones para crear series automáticas así como tendencias lineales o geométricas.

Si escogemos la opción **Series...** nos aparecerá un menú donde podemos crear este tipo de series.

Otra forma de crear series es arrastrar el cuadro de llenado, pero con el botón derecho del ratón. Al soltar el botón, Excel mostrará un menú con varias opciones.

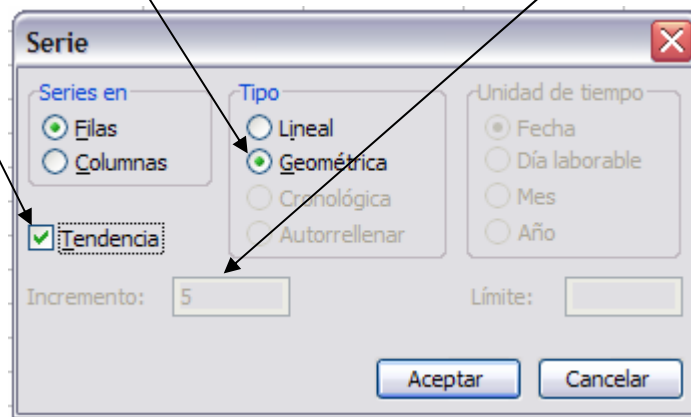
Una tercera forma de crear series es haciendo clic en el menú **Edición – Rellenar – Series**.



Escribe en cualquier celda el número **1**.

Posiciona el cursor en el cuadro de autollenado, presiona el botón derecho del ratón y no lo dejes de presionar y arrastra hacia abajo cuatro o cinco celdas el cuadro de llenado con el botón derecho del ratón y escoge la opción **Series...**

Escribe en la casilla inferior **Incremento** el número **5** y activa la casilla **Tendencia - Geométrica**.



Selecciona el botón **Aceptar**

Excel ha creado una tendencia geométrica a partir del valor inicial.

En el siguiente ejemplo y partiendo del valor **1**, la columna A tiene una serie del tipo **Geométrica** con un **incremento** de 5.

La columna B tiene una serie del tipo **Lineal** con un incremento de 5

La columna C tiene una serie del tipo **Lineal** con un incremento de 1

	A	B	C
1	1	1	1
2	5	6	2
3	25	11	3
4	125	16	4
5	625	21	5

Prueba sin miedo a crear series lineales, geométricas, tendencias y compara y estudia los resultados.

## Copiar y mover celdas

*Otra forma de copiar o mover celdas es situando el puntero del ratón en el mismo borde de la selección. Observa la forma que adopta:*



*Arrastrando de esta forma la selección, moveremos las celdas a otra ubicación. Si lo arrastramos manteniendo pulsada la tecla de Control, lo que haremos será copiar las celdas.*

## Pegado especial

Esta orden se encuentra ubicada en el menú **Edición** y nos permite realizar pegados más específicos que con la orden **Pegar** habitual. Por ejemplo, imaginemos que tenemos una serie de celdas donde hay fórmulas que han dado un resultado. Es posible que queramos copiar y pegar el resultado en otra parte de la hoja. Si realizamos una acción de Copiar y Pegar normal y corriente, lo que se pegarán serán las fórmulas con el resultado. En cambio, con la orden **Pegado especial** podemos hacer que sólo se peguen los valores de los resultados, pero no las fórmulas.

Existen, por supuesto otras posibilidades de pegado especial. Vamos a ver un ejemplo:

*Escribe varios valores en varias celdas. Selecciónalos y pulsa el botón **Copiar**.*

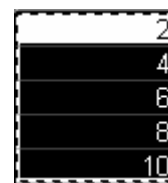


*Por ejemplo:*

*Accede a **Edición – Pegado especial...***

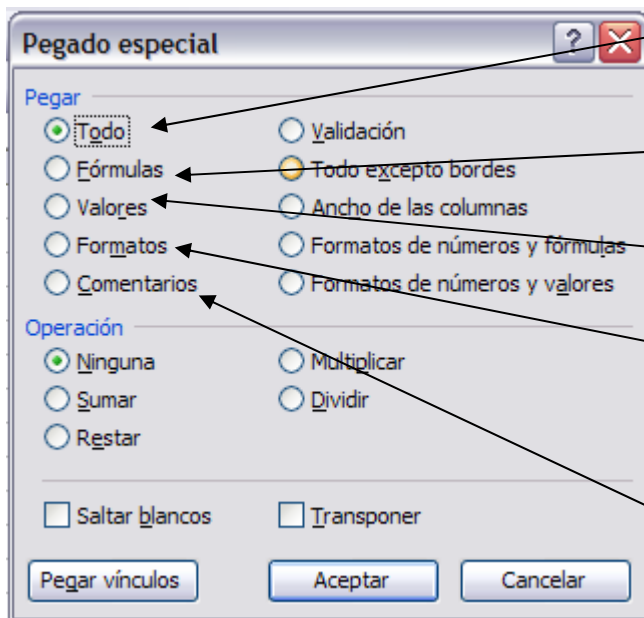
*Del menú que aparece escoge la opción **Operación – Sumar** y acepta.*

Observa que el contenido del portapapeles se ha sumado a las celdas de la hoja. En este caso, hemos doblado las cantidades que había en las hojas.





Las opciones de este menú son:



**Todo:** pega todos los atributos del portapapeles. Es como el pegado normal.

**Fórmulas:** pega sólo las fórmulas de la celda origen.

**Valores:** pega sólo los valores de la celda origen.

**Formatos:** no se pegarán números ni fórmulas. Sólo el formato (negrita, cursiva, etc.) de las celdas originales.

**Comentarios:** sólo se copian los comentarios de las celdas

**Validación:** se pegan las reglas de

validación de entrada de datos.

**Todo excepto bordes:** pega valores, formatos y fórmulas pero no bordes.

En la sección de **Operación** se muestran varias operaciones que pueden realizarse en el área de pegado como hemos visto en la práctica anterior.

**Saltar blancos.** Si está activada, la información que se pega no se pegará en las celdas en blanco.

**Transponer.** Para transponer una selección de celdas. Esta opción cambia la posición de las filas por columnas.

**Pegar vínculos.** Establece un vínculo con la fuente de datos. Si los datos originales cambian, también cambiarán los datos pegados.

*Escribe un rango de datos como el ejemplo:*

*Seleccionalo y pulsa en el botón **Copiar***

*Selecciona ahora el rango de celdas **B1: F1***

	A
1	1
2	2
3	3
4	4
5	5

	A	B	C	D	E	F
1	1					
2	2					
3	3					
4	4					
5	5					



## Accede a Edición – Pegado especial

Activa la casilla **Transponer** y acepta.

	A	B	C	D	E	F
1	1	1	2	3	4	5
2	2					
3	3					
4	4					
5	5					

Hemos seleccionado cinco celdas hacia la derecha porque de lo contrario no funcionaría la acción de transponer. Es decir, que hemos de seleccionar para la zona del pegado especial el mismo número de celdas que el rango original.

## Insertar y eliminar filas y columnas

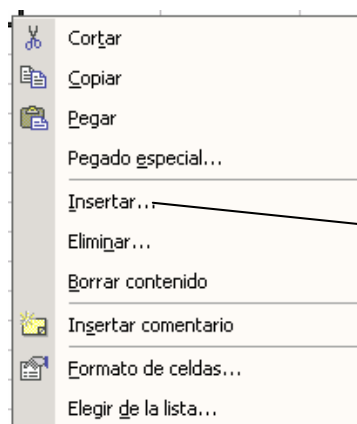
Al insertar filas o columnas en Excel, las celdas se desplazan para dejar sitio a las nuevas celdas. Es muy fácil insertar una fila o una columna.

Imaginemos que tenemos una lista cualquiera de datos y queremos insertar una fila nueva entre la fila 2 y la fila 3.

*Pulsa un clic a la izquierda de la fila, en el número de fila.  
Debe seleccionarse la misma:*

	A	B
1	GASTOS	
2	Oficina	3233
3	Luz	5600
4	Teléfono	12000
5	Agua	3000

Accede a **Insertar – Filas** o bien pulsa el botón derecho del mouse sobre el número de fila y escoge **Insertar**



*Se habrá insertado una nueva fila.*

	A	B
1	GASTOS	
2	Oficina	3233
3		
4	Luz	5600
5	Teléfono	12000
6	Agua	3000

De la misma forma podríamos borrar una fila completa. (Seleccionándola y accediendo

a **Edición – Eliminar**).

Al igual que las filas, también podemos insertar y eliminar columnas de la misma forma.

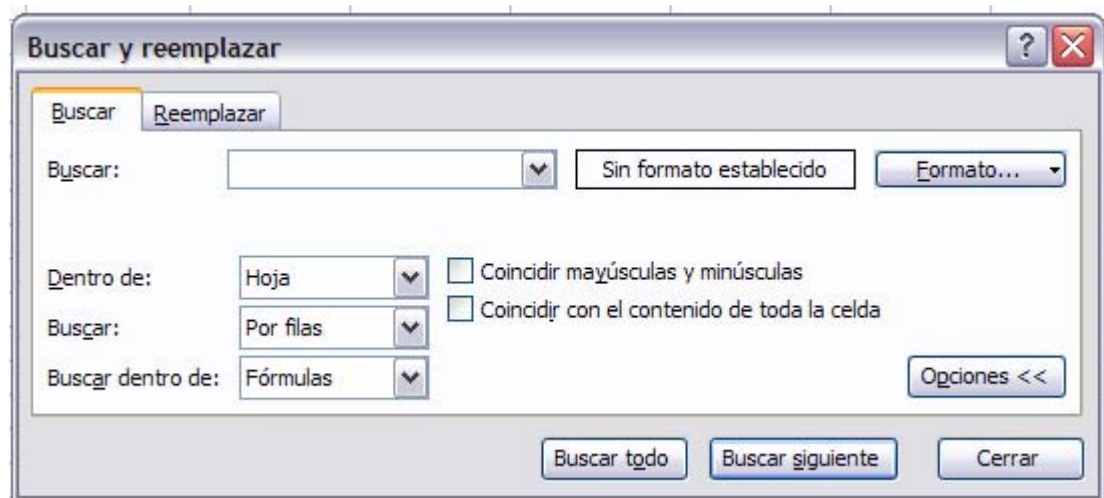
Se selecciona la letra de la columna.

Se accede a **Insertar – Columnas** si se quieren insertar.

Se accede a **Edición – Eliminar** si se quieren eliminar.

## Buscar y reemplazar datos

Al igual que otros programas de Windows, con Excel es posible buscar algún dato en el libro de trabajo desde **Edición – Buscar**. Aparecerá la típica pantalla desde donde podemos escribir alguna palabra que Excel se encargará de buscar.



Asimismo, podemos indicarle que reemplace un dato por otro en todo el libro. Esto último podemos hacerlo desde el mismo cuadro de diálogo de **Buscar** o bien desde la opción **Edición – Reemplazar**.

## Selección de celdas no-adyacentes

Si lo que deseas es seleccionar un rango de celdas que no estén juntas, deberás hacerlo pulsando al mismo tiempo que seleccionas, es decir, presiona la tecla **[Ctrl]** y sin dejar de presionar la tecla control, selecciona las celdas con el ratón, cuantas veces necesite.

	A	B	C
1			
2			
3			
4			

## Llenar datos en un rango

Una de las formas de escribir en cada una de las celdas de un rango, en vez de la forma habitual, es:

Seleccionar el rango.

Escribir el dato de la primera celda.

Pulsar Intro.

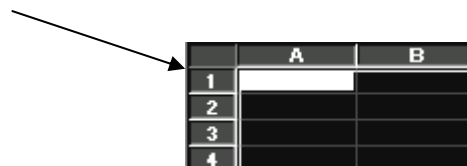
Escribir el dato de la segunda celda...

Continuar sucesivamente hasta el final del rango.

## Borrar todos los datos de la hoja

Una opción rápida para borrar todos los datos de una hoja es cerrando el libro sin grabarlo y creando uno nuevo, pero es posible que tengamos datos en otras hojas del libro que no queramos desperdiciar. También podríamos eliminar la hoja, pero un buen sistema sería el siguiente:

Pulsa en el cuadro de la esquina superior derecha (encima de los rótulos de las filas y a la izquierda de los rótulos de las columnas). Verás que toda la hoja queda seleccionada.



Pulsa la tecla **Supr** del teclado.

Pulsa un clic en cualquier parte de la hoja para quitar la selección.

La totalidad de los datos se han borrado.



## Inmovilizar paneles

En hojas muy extensas puede ocurrir que tengamos una o varias filas o columnas con rótulos de nombres y que al desplazar la hoja y debido a su longitud, perdamos de vista esos rótulos que nos pueden servir como referencia. Observa el ejemplo:

	A	B	C	D	E
1		ENERO	FEBRERO	MARZO	ABRIL
2	JOSÉ	80	86	12	54
3	JUAN	48	57	40	60
4	PEDRO	77	27	60	86
5	RAÚL	73	86	99	10
6	VERONICA	60	24	99	25
7	ELIZABETH	48	53	83	14
8	JUDITH	46	36	1	27
9	GUISCELA	11	21	91	53
10	DIANA	97	2	67	22

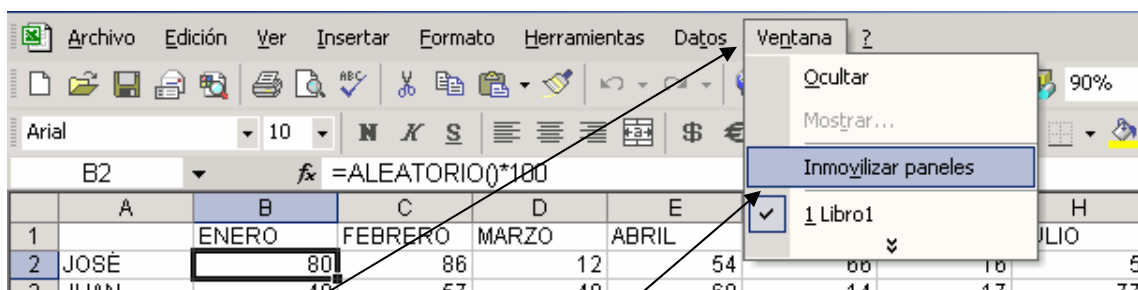
Imagínate que en vez de 4 columnas de datos, son 200 columnas. Al estar el cursor situado cerca de la columna A, tenemos como referencia de los datos a los nombres de dicha columna, pero si desplazamos la pantalla hacia la izquierda, perderíamos la referencia de los nombres.

J	K	L	M
SEPTIEMBR	OCTUBRE	NOVIEMBRE	DICIEMBRE
9	59	51	34
38	73	26	77
72	62	10	98
86	92	73	1
32	20	79	32
64	77	28	48
55	62	29	65
29	92	39	39
36	44	62	78

Para que no ocurra esto tendríamos que:

Situar el cursor en la celda B2.

Esta celda contiene por encima los rótulos de los meses y a su izquierda contiene los rótulos de las personas.



Accede a **Ventana** – **Inmovilizar paneles**

Veremos unas líneas negras que significan la división que hemos hecho. Lo que haya por encima y a la izquierda de esas líneas será lo que quede inmovilizado.

	A	B	C	D	E
1		ENERO	FEBRERO	MARZO	ABRIL
2	JOSÉ	80	86	12	54
3	JUAN	48	57	40	60
4	PEDRO	77	27	60	86
5	RAÚL	73	86	99	10
6	VERONICA	60	24	99	25
7	ELIZABETH	48	53	83	14
8	JUDITH	46	36	1	27
9	GUISCELA	11	21	91	53
10	DIANA	97	2	67	22
11					

Ahora podríamos desplazarnos hacia la derecha y siempre veríamos la columna izquierda que nos serviría como referencia. De la misma forma, si nos desplazamos hacia abajo, veremos la fila de los meses inmovilizada.

	A	J	K	L	M
1		SEPTIEMBRE	OCTUBRE	NOVIEMBRE	DICIEMBRE
2	JOSÉ	9	59	51	34
3	JUAN	38	73	26	77
4	PEDRO	72	62	10	98
5	RAÚL	86	92	73	1
6	VERONICA	32	20	79	32
7	ELIZABETH	64	77	28	48
8	JUDITH	55	62	29	65
9	GUISCELA	29	92	39	39
10	DIANA	36	44	62	78
11					

Para anular la inmovilización de los paneles, deberíamos acceder a

## Ventana – Movilizar paneles

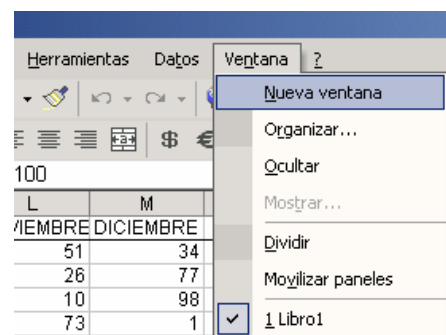
y las líneas de inmovilización desaparecerían, quedando la hoja como estaba antes.

## División en ventanas

En libros extensos es posible crear una o varias ventanas del mismo libro y trabajar en una u otra. Para ello debemos acceder a:


### Ventana – Nueva ventana

Con esta orden se habrá creado una nueva ventana del mismo libro. Si al principio te cuesta controlar qué ventana estás viendo, puedes organizarlas con **Ventana – Organizar** y escoger entre mosaico o



cascada. Es importante observar el nombre de la ventana en la barra azul del título; aparece junto a dos puntos y el número de ventana. En la siguiente imagen, se muestran dos ventanas de un mismo libro, donde la ventana de la izquierda es la activa (azul).

Libro1:2					Libro1:1				
	A	B	C	MA		A	B	C	D
1		ENERO	FEBRERO	MA	1		ENERO	FEBRERO	MARZO
2	JOSÉ	80	86		2	JOSÉ	80	86	
3	JUAN	48	57		3	JUAN	48	57	
4	PEDRO	77	27		4	PEDRO	77	27	
5	RAÚL	73	86		5	RAÚL	73	86	
6	VERONICA	60	24		6	VERONICA	60	24	
7	ELIZABETH	48	53		7	ELIZABETH	48	53	
8	JUDITH	46	36		8	JUDITH	46	36	
9	GUISCELA	11	21		9	GUISCELA	11	21	
10	DIANA	97	2		10	DIANA	97	2	
11					11				
12					12				
13					13				
14					14				

Si escribimos algo en una de las dos ventanas, veremos cómo en la otra se va escribiendo exactamente lo mismo. Para pasar de una ventana a otra pulsaremos un clic, y para cerrar una de las dos ventanas, simplemente pulsaremos el botón de cerrar  teniendo en cuenta que **el cierre de la última ventana supone el cierre del libro.**

## UNIDAD 4

En esta unidad comenzaremos a estudiar la parte más importante de Excel, las fórmulas, con múltiples ejemplos prácticos.

### La sintaxis de una fórmula

Una **fórmula** es una expresión que introducimos en una celda y que relaciona valores y fórmulas de otras celdas para producir un resultado. Una fórmula comienza siempre con el signo igual (=) y puede contener textos, números, referencias de celdas, etc.

En la celda que contiene una fórmula se visualiza siempre el resultado de la misma y la fórmula en sí se visualiza en la barra de fórmulas.

La fórmula combina diferentes operadores para realizar los cálculos. Estos operadores son:

#### Aritméticos:

Suma	+
Resta	-
Multiplicación	*
División	/
Porcentaje	%
Exponente	^

#### De comparación:

Igual	=
Distinto	<>
Mayor	>
Menor	<
Mayor o igual	>=
Menor o igual	<=

#### De texto:

Concatenación &

Este último operador sirve básicamente para unir cadenas de texto y producir un nuevo valor a partir de esa unión. P.Ejemplo: Zorro&Rojo = ZorroRojo

Ejemplos de fórmulas son:

=12+5	Suma los valores numéricos 12 y 5
=C1+C5	Suma el contenido de las celdas C1 y C5
=(C1+C5)-A2	Suma el contenido de las celdas C1 y C5 y al resultado le

<b>=Ventas-Gastos</b>	resta A2.
<b>=2^3</b>	Resta dos rangos de celdas llamados Ventas y Gastos
	Eleva al cubo el número 2

## Prioridad en las fórmulas

Es muy importante señalar que en una fórmula, la introducción de algunos de los diferentes operadores tiene prioridad sobre otros. Observa el orden de prioridad de los operadores.

<b>Mayor prioridad</b>	Porcentaje
	Exponente
	Multiplicación y división
	Suma y resta
	Unión de texto
<b>Menor prioridad</b>	Comparación

Así, si introducimos la fórmula:

**= 10 + 2 \* 10** producirá un resultado de **30**, pues primero se realiza la operación de multiplicación de **2 \* 10** y finalmente se le suma el primer 10.

**= (10 + 2) \* 10** producirá un resultado de **120**, pues en este caso se producirá en primer lugar la multiplicación del interior del paréntesis multiplicando su resultado por el último 10.

## Mensajes de error

En algún momento puede producirse el hecho de que nos equivoquemos en la realización de una fórmula y que ésta intente realizar cálculos con datos erróneos. Por ejemplo, podemos intentar **=C1+C2** habiendo un texto en C1 y un número en C2, por lo que Excel devolverá un mensaje de error. Observa los siguientes mensajes de error y su causa:

<b>#¡DIV/0!</b>	Se está intentando dividir un número entre 0
<b>#N/A</b>	Valor no disponible
<b>#¡NOMBRE?</b>	Se ha utilizado un nombre que Excel no reconoce
<b>#¡NULO!</b>	Intersección no válida de dos áreas
<b>#¡NUM!</b>	Número utilizado de forma incorrecta
<b>#¡REF!</b>	Referencia no válida a una celda
<b>#¡VALOR!</b>	Operando o argumento erróneo
<b>#####</b>	Columna demasiado estrecha para ver los datos

Para ver con más claridad el uso de fórmulas, hagamos el siguiente ejercicio:

*Sitúate en la celda B2, teclea el número 100000 y presiona enter.*

Sitúate en la celda B3, teclea el número 500000 y presiona enter.

Sitúate en la celda B4, teclea el número 55000 y presiona enter.

Sitúa el cursor en la celda B6.

Escribe la siguiente fórmula:

**=B2+B3+B4**

Pulsa **Intro**

	A	B	C	D
1		Enero	Febrero	Marzo
2	Ventas	100000	15000	40000
3	Ingresos	500000	70000	750000
4	Varios	55000	45000	12000
5				
6	TOTALES			

Aparecerá el resultado de la fórmula. Cuando trabajamos con fórmulas, Excel calcula siempre el contenido de la fórmula que estamos utilizando. En este caso, podríamos introducir la fórmula. **=100000+50000+25000** pero siempre daría el mismo resultado porque lo que hacemos es calcular una suma con números fijos. Por eso utilizaremos los nombres de las celdas. La ventaja será que si posteriormente cambiamos algún dato de las celdas, la fórmula se recalculará automáticamente y volverá a darnos el resultado actualizado.

A continuación podríamos introducir la misma fórmula bajo la columna de los números de los gastos, pero lo que haremos será utilizar la potente función de copia de Excel.

Sitúa el cursor en la celda B6 y pulsa el botón **Copiar** de la barra de herramientas (o bien la opción **Edición – Copiar**).

Sitúa el cursor en la celda C6 y pulsa el botón **Pegar** de la barra de herramientas (o bien la opción **Edición – Pegar**).

La fórmula se ha copiado, pero Excel ha actualizado las celdas de la fórmula a la columna donde se encuentra el cursor actualmente.

Sitúa el cursor en la celda A9 y escribe el siguiente texto:

**BENEFICIOS:**

Sitúa el cursor en la celda A10 y escribe la siguiente fórmula:

**=B7-E7**

Pulsa **Intro** y verás que el resultado es negativo, es decir, los gastos han sido superiores a los ingresos.

Graba la hoja. Puedes darle el nombre que desees.

Accede a **Archivo – Cerrar**

Accede a **Archivo – Nuevo** y acepta el nuevo libro de trabajo.

Copia la siguiente hoja:

	A	B	C	D
1		Enero	Febrero	Marzo
2	Ventas	100000	15000	40000
3	Ingresos	500000	70000	750000
4	Varios	55000	45000	12000
5				
6	TOTALES			

Sitúa el cursor en la celda **B6** y escribe la fórmula:

**=B2+B3+B4**

(A partir de ahora supondremos que has pulsado **Intro** para validar la fórmula).

Vuelve a situarte en **B6**

Sitúa el cursor del ratón en la esquina inferior derecha de la celda, de forma que sin pulsar nada, aparezca una cruz negra. Cuando la veas, pulsa clic y sin soltar el ratón, “arrastra” hacia la derecha hasta la celda **D6**

Si ha funcionado correctamente, la fórmula de la celda inicial se habrá copiado en las dos celdas de al lado, dando como resultado, la suma de cada columna.

	A	B	C	D
1		Enero	Febrero	Marzo
2	Ventas	100000	15000	40000
3	Ingresos	500000	70000	750000
4	Varios	55000	45000	12000
5				
6	TOTALES	655000	130000	802000

Para quitar la selección de color negro, puedes pulsar un clic en cualquier parte de la hoja.

## Referencias

Cuando copiamos fórmulas de la forma que acabamos de ver, el contenido de la fórmula se actualiza a medida que copiamos en horizontal o en vertical. Si te sitúas en las celdas C6 y D6 y miras en la barra de fórmulas, observarás que cada celda contiene la fórmula de su columna correcta. La referencia indica la posición de la celda contenida en la fórmula. Observa la siguiente hoja:

	A	B	C	D	E	F
1		Enero	Febrero	Marzo		Aumento fijo
2	Ventas	100000	15000	40000		500
3	Ingresos	500000	70000	750000		
4	Varios	55000	45000	12000		
5						
6	TOTALES	655500	130000	802000		

En este caso, en la primera fórmula de la celda **B6** hemos sumado la columna B, pero también hemos incluido en la fórmula la celda **F1** de forma que sume el contenido de esta celda en la suma de la columna. En la primera celda no pasa nada, pero si volvemos a copiar la fórmula en las celdas de al lado, observaremos en la celda **C6** lo siguiente: **=C2+C3+C4+G2**. Es decir, Excel ha copiado la fórmula, pero también ha desplazado la referencia de la celda F2 y ahora la ha convertido en G2. No hace falta mencionar que en G2 no hay ningún dato. Excel ha tomado las referencias de la primera celda como posiciones **relativas** y las ha copiado hacia su derecha. En nuestro ejemplo, no nos interesa que la celda **F2** se modifique a medida que copiamos la fórmula.

Para que no ocurra esto, debemos convertir la celda **F2** en referencia **absoluta** es decir, que aunque copiemos la fórmula en otras posiciones, la referencia a la celda **F2** no cambie nunca.

**Celdas relativas:** indican la posición de la celda como desplazamiento a partir de la cual se está introduciendo la fórmula. Si las celdas referenciadas cambian de ubicación, Excel ajusta las referencias para adaptarlas a la nueva posición.

**Celdas absolutas:** indican posiciones que no cambian. Una celda se convierte en absoluta añadiendo antes y después de la letra de la columna el signo pesos (\$). Por ejemplo: **\$B\$6**

**Celdas mixtas:** combina los dos tipos de referencia anteriores. Por ejemplo: **\$B6, A\$7**.

Podemos convertir una celda en absoluta posicionando el cursor al lado del nombre de la columna y pulsando la tecla **F4**. Esto añade automáticamente el signo dólar a la columna y la convierte en absoluta.

Siguiendo con nuestro ejemplo, si modificamos la fórmula de la primera celda como sigue: **=B2+B3+B4+\$F\$2** y la volvemos a copiar hacia la derecha, observaremos que





Excel ha actualizado las columnas a las nuevas posiciones de las fórmulas (relativas), pero la celda **F2** no cambia en la copia (absoluta).

## UNIDAD 5

En esta Unidad veremos una de las posibilidades más potentes de Excel: las funciones. Veremos algunos ejemplos prácticos para comenzar.

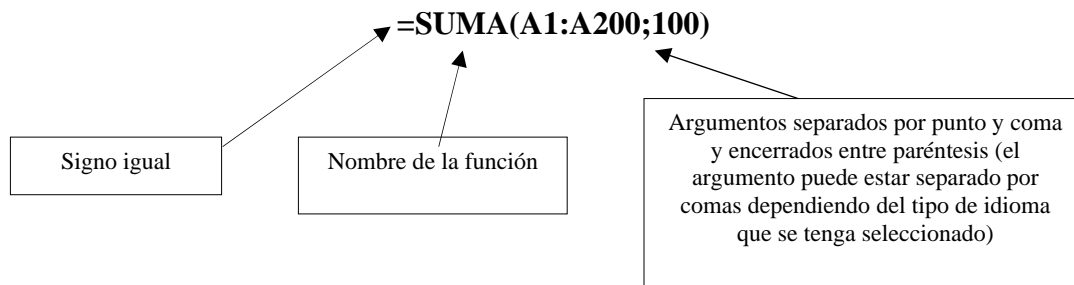
### Funciones

Una **función** es una fórmula ya escrita y preparada para realizar cálculos y simplificar el uso de fórmulas extensas. Las funciones tienen un nombre propio y existen multitud de funciones. Imagínate sumar un rango de 200 celdas con una fórmula del tipo  $=A1+A2+A3+A4...$

Existen funciones que realizan complejos cálculos financieros, estadísticos, matemáticos, etc, y que permiten ahorrar trabajo y tiempo en la escritura de una fórmula.

### Sintaxis de una función

Las funciones deben mantener unas reglas de sintaxis tal y como se indica en el siguiente ejemplo:



En el ejemplo, se sumará todo el rango A1:A200 y aparte el número 100. Es decir, que dentro de los paréntesis que forman el contenido de la función, hay **dos argumentos** a sumar.

### La función Autosuma

Es quizá la función más utilizada en una hoja de cálculo. Por ello, Excel proporciona un botón exclusivo para la función Autosuma en la barra de herramientas.

Para utilizar una función, podemos escribirla manualmente o bien utilizar el Asistente para funciones que veremos posteriormente y que nos irá guiando paso a paso en la construcción de la función. Para clarificar el uso de esta función, hagamos el siguiente ejercicio:

Escribe en una hoja nueva unos cuantos números y después coloca el cursor bajo esa misma lista:

	A
1	4123
2	2134
3	4323
4	6234
5	

Pulsa el botón **Autosuma** situado en la barra de herramientas estándar



Observa que Excel detecta lo que queremos sumar y lo marca con puntos suspensivos intermitentes. Ahora podemos aceptar pulsando **Intro** o bien seleccionar con el ratón la zona que queremos sumar.

Pulsa **Intro**.

Otra forma de hacerlo es la siguiente:

Borra el contenido de la celda que contiene la fórmula.

Selecciona toda el área numérica, última celda incluida:

	A
1	4123
2	2134
3	4323
4	6234
5	

Pulsa el botón **Autosuma**.

En este caso marcamos directamente el rango que queremos sumar, por lo que Excel lo suma directamente.

Con el cursor situado en la celda que contiene la fórmula, observa la barra de fórmulas.

=SUMA(A1:A4)

La función tiene entre paréntesis la celda inicial del rango a sumar y la celda final separadas por dos puntos. Desde aquí podemos modificar manualmente el rango.



## La función PROMEDIO

Otra interesante función es la llamada **=PROMEDIO()**. Funciona exactamente igual que la suma, pero no existe ningún botón, por lo que debemos introducirla manualmente. Cuando introducimos una función mediante el teclado, podemos escribirla por completo o hacer lo siguiente:

*Borra el contenido de la última fórmula*

*Escribe lo siguiente:*

**=PROMEDIO(**

*Selecciona con el ratón el rango de números. Fíjate cómo la fórmula va tomando dicho rango y se va escribiendo sola.*

*Escribe ) para cerrar la fórmula.*

*El resultado obtenido es la media de los datos numéricos.*

## El asistente para funciones

Existen muchos tipos de funciones: matemáticas, estadísticas, de fecha, científicas, etc., alguna de las cuales contiene una sintaxis bastante más difícil que la autosuma, por ejemplo. Existen funciones que realizan complejos cálculos y que tan sólo nos piden unos datos específicos.

Si no recordamos la sintaxis de una función, podemos hacerlo con el **Asistente de funciones** el cual, nos guiará paso a paso hasta obtener el resultado buscado, como lo veremos a continuación.

*Borra el contenido de la última fórmula y sitúa el cursor en ella.*

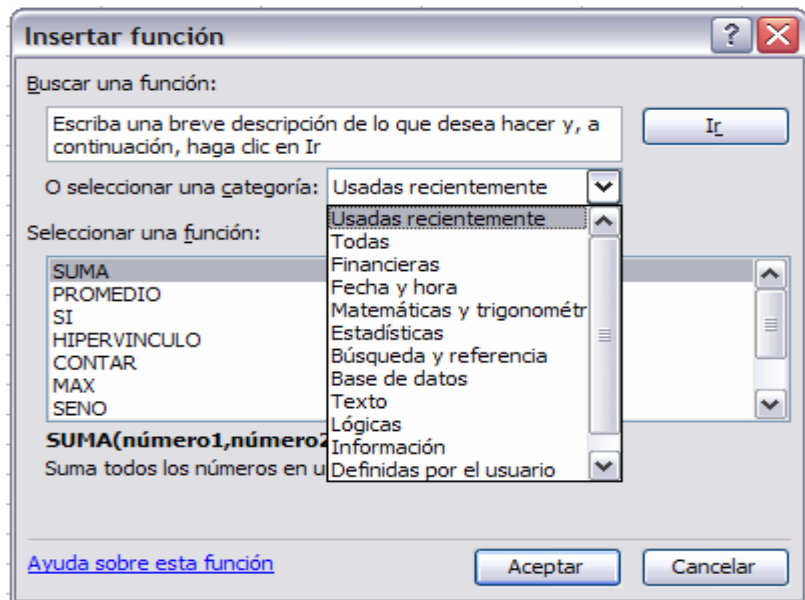
*Pulsa el botón **Pegar función** de la barra de herramientas estándar*



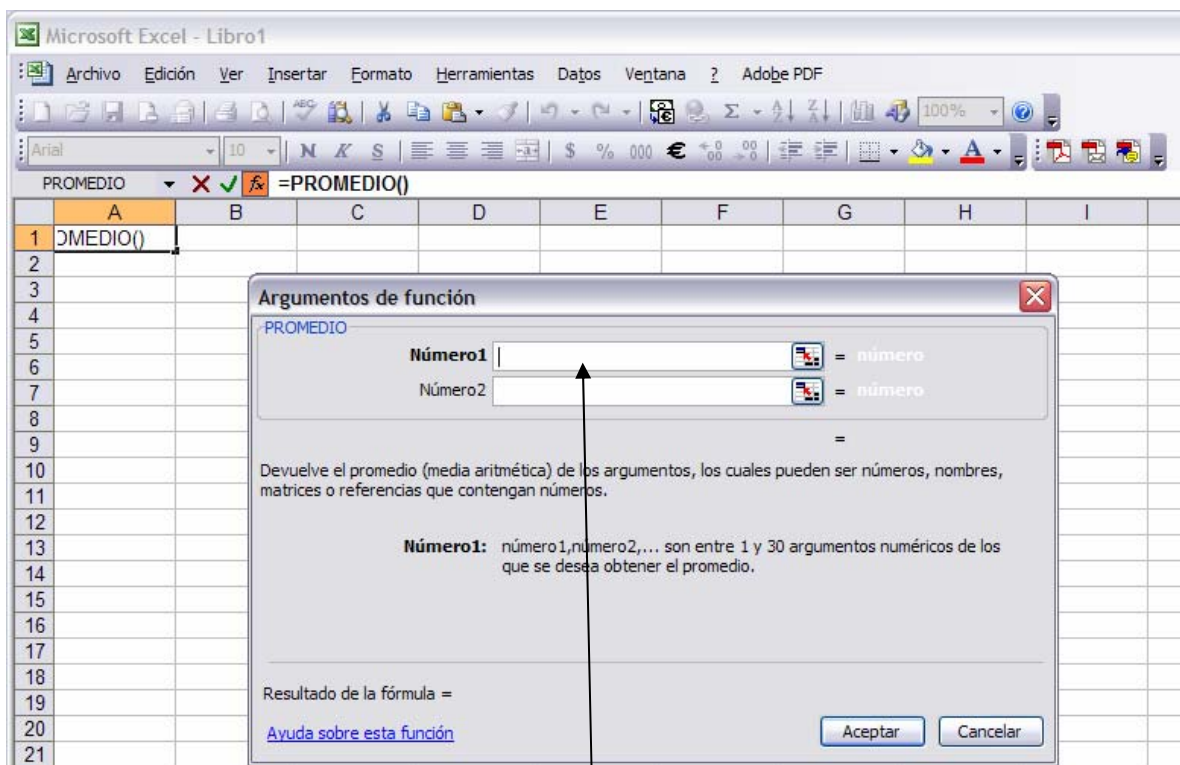
*Pulsa a la izquierda la opción **Estadísticas**.*



Sube la lista deslizable de la ventana derecha hasta encontrar la función **PROMEDIO** y pulsa un clic sobre ella.



Observa la línea de estado de la ventana; nos explica para qué sirve esa función. Acepta.



Ahora nos pide qué celdas o rango de celdas queremos utilizar para saber el resultado del promedio de datos. Podemos pulsar clic en las celdas que nos interesen, escribirlas a mano o bien selecciona un rango de datos de la hoja. Selecciona el rango adecuado y acepta.

## Otros ejemplos de funciones: MAX, MIN

Haz una sencilla hoja de cálculo como la que sigue:

	A	B	C	D
1	Artículos en almacén			
2				
3	<b>Código</b>	<b>Cantidad</b>	<b>Precio unitario</b>	<b>Precio total</b>
4	A1	23	570	
5	A2	32	875	
6	A3	11	450	
7	A4	45	235	
8	A5	23	450	
9	A6	22	590	

Sitúa el cursor en **D4** y escribe la fórmula: **=B4\*C4**. Cópiala hacia abajo.

Escribe al lado de la hoja las nuevas celdas de texto:

	A	B	C	D	E	F
1	Artículos en almacén					
2						
3	<b>Código</b>	<b>Cantidad</b>	<b>Precio unitario</b>	<b>Precio total</b>	Valoración del almacén	
4	A1	23	570	13110	Promedio general	
5	A2	32	875	28000	Cantidad máxima	
6	A3	11	450	4950	Cantidad mínima	
7	A4	45	235	10575	Número de elementos	
8	A5	23	450	10350		
9	A6	22	590	12980		

Escribe las fórmulas de las celdas:

Celda	Fórmula
F3	=SUMA(D4:D9)
F4	=PROMEDIO(D4:D9)
F5	=MAX(D4:D9)
F6	=MIN(D4:D9)
F7	=CONTAR(D4:D9)

Selecciona el rango de los resultados y conviértelo en formato moneda.



Como habrás imaginado, hemos obtenido el valor máximo, mínimo y además hemos contado el número de elementos numéricos que aparecen en el rango D4:D9.

## Sugerir una función

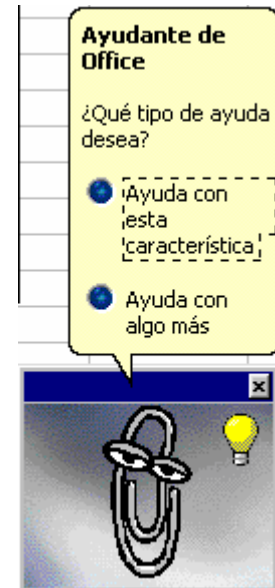
En ocasiones, podemos conocer el cálculo que queremos realizar, pero no si existe alguna función que Excel nos pueda aportar para obtener el resultado.

En este caso, podemos hacer que sea el propio Excel el que nos *sugiera* una función a utilizar. Cuando ocurra esto, podemos pulsar el botón **Pegar función** y seguidamente pulsar el botón de ayuda que aparece en la parte inferior del cuadro de diálogo. Aparecerá el asistente de Excel.

Pulsando el botón **Ayuda con esta característica** podemos escribir una descripción de lo que queremos hacer y posiblemente Excel nos ayude. Por ejemplo:

*Pulsa dicho botón y escribe en la casilla que aparece: **Desviación típica**.*

Excel nos muestra una lista de funciones recomendadas para obtener los resultados que buscamos.



## Funciones anidadas

Se llaman así aquellas funciones que actúan como argumento de otra función, es decir, que se encuentran dentro de otra función. En el proceso de cálculo, Excel realiza primero el cálculo de la función interior y después, el resultado de la función exterior teniendo ya en cuenta el resultado que se ha obtenido con la función interior.

Por ejemplo, la función:

**=RAIZ(POTENCIA(20;3))**

Primero calculará el resultado de la función interior, o sea, de la potencia, cuyo resultado es 8.000 y luego se calculará el resultado de la exterior, teniendo en cuenta ya este resultado.

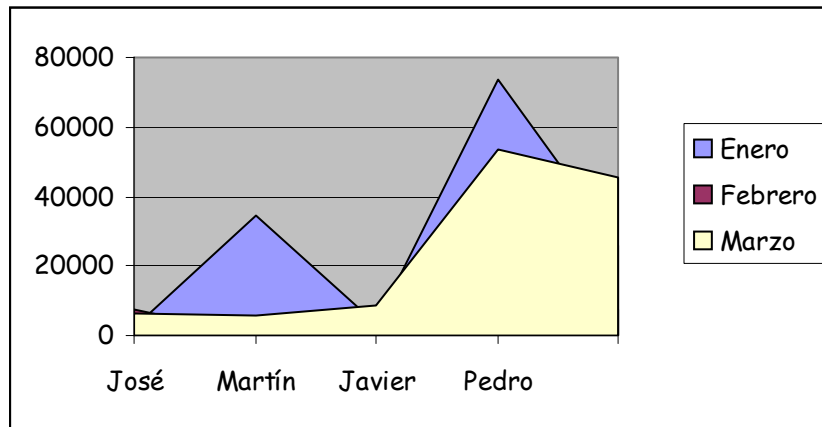
## UNIDAD 6

Excel ofrece la posibilidad de trabajar con gráficos en sus hojas de cálculo que nos proporcionarán una visión más clara y gráfica del contenido de las tablas de datos.

Un gráfico en Excel es la representación gráfica de un conjunto de datos de una hoja de cálculo. Podemos crear diferentes tipos de gráficos (barras, columnas, líneas, etc.) dependiendo de la información visual que queramos conseguir. Los datos utilizados en su creación, pueden variar y el gráfico se actualizará automáticamente.

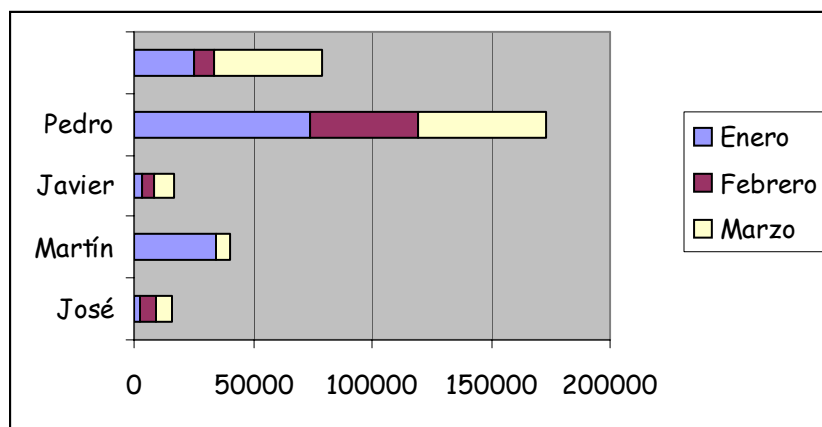
Éstas son las características de algunos de los gráficos más utilizados:

Gráfico de áreas



Representan la evolución de las series a lo largo del tiempo. Muestran el volumen de cada serie y el total acumulado de las mismas.

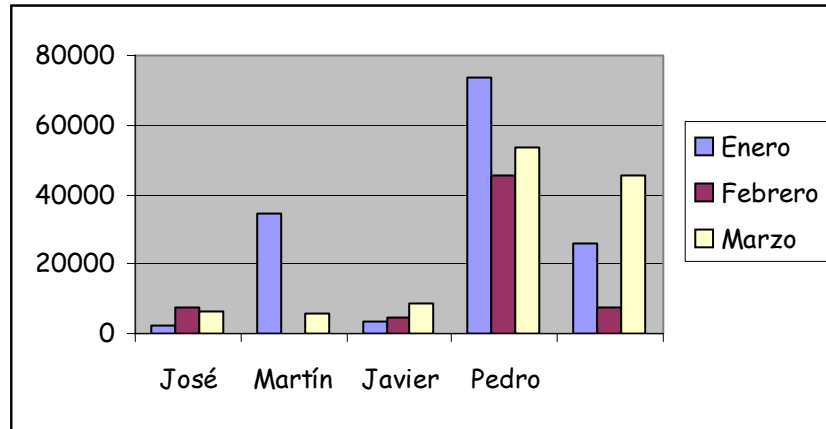
Gráfico de barras





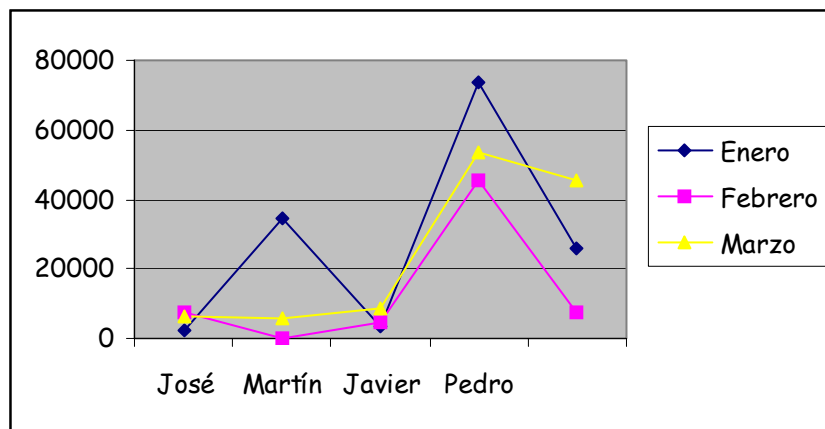
Comparan las series. El eje X se representa verticalmente y el eje Y horizontalmente. Las barras apiladas (ejemplo) representan la relación de cada punto con el total.

### Gráfico de columnas



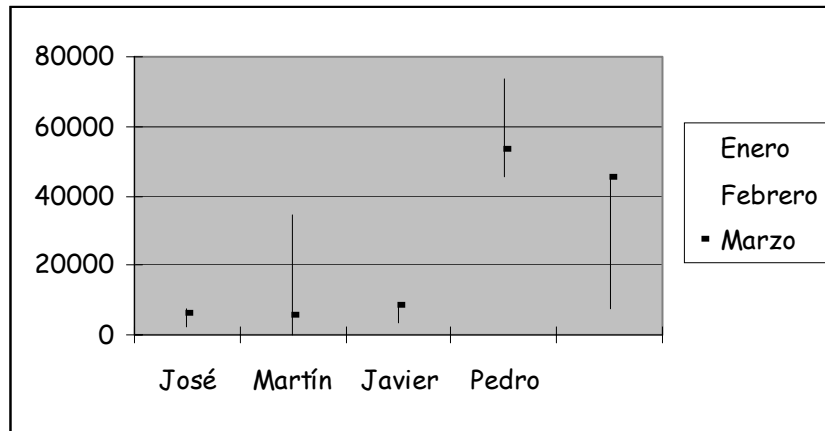
Representa las series en barras verticales y permite compararlas y analizar las diferencias de valores entre los puntos a través del tiempo. Es un gráfico ideal para observar los datos en un intervalo de tiempo dado.

### Gráfico de líneas



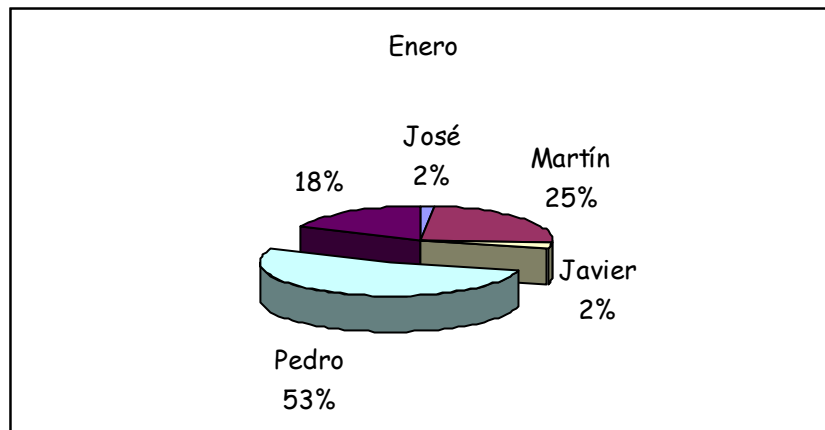
Estudia las tendencias de los valores a lo largo de un período de tiempo, resaltando la velocidad del cambio.

## Gráfico bursátil



Especial para representar datos bursátiles. Si se desean representar los valores bursátiles de **apertura**, **máximo**, **mínimo** y **cierre**, se tienen que seleccionar 4 filas o columnas de datos correspondientes a dichos valores. Es un gráfico ideal para estudiar las fluctuaciones.

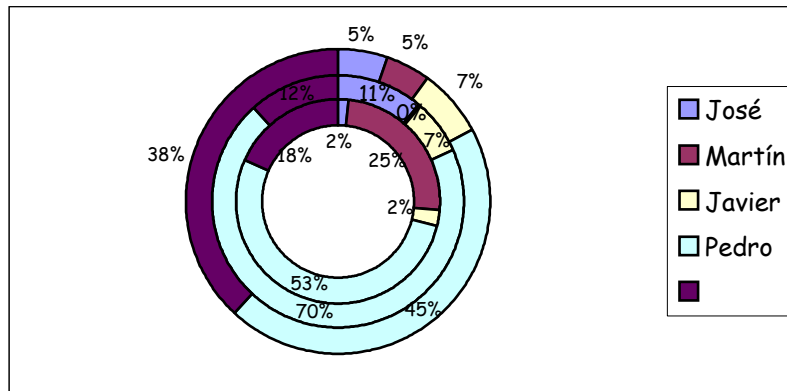
## Gráfico circular o de sectores



Representa una sola serie de datos que son analizados y cuyo valor se expresa en porcentaje. Se utilizan también para resaltar algún valor concreto.

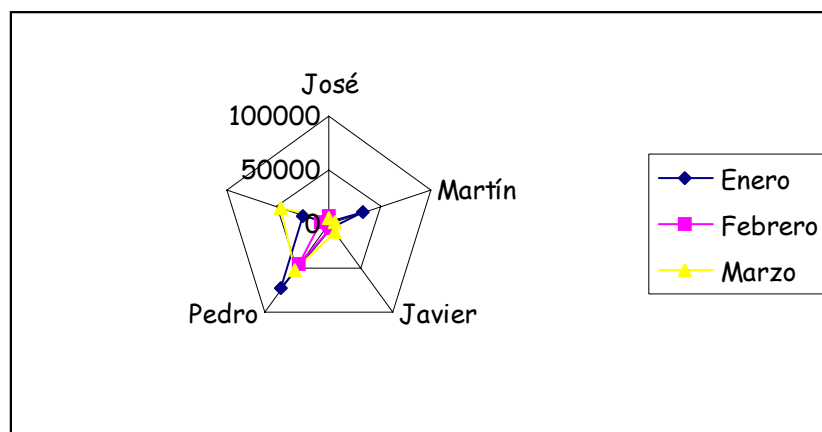


## Gráfico de anillos



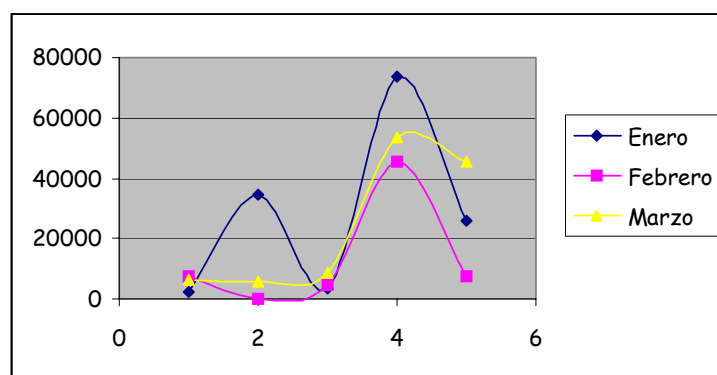
Similar al de sectores, no se limita a una sola serie, sino que puede representar tantas como deseemos. Las series son los anillos y los colores representan cada categoría.

## Gráfico de radar



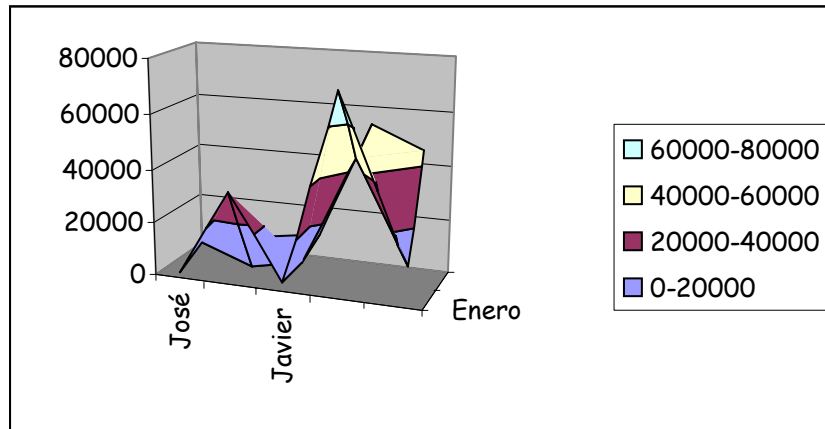
Cada categoría forma un eje y cada eje sale del punto central. Si existen varias series, todos sus puntos se unirán con una línea. No se podrán intercambiar los valores del orden una vez creado.

## Gráfico XY (gráfico de dispersión)



Trabajan con dos ejes de valores. Se selecciona la primera columna del rango para los valores de eje X y la segunda para los del eje Y. Se usan para analizar tendencias de los valores a través del tiempo, y sus posibles relaciones entre series.

### Gráfico de superficie (3-D)



Nos sirve para trabajar con grandes cantidades de datos y su combinación. Inicia mediante colores, las zonas con valores más parecidos.

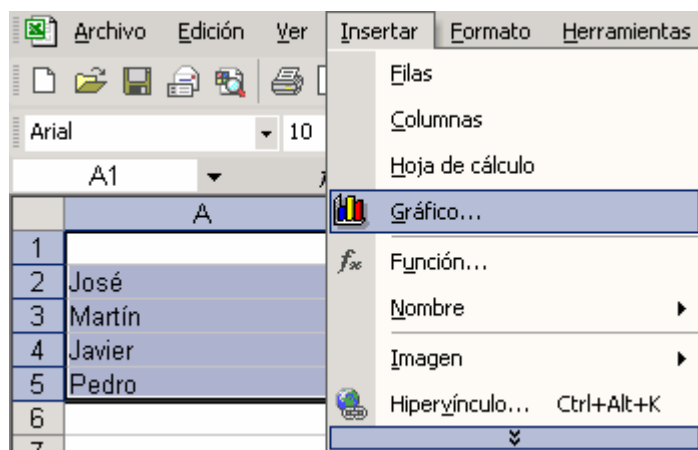
## Crear y modificar un gráfico

*Crea una hoja como la que sigue:*

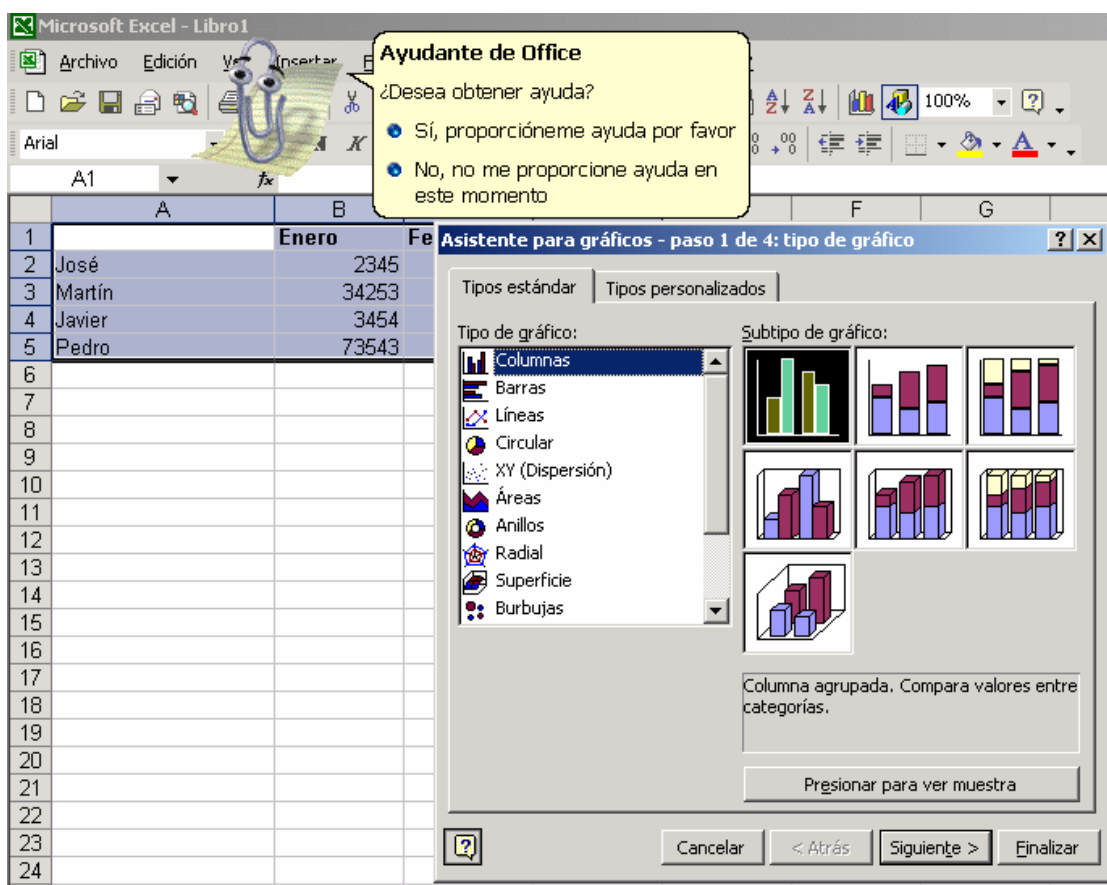
	A	B	C	D
1	.	<b>Enero</b>	<b>Febrero</b>	<b>Marzo</b>
2	José	2345	7254	6244
3	Martín	34253	246	5724
4	Javier	3454	4562	8654
5	Pedro	73543	45734	53624
6	.	25624	7635	45734

*Selecciona el rango **A1:D6***

*Accede a **Insertar – Gráfico***

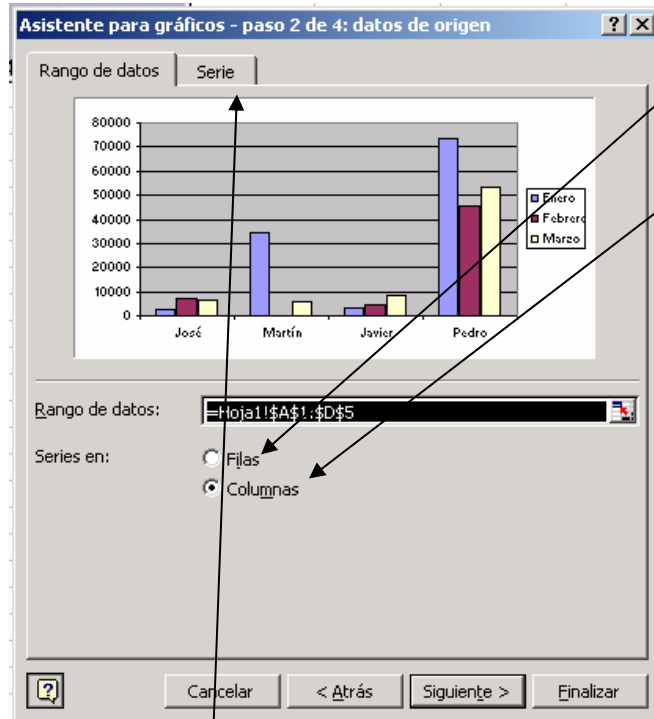


Aparece un asistente para la creación del gráfico. En este primer paso podemos elegir el tipo de gráfico que queremos.



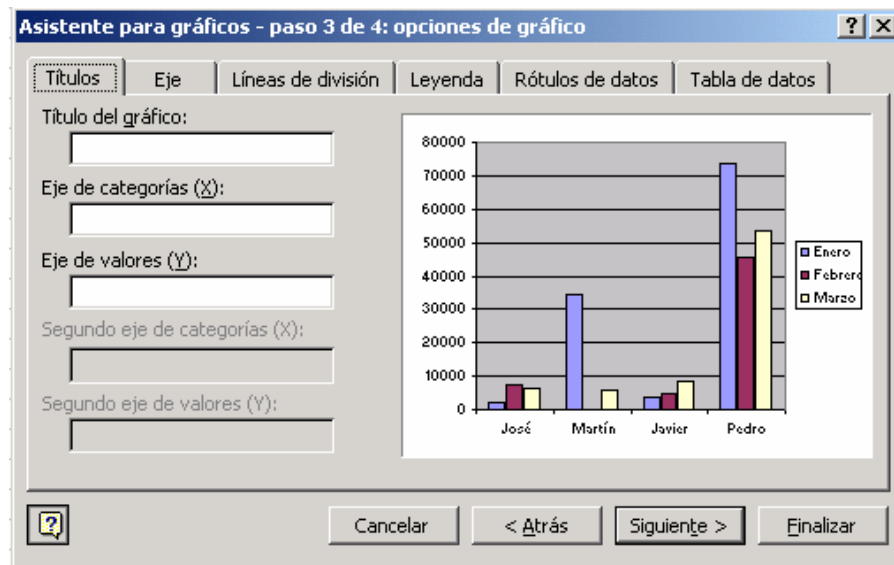
*Pulsa el botón **Siguiente***

Aquí se nos muestra el rango de datos que hemos seleccionado previamente. Podemos cambiarlo o dejar el que ya está seleccionado. Si pulsas en las casillas **Filas** y **Columnas** los datos del gráfico se transponen para mostrar en el eje de las X los rótulos de la primera fila o la primera columna. Deja la opción **Columnas** activada.



La pestaña superior **Serie** nos muestra las series que están seleccionadas en este momento y que corresponden a los meses. Las series nos muestran los colores correspondientes a cada mes porque hemos seleccionado tres columnas. La representación de las series se llama **Leyenda**.

*Pulsa el botón **Siguiente**.*





Aquí podemos modificar varias opciones como títulos, leyenda, etc.

*Escribe en la casilla **Título del gráfico** el texto: **VENTAS TOTALES**. Si esperas unos segundos, aparecerá la simulación en la ventana de la derecha.*

*Escribe como título del eje de las X el texto: **AGENTES**.*

*Escribe como título del eje de las Y el texto: **Ventas en miles**.*

La pestaña superior **Ejes** nos permite activar y desactivar la visualización de los ejes. Puedes activar o desactivar las distintas opciones para comprobar el resultado en la ventana de simulación.

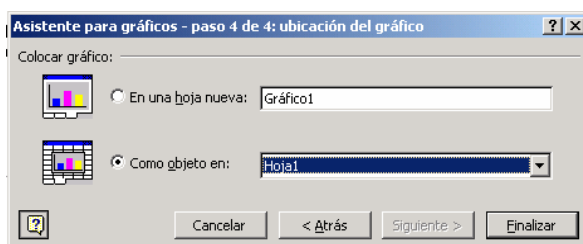
La pestaña **Líneas de división** permite activar o desactivar las líneas horizontales o verticales de división. Prueba también a activar o desactivar las distintas opciones.

La pestaña **Leyenda** permite activar, desactivar y modificar la posición de la leyenda.

La pestaña **Rótulos de datos** permite varios modelos de visualización de los rótulos de datos.

La pestaña **Tabla de datos** si está activada, nos muestra en miniatura la tabla origen de los datos del gráfico.

*Pulsa el botón **Siguiente**.*



Finalmente podemos optar por crear el gráfico en la misma hoja, el cual se podrá modificar como si de un objeto cualquiera se tratara (mover, cambiar el tamaño, modificar el contenido...) o bien crear el gráfico en una hoja completamente nueva, lo cual añadiría una hoja sólo para mostrar el gráfico.

*Deja la opción **Como objeto en..** seleccionada y pulsa el botón **Terminar**.*



El gráfico aparece en la misma hoja de trabajo. Ahora podemos estirar su tamaño desde uno de los nodos de control, moverlo arrastrando desde el interior del gráfico, etc.

Si pulsamos un clic fuera del gráfico en cualquier parte de la pantalla de trabajo, se observa que la marca negra de selección desaparece. Si volvemos a pulsar un clic en el interior del gráfico, vuelve a aparecer.

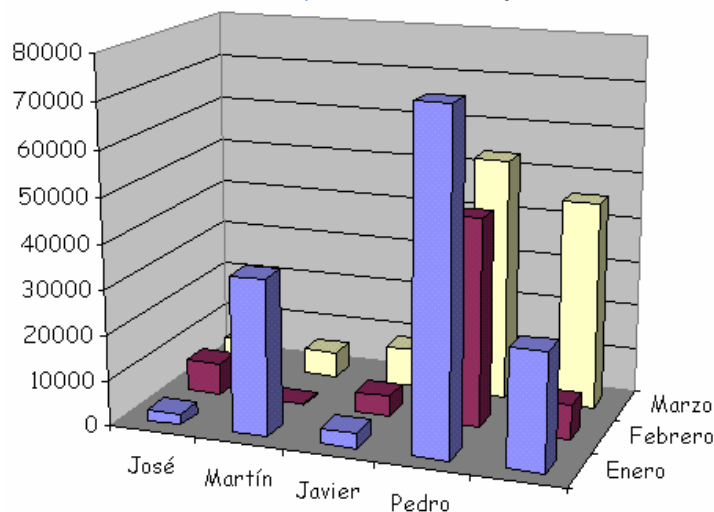
Para modificar cualquier parte del gráfico podemos pulsar doble clic. Por ejemplo, si pulsamos doble clic sobre el fondo gris del gráfico, aparecerá un cuadro de diálogo desde el cual podemos cambiar los colores del mismo.

Puedes también pulsar un clic sobre alguno de los tres títulos que hemos colocado (título principal, eje X y eje Y) y observarás que puedes modificar dicho título.

Si el gráfico está seleccionado (marcado en negro) puedes abrir algún menú y observarás que algunas opciones han cambiado. Éstas afectan al gráfico.

Si se desea borrar un gráfico, sólo hay que tenerlo seleccionado y pulsar la tecla Supr del teclado. A veces, en vez de modificar los datos de un gráfico es mejor y más rápido crearlo de nuevo.

*Borra el gráfico actual y crea el siguiente:*







Observa que hemos desactivado la opción **Leyenda**. Al ser un gráfico tridimensional, se crea un nuevo eje llamado **Eje Z** que muestra las series en dicho eje, por lo que la leyenda no es necesaria.

*Añada a la hoja la siguiente columna, con fórmula incluida:*

	A	B	C	D	E
1		Enero	Febrero	Marzo	TOTALES
2	José	2345	7254	6244	15843
3	Martín	34253	246	5724	40223
4	Javier	3454	4562	8654	16670
5	Pedro	73543	45734	53624	172901

Bien. Ahora vamos a crear un gráfico con los nombres de los agentes y los totales en forma de columnas. No es necesario en este caso seleccionar todo el rango de datos porque sólo nos interesa la última columna como datos de la serie.

Selecciona la primera columna.

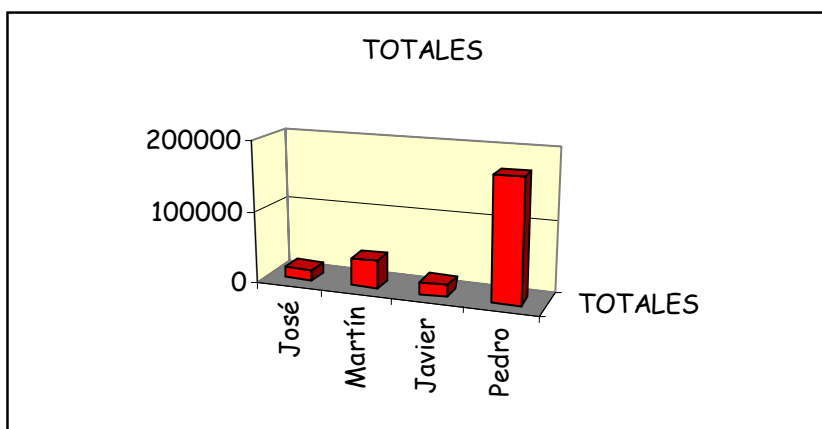
	A	B	C	D	E
1		Enero	Febrero	Marzo	TOTALES
2	José	2345	7254	6244	15843
3	Martín	34253	246	5724	40223
4	Javier	3454	4562	8654	16670
5	Pedro	73543	45734	53624	172901

*Manteniendo la tecla **Control** pulsada y sin soltarla, selecciona la columna de los totales.*

	A	B	C	D	E
1		Enero	Febrero	Marzo	TOTALES
2	José	2345	7254	6244	15843
3	Martín	34253	246	5724	40223
4	Javier	3454	4562	8654	16670
5	Pedro	73543	45734	53624	172901

*Selecciona el modelo **Columnas 3D** y sigue los pasos del asistente hasta la finalización de la creación del gráfico. Recuerda desactivar en este caso la **Leyenda**.*

*Cambia los colores del fondo y de las barras*



*pulsando doble clic sobre ellos.*

*Con el gráfico seleccionado, abre el menú **Gráfico***

Las opciones de este menú son las siguientes:

**Tipo de gráfico:** podemos cambiar el estilo del gráfico.

**Datos de origen:** para cambiar el rango del origen de los datos.

**Opciones de gráfico:** podemos modificar los títulos, leyenda, etc.

**Ubicación:** para crearlo en una hoja nueva o dejarlo en la actual.

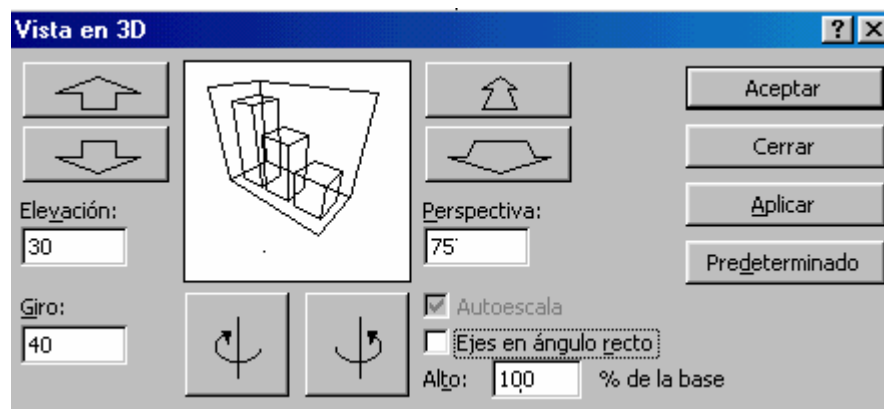
**Agregar datos/línea de tendencia:** para añadir nuevos datos pertenecientes a nuevas columnas o filas.

**Vista 3D:** permite cambiar la visualización en tres dimensiones.

*Accede a **Vista 3D***

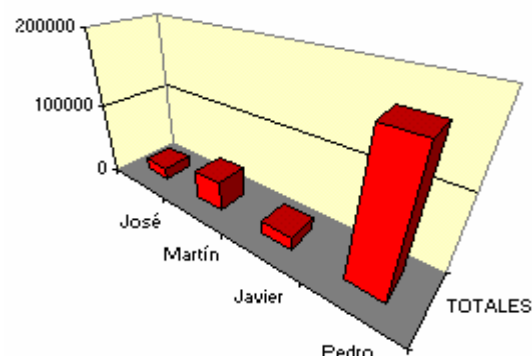
Ahora podemos cambiar la elevación del gráfico, la rotación, perspectiva, etc.

*Prepáralo como ves en la imagen y acepta.*



TOTALES

Dependiendo de los cambios introducidos en el cuadro de diálogo, tendremos que modificar el tamaño del gráfico porque quizás no se vea demasiado bien.





## UNIDAD 7

El concepto de **Matriz** viene de los lenguajes de programación y de la necesidad de trabajar con varios elementos de forma rápida y cómoda. Podríamos decir que una matriz es una serie de elementos formando filas (matriz bi-dimensional) o filas y columnas (matriz tri-dimensional).

La siguiente tabla representa una matriz unidimensional (vector):

1	2	3	4	5
---	---	---	---	---

ahora una matriz bidimensional:

1,1	1,2	1,3	1,4	1,5
2,1	2,2	2,3	2,4	2,5
3,1	3,2	3,3	3,4	3,4

Observa por ejemplo el nombre del elemento **3,4** que significa que está en la posición de fila 3, columna 4. En Excel, podemos tener un grupo de celdas en forma de matriz y aplicar una fórmula determinada en ellas de forma que tendremos un ahorro del tiempo de escritura de fórmulas.

En Excel, las fórmulas que hacen referencia a matrices se encierran entre corchetes {}. Hay que tener en cuenta al trabajar con matrices lo siguiente:

- No se puede cambiar el contenido de las celdas que componen la matriz
- No se puede eliminar o mover celdas que componen la matriz
- No se puede insertar nuevas celdas en el rango que compone la matriz

*1. Crea la siguiente hoja:*

	A	B	C	D	E
1					
2		Art.1	Art.2	Art.3	Art.4
3	Unidades	12	15	17	13
4	Precio	45	69	45	33
5	Total Unidad	540	1035	765	429
6					
7	TOTAL	2769			

Si te sitúas en la celda **B4**, observarás que hemos hecho una simple multiplicación para calcular el precio total de las unidades. Lo mismo pasa con las demás fórmulas.

En vez de esto, podríamos haber combinado todos los cálculos posibles en uno solo utilizando una fórmula matricial.

Una fórmula matricial se tiene que aceptar utilizando la combinación de teclas **CTRL+MYSC+Intro** y Excel colocará los corchetes automáticamente.

*Borra las celdas adecuadas para que quede la hoja de la siguiente forma:*

	A	B	C	D	E
1					
2		<b>Art.1</b>	<b>Art.2</b>	<b>Art.3</b>	<b>Art.4</b>
3	Unidades	12	15	17	13
4	Precio	45	69	45	33
5	Total Unidad				
6					
7	<b>TOTAL</b>				

*Sitúa el cursor en la celda **B7** e introduce la fórmula:*

**=SUMA(B3:E3\*B4:E4)**

*Acepta la fórmula usando la combinación de teclas adecuadas.*

Observa cómo hemos obtenido el mismo resultado tan sólo con introducir una fórmula.

	A	B	C	D	E
1					
2		<b>Art.1</b>	<b>Art.2</b>	<b>Art.3</b>	<b>Art.4</b>
3	Unidades	12	15	17	13
4	Precio	45	69	45	33
5	Total Unidad				
6					
7	<b>TOTAL</b>	<b>2769</b>			

Observa la misma en la barra de fórmulas. Ahora hay que tener cuidado en editar celdas que pertenezcan a una matriz, ya que no se pueden efectuar operaciones que afecten sólo a un rango de datos. Cuando editamos una matriz, editamos todo el rango como si de una sola celda se tratase.

## Constantes matriciales

Al igual que en las fórmulas normales podemos incluir referencias a datos fijos o constantes, en las fórmulas matriciales también podemos incluir datos constantes. A estos datos se les llama **constantas matriciales** y se debe incluir un separador de columnas (símbolo ;) y un separador de filas (símbolo \).

Por ejemplo, para incluir una matriz como constante matricial:

30    25  
31    18

Debemos escribir: {30;25\31;18}

*Escribe estas celdas en la hoja2*

	A	B	C	D
1	5	5		
2	5	5		
3				
4	1	2		

*Selecciona el rango C1:D2*

*Escribe la fórmula: =A1:B2\*{10;20\30;40}*

*Acepta la fórmula con la combinación de teclas adecuada.*

Observa que Excel ha ido multiplicando los valores de la matriz por los números introducidos en la fórmula:

C1			=	{=A1:B2*{10;20\30;40}}	
	A	B	C	D	
1	5	5	50	100	
2	5	5	150	200	
3					
4	1	2			

Cuando trabajamos por fórmulas matriciales, cada uno de los elementos de la misma, debe tener idéntico número de filas y columnas, porque de lo contrario, Excel expandiría las fórmulas matriciales. Por ejemplo:

= {1;2;3}\*{2\3} se convertiría en = {1;2;3\1;2;3}\*{2;2;2\3;3;3}

Selecciona el rango **C4:E5**

Introduce la fórmula: **=A4:B4+{2;5;0\3;9;5}** y acéptala.

C4			=	{=A4:B4+{2;5;0\3;9;5}}	
	A	B	C	D	E
1	5	5	50	100	
2	5	5	150	200	
3					
4	1	2	3	7	#N/A
5			4	11	#N/A

Observemos que Excel devuelve un mensaje de error diciendo que el rango seleccionado es diferente al de la matriz original.

Graba si lo deseas la hoja.

## Vínculos y referencias en Excel

Excel permite utilizar en sus fórmulas referencias a otras celdas, hojas o incluso libros de trabajo. A veces es más práctico dividir el trabajo en pequeños libros y posteriormente unirlos en uno. Imagínate una empresa con tres sucursales, las cuales llevan por separado una serie de hojas. En un momento dado, interesaría unir las todas en una sola hoja a modo de resumen.

Excel permite varios tipos de referencias en sus fórmulas:

- **Referencias externas:** cualquier referencia a celdas y rangos de otros libros de trabajo.
- **Libro independiente:** un libro que contiene vínculos con otros libros, y por lo tanto **depende** de los datos de los otros libros.
- **Libro de trabajo fuente:** libro que contiene los datos a los que hace referencia una fórmula de un libro dependiente a través de una referencia externa.

Por ejemplo, la referencia:

**'C:\Mis documentos\[Ventas.xls]Enero'!A12**

Haría referencia a la celda **A12** de la hoja **Enero** del libro **Ventas.xls** que está guardado en la carpeta **Mis documentos** de la unidad **C:**

*Crea en un libro nuevo la siguiente hoja:*

	A	B	C	D
1	<b>Empresa 1</b>			
2				
3		<b>Enero</b>	<b>Febrero</b>	<b>Marzo</b>
4	Gastos	150000	175000	180000
5	Compras	300000	180000	135000
6	Nóminas	450000	450000	450000

*Guarda el libro con el nombre: **Empresa1***

*Cierra el libro de trabajo.*

*En un nuevo libro de trabajo, crea la siguiente hoja:*

	A	B
1	<b>Resumen empresa 1</b>	
2		
3		<b>TOTAL</b>
4	<b>Enero</b>	
5	<b>Febrero</b>	
6	<b>Marzo</b>	

*Sitúate en la celda **B4**.*

*Escribe la fórmula: (suponiendo que la tengas guardada en la carpeta **Mis documentos**:*

**= 'C:\Mis documentos\[empresa1.xls]Hoja1'!B4:D4**

*Cópiala dos celdas hacia abajo.*

*Graba el libro con el nombre: **empresa2.xls***

*Abre el libro **empresa1.xls***

*Accede a **Ventana – Organizar** y acepta la opción **Mosaico**.*

Ahora tenemos dos ventanas correspondientes a los dos libros de trabajo abiertos. Para pasar de una a otra, debemos activarla con un clic en su título o en cualquier parte de la misma. Por ejemplo, si deseamos situar el cursor en la ventana inactiva, primero debemos pulsar un clic para activarla y después otro clic para situar ya el cursor.

*Sitúa el cursor en la celda **B4** del libro **empresa2**.*





Observa la barra de fórmulas. Ahora no vemos el camino marcado que hace referencia a un archivo grabado en disco. Cuando tenemos abiertos los archivos, no se observa el camino de unidades y carpetas.

Si ahora modificamos cualquier dato del libro **empresa1**, se actualizarán las fórmulas del libro **empresa2**.

*Cierra los dos libros.*

## Auditoria de hojas

Esta sencilla opción sirve para saber a qué celdas hace referencia una fórmula determinada, posibles errores en fórmulas, etc.

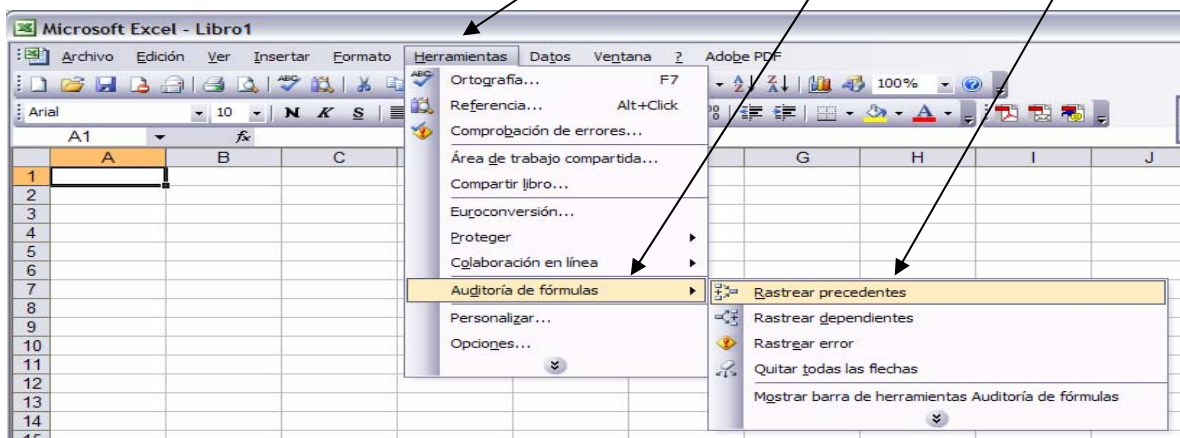
*Crea un libro nuevo.*

*Crea una sencilla hoja con sus fórmulas:*

	A	B	C	D	E
1					
2				<b>TOTAL</b>	<b>10% Desc</b>
3	Elemento1	100	100	200	20.00
4	Elemento2	200	200	400	40.00
5	Elemento3	300	300	600	60.00

*Sitúa el cursor en la celda D2*

*Selecciona de la barra de menús “Herramientas” - “Auditoría” - “Rastrear precedentes”*





	A	B	C	D	E
1					
2				<b>TOTAL</b>	<b>10% Desc</b>
3	Elemento1	100	100	200	20.00
4	Elemento2	200	200	400	40.00
5	Elemento3	300	300	600	60.00

Selecciona de la barra de menús “Herramientas” – “Auditoría” – “Rastrear dependientes”

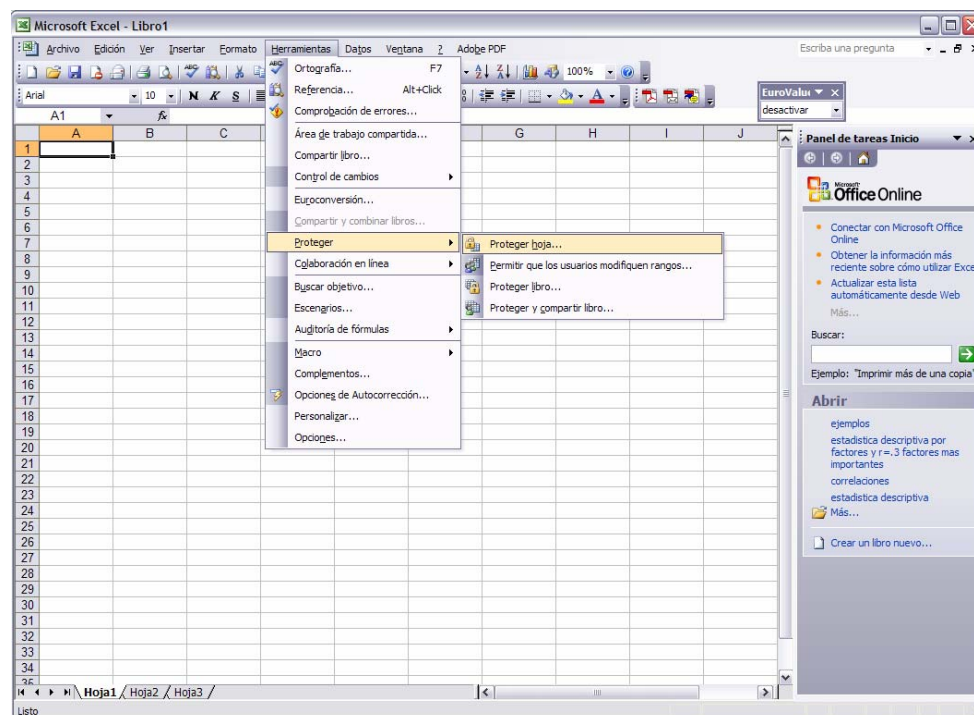
	A	B	C	D	E
1					
2				<b>TOTAL</b>	<b>10% Desc</b>
3	Elemento1	100	100	200	180.00
4	Elemento2	200	200	400	360.00
5	Elemento3	300	300	600	540.00

Excel nos muestra que la fórmula hace referencia al rango B2:C2 (precedentes) y que a su vez, otra celda, la E2, depende del resultado de la celda actual (dependientes).

A través de esta opción podemos localizar qué celdas dependen de otras en sus fórmulas, a qué celdas hace referencia la fórmula, etc., incluso podemos, en caso de error, localizar el mismo (opción Rastrear error).

Accede a **Herramientas – Auditoría – Quitar todas las flechas**

## Protección de hojas



La protección de hojas nos permite proteger contra borrados accidentales algunas celdas que consideremos importantes. Podemos proteger toda la hoja, el libro entero, o bien sólo algunas celdas.

*Accede a **Herramientas – Proteger – Proteger hoja** y acepta el cuadro de diálogo que aparece.*

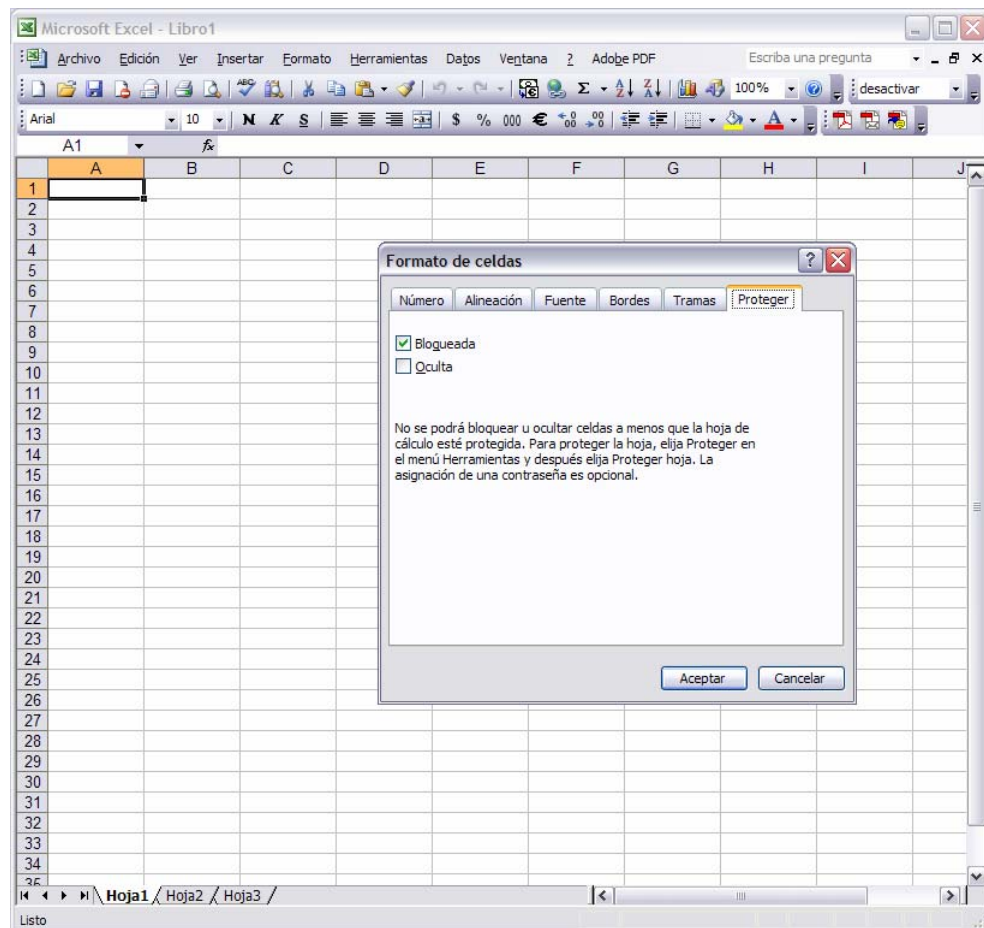
*Intenta borrar con la tecla **Supr** cualquier celda que contenga un dato.*

La hoja está protegida por completo. Imaginemos ahora que sólo deseamos proteger las celdas que contienen las fórmulas, dejando libres de protección el resto de celdas.

*Desprotege la hoja siguiendo el mismo método que antes.*

*Selecciona el rango **B2:C4** y accede a **Formato – Celdas – (Pestaña proteger)**.*

*A continuación aparecerá una ventana con varias opciones, en la cual escogemos la de **Proteger**, haciendo clic en la pestaña **Proteger**.*



*Desactiva la opción **Bloqueada** y acepta el cuadro.*

*Vuelve a proteger la hoja desde **Herramientas – Proteger – Proteger hoja**.*

*Cambia algún valor del rango **B2:C4***

*Intenta cambiar algo o borrar alguna celda del resto de la hoja.*

Con la opción anterior (Bloqueada), hemos preparado un rango de celdas para que esté libre de protección cuando decidamos proteger toda la hoja. De esta forma no habrá fallos de borrados accidentales en celdas importantes.

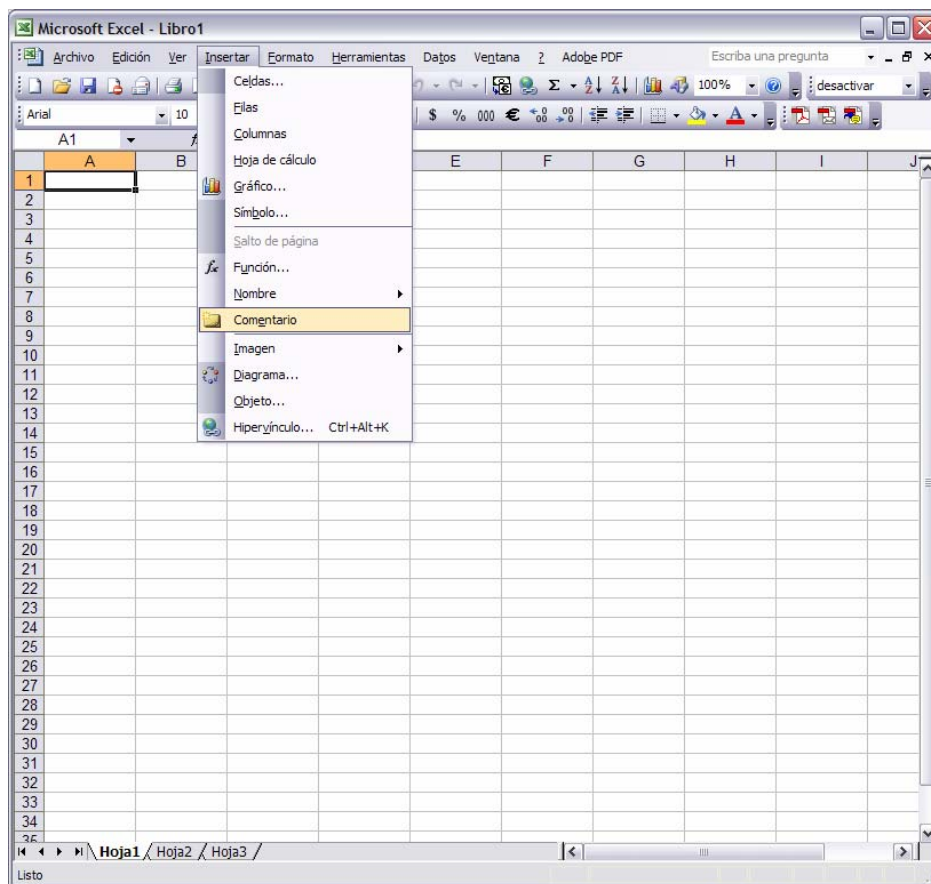
Si escribimos una contraseña al proteger la hoja, nos la pedirá en caso de querer desprotegerla posteriormente.

Si elegimos la opción **Proteger libro**, podemos proteger la estructura entera del libro (formatos, anchura de columnas, colores, etc...)

## Insertar comentarios

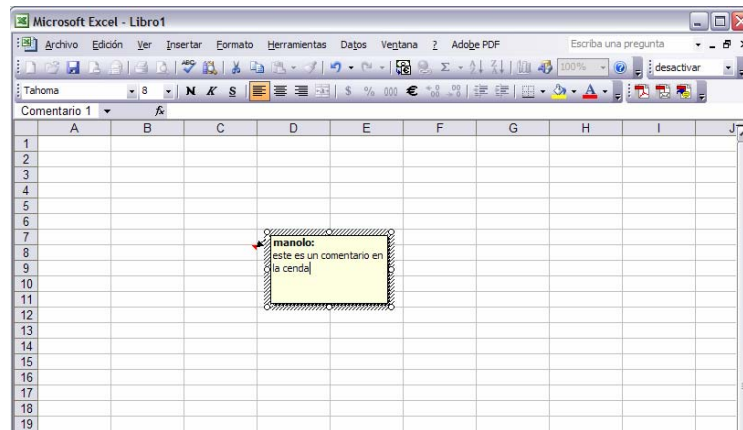
Es posible la inserción de comentarios en una celda a modo de anotación personal. Desde la opción **Insertar – Comentario** podemos crear una pequeña anotación.

*Sitúa el cursor en **E1** y accede a "Insertar" – "Comentario".*



*Escribe el siguiente texto:*

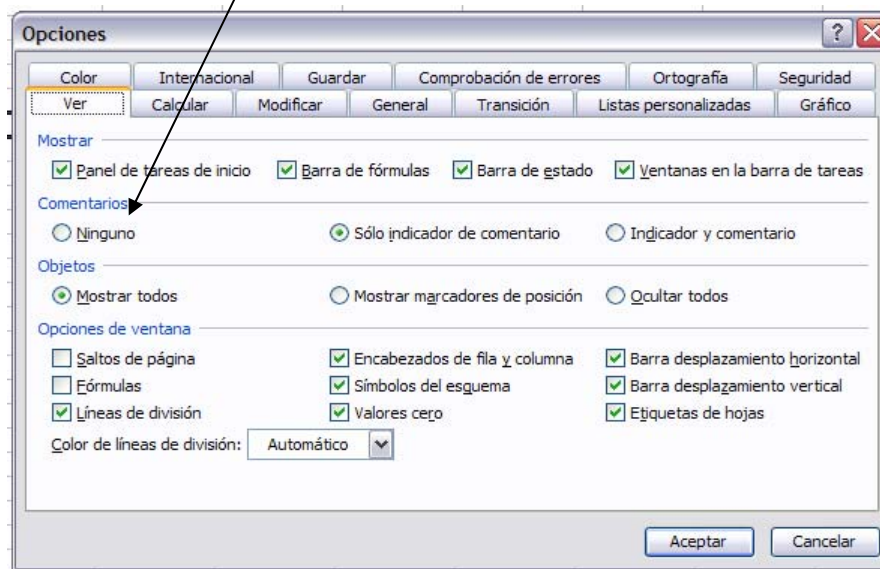
*Éste es un comentario en una celda*



*Pulsa clic fuera de la casilla amarilla.*

Dependiendo de qué opción esté activada en el menú **Herramientas – Opciones – Ver**, podemos desactivar la visualización de una marca roja, la nota amarilla, activar sólo la marca, o todo.

*Accede a “Herramientas” – “Opciones” y observa en la pestaña **Ver** (sección **Comentarios**) las distintas casillas de opción. Prueba a activar las tres saliendo del cuadro de diálogo y observa el resultado.*



*Finalmente, deja la opción **Sólo indicador de comentario** activada.*

*Sitúa el cursor sobre la celda que contiene el comentario.*

*Pulsa el botón derecho del ratón sobre esa misma celda.*



Desde aquí o bien desde **Edición**, podemos modificar o eliminar el comentario.

## Subtotales

En listas de datos agrupados por un campo, es útil mostrar a veces no sólo el total general de una columna, sino también los sub-totales parciales de cada elemento común.

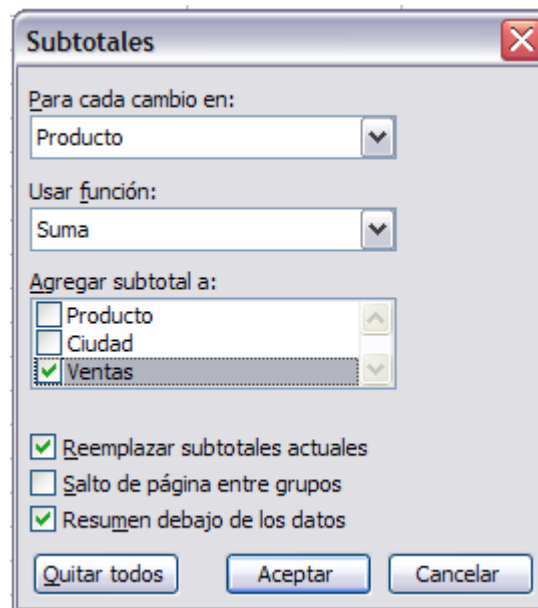
*Crea una hoja sencilla:*

	A	B	C
1	Producto	Ciudad	Ventas
2	Microkill	Durango	1,200,000
3	Espectrum	Monterrey	1,000,000
4	Amibac Solución	Gomez Palacio	800,000
5	Amibac Jabón	Zacatecas	1,500,000
6	Puribac	Aguascalientes	2,300,000
7			

*Ordénala por **Producto**.*

*Selecciona todo el rango de datos (A1:C6)*

*Accede a "Datos" – "Subtotales".*



Excel nos muestra por defecto una configuración para crear sub-totales agrupados por **Producto** (casilla **Para cada cambio en**), utilizando la función **SUMA** y añadiendo el resultado bajo la columna **Ventas**.

*Acepta el cuadro.*

Observa la agrupación que ha hecho Excel, calculando las ventas por marcas y obteniendo las sumas parciales de cada una de ellas.

	1	2	3		A	B	C
	1				<b>Producto</b>	<b>Ciudad</b>	<b>Ventas</b>
	2				Amibac Jabón	Zacatecas	1,500,000
	3				<b>Total Amibac Jabón</b>		1,500,000
	4				Amibac Solución	Gomez Palacio	800,000
	5				<b>Total Amibac Solución</b>		800,000
	6				Spectrum	Monterrey	1,000,000
	7				<b>Total Spectrum</b>		1,000,000
	8				Microkill	Durango	1,200,000
	9				<b>Total Microkill</b>		1,200,000
	10				Puribac	Aguascalientes	2,300,000
	11				<b>Total Puribac</b>		2,300,000
	12				<b>Total general</b>		6,800,000
	13						

En el margen izquierdo de la ventana se muestran unos controles para obtener mayor o menor nivel de resumen en los subtotales.

Pulsa los botones **1 2 3** y observa el resultado.

Vuelve a **"Datos" – "Subtotales"**.

Abre la lista de **"Usar función"** y elige la función **PROMEDIO**.

Desactiva la casilla **Reemplazar subtotales actuales** porque borraría los que ya hay escritos.

	1	2	3	4		A	B	C
	1					<b>Producto</b>	<b>Ciudad</b>	<b>Ventas</b>
	2					Amibac Jabón	Zacatecas	1,500,000
	3					<b>Total Amibac Jabón</b>		1,500,000
	4					<b>Promedio Amibac Jabón</b>		1,500,000
	5					Amibac Solución	Gomez Palacio	800,000
	6					<b>Total Amibac Solución</b>		800,000
	7					<b>Promedio Amibac Solución</b>		800,000
	8					Spectrum	Monterrey	1,000,000
	9					<b>Total Spectrum</b>		1,000,000
	10					<b>Promedio Spectrum</b>		1,000,000
	11					Microkill	Durango	1,200,000
	12					<b>Total Microkill</b>		1,200,000
	13					<b>Promedio Microkill</b>		1,200,000
	14					Puribac	Aguascalientes	2,300,000
	15					<b>Total Puribac</b>		2,300,000
	16					<b>Promedio Puribac</b>		2,300,000
	17					<b>Total general</b>		6,800,000
	18					<b>Promedio general</b>		1,360,000
	19							

Acepta.

Pulsa un clic uno a uno en los 4 botones **1 2 3 4** observa el resultado.

Accede a **"Datos" – "Subtotales"** y pulsa en **Quitar todos**.





Si se quisiera crear subtotales por otro campo (por ejemplo el campo **Ciudad**), deberíamos primero ordenar la lista por ese campo para que Excel pueda agrupar posteriormente la tabla.

## Tablas dinámicas

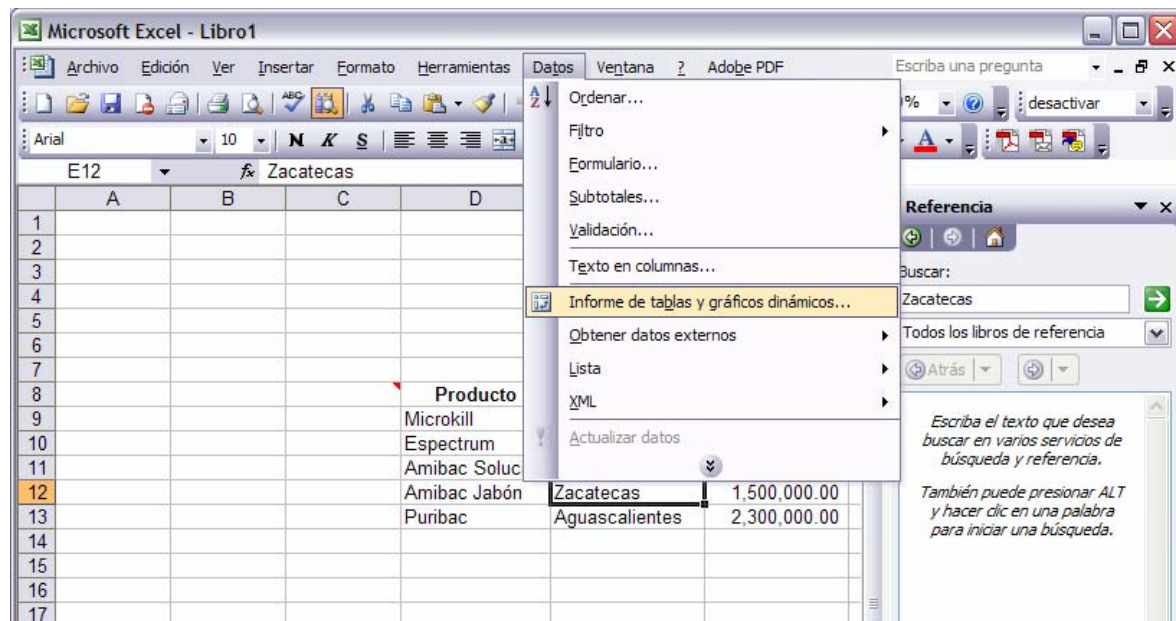
Una tabla dinámica nos permite modificar el aspecto de una lista de elementos de una forma más fácil, cómoda y resumida. Además, podemos modificar su aspecto y mover campos de lugar.

Para crear tablas dinámicas hemos de tener previamente una tabla de datos preparada y posteriormente acceder a **“Datos” – “Asistente para tablas dinámicas”**.

*Crea la siguiente tabla de datos:*

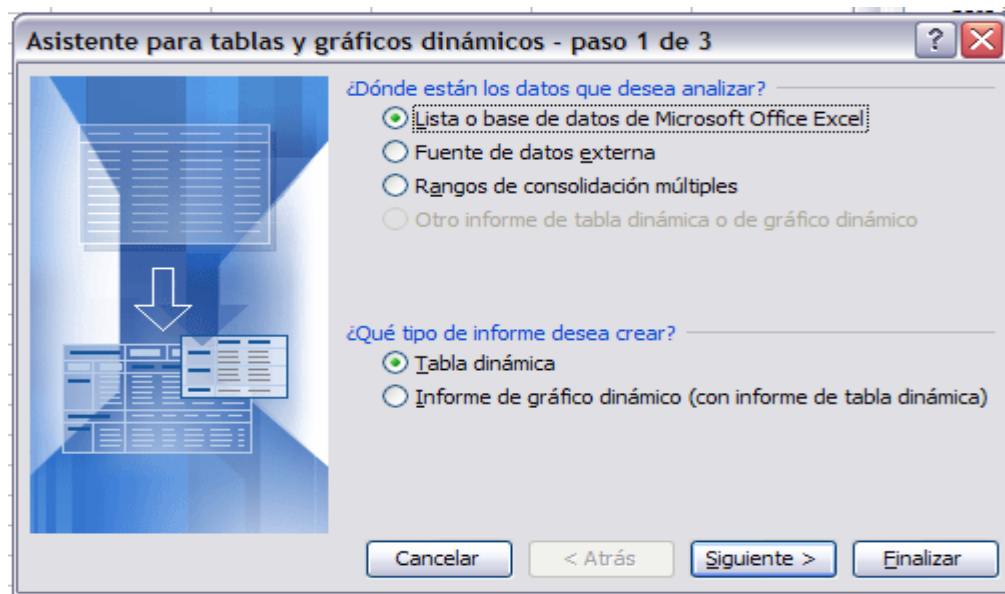
*Selecciona toda la tabla y accede a **“Datos” – “Informe de Datos y Tablas Dinámicas”**.*

	A	B	C
1	Producto	Mes	Precio
2	Producto1	Enero	1500
3	Producto2	Febrero	1450
4	Producto3	Marzo	1600
5	Producto4	Abril	1700
6	Producto5	Mayo	1400
7	Producto6	Junio	1350



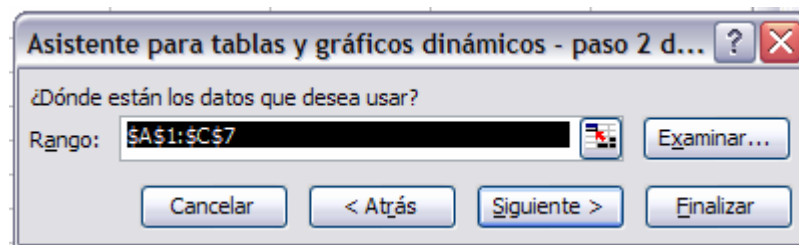
En primer lugar aparece una pantalla que representa el primer paso en el asistente para tablas dinámicas. Aceptaremos la tabla que hay en pantalla.



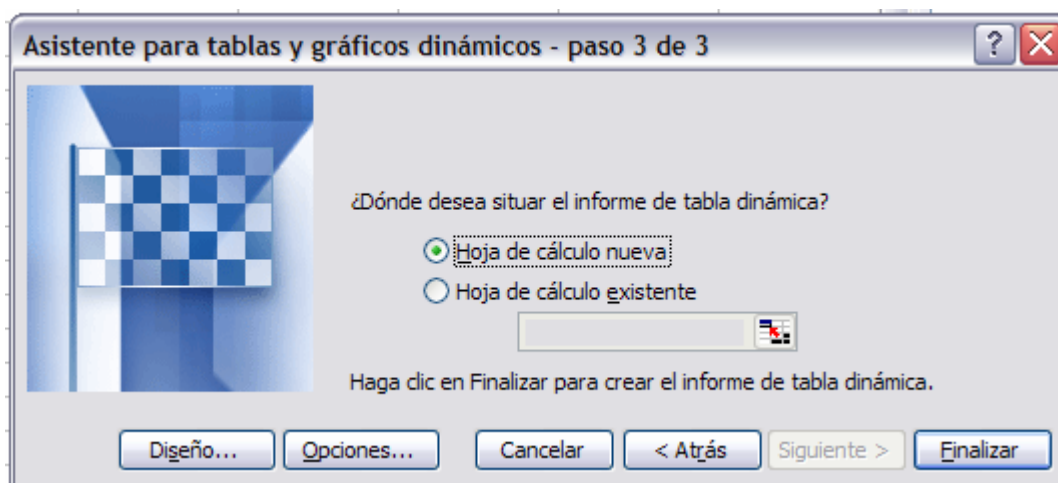


*Pulsa en **Siguiendo**.*

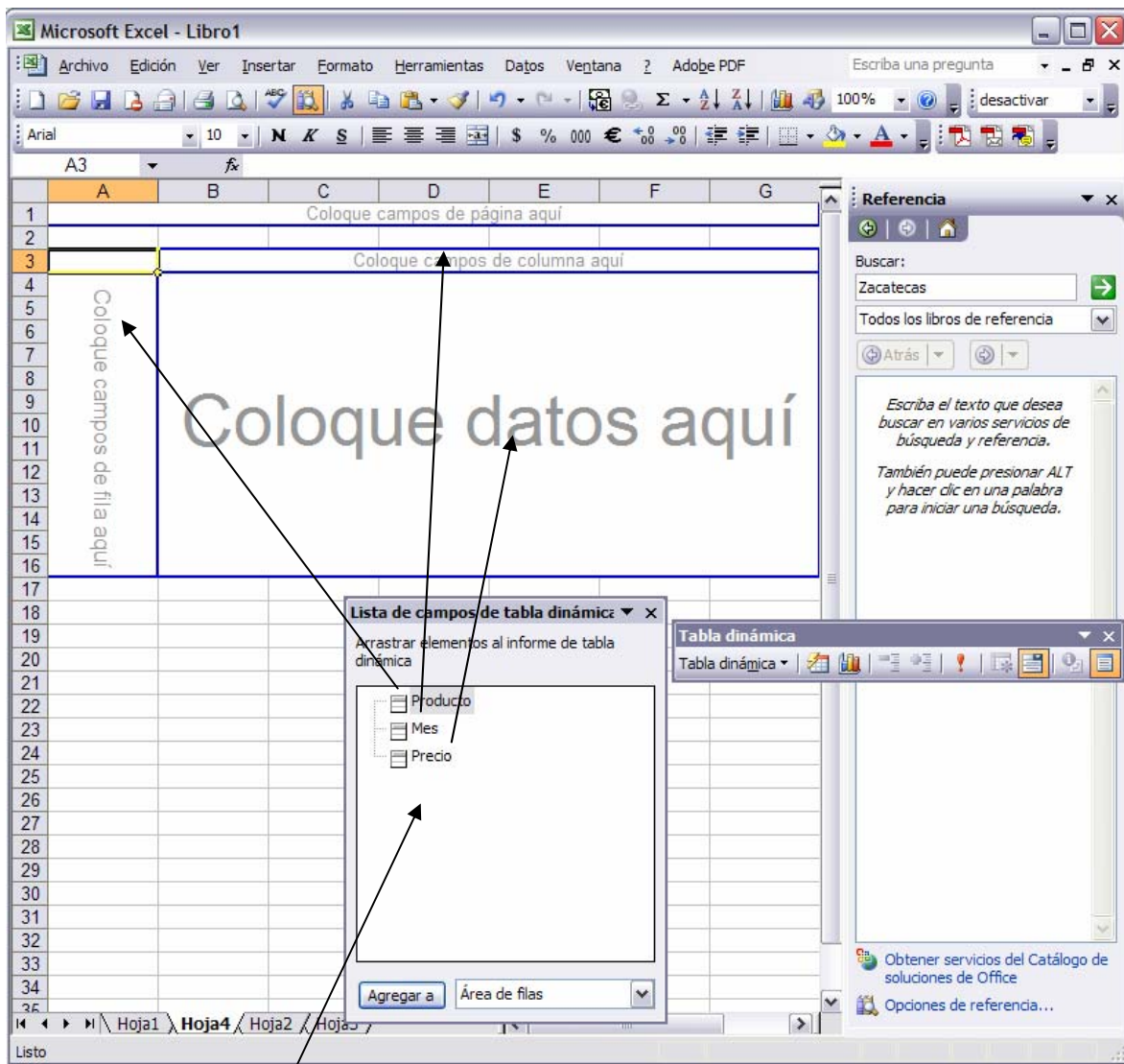
*Acepta el rango pulsando en **Siguiendo**.*



En el siguiente paso, Excel nos muestra la pantalla en donde nos pregunta en donde quiere colocar la tabla dinámica.



A continuación nos preguntara el diseño de la tabla y en donde queremos que se inserten los campos. Se mostrara una ventana como la siguiente:



Los campos del origen de los datos están situados en la parte derecha del cuadro de diálogo. Aquí veremos la estructura final que tendrá la tabla. Lo que hay que hacer es “arrastrar” los campos de la derecha hacia la posición deseada en el interior de la tabla.

*Arrastra los campos de la parte derecha según se ve en la ilustración:*

	Mes	COLUMNA
Producto		Suma de Precio
FILA		DATOS



A continuación vera la siguiente ventana en donde muestra la tabla dinámica terminada.

The screenshot shows the Microsoft Excel 2003 interface. The main window displays a PivotTable with the following data:

Suma de Precio	Mes	Enero	Febrero	Marzo	Abril	Mayo	Junio	Total general
Producto 1		1500						1500
Producto 2			1450					1450
Producto 3				1600				1600
Producto 4					1700			1700
Producto 5						1400		1400
Producto 6							1350	1350
Total general		1500	1450	1600	1700	1400	1350	9000

Overlaid on the spreadsheet are three task panes:

- Referencia**: A search pane with a text input field containing "Zacatecas" and a search button.
- Lista de campos de tabla dinámica**: A pane showing a list of fields: "Producto", "Mes", and "Precio". Below the list is a button labeled "Agregar a" and a dropdown menu set to "Área de filas".
- Tabla dinámica**: A small pane showing the name of the current PivotTable, "Tabla dinámica".

Observa el resultado de la tabla dinámica. Hemos diseñado la estructura para que nos muestre los productos en su parte izquierda, los meses en columnas, y además, el precio de cada producto en la intersección de la columna.

Observa también que se han calculado los totales por productos y por meses.

	A	B	C	D	E	F	G	H
1	Suma de Precio	Mes						
2	Producto	Enero	Febrero	Marzo	Abril	Mayo	Junio	Total general
3	Producto1	1500						1500
4	Producto2		1450					1450
5	Producto3			1600				1600
6	Producto4				1700			1700
7	Producto5					1400		1400
8	Producto6						1350	1350
9	Total general	1500	1450	1600	1700	1400	1350	9000

Si modificamos algún dato de la tabla original, podemos actualizar la tabla dinámica desde la opción “**Datos**” – “**Actualizar datos**” que se encuentra en la Barra de menús siempre que el cursor esté en el interior de la tabla dinámica.

Al actualizar una tabla, Excel compara los datos originales. Pero si se han añadido nuevas filas, tendremos que indicar el nuevo rango accediendo al paso 2 del Asistente. Esto podemos hacerlo accediendo nuevamente a “**Datos**” – “**Asistente para tablas dinámicas**” y volviendo atrás un paso.

Es posible que al terminar de diseñar la tabla dinámica nos interese ocultar algún subtotal calculado. Si es así, debemos pulsar **doble clic** en el campo gris que representa el nombre de algún campo, y en el cuadro de diálogo que aparece, elegir la opción **Ninguno**. Desde este mismo cuadro podemos también cambiar el tipo de cálculo.

Es posible también mover los campos de sitio simplemente arrastrando su botón gris hacia otra posición. Por ejemplo, puede ser que queramos ver la tabla con la disposición de los campos al revés, es decir, los productos en columnas y los meses en filas.

Si no está al crear la tabla, podemos activar la visualización de la barra de herramientas para tablas dinámicas (Ver- Barras de herramientas – Tablas dinámicas).



Desde aquí podemos realizar operaciones de actualización, selección de campos, ocultar, resumir, agrupar, etc. Puedes practicar sin miedo los diferentes botones de la barra.

## Búsqueda de objetivos

Hay veces en los que al trabajar con fórmulas, conocemos el resultado que se desea obtener, pero no las variables que necesita la fórmula para alcanzar dicho resultado. Por ejemplo, imaginemos que deseamos pedir un préstamo al banco de 2,000,000 de pesos y disponemos de dos años para pagarlo. Veamos cómo se calcula el pago mensual:

La función **=PAGO(interés/12;período\*12;capital)** nos da la cuota mensual a pagar según un capital, un interés y un período en años.

Escribe los siguientes datos:

B5		fx =PAGO(B2/12,B3*12,B1)	
	A	B	C
1	Capital	2,000,000	
2	Interés	4.50%	
3	Años	2	
4			
5	Pago Mensual	-\$87,295.62	
6			

Escribe en la celda **B5** la fórmula: **=PAGO(B2/12;B3\*12;B1)**.

Vemos que la cuota a pagar es de **87.296.62 pesos**.

La función **=PAGO()** siempre nos dará el resultado en números negativos. Si queremos convertirlo en resultado positivo, debemos encerrar la función en otra función: la función **=ABS()**. Esta función convierte cualquier número en positivo (valor absoluto).

Modifica la función y escribe: **=ABS(PAGO(B2/12;B3\*12;B1))**

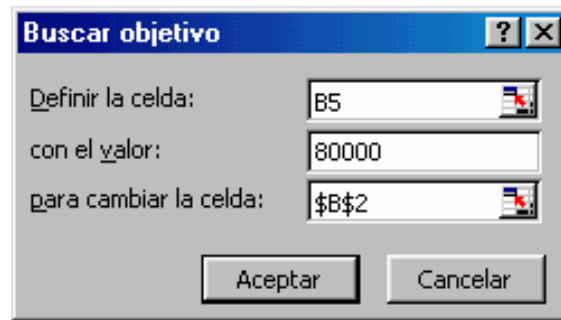
Ahora podemos variar los valores de las tres casillas superiores para comprobar diferentes resultados. Pero vamos a lo que vamos: imaginemos que sólo disponemos de 80,000 pesos para pagar cada mes. El banco actual nos ofrece un interés del 4,5%, así que vamos a ver qué interés tendríamos que conseguir para llegar a pagar las 80,000 que podemos pagar. Podríamos ir cambiando manualmente la celda del interés hasta conseguir el resultado requerido, pero a veces hay cálculos complejos y nos llevaría tiempo ir probando con decimales hasta conseguirlo.

Para ello, tenemos la opción **Buscar objetivos**, a través de la cual Excel nos proporcionará el resultado buscado.

Sitúa el cursor en **B5** si no lo está ya.

Accede a la opción de **"Herramientas" – "Buscar objetivos"** que se encuentra en la barra de menús.

Rellena las casillas como ves a continuación y acepta el cuadro.



Excel avisa que ha hallado una solución al problema.

*Acepta este último cuadro de diálogo.*

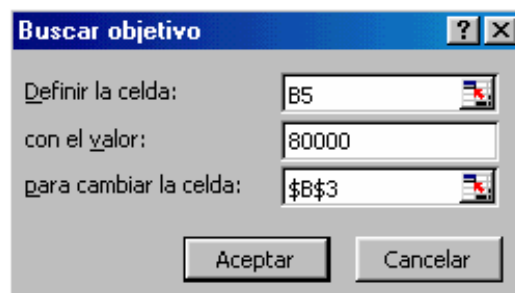
Sin embargo, si observas la celda del interés, aparece en negativo, por lo que el resultado no ha sido el esperado (evidentemente, el banco no nos va a pagar el interés a nosotros), por lo que nos vemos obligados a cambiar otra celda.

El capital no podemos cambiarlo. Necesitamos los 2.000.000, así que vamos a intentarlo con los años.

*Desahaz la última acción desde*



*Vuelve a preparar las siguientes casillas:*



*Acepta la solución de Excel.*

Observa que han aparecido decimales, pero ya sabemos que podemos cambiar el número de meses a pagar si es que no podemos tocar el interés. Quita los decimales. Necesitaremos dos años y dos meses.

Posiblemente otro banco nos ofrezca un interés más bajo, por lo que podemos volver a buscar un nuevo valor para el período.

Para trabajar con la opción de **Buscar objetivos**, hay que tener presente lo siguiente:

- Una celda cambiante (variable) debe tener un valor del que dependa la fórmula para la que se desea encontrar una solución específica.
- Una celda cambiante no puede contener una fórmula.
- Si el resultado esperado no es el deseado, debemos deshacer la acción.

## Tablas de datos de una y dos variables

Existe otro método para buscar valores deseados llamado **tablas de variables**. Existen dos tipos de tablas:

- **Tabla de una variable:** utilizada cuando se quiere comprobar cómo afecta un valor determinado a una o varias fórmulas.
- **Tabla de dos variables:** para comprobar cómo afectan dos valores a una fórmula.

A continuación modificaremos la tabla de amortización del préstamo de forma que Excel calcule varios intereses y varios años al mismo tiempo. Para crear una tabla hay que tener en cuenta:

- La celda que contiene la fórmula deberá ocupar el vértice superior izquierdo del rango que contendrá el resultado de los cálculos.
- Los diferentes valores de una de las variables deberán ser introducidos en una columna, y los valores de la otra variable en una fila, de forma que los valores queden a la derecha y debajo de la fórmula.
- El resultado obtenido es una matriz, y deberá ser tratada como tal (ver lección 8)

*Prepara la siguiente tabla. En ella, he puesto varios tipos de interés y varios años para ver distintos resultados de una sola vez.*

	A	B	C	D	E	F	G
1	Capital	2,000,000					
2	Interés	4.50%					
3	Años	2					
4							
5	Pago Mensual	\$87,295.62	1	2	3	4	5
6		4.50%					
7		4.25%					
8		4.00%					
9		3.75%					
10							

*Selecciona el rango **B5:F9** y accede, selecciona de la barra de menús la opción “Datos” – “Tabla”.*





Rellena las casillas como ves a continuación y acepta.

**Tabla** [?] [X]

Celda de entrada (fila):  [icon]

Celda de entrada (columna):  [icon]

Debes seleccionar el rango **C6:F9** y arreglarlo de forma que no se vean decimales, formato millares y ajustar el ancho de las columnas.

	A	B	C	D	E	F
1	Capital	2,000,000				
2	Interés	4.50%				
3	Años	2				
4						
5	Pago Mensual	\$87,295.62	1	2	3	4
6		4.50%	170,757.04	87,295.62	59,493.85	45,606.97
7		4.25%	170,528.34	87,072.56	59,270.65	45,382.20
8		4.00%	170,299.81	86,849.84	59,047.97	45,158.11
9		3.75%	170,071.45	86,627.47	58,825.81	44,934.71
10						

De esta forma podemos comprobar de una sola vez varios años y varios tipos de interés.

## Escenarios

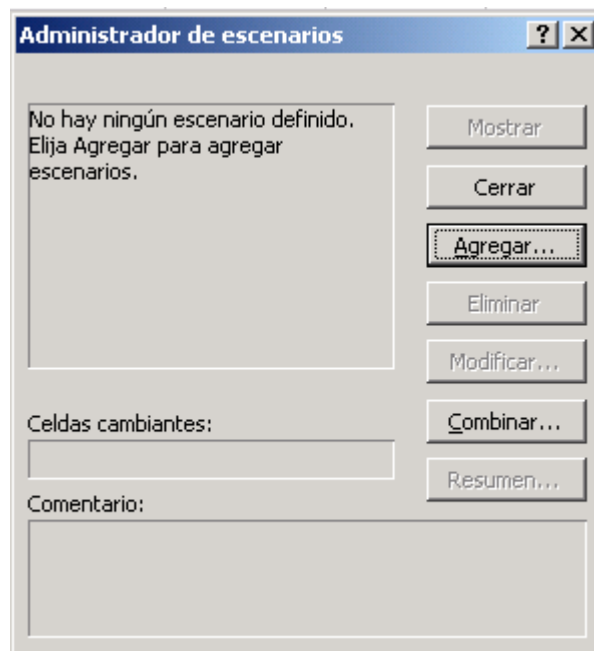
Un **Escenario** es un grupo de celdas llamadas **Celdas cambiantes** que se guarda con un nombre.

Haz una copia de la hoja con la que estamos trabajando y en la copia, modifica los datos:

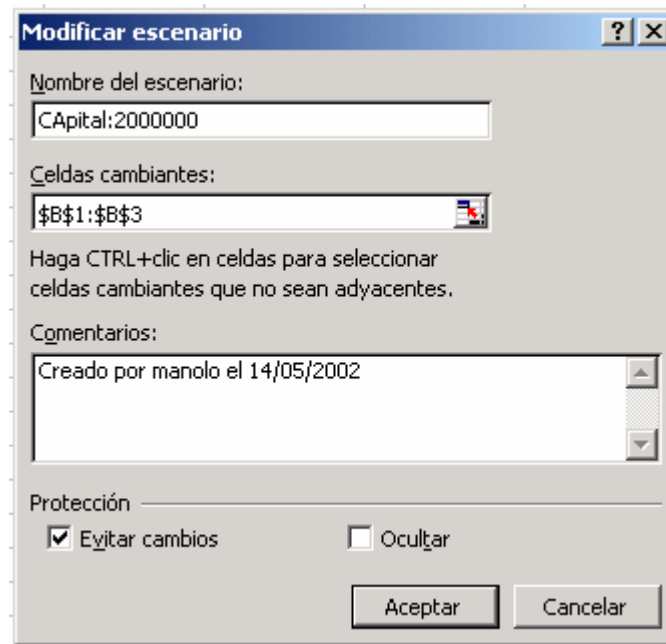
	A	B
1	Capital	2,000,000
2	Interés	4.50%
3	Años	2
4		
5	Pago Mensual	\$87,295.62
6		

Selecciona de la barra de menús la opción de **“Herramientas” – “Escenarios”** y pulsa en **“Agregar”**.





*Rellena las casillas tal y como ves en la página siguiente:*



*Acepta el cuadro de diálogo.*

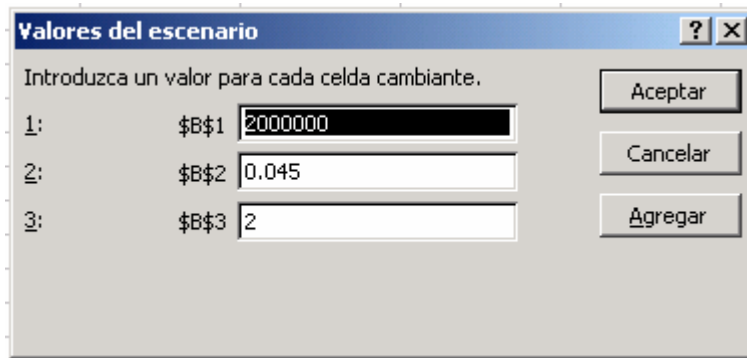
*Vuelve a aceptar el siguiente cuadro de diálogo.*

*Vuelve a pulsar en **Agregar**.*

*Esríbele el nombre:*

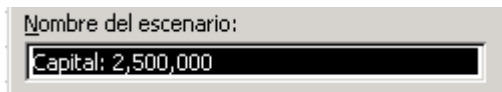


*Acepta y modifica el siguiente cuadro:*



*Acepta y agrega otro escenario.*

*Vuelve a escribir igual que antes:*



*Acepta y modifica la línea del interés:*



*Acepta.*

Acabamos de crear tres escenarios con distintas celdas cambiantes para un mismo modelo de hoja y una misma fórmula.

*Selecciona el primer escenario de la lista y pulsa en **Mostrar**. Observa el resultado en la hoja de cálculo.*

*Haz lo mismo para los otros dos escenarios. Muéstralos y observa el resultado.*

Podemos también crear un resumen de todos los escenarios existentes en una hoja para observar y comparar los resultados.

*Pulsa en **Resumen** y acepta el cuadro que aparece.*

Observa que Excel ha creado una nueva hoja en formato de subtotales (o en formato tabla dinámica si se hubiera elegido la otra opción). Esta hoja puede ser tratada como una hoja de subtotales expandiendo y encogiéndolos niveles.

1	2								
1	2	A	B	C	D	E	F	G	H
1	2								
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									

Resumen de escenario			
Valores actuales:		Capital: 2,000,000	Capital: 2,500,000
Celdas cambiantes:			
\$B\$1	2,000,000	2,000,000	2,000,000
\$B\$2	4.50%	4.50%	5.00%
\$B\$3	2	2	2
Celdas de resultado:			
\$B\$5	\$87,295.62	\$87,295.62	\$87,742.78

Notas: La columna de valores actuales representa los valores de las celdas cambiantes en el momento en que se creó el Informe resumen de escenario. Las celdas cambiantes de cada escenario se muestran en gris.

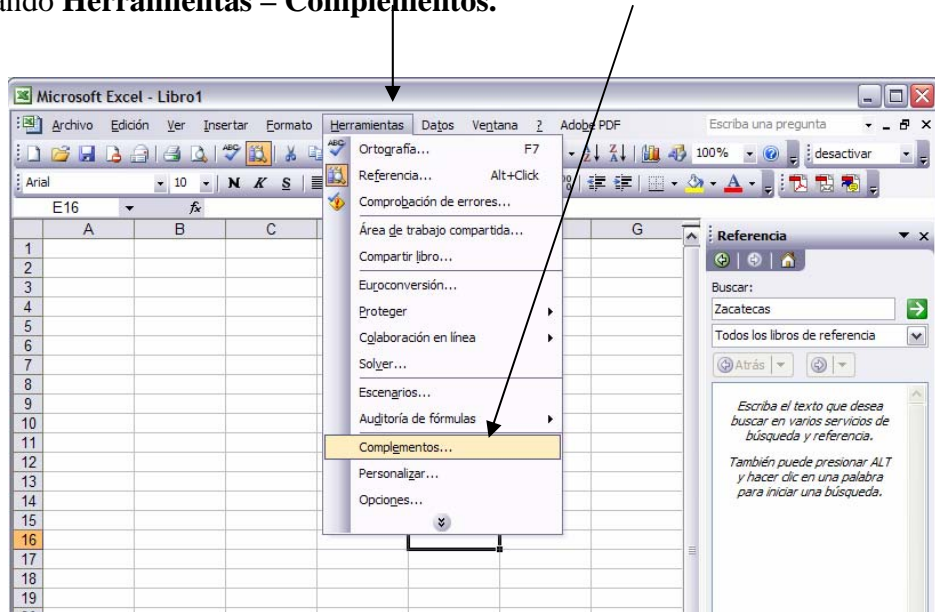
## Solver

En estos tiempos donde se habla de la tecnología, información, sociedad del conocimiento, etc., una de las herramientas más poderosas que viene incluida en la Hoja de cálculo de Excel, es el Solver, y se puede ubicar en el menú principal en la opción Herramientas, al pulsar este icono aparecerán varias opciones y ahí encontrarán dicha instrucción, ella resuelve problemas lineales y enteros utilizando el método más simple con límites en las variables y el método de ramificación y límite, implantado por John Watson y Dan Fylstra de Frontline Systems, Inc. Es de hacer notar que estos problemas se presentan en las ciencias administrativas y es requisito indispensable en casi todas las áreas de ciencias sociales, ingeniería, y en cualquiera de las carreras universitarias como Ciencias Estadísticas, Economía, Administración, entre otras, allí se estudia en una cátedra llamada Investigación de Operaciones, en ella se construyen modelos para el análisis y la toma de decisiones administrativas.

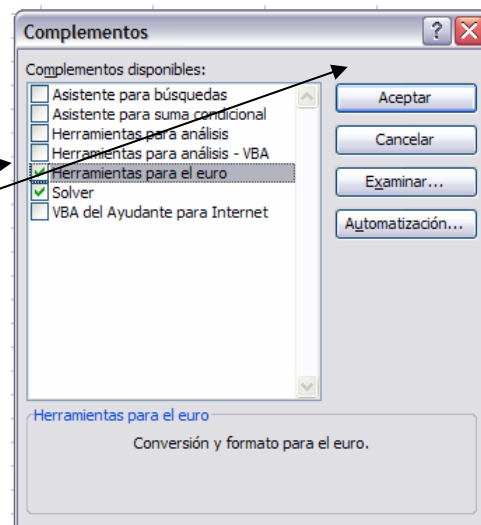
En tiempos remotos se utilizaban algoritmos muy complejos entre ellos el del método simplex y el dual, estas técnicas manualmente son complejas, pero con la tecnología aparecieron programas para resolver estos problemas entre ellos se encuentra el más conocido que es el "LINDO", pero hoy tenemos la oportunidad de resolverlos muy fácilmente mediante la hoja de cálculo de Excel y el paquete agregado llamado "SOLVER" que optimiza los modelos sujetos a restricciones, como los modelos de programación lineal y no lineales, la cual permite obtener las soluciones óptimas para un modelo determinado, y dependiendo de los niveles de la organización se tomen las mejores decisiones para resolver los conflictos de una empresa.

La función **Buscar Objetivo** que vimos anteriormente, posee gran utilidad para resolver problemas que involucren un valor objetivo exacto y que depende de un único valor desconocido. Para los problemas más complejos, se debe usar la función **Solver**. **Solver** puede manejar problemas que involucren muchas variables y pueden ayudar a encontrar las combinaciones de variables que maximicen o minimicen una celda objetivo, esto es muy importante para materias como la de programación lineal, que se lleva en muchas ingenierías y escuelas de matemáticas. También permite especificar una o más restricciones que deben cumplirse para que la solución sea válida.

**Solver** es un complemento disponible en Excel. Si usted realiza una instalación completa, en el menú de las **Herramientas** se incluirá el comando **Solver**. Si no se encuentra haz el comando **Herramientas – Complementos**.



A continuación aparecerá una ventana en donde se muestra los programas instalados que se pueden utilizar, entre ellos se encuentra el **Solver**, seleccionamos en la casilla de verificación el programa **Solver** y hacemos clic en **Aceptar**.



## Habilitación de la opción Solver

Como un ejemplo del tipo de problemas que con **Solver** se puede resolver, imagine que está planeando una campaña de publicidad para un producto nuevo. Su presupuesto total son \$12,000,000; usted quiere exponer 800 millones de veces por lo menos sus anuncios a los lectores potenciales; y usted ha decidido poner anuncios en seis publicaciones, llamadas Pub1 a Pub6. Cada publicación alcanza un número diferente de lectores y con cargos (pagos) en una proporción diferente por página. (Para mantener este análisis más simple, ignoraremos el problema de posibles descuentos por volumen). Su trabajo es llegar al mayor



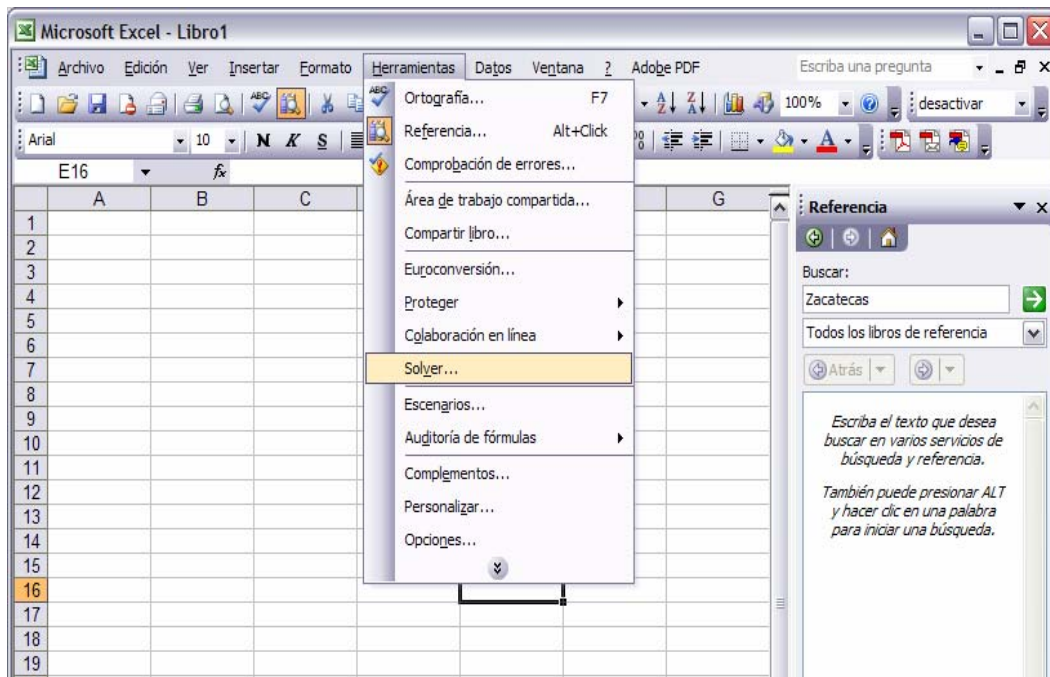
número de lectores objetivos posibles, al menor costo, bajo las siguientes condiciones adicionales:

- Por lo menos seis anuncios deben existir en cada publicación
- No más de un tercio de los pesos disponibles para publicidad deben gastarse en una publicación.
- Su costo total por poner anuncios en Pub3 y Pub4 no debe exceder \$7.500.000.

	A	B	C	D	E	F	G
			Audiencia por	Número de			
		Costo por	anuncio	anuncios		Porcentaje	Audiencia total
1	Publicación	anuncio	(millones)	situados	Costo total	del total	(millones)
2	Pub1	\$ 141,420	9.9		\$ 0	#DIV/0!	0.0
3	Pub2	\$ 124,410	8.4		\$ 0	#DIV/0!	0.0
4	Pub3	\$ 113,100	8.2		\$ 0	#DIV/0!	0.0
5	Pub4	\$ 70,070	5.1		\$ 0	#DIV/0!	0.0
6	Pub5	\$ 53,000	3.7		\$ 0	#DIV/0!	0.0
7	Pub6	\$ 52,440	3.6		\$ 0	#DIV/0!	0.0
8	<b>Total</b>				<b>\$ 0</b>		<b>0.0</b>
9	<b>Total Pub3 + Pub4</b>				<b>\$ 0</b>		
10							
11	<b>Restricciones:</b>						
12					Presupuesto para publicidad total		\$ 12,000,000
13					Presupuesto total para Pub3 + Pub4		\$ 7,500,000
14					Mínimo de audiencia total (millones)		800
15					% máximo de presupuesto a gastar en una publicación		33.33%
16					Número mínimo de anuncios por publicación		6
17							

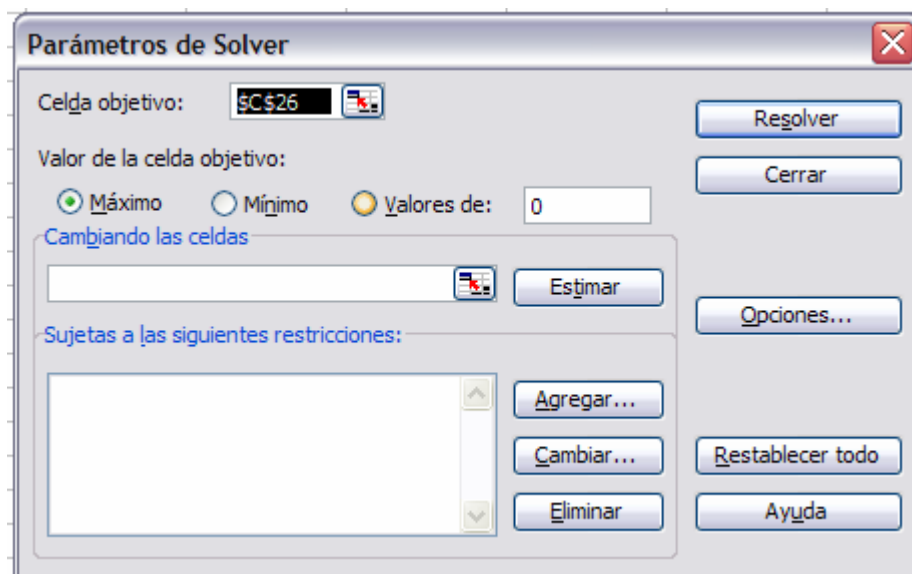
Tú puedes ser capaz de solucionar este problema sustituyendo muchas alternativas hasta lograr una respuesta satisfactoria, manteniendo el cuidado de respetar las condicionantes antes definidas, y revisando el impacto de sus cambios en el gasto total. De hecho, eso es lo que **Solver** hará por ti, pero lo hará mucho más rápido, empleando técnicas analíticas para determinar la solución óptima sin tener que probar cada alternativa posible.

Para usar **Solver**, escoja el comando **Solver** del menú **Herramientas**, encontrando una situación similar a la planteada en la siguiente figura.



Para definir los parámetros de **Solver** se deben completar tres secciones: su objetivo (de acuerdo al ejemplo sería minimizar el gasto total), sus variables o celdas de cambio (en el ejemplo sería el número de anuncios que se pondrán en cada publicación), y sus restricciones o limitantes que en el problema se plantean.

## Especificando el Objetivo



En la sección **Celda objetivo**, se indica la meta u objetivo a lograr. En este ejemplo, usted quiere minimizar el costo total, lo que a su vez se complementa en la sección **Valor de la celda objetivo** que en el ejemplo es **Mínimo** la elección.

Usted puede indicar el lugar en donde está definida la función objetivo ya sea de tres formas diferentes:

1. Indicando las coordenadas de una celda;
2. Tecleando un nombre que se ha asignado a una celda;
3. Buscando y seleccionando la celda directamente en la hoja de cálculo.

Si usted asigna un nombre a la celda objetivo, **Solver** lo utilizará de igual forma en sus informes, aún cuando usted especifica las coordenadas de la celda en lugar de su nombre en la sección **Celda Objetivo**. Si usted no especifica nombres para las celdas, en los informes de **Solver** se incluirán nombres basados en los títulos de la columna más cercana y en el texto que encabeza la fila, pero estos nombres no aparecen en las secciones de la función **Solver**. Para tener mayor claridad al leer los resultados, es una buena idea nombrar todas las celdas importantes de su modelo antes de ejecutar **Solver**.

En este ejemplo, usted quiere que **Solver** determine en la celda objetivo el menor valor posible, para lo que se selecciona **Min**. En otros problemas, usted podría querer maximizar una celda objetivo a su mayor valor posible seleccionando la opción **Max** por ejemplo si su celda objetivo expresara ganancias o utilidades. O usted podría querer que **Solver** encontrara una solución igual a algún valor particular, en dicho caso usted seleccionaría en **Valores de** la cantidad o una celda de referencia. Observe que seleccionando la opción **Valores de**, y considerando sólo una celda como variable de cambio, sin especificar restricciones, usted puede usar **Solver** como la función **Buscar objetivo**.

Usted no tiene que especificar un objetivo. Si deja la **Celda objetivo** en blanco, pulsando el botón **Opciones**, y selecciona la opción **Mostrar resultado de Iteraciones**, usted puede visualizar paso a paso las combinaciones o iteraciones de celdas de cambio en relación a las



restricciones. Usted podrá conseguir una respuesta que respete las restricciones pero no necesariamente será la solución óptima.

### Especificando las Celdas de Cambio (Variables de cambio)

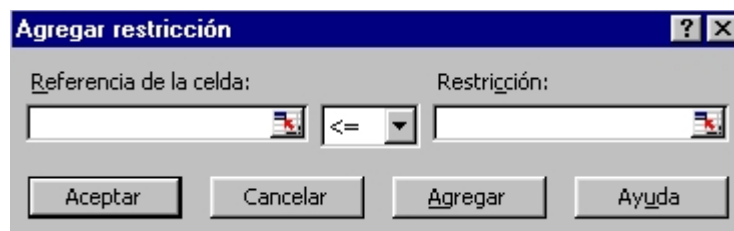
El próximo paso es indicar en **Solver** cuáles serán las celdas de cambio, lo que se señala en la sección **Cambiando las Celdas**. En el ejemplo de campaña publicitaria, las celdas cuyos valores pueden ser ajustados son aquellas que especifican el número de anuncios a ser puesto en cada publicación. Estas celdas quedan en el rango D2:D7.

Como se mencionó anteriormente, se puede proporcionar esta información tecleando las coordenadas de la celda, indicando el nombre de la celda, o seleccionando las celdas en la hoja de la planilla. Si las variables no están en celdas adyacentes, usted puede separar las celdas de cambio (o rangos) con comas (si usted selecciona celdas no adyacentes, mantenga presionada la tecla Ctrl mientras selecciona cada celda o rango). Alternativamente, usted puede pulsar el botón **Estimar**, y **Solver** propondrá las celdas de cambio más apropiadas de acuerdo a la celda objetivo especificada.

Usted puede especificar al menos una celda de cambio; de otra forma **Solver** no tendría nada que hacer. Si usted especifica una celda objetivo (tal como lo hace en la mayoría de los casos), usted debe especificar celdas de cambio que sean precedentes a la celda objetivo, o sea que estén vinculadas; es decir, celdas que en la fórmula de la celda objetivo dependen en forma prioritaria en su cálculo. Si el valor de la celda objetivo no depende de las variables, **Solver** no podrá resolver nada.

### Definiendo las Restricciones

El último paso, especificar las restricciones es optativo. Para especificar una restricción, pulse el botón **Agregar** en los **Parámetros de Solver**. La figura siguiente muestra en la cuarta restricción cómo usted expresa la restricción que los gastos totales de publicidad (valor en celda E8 en el modelo) debe ser menor o igual que el presupuesto total (el valor en celda G11).



Como puede ver, una restricción consiste de tres componentes: una celda de referencia, un operador de la comparación, y un valor de restricción. Usted especifica la referencia de la celda en la opción **Referencia de la Celda**, selecciona un operador de la comparación de la lista central, y especifica el valor de la restricción en el lado derecho. Después de especificar



una restricción de esta manera, usted puede pulsar el botón **Aceptar** para volver a los **Parámetros de Solver** o pulsar el botón **Agregar** para especificar otra restricción.

La figura siguiente muestra todos los **Parámetros de Solver** definidos. Note que las restricciones se listan en orden alfabético, no necesariamente en el orden en el que usted las definió.



Dos de las restricciones tienen un rango de referencias en el lado izquierdo del operador de la comparación.

- La expresión:  $\$D\$2:\$D\$7 \geq \$G\$15$  define que el valor de cada celda en D2:D7 debe ser mayor o igual a 6 y;
- La expresión  $\$F\$2:\$F\$7 \leq \$G\$14$  define que el valor de cada celda en F2:F7 no debe ser mayor que un 33,33 por ciento.

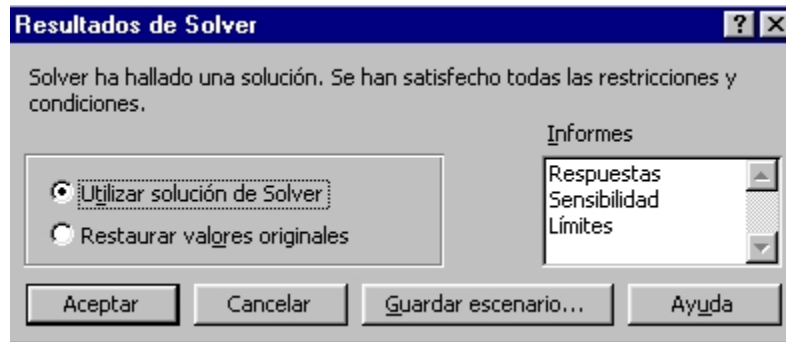
Cada una de estas expresiones es una manera de simplificar seis restricciones separadas, en sólo una. Si usted usa este tipo de simplificación, el valor de la restricción en el lado derecho del operador de la comparación debe ser un rango de las mismas dimensiones que el rango del lado izquierdo, una referencia de celda simple, o un valor constante.

Después de completar los **Parámetros de Solver**, pulse el botón **Resolver**. En la medida que **Solver** trabaja, aparecen mensajes en la barra de estado. **Solver** determina valores por ensayo en las celdas de cambio, recalcula la planilla, y entonces prueba los resultados. Comparando el resultado de cada iteración con el de la iteración predecesora, **Solver** determina el conjunto de valores que satisfagan el objetivo así como las restricciones.

En el ejemplo de campaña de anuncios, Solver tiene éxito encontrando un valor óptimo para la celda objetivo considerando todas las restricciones, desplegando el mensaje mostrado en



que se muestra a continuación. Los valores desplegados en la planilla de cálculo en ese momento producen la solución óptima. Usted puede dejar estos valores en la planilla de cálculo seleccionando Aceptar, o puede restaurar los valores que sus variables tenían antes de que usted activara Solver pulsando el botón Cancelar o seleccionando Restaurar Valores Originales y luego Aceptar. Usted también tiene la opción de asignar los valores de la solución a un escenario determinado.



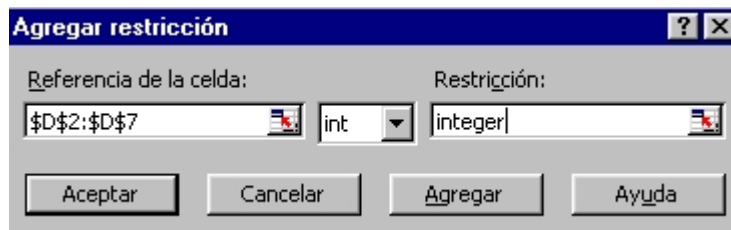
Los valores de la solución mostrados en la imagen siguiente indican que usted puede mantener sus costos de campaña de anuncio a un mínimo con 16,6 anuncios en Pub1, 6,0 en Pub2, 32,9 en Pub3, 53,2 en Pub4, 6,0 en Pub5, y 6,0 en Pub6. Esta combinación de colocaciones expone a su público objetivo a 800 millones de veces (asumiendo que el número de lectores de las publicaciones son correctos). Desgraciadamente, como no es posible ejecutar una fracción de un anuncio, la solución no es práctica.

	A	B	C	D	E	F	G
			Audiencia por anuncio (millones)	Número de anuncios situados	Costo total	Porcentaje del total	Audiencia total (millones)
1	Publicación	Costo por anuncio					
2	Pub1	\$ 141,420	9.9	16.6	\$ 2,348,441	21.0%	164.4
3	Pub2	\$ 124,410	8.4	6.0	\$ 746,460	6.7%	50.4
4	Pub3	\$ 113,100	8.2	32.9	\$ 3,726,423	33.3%	270.2
5	Pub4	\$ 70,070	5.1	53.2	\$ 3,726,423	33.3%	271.2
6	Pub5	\$ 53,000	3.7	6.0	\$ 318,000	2.8%	22.2
7	Pub6	\$ 52,440	3.6	6.0	\$ 314,640	2.8%	21.6
8	Total				\$ 11,180,386		800.0
9	Total Pub3 + Pub4				\$ 7,452,846		
10							
11	Restricciones:	Presupuesto para publicidad total					\$ 12,000,000
12		Presupuesto total para Pub3 + Pub4					\$ 7,500,000
13		Mínimo de audiencia total (millones)					800
14		% máximo de presupuesto a gastar en una publicación					33.33%
15		Número mínimo de anuncios por publicación					6
16							

Usted puede solucionar el resultado no entero de dos maneras: redondeando, o agregando nuevas restricciones que obliguen a los resultados a adoptar números enteros. En la próxima sección se discute lo que pasa si usted exige a **Solver** entregar una solución entera.

## Especificando una restricción de números enteros

Para estipular que sus variables de colocación de anuncios se restrinjan a números enteros, usted ejecuta Solver como de costumbre y pulsa el botón Agregar en los Parámetros de Solver. En Agregar, usted selecciona el rango que contiene sus anuncios de colocación (D2:D7). Luego, despliega la lista central del cuadro de diálogo y seleccione int. Solver inserta la palabra integer en la Restricción, tal como se muestra en la imagen que se muestra a continuación. Pulse el botón Aceptar para volver a los Parámetros de Solver.



Cuando usted pulsa el botón Resolver en los Parámetros de Solver para ejecutar el problema con la nueva restricción de números enteros, usted puede lograr una solución óptima con números enteros disminuyendo las Pub1 de 16,6 a 16, las Pub3 de 32,9 a 32, las Pub4 de 53,2 a 53, y aumentando las Pub5 y Pub6 a 9 y 7, respectivamente. Estos valores generan un número de lectores de 800 millones a un costo de \$ 11.186.170.

### ¿Usted necesita Restricciones con Números Enteros?

Al agregar una restricción para números enteros a **Solver**, puede aumentar geoméricamente la complejidad del problema, pudiendo generarse retrasos posiblemente inaceptables. El ejemplo discutido en este capítulo es relativamente simple y no toma una cantidad excesiva de tiempo para la resolución, pero un problema más complejo con restricción para números enteros podría plantear un desafío mayor para **Solver**. Ciertos problemas pueden resolverse sólo usando una restricción para enteros. En particular, las soluciones enteras son útiles para problemas en los que las variables pueden asumir sólo dos valores, como 1 ó 0 (sí o no), pero usted también puede usar la opción bin en la selección de la Restricción.

## Guardando y Re Utilizando los Parámetros de Solver

Cuando usted guarda un libro de trabajo después de usar el **Solver**, todo los valores que utilizó en los **Parámetros de Solver** se graban junto con sus datos de la planilla de cálculo. Usted no necesita especificar nuevamente el problema si quiere continuar trabajando con él durante una sesión posterior de Excel.

Para guardar más de un set de parámetros de **Solver** con una hoja de cálculo dada, usted debe usar la opción de **Guardar Modelo**. Para usar esta opción, siga estos pasos:

1. Elija **Solver** del menú de Herramientas.
2. Presione el botón Opciones, entonces en las Opciones de Solver mostrada en la imagen que se presenta a continuación, pulse el botón Guardar Modelo. Excel sugiere una celda en donde guardar los parámetros de Solver en la hoja de cálculo.



3. Especificar una celda vacía o teclear su referencia y entonces pulsar el botón Aceptar. Si usted especifica una sola celda, Solver pega en un rango el modelo, empezando en la celda indicada e inserta fórmulas en tantas celdas debajo de él como sea necesario. (Esté seguro que las celdas debajo de la celda indicada no contienen datos). Si usted especifica un rango, Solver llena sólo las celdas especificadas de los parámetros del modelo. Si el rango es demasiado pequeño, algunos de sus parámetros no se guardarán.
4. Para reutilizar los parámetros guardados, pulse el botón Opciones en los Parámetros de Solver, pulse el botón a Cargar Modelo, y entonces especifique el rango en el que usted guardó los parámetros de Solver.

Usted encontrará más fácilmente al guardar y reutilizar los parámetros de **Solver** si asigna un nombre a cada rango del modelo, después de que usted usa la opción de **Guardar Modelo**.

### ***Asignando los Resultados de Solver a un Escenario***

Una mejor manera de guardar sus parámetros de **Solver** es salvarlos como un Escenario empleando **Guardar Escenarios**. Como usted podría haber notado, el mensaje de **Resultados de Solver**. Solver incluye la opción **Guardar Escenario**. Pulsando este botón activa el Administrador de Escenarios y le permite asignar un nombre al escenario junto con



los valores actuales de sus celdas de cambio. Esta opción proporciona una manera excelente para explorar y realizar un análisis posterior en una variedad de posibles resultados.

## Otras Opciones de Solver

En la imagen de Opciones de Solver mostrada anteriormente, se observa que el Solver contienen varias alternativas que podrían necesitar alguna explicación. Con el Tiempo y las Iteraciones, usted le indica a Solver cuán duro es trabajar en la solución. Si Solver alcanza el tiempo límite o el número límite de iteraciones antes de encontrar una solución, el cálculo se detiene y Excel le pregunta si usted quiere continuar. Las opciones predefinidas son normalmente suficientes para resolver la mayoría de los problemas, pero si usted no alcanza una solución con estas opciones, puede probar ajustándolas.

La opción **Precisión** es usada por Solver para determinar lo cercano que se quiere que los valores en las celdas en donde se ha definido la restricción tenga con los límites impuestos en dicha restricción. El valor máximo es 1, lo que representa la precisión más baja. Especificando un valor menor que la cifra por defecto 0,000001, origina mayores tiempos en lograr la solución.

La opción **Tolerancia** sólo se aplica a problemas que usan restricción para números enteros y representa un porcentaje de error permitido en la solución.

Las opciones **Estimación**, **Derivadas**, y **Hallar** quedan mejor en su elección predefinidas, a menos que usted entienda técnicas de optimización lineal. Si usted quiere más información sobre estas opciones, remítase a la ayuda en línea de Excel.

## La Opción Adoptar Modelo Lineal

Un problema de optimización lineal es uno en el que el valor de la celda objetivo es una función lineal de cada celda de cambio; es decir, si usted hace un gráfico de XY del valor de la celda objetivo contra todos los valores posibles de cada celda de cambio, sus gráficos serán líneas rectas. Si alguno de sus gráficos produce curvas en lugar de las líneas rectas, el problema es no lineal.

La opción **Adoptar Modelo Lineal** sólo puede activarse para los modelos en los que todas las relaciones son lineales. Modelos que usan adición y substracción simple y funciones de la hoja de cálculo como **SUMA** son lineales por naturaleza. Sin embargo, la mayoría de los modelos son no lineales. Ellos son generados multiplicando celdas cambiantes por otras celdas cambiantes, usando factores exponenciales o de crecimiento, o usando funciones de la hoja de cálculo no lineales como PMT.



**Solver** puede manejar problemas de optimización lineal y no lineales. Puede resolver problemas lineales más rápidamente si usted selecciona en la opción **Adoptar Modelo Lineal**. Si usted selecciona esta opción para un problema no lineal y entonces intenta resolver el problema, **Solver** despliega un mensaje **No se Satisfacen Las Condiciones Para Adoptar un Modelo Lineal**. Si usted no está seguro de la naturaleza de su modelo, es mejor no usar esta opción.

Si usted selecciona **Adoptar un Modelo Lineal** y entonces elige la opción de **Informe de Sensibilidad**, **Solver** produce un **Informe de Sensibilidad** en una forma ligeramente diferente que para los problemas no lineales.

### La Importancia de Usar Valores Iniciales Apropriados

Si su problema es no lineal, usted debe estar consciente de un detalle muy importante: su opción de valores iniciales puede afectar la solución generada por **Solver**. Con problemas no lineales usted debe hacer siempre lo siguiente:

- Ponga sus celdas de cambio con valores razonablemente cercanos al valor óptimo antes de ejecutar el problema.
- Pruebe valores iniciales alternativos de ver qué impacto puedan tener en la solución de **Solver**.

### La Opción de Mostrar los Resultados de las Iteraciones

Si usted está interesado en explorar muchas combinaciones de sus celdas de cambio, en lugar de sólo la combinación que produce el resultado óptimo, usted puede obtener ventajas al revisar los **Resultados de las Iteraciones**. Simplemente pulse el botón **Mostrar Resultado de Iteraciones** en las **Opciones de Solver**. Después de cada iteración, podrá guardar el resultado en un escenario, continuando con la próxima iteración.

Usted debe estar consciente que cuando usa **Mostrar Resultado de Iteraciones**, **Solver** hace una pausa en soluciones que no cumplen todas sus restricciones así como para soluciones subóptimas.

### Generando Informes

Además de insertar valores óptimos en las celdas de cambio de su problema, **Solver** puede resumir sus resultados en tres informes: **Respuestas**, **Sensibilidad** y **Límites**. Para generar uno o más informes, seleccione los nombres de los informes. Seleccione los informes que usted quiere y entonces pulse el botón **Aceptar**. (Empleando la tecla Ctrl podrá seleccionar más de uno). Cada informe se guarda en un hoja de cálculo separado en el libro de trabajo actual, con la etiqueta identificada con el nombre del informe.



## El Informe de Sensibilidad

El informe de **Sensibilidad** proporciona información sobre cuán sensible es la celda objetivo a los cambios en sus restricciones. Este informe tiene dos secciones: uno para sus celdas de cambio y uno para sus restricciones. La columna derecha en cada sección proporciona la información de sensibilidad.

Cada celda de cambio y restricciones se lista en una fila separada. La sección de la celda de cambio incluye un valor de **Gradiente Reducido** que indica cuánto la celda objetivo sería afectada por un aumento de una unidad en la celda de cambio correspondiente. Igualmente, el **Multiplicador de Lagrange** en la sección de las **Restricciones** indica cuánto la celda objetivo sería afectada por un aumento de una unidad en el valor de la restricción correspondiente.

## El Informe de Sensibilidad para un Modelo Lineal

Si usted selecciona **Adoptar Modelo Lineal** en las **Opciones**, el informe de **Sensibilidad** incluye varias columnas adicionales de información.

Para las celdas cambiantes, la columna del **Costo Reducido** muestra el aumento en el valor de la celda objetivo por cada unidad que cambie la celda de cambio. La columna del **Coefficiente Objetivo** muestra el grado en que la celda de cambio y la celda objetivo están relacionadas. El **Aumento** y **Disminución Permissible** muestran la cantidad que el **Coefficiente Objetivo** debe cambiar antes de que las celdas cambiantes sean afectadas.

Para las restricciones, la columna de **Precio Sombra** indica el aumento en el valor objetivo por cada unidad que se incrementan las restricciones. La columna del **Lado Derecho de la Restricción** (RHS) simplemente despliega el valor de la restricción usado en el problema. Y el **Aumento Aceptable** y la **Disminución Permissible** muestran la cantidad que el valor de la restricción (mostrado en la columna del Lado Derecho de la Restricción) debe cambiar antes de que las celdas cambiantes sean afectadas.

## El Informe de Respuesta

El informe de **Respuesta** lista la celda objetivo, las celdas de cambio, y las restricciones. Este informe también incluye información sobre el estado del valor de holgura para cada restricción. Los estados pueden ser **Opcional**, **Obligatorio** o **No Satisfecho**. El valor de holgura es la diferencia entre el valor de la solución de las celdas de restricción y el número que aparece en el lado derecho de la fórmula de restricción. Una restricción **Obligatoria** es una para el que el valor de holgura es 0. Una restricción **Opcional** es una restricción que estaba satisfecha con un valor de holgura diferente a 0.

## El Informe de Límites

El informe de **Límites** dice cuánto pueden aumentar o disminuirse los valores de sus celdas de cambio sin transgredir las restricciones de su problema. Para cada celda de cambio, este informe lista el valor óptimo así como los valores más bajos y más altos que pueden usarse sin violar la restricción.

## Cuando el Solver es Incapaz de Resolver

El **Solver** es poderoso pero no milagroso. No podría resolver cada problema que usted plantee. Si **Solver** no puede encontrar la solución óptima a su problema, presenta un mensaje de realización infructuoso en la respuesta a la **Solución de Solver**.

**Los mensajes de la realización infructuosa más comunes son los siguientes:**

- **Solver** no podría encontrar una solución factible. **Solver** es incapaz de encontrar una solución que satisfice todas sus restricciones. Esto puede pasar si las restricciones están lógicamente en conflicto (por ejemplo, si en restricciones separados usted pregunta que Pub1 sea mayor que 5 y menor que 3) o si no todas las restricciones pueden satisfacerse (por ejemplo, si usted insiste que se logre una campaña de publicidad para 800 millones de lectores con \$1 millones de presupuesto).
- En algunos casos, **Solver** devuelve también este mensaje si el valor inicial de sus celdas de cambio están lejos de sus valores óptimos. Si usted piensa que sus restricciones están lógicamente consistentes y su problema es soluble, pruebe cambiando el valor inicial y reejecutando **Solver**.
- El límite máximo de iteración fue alcanzado; ¿continúa? Para evitar que su computadora trabaje indefinidamente con un problema insoluble, **Solver** se diseñó para hacer una pausa y presentar este mensaje cuando ha realizado su número predefinido de iteraciones sin llegar a una solución. Cuando usted ve este mensaje, usted puede reiniciar la búsqueda para una solución pulsando el botón **Continuar**, o usted puede pulsar el botón **Detener**. (Usted también puede asignar los valores actuales a un escenario).
- Si usted pulsa el botón **Continuar**, **Solver** empieza resolviendo de nuevo y no se detiene hasta que encuentre una solución, se rinde o alcanza su límite de tiempo de máximo. Si sus problemas frecuentemente exceden el límite de iteraciones de **Solver**, usted puede aumentar el valor predefinido escogiendo en las **Opciones** un nuevo valor en las Iteraciones.
- El límite de tiempo de máximo fue alcanzado; ¿continúa? Este mensaje es similar al mensaje del límite de iteración. **Solver** se diseñó para hacer una pausa después que un periodo de tiempo predefinido ha pasado. Usted puede aumentar este valor por defecto escogiendo en las **Opciones** la alternativa **Tiempo**.



Para dejar más en claro la utilización del Solver para resolver problemas de programación lineal haremos el siguiente ejemplo:

Para construir el modelo de programación lineal lo primero que debemos de hacer es:

- 1.- Definir Variables de decisión
- 2.- Definir la función objetivo
- 3.- Definir las restricciones

Utilidad o Pérdida =  $PX - CX - F$

Max  $Z = PX - CX - F$

Donde

$X \leq U$        $P = \text{Precio}$

$X \leq D$        $C = \text{Costo por unidad}$

$X \geq 0$        $X = \text{Unidades vendidas}$

$F = \text{Costo fijo}$

### Ejemplo

Andrés Z. Es presidente de una microempresa de inversiones que se dedica a administrar las carteras de acciones de varios clientes. Un nuevo cliente ha solicitado que la compañía se haga cargo de administrar para él una cartera de \$100.000. A ese cliente le agradaría restringir la cartera a una mezcla de tres tipos de acciones únicamente, como podemos apreciar en la siguiente tabla. Formule usted un modelo de Programación Lineal para mostrar cuántas acciones de cada tipo tendría que comprar Andrés con el fin de maximizar el rendimiento anual total estimado de esa cartera.

Acciones	Precio (\$)	Rendimiento Anual Estimado por Acción (\$)	Inversión Posible (\$)
Naves a	60	7	60.000
Telectricidad	25	3	25.000
Rampa	20	3	30.000

Para solucionar este problema debemos seguir los pasos para la construcción de modelos de programación lineal (PL):

- 1.- Definir la variable de decisión.

2.- Definir la función objetivo.

3.- Definir las restricciones.

Luego construimos el modelo:

$$\text{MAX } Z = 7X_1 + 3X_2 + 3X_3$$

$$60X_1 + 25X_2 + 20X_3 \leq 100.000$$

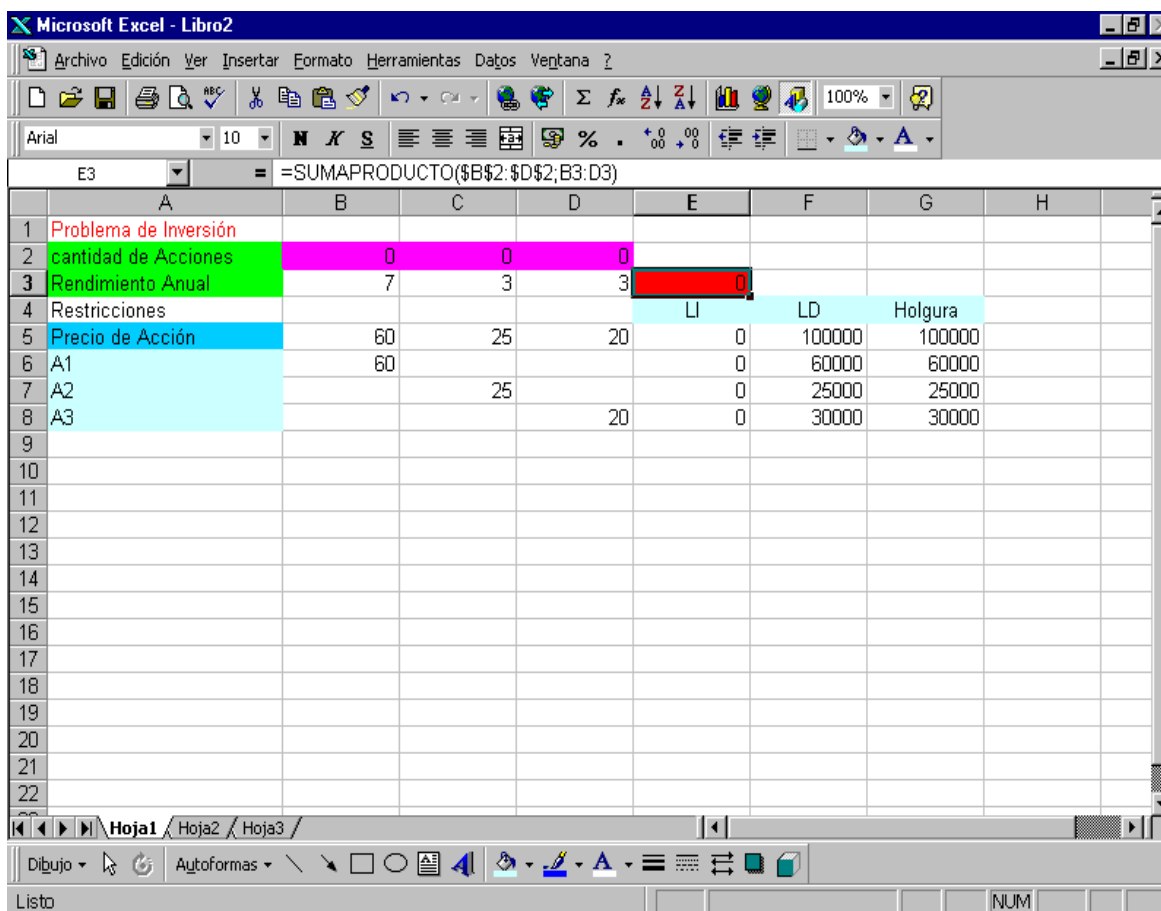
$$60X_1 \leq 60.000$$

$$25X_2 \leq 25.000$$

$$20X_3 \leq 30.000$$

$$X_i \geq 0$$

A continuación se construye el modelo en una hoja de cálculo de Excel de la siguiente manera:



	A	B	C	D	E	F	G	H
1	Problema de Inversión							
2	cantidad de Acciones	0	0	0				
3	Rendimiento Anual	7	3	3	0			
4	Restricciones				LI	LD	Holgura	
5	Precio de Acción	60	25	20	0	100000	100000	
6	A1	60			0	60000	60000	
7	A2		25		0	25000	25000	
8	A3			20	0	30000	30000	
9								
10								
11								
12								
13								
14								
15								
16								
17								
18								
19								
20								
21								
22								

En la fila 2 se coloca la variable de decisión, la cual es el número de acciones y sus valores desde la B2 hasta la D2.



En la fila 3 el rendimiento anual y sus valores desde B3 hasta D3. En la celda E3 colocaremos una fórmula la cual nos va indicar el rendimiento anual total, =sumaproducto(\$B\$2:\$D\$2;B3:D3).

Desde la fila B5 hasta la D8 colocaremos los coeficientes que acompañan a las variables de decisión que componen las restricciones.

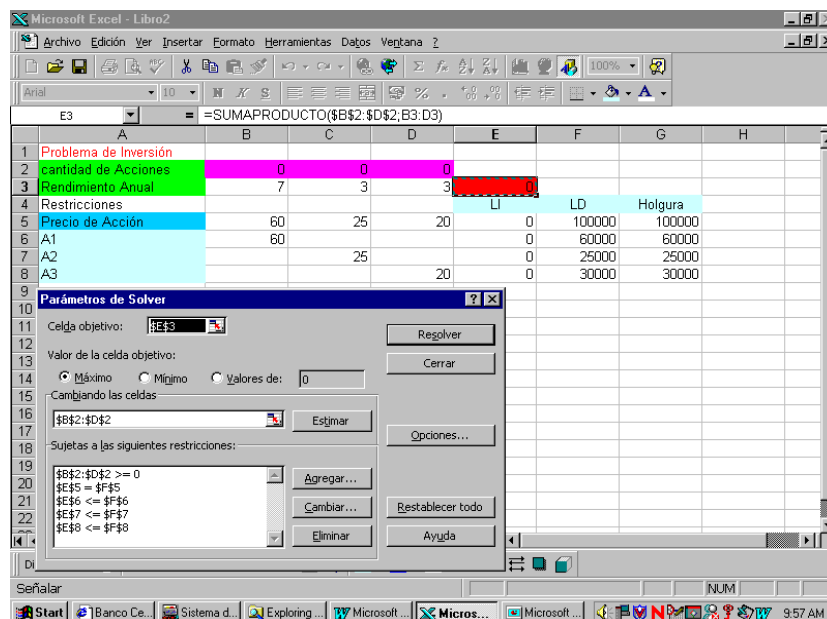
Desde la E5 hasta la E8 se encuentra la función de restricción (LI) y no es mas que utilizar la siguiente fórmula =sumaproducto(\$B\$2:\$D\$2;B5:D5), la cual se alojaría en la celda E5, luego daríamos un copy hasta la E8.

Desde la F5 hasta F8 se encuentran los valores de las restricciones. Desde la G5 hasta G8 se encuentra la holgura o excedente.

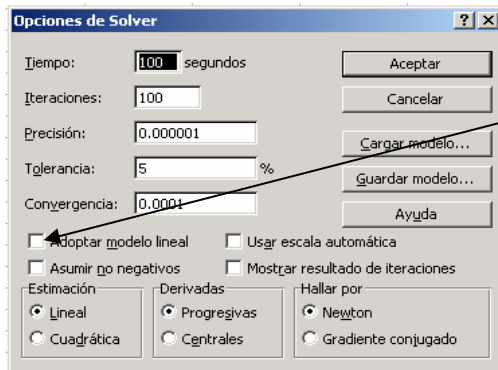
Una vez completada la hoja de cálculo con el modelo respectivo, **grabe su hoja**, y seleccione de la barra de menú “**Herramientas**” – “**Solver**”, ahí tendrá que especificar dentro del cuadro de diálogo de Solver:

- La celda que va a optimizar
- Las celdas cambiantes
- Las restricciones

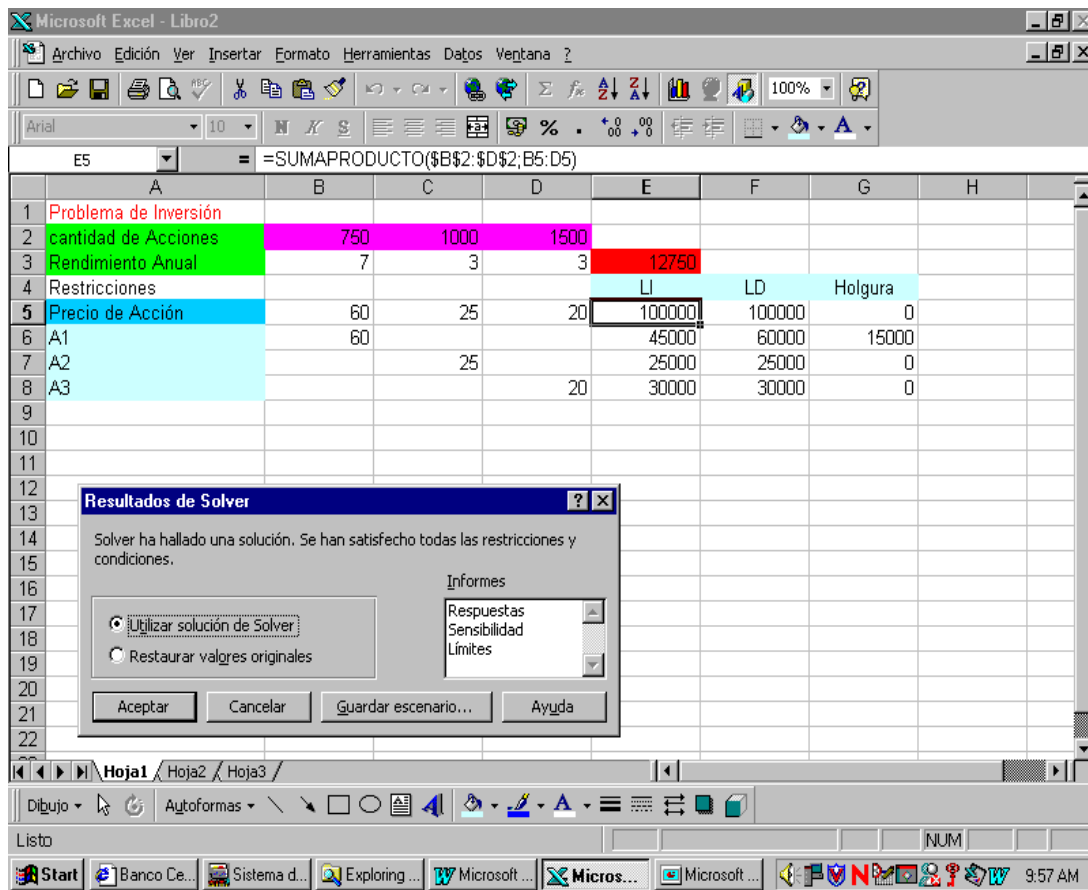
Así tendremos la siguiente pantalla:



Como se puede observar en la celda objetivo se coloca la celda que se quiere optimizar, en las celdas cambiantes las variables de decisión y por último se debe complementar con las restricciones. Una vez realizados estos pasos deben pulsar el icono de "Opciones".



Debe hacer clic en "Asumir modelo lineal" y enseguida el botón de "Aceptar". Luego haga clic en el botón de "Resolver" para realizar la optimización, lea detenidamente el mensaje de terminación de Solver y ahí observará si se encontró una solución o hay que modificar el modelo, en caso de haber encontrado una solución óptima usted podrá aceptar o no dicha solución, luego tendrá oportunidad de realizar un informe de análisis de sensibilidad para luego tomar la mejor decisión.



En nuestro ejemplo el máximo rendimiento anual fue de \$12,750 y la cantidad de acciones a comprar serían 750, 1000 y 1500 para Navesa, Telectricidad y Rampa respectivamente. De esta forma podemos observar la potencia que tiene el Solver. (Para mayor información sobre el tema, en la ayuda de la hoja de cálculo de Excel o en libro de Investigación de Operaciones Taha, Ed. Limusa).



## UNIDAD 8

### Acceso a datos del exterior

A veces puede ocurrir que necesitemos datos que originalmente se crearon con otros programas especiales para ese cometido. Podemos tener una base de datos creada con Access o de dBASE que son dos de los más conocidos gestores de bases de datos y posteriormente querer importar esos datos hacia Excel para poder trabajar con ellos.

Para ello necesitaremos una aplicación especial llamada **Microsoft Query** que nos permitirá acceso a datos externos creados desde distintos programas.

También es posible que sólo nos interese acceder a un conjunto de datos y no a todos los datos de la base por completo, por lo que utilizaremos una **Consulta** que son parámetros especiales donde podemos elegir qué datos queremos visualizar o importar hacia Excel.

Si deseamos acceder a este tipo de datos, es necesario haber instalado previamente los controladores de base de datos que permiten el acceso a dichos datos. Esto lo puedes comprobar desde el **Panel de Control** y accediendo al icono:



donde te aparecerá un cuadro de diálogo con los controladores disponibles:

Orígenes de datos de usuario:	
Nombre	Controlador
dBASE Files	Microsoft dBase Driver (*.dbf)
Excel Files	Microsoft Excel Driver (*.xls)
FoxPro Files	Microsoft FoxPro Driver (*.dbf)
MS Access 97 Database	Microsoft Access Driver (*.mdb)
Text Files	Microsoft Text Driver (*.txt; *.csv)

## Creación de una consulta de datos

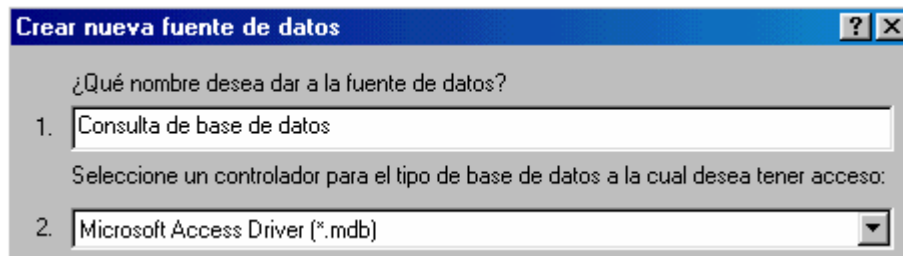
Para comenzar, es necesario definir previamente la consulta que utilizaremos indicando la fuente de datos y las tablas que queremos importar. Si no tienes nociones de la utilización de los programas gestores de bases de datos, no te preocupes porque sólo vamos a extraer datos de ellos.

Para hacerlo sigamos los siguientes pasos:

### *Accede a Datos – Obtener datos externos – Crear nueva consulta*

Aparecerá la pantalla de **Microsoft Query**. Ahora podemos dar un nombre a la nueva consulta.

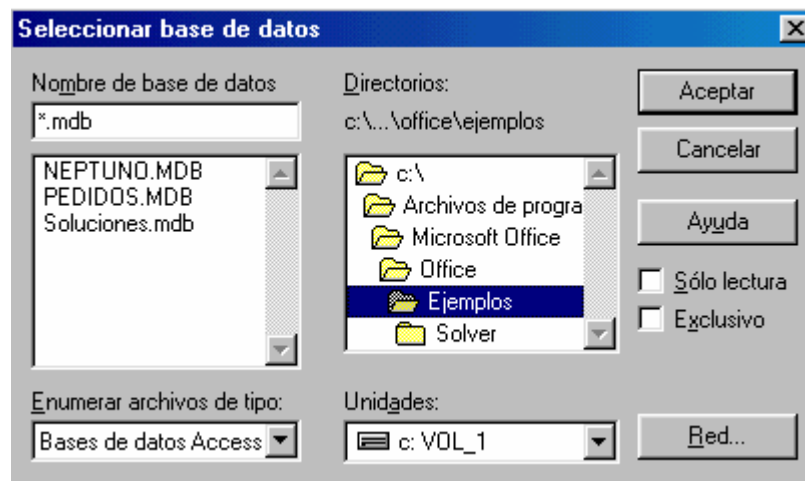
*Pulsa en Añadir y añade los siguientes datos:*



*Haz clic en Conectar.*

*Clic en Seleccionar.*

Ahora debemos indicarle la ruta donde buscará el archivo a importar. Nosotros hemos elegido la base de datos **Neptuno.MDB** que viene de ejemplo en la instalación de Microsoft Office 97. La puedes encontrar en la carpeta **C:\Archivos de programa\Microsoft Office\Office\Ejemplos**. Observa la siguiente ilustración:



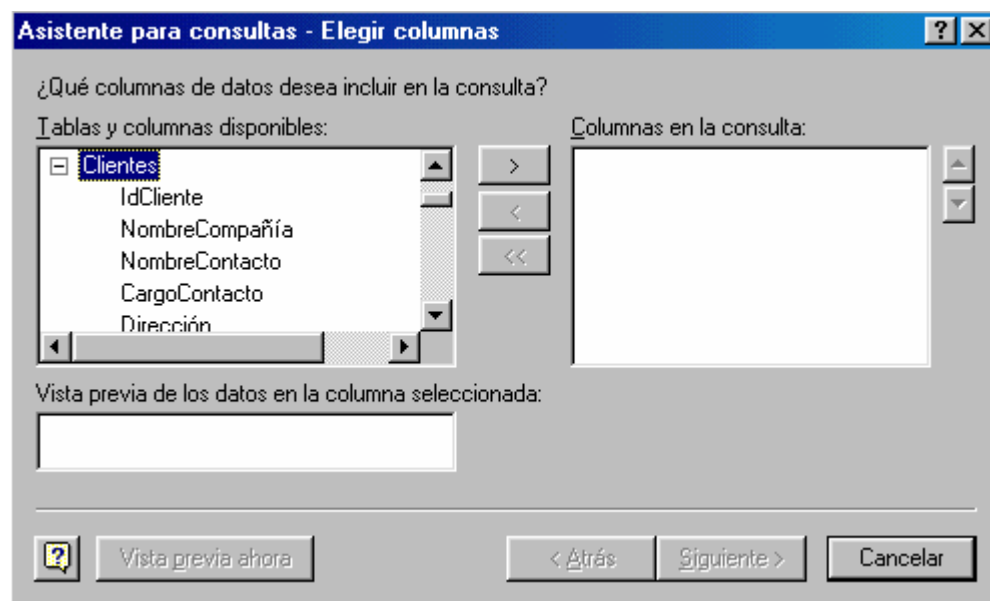
Selecciona la base de datos **SOLUCIONES.MDB** y acepta.

Acepta también el cuadro de diálogo que aparece (el anterior)

Selecciona la tabla **CLIENTES**



Acepta los cuadros de diálogo que quedan hasta que aparezca en pantalla el asistente de creación de consultas, tal y como aparece en la página siguiente:



Carga los campos **IdCliente**, **Dirección**, **Ciudad** y **Teléfono** seleccionando clic en el campo y pulsando el botón



Continúa al paso **Siguiente**.

Ahora podemos elegir de entre los campos alguna condición para la importación de los datos. Es posible que sólo nos interesen los clientes cuya población sea Barcelona. Si no modificamos ninguna opción, Excel importará todos los datos.





*Modifica las casillas de la siguiente forma:*

*Pulsa en **Siguiente**.*

*Elige el campo **IdCliente** como campo para la ordenación y **Siguiente**.*

A continuación podríamos importar los datos directamente a Excel, pero vamos a ver cómo funciona la ventana de Query. También podríamos guardar la consulta.

*Elige la opción **Ver datos...***

*Pulsa en **Finalizar**.*

## Microsoft Query

Aparece la pantalla de trabajo de Microsoft Query. Desde esta pantalla podemos modificar las opciones de consulta, el modo de ordenación, añadir o eliminar campos, etc.

Observa las partes de la pantalla: en la parte superior tenemos la típica barra de botones. En la parte central el nombre y los campos de la tabla que hemos elegido, así como la ventana de criterios de selección, y en la parte inferior los campos en forma de columna.

Podemos añadir campos a la consulta seleccionándolos de la tabla y arrastrándolos hacia una nueva columna de la parte inferior. En nuestro caso, vemos que sólo hay un cliente que cumpla la condición de ser de la ciudad de Barcelona.

*Borra el criterio **Barcelona** de la casilla de criterios.*

*Pulsa el botón **Ejecutar consulta ahora** situado en la barra de herramientas superior y observa el resultado.*

*En la barra de menú, selecciona “**Archivo**” – “**Devolver datos a Microsoft Excel**”.*

*Acepta el cuadro de diálogo que aparece.*



## Devolver datos a Excel

Ahora podemos tratar los datos como si fueran columnas normales de Excel, pero con la ventaja que también podemos modificar algunos parámetros desde la barra de herramientas que aparece.



A través de esta barra tendremos siempre la posibilidad de actualizar la consulta, haya o no haya ocurrido alguna modificación en ella.

Fíjate que es posible porque el programa almacena en un libro de trabajo la definición de la consulta de donde son originarios los datos, de manera que pueda ejecutarse de nuevo cuando deseemos actualizarlos.

Si desactivamos la casilla Guardar definición de consulta y guardamos el libro, Excel no podrá volver a actualizar los datos externos porque éstos serán guardados como un rango estático de datos.

También podemos indicar que se actualicen los datos externos cuando se abra el libro que los contiene; para ello hay que activar la casilla **Actualizar al abrir el archivo**.

Recuerda que, para que sea posible la actualización de los datos externos, se necesita almacenar la consulta en el mismo libro o tener la consulta guardada y ejecutarla de nuevo.



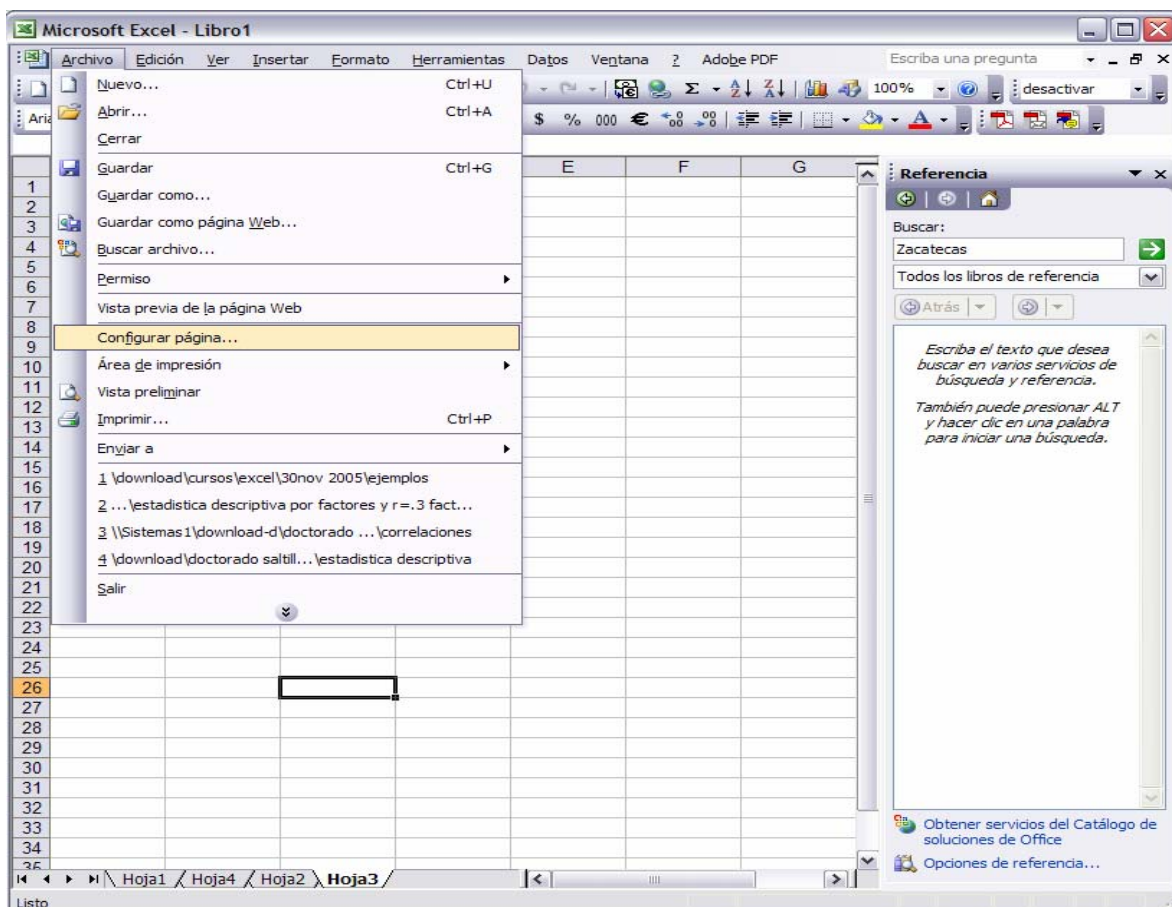
## UNIDAD 9

### Impresión en Excel

Una de las más importantes funciones de Excel, es imprimir, pues es la forma en que podemos mostrar nuestros resultados a otras personas, para ello se deben de tener en cuenta varias opciones con las que cuenta el Excel para configurar nuestros archivos para un tipo de impresora en particular.

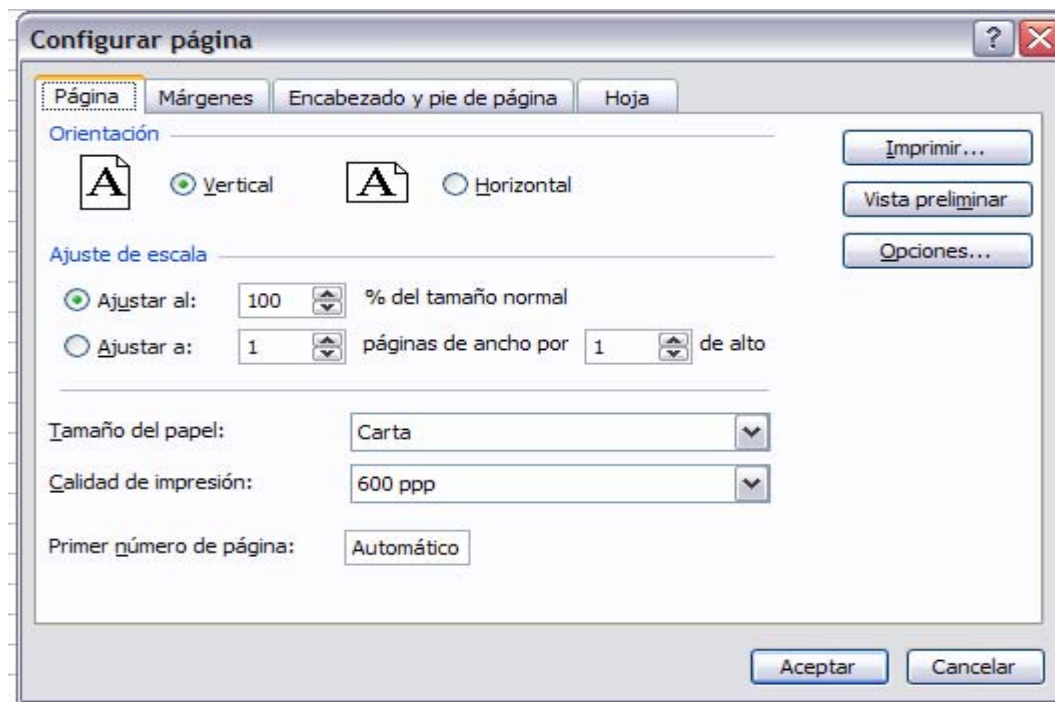
### Configuración de una hoja

Para configurar el tamaño de página, márgenes, encabezados y pies de página, accedemos a la barra de menú la opción de “Archivo” – “Configurar página”.



Aquí obtenemos la siguiente ventana la cual tiene 4 pestañas en donde se pueden configurar la página, los márgenes, los encabezados y los pies de página y la configuración de la hoja.

Desde este cuadro de diálogo podemos establecer el tamaño del papel, orientación en la impresora, cambiar la escala de impresión, colocar encabezados, etc.



Cuando la pestaña de Página está seleccionada, podemos configurar las siguientes opciones:

**Orientación:** permite ajustar la página para que su impresión sea en modo vertical o modo horizontal.


**Escala:** esta opción nos permite ajustar una hoja para que pueda ser impresa en la cantidad de hojas requeridas, o en una hoja cuando está ajustada en un página de alto por una página de ancho, esta opción es muy útil, cuando la hoja que se está imprimiendo se sale de una hoja por una columna.

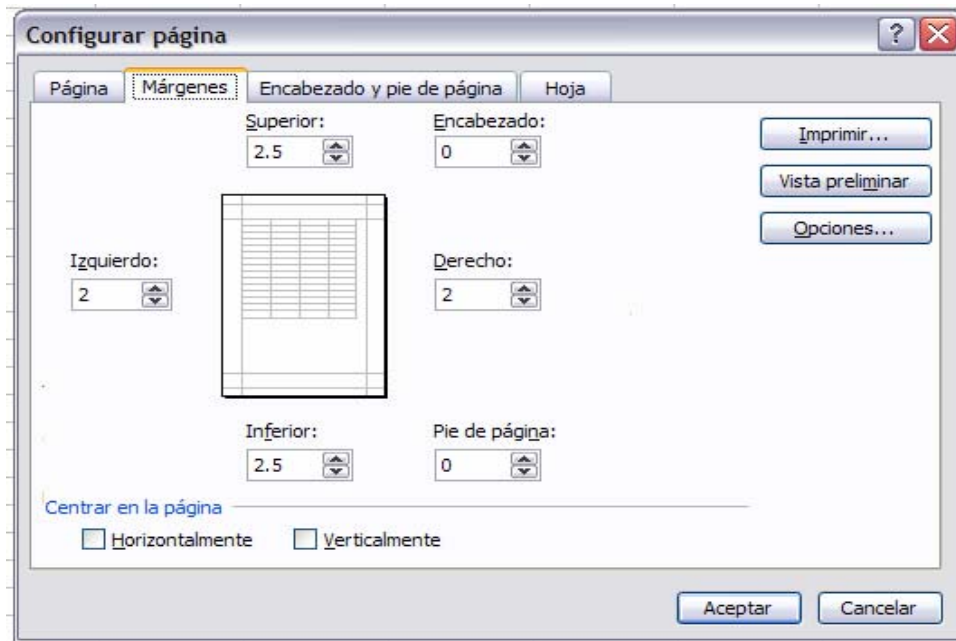
**Tamaño de papel:** en esta opción podemos definir el tamaño de papel que utilizaremos, hay que tener en cuenta que cuando no configuramos esta opción, muchas veces se nos imprime todo el texto, pues alimentamos nuestra impresora con papel tamaño carta, el cual es menos largo que el papel de tamaño A4, como en el caso que seleccionemos papel tamaño legal y pongamos tamaño oficio, el cual es media pulgada más pequeño.

**Calidad de Impresión:** ésta depende del tipo de impresora que se esté utilizando, ya sea láser, inyección de tinta, margarita, etc.

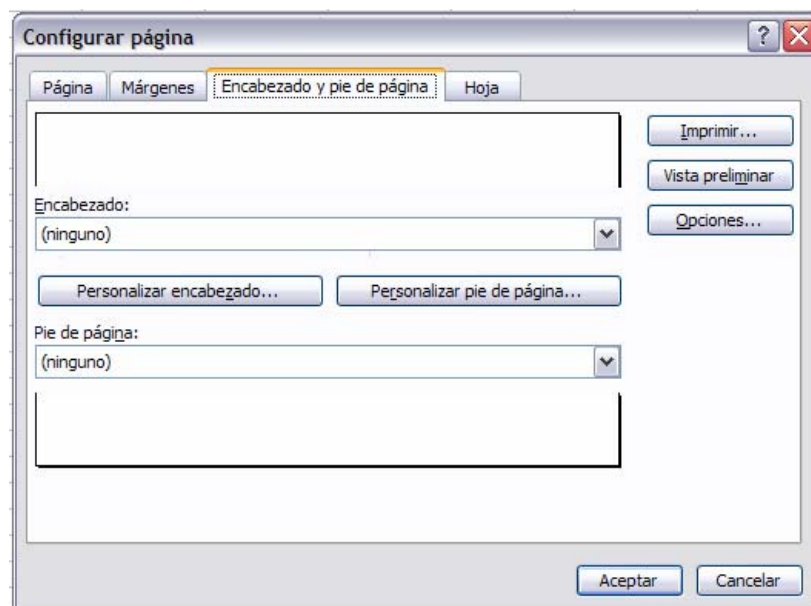
**Primer número de página:** Esta opción nos permite numerar las hojas desde un número deseado.



Utilizando la última hoja que tenemos en pantalla, veamos qué hacer en el caso de impresión de una hoja. En principio tenemos el botón Vista preliminar situado en la barra superior de herramientas que nos permite obtener una visión previa  del resultado de la hoja antes de imprimir.

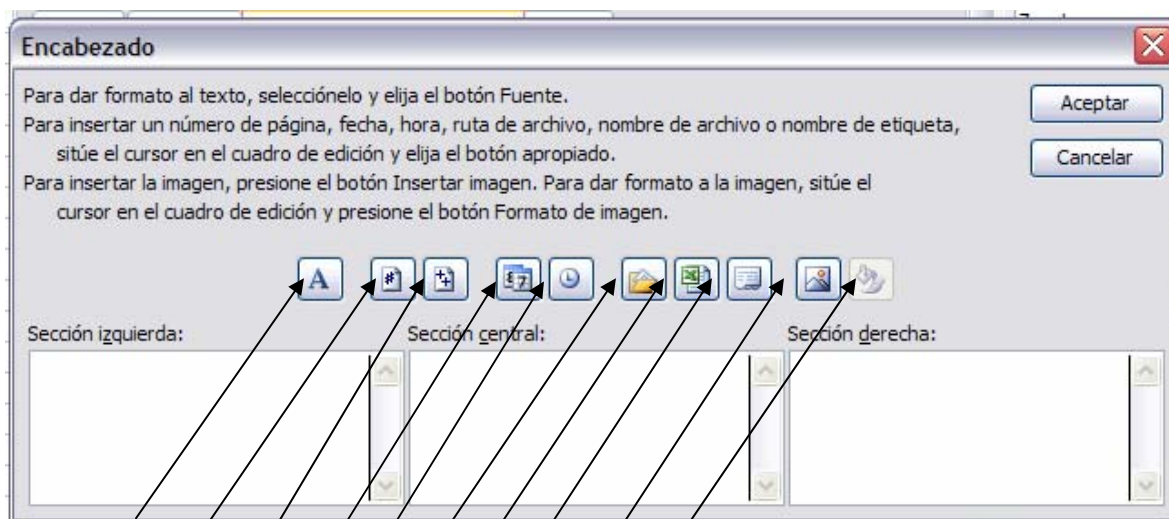


La segunda pestaña se refiere a los márgenes en la hoja que estamos utilizando. Aquí podemos personalizar nuestros márgenes, tenemos 4, el superior, el inferior, el izquierdo y el derecho, además de el margen que tiene el pie de página y el margen desde el borde del encabezado. Además se pueden centrar las hojas horizontalmente y verticalmente, seleccionando las cajas de opciones que se encuentran en la parte inferior de esta ventana.





En la pestaña de Encabezados y Pies de Página podemos configurar los que se encuentra en la parte superior y en la sección inferior de las páginas, esto significa que se imprimirá lo que deseamos en todas las hojas. Si presionas el botón “Personalizar el encabezado”, obtendrás la siguiente ventana:



**Fuente:** cambia el tipo de fuente, el tamaño de fuente, el estilo del texto seleccionado en los cuadros sección izquierda, sección central o sección derecha.

**Número de página:** inserta los números de página en el encabezado o en el pie de página al imprimir una hoja de cálculo, Microsoft Excel actualiza estos números automáticamente al agregar o eliminar datos o al establecer saltos de página.

**Número de páginas:** inserta el número total de páginas en la hoja activa y ajusta los números de página de forma automática al imprimir las hojas de cálculo.

**Fecha:** inserta la fecha actual.

**Hora:** inserta la hora actual.

**Ruta y nombre de archivo:** inserta la ruta y el nombre del archivo activo.

**Nombre de archivo:** inserta el nombre del archivo activo

**Nombre de hoja:** inserta nombre de la hoja activa:

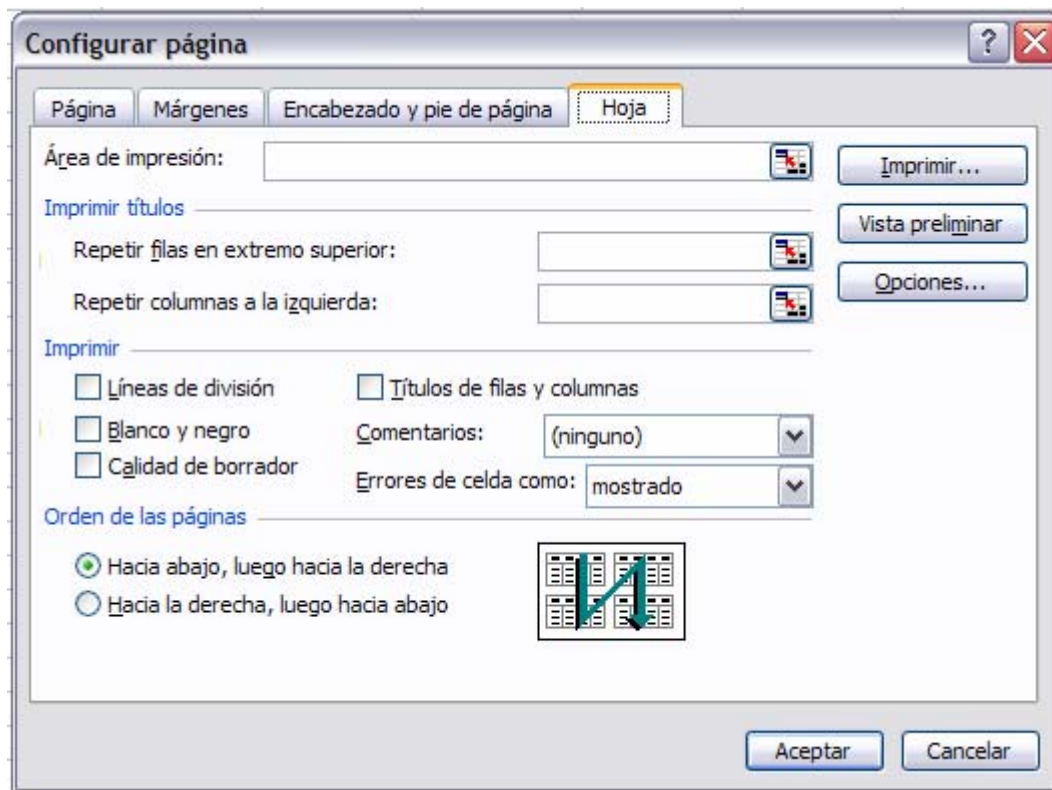
**Botón de insertar imagen:** permite elegir la imagen que desea insertar en la hoja de cálculo.

**Botón formato de imagen:** permite ajustar el tamaño, girar, establecer la escala, recortar y ajustar la imagen que ha elegido insertar en la hoja de cálculo activa.

Si presionas el botón “Personalizar pie de página”, obtendrás la misma ventana que vimos anteriormente, solamente que la configuración es para el pie de página.



Si seleccionamos la pestaña de **Hoja** en la ventana de configurar impresión, aparecerá una ventana como ésta:



**Área de impresión:** sirve para seleccionar el rango de la hoja de cálculo que desee imprimir, haz clic en el cuadro de área de impresión y arrastra hacia las áreas de las hojas de cálculo que desees imprimir. El botón Contraer diálogo, situado en el extremo derecho de este cuadro de diálogo, desplazará de forma temporal el cuadro de diálogo para que puedas especificar el rango mediante la selección de celdas de la hoja de cálculo. Cuando haya finalizado, puedes hacer otra vez clic otra vez en el botón para presentar todo el cuadro de diálogo.

**Repetir filas (renglones) en el extremo superior:** esta opción sirve para imprimir los mismos renglones como títulos en cada página de una hoja de cálculo impresa. Si desea especificar alguna fila como título horizontal de cada página, selecciona repetir filas en el extremo superior.

**Repetir columnas en el extremo superior:** esta opción sirve para imprimir las mismas columnas como títulos en cada página de una hoja de cálculo impresa, si desea especificar alguna fila como título horizontal de cada página, seleccione repetir filas en el extremo superior.

**Líneas de división:** imprime las líneas de división de celdas horizontales y verticales en las hojas de cálculo.



**Blanco y Negro:** si tu hoja está en colores, esta opción te permite imprimir sólo en blanco y negro.

**Calidad borrador:** con esta opción seleccionada, excel sólo imprimirá los valores, pero no imprimirá la mayoría de los gráficos ni las líneas de división de la hoja.

**Títulos:** Imprime los números de renglones y letras de las columnas.

**Comentarios:** imprime los comentarios como una hoja aparte o al final de la impresión, recuerda que los comentarios se insertaron en la hoja haciendo clic en el botón derecho del ratón.

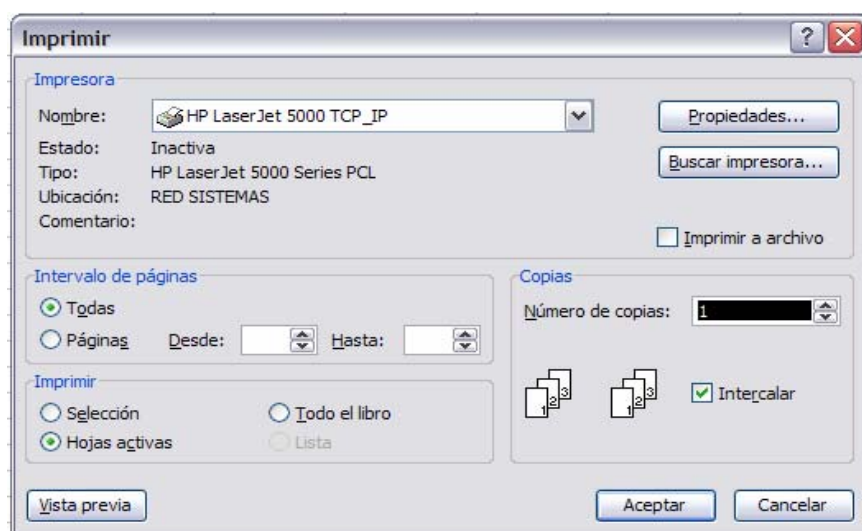
**Errores:** permite especificar cómo aparecerán los errores en el documento impreso. ya sea como un espacio en blanco, “—“ o ·N/A.

**Orden:** ésta es la forma en como se imprimirán las hojas de cálculo, cuando es más de una, existen 2 opciones de arriba abajo y luego de izquierda a derecha, la segunda opción, es primero imprimir de izquierda a derecha y después de arriba abajo.

El botón de opciones es para configurar la impresora, para más información consulte el cuaderno informático 1 de PC y Windows.

## Imprimir

Para imprimir el documento selecciona de la barra de menú, la opción de “**Archivo**” – “**Imprimir**”, a continuación aparecerá una ventana como la que se presenta a continuación:





Lo primero que observamos es el nombre de la impresora; la que se está utilizando, en el ejemplo que estamos viendo, es la impresora Epson 850 y está conectada en red en la máquina que se llama máquina1.

**Estado:** esto se refiere a si la impresora está imprimiendo algún trabajo de alguna otra computadora en la red, o de la misma computadora.

**Tipo:** es el nombre que se le dio a la impresora.

**Ubicación:** en donde se encuentra ubicada la impresora, si está conectada al puerto paralelo aparecerá LPT1:, en la imagen mostrada, la impresora está conectada a el puerto de red <\\sistemas1\\epson850>

**Intervalos de páginas:** esta opción sirve para si quiero imprimir todo el documento, o sólo deseo imprimir una hoja o un rango de hojas.

**Imprimir:** en esta opción podemos seleccionar si queremos imprimir todo el libro, la hoja activa o una selección de hojas, recuerda que esto es para imprimir lo que se encuentra dentro de las hojas, pero es diferente a lo que seleccionaste en la hoja, es decir, el rango de impresión.

**Copias:** esta opción permite configurar la cantidad de copias que desea imprimir, además puede intercalar las copias, si es que imprime 2 juegos de un mismo documento en la computadora, si seleccionas la opción de intercalar, cuando imprimas un documento, la computadora tendrá que imprimir 2 veces el documento y guardarlo en el disco duro mientras se imprime, éste se llama cola de impresión, si no seleccionas la opción de intercalar, la impresora imprimirá 2 o más veces (según la cantidad de copias que seleccionaste en la impresora), pero sólo imprime 1 vez el documento y lo guarda en el disco duro, las demás copias las imprime la impresora como si fuera una copiadora, esta opción es buena cuando imprimimos en una impresora láser, pues hay impresoras que imprimen hasta 24 páginas por minuto, si utilizas una impresora de inyección de tinta, esto no ocurre, por lo que es mejor que selecciones la opción de intercalar copias, pues estas impresoras no imprimen las demás copias como copiadora.

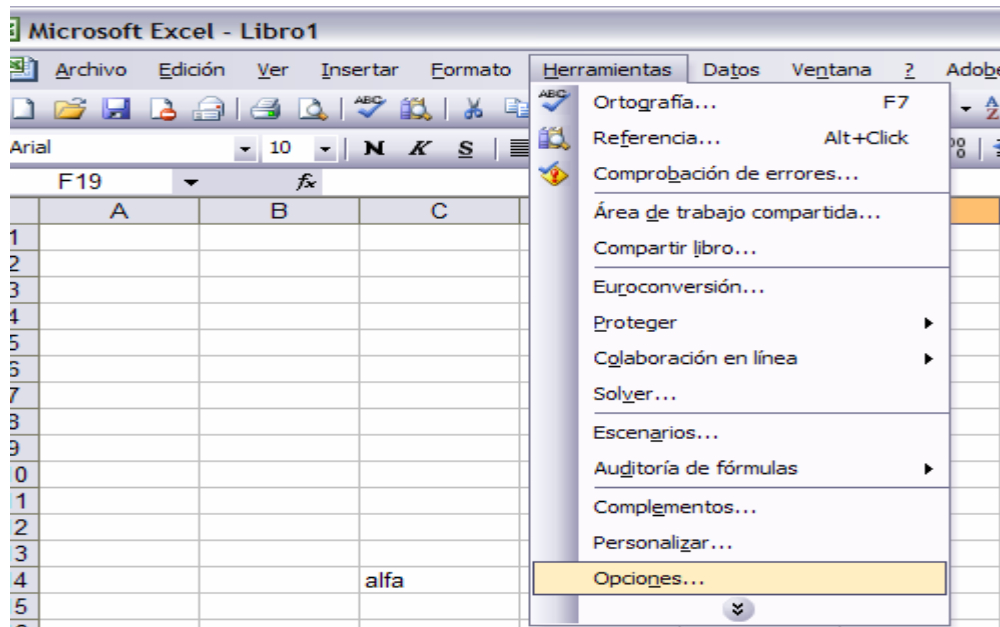
**Botón de Propiedades:** sirve para configurar la impresora en la cual vamos a imprimir.

**Botón rebuscar impresora:** sirve para encontrar una impresora que no tenga instalada en mi computadora.

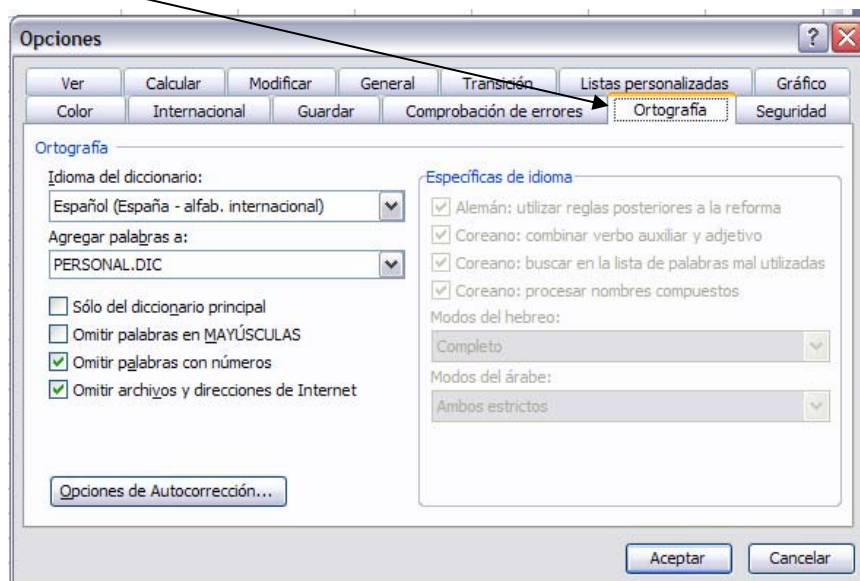
## Corrección ortográfica

Excel XP 2003 incorpora un corrector ortográfico que podemos activar al ir escribiendo texto sobre la marcha o bien una vez hayamos terminado de escribir.

El corrector que actúa sobre la marcha podemos encontrarlo en **Herramientas –Opciones**.

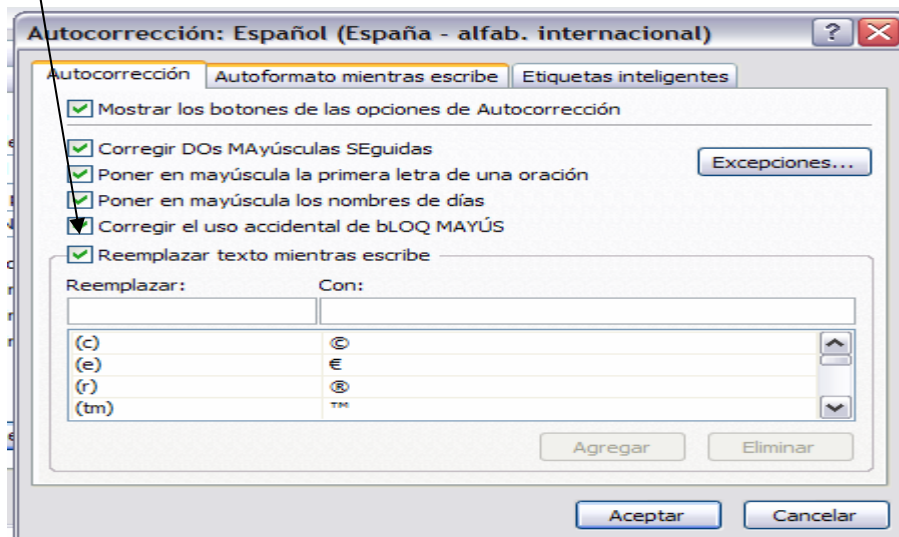


A continuación aparecerá una ventana con las opciones de ortografía, haz clic en la pestaña de ortografía.

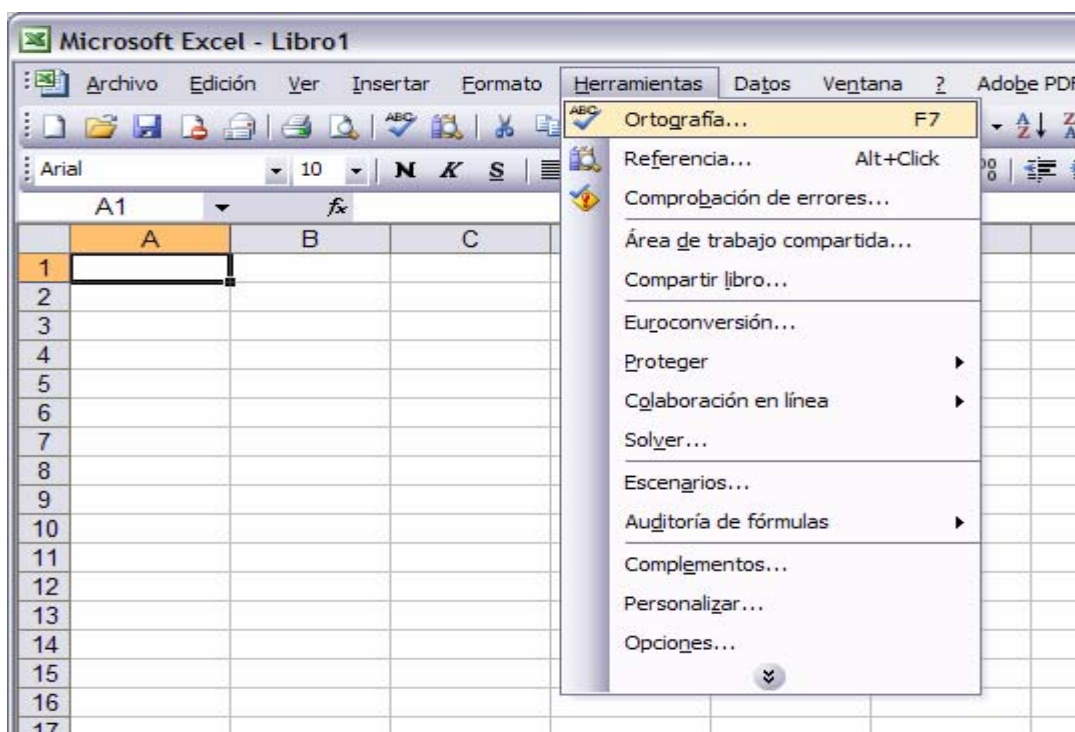


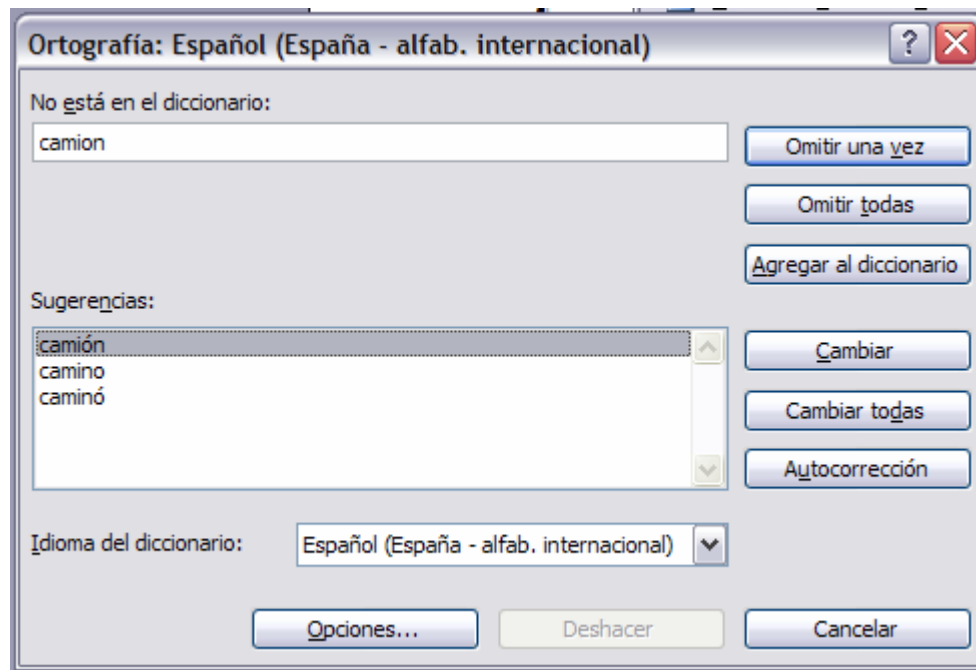


En seguida selecciona las opciones de auto corrección, pulsando en el botón **Opciones de Autocorrección**. Y te aparecerá la ventana que se muestra a continuación, selecciona la opción de reemplazar mientras escribe y haz clic en aceptar.



Otro método es corregir una vez finalizado el trabajo desde Herramientas – Ortografía. Aparecerá un menú que nos irá indicando las palabras que Excel considera con falta de ortografía. Podemos omitirlas o bien cambiarlas por las que nos ofrece el programa.

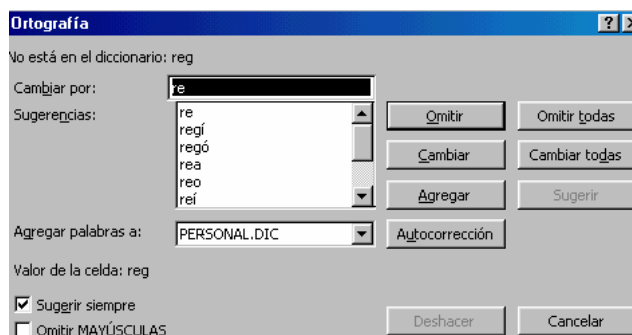




En este menú aparece un cuadro de diálogo donde podemos añadir palabras para que Excel las cambie automáticamente por otras.

Si elegimos la opción Agregar palabras a..., podemos elegir el diccionario que queremos introducir la palabra que no se encuentra en el diccionario principal de Excel. Por omisión, disponemos del diccionario PERSONAL.DIC, que se encuentra vacío hasta que le vamos añadiendo palabras nuevas.

A partir de introducir una nueva palabra en el diccionario, ésta deja de ser incorrecta. Hay que hacer notar que Excel comparte los diccionarios con otras aplicaciones de Office, por lo que si hemos añadido palabras, éstas estarán disponibles en una futura corrección desde Word, por ejemplo.



## UNIDAD 10

---

### MACROS

#### Macro

Una macro es un conjunto de instrucciones que sirve para automatizar procesos. Refiriéndonos a Excel, supongamos que realizamos frecuentemente la acción de seleccionar un rango para aplicarle negrita, cambio de fuente y centrado. En lugar de hacer estas acciones manualmente, se puede elaborar una macro e invocarla para que ejecute los tres procesos automáticamente.

#### Objetos, propiedades y métodos

A la hora de trabajar con macros en Excel, deben tenerse claros ciertos conceptos de lo que se llama programación orientada a objetos (OOP). No se hablará demasiado sobre la OOP, pero se definirán a continuación los conceptos de **Objeto**, **Propiedades** y **Métodos**.

#### Objeto

Cuando en el mundo real nos referimos a objeto significa que hablamos de algo más o menos concreto que puede ser cualquier cosa. Si se quiere concretar un poco más podemos referirnos a objeto coche, objeto silla, objeto casa, etc. En OOP, la generalización (o definición) de un objeto se llama **Clase**, así la clase coche sería como la representante de todos los coches del mundo, mientras que un objeto coche sería un coche en abstracto. De momento, no definiremos ni estudiaremos las clases sino que nos concentraremos en los objetos, pero tenga en cuenta que cualquier objeto está definido por una clase.

Cuando decimos que la clase coche representa a todos los coches del mundo significa que define cómo es un coche, cualquier coche. Dicho de otra forma y para aproximarnos a la definición informática, la clase coche define algo que tiene cuatro ruedas, un motor, un chasis,... entonces, cualquier objeto real de cuatro ruedas, un motor, un chasis, etc. es un objeto de la clase coche.

#### Propiedades

Cualquier objeto tiene características o propiedades como por ejemplo el color, la forma, peso, medidas, etc. Estas propiedades se definen en la clase y luego se particularizan en cada objeto. Así, en la clase coche se podrían definir las propiedades Color, Ancho y Largo, luego al definir un objeto concreto como coche ya se particularizarían estas propiedades a, por ejemplo, Color=Rojo, Ancho=2 metros y Largo =3,5 metros.



## Métodos

La mayoría de los objetos tienen comportamientos o realizan acciones, por ejemplo, una acción evidente de un objeto coche es el de moverse o lo que es lo mismo, trasladarse de un punto inicial a un punto final. Cualquier proceso que implica una acción o pauta de comportamiento por parte de un objeto se define en su clase para que luego pueda manifestarse en cualquiera de sus objetos. Así, en la clase coche se definirían en el método mover todos los procesos necesarios para llevarlo a cabo (los procesos para desplazar de un punto inicial a un punto final), luego cada objeto de la clase coche simplemente tendría que invocar este método para trasladarse de un punto inicial a un punto final, cualesquiera que fueran esos puntos.

Repasemos a continuación todos estos conceptos pero ahora desde el punto de vista de algunos de los objetos que nos encontraremos en **Excel** como **Worksheet** (Objeto hoja de cálculo) o **Range** (Objeto casilla o rango de casillas).

Un objeto **Range** está definido por una clase donde se definen sus propiedades, recordemos que una propiedad es una característica, modificable o no, de un objeto. Entre las propiedades de un objeto **Range** están **Value**, que contiene el valor de la casilla, **Column** y **Row** que contienen respectivamente la fila y la columna de la casilla, **Font** que contiene la fuente de los caracteres que muestra la casilla, etc.

**Range**, como objeto, también tiene métodos, recordemos que los métodos sirven para llevar a cabo una acción sobre un objeto. Por ejemplo el método **Activate**, hace activa una celda determinada, **Clear**, borra el contenido de una celda o rango de celdas, **Copy**, copia el contenido de la celda o rango de celdas en el portapapeles.

## Conjuntos

Un conjunto es una colección de objetos del mismo tipo, para los que conozcan algún lenguaje de programación es un array de objetos. Por ejemplo, dentro de un libro de trabajo puede existir más de una **Hoja (Worksheet)**, todas las hojas de un libro de trabajo forman un conjunto, **el conjunto Worksheets**.

Cada elemento individual de un conjunto se referencia por un índice, de esta forma, la primera, segunda y tercera hoja de un libro de trabajo, se referenciarán por **Worksheets(1)**, **Worksheets(2)** y **Worksheets(3)**. (**Hoja1**, **Hoja2**, **Hoja3**)

## Objetos de Objetos

Es muy habitual que una propiedad de un objeto sea otro objeto. Siguiendo con el coche, una de las propiedades del coche es el motor, y el motor es un objeto con propiedades como cilindrada, caballos, número de válvulas, etc. y métodos, como **aumentar\_revoluciones**, **tomar\_combustible**, **mover\_pistones**, etc.



En Excel, el objeto **WorkSheet** tiene la propiedad **Range** que es un objeto, **Range** tiene la propiedad **Font** que es también un objeto y **Font** tiene la propiedad **Bold** (negrita). Tenga esto muy presente ya que utilizaremos frecuentemente Propiedades de un objeto que serán también Objetos. Dicho de otra forma, hay propiedades que devuelven objetos, por ejemplo, la propiedad **Range** de un objeto **WorkSheet** devuelve un objeto de tipo **Range**.

### **Programación Orientada a Objetos o Programación Basada en Objetos**

Hay una sutil diferencia entre las definiciones del título. Programación orientada a Objetos, significa que el programador trabaja con objetos fabricados por él mismo, es decir, el programador es quien implementa las clases para luego crear objetos a partir de ellas. Lo que haremos nosotros, por el momento, será utilizar objetos ya definidos por la aplicación Excel (WorkSheets, Range,...) sin implementar ninguno nuevo, por lo que en nuestro caso es más correcto hablar de programación basada en objetos. Observe que ésta es una de las grandes ventajas de la OOP, utilizar objetos definidos por alguien sin tener que conocer nada sobre su implementación, sólo debemos conocer sus propiedades y métodos y utilizarlos de forma correcta.

Bueno, después de esta extensa pero necesaria introducción pasemos ya a hacer alguna cosa en Excel. No es necesario que se aprenda lo anterior al pie de la letra y tampoco es necesario que lo comprenda al cien por cien, sólo téngalo presente para las definiciones que vienen a continuación y verá cómo va asimilando los conceptos de Objeto, propiedades, métodos, etc.

### **Editor de Visual Basic**

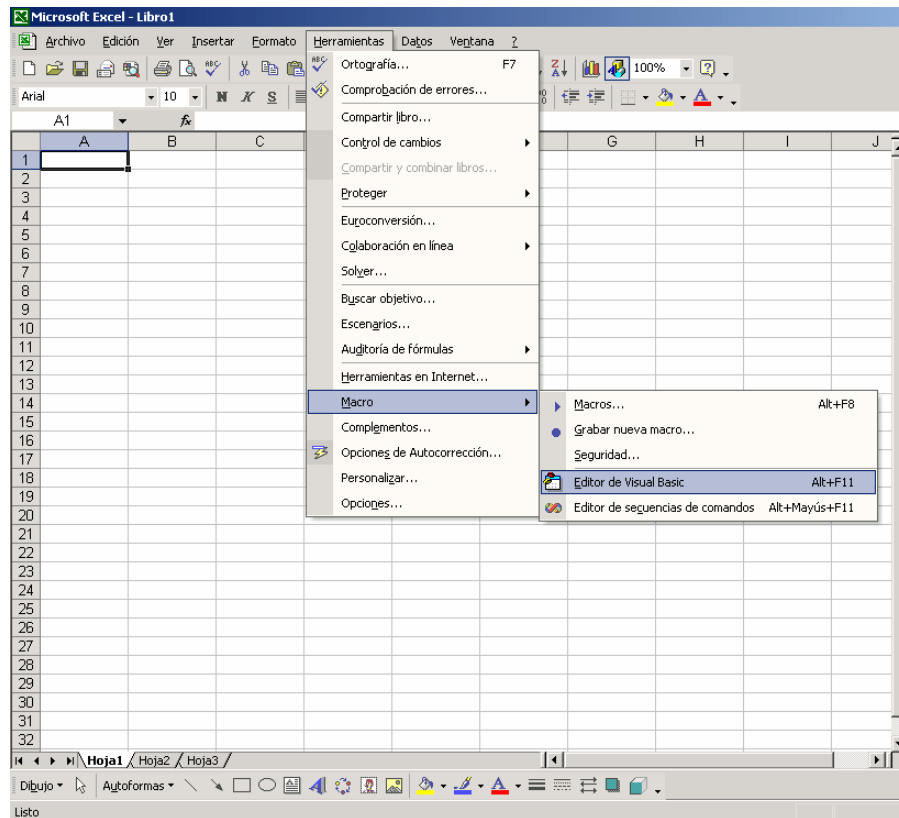
El editor de Visual Basic es la aplicación que utilizaremos para construir las macros que interactuarán junto con los libros de trabajo. A continuación prepararemos un archivo en el que escribiremos las primeras instrucciones en Visual Basic.

### ***Preparar un archivo nuevo***

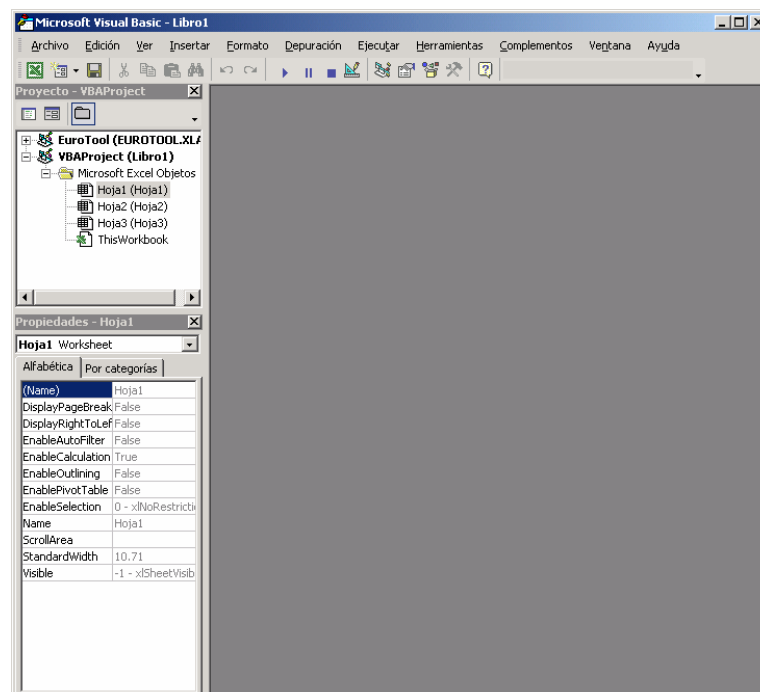
Para entrar en el editor de Visual Basic, ejecute los pasos siguientes.

Selecciona en la barras de menú “**Herramientas**” – “**Macro**” – “**Editor de Visual Básic**”. Se abrirá la ventana siguiente:





Maximiza la ventana para trabajar más cómodamente y procura tener activadas la ventana **Explorador de proyectos** y la ventana **Propiedades**.



### ***Inserta un nuevo módulo***

Un módulo sirve para agrupar procedimientos y funciones. El procedimiento y la función son entidades de programación que sirven para agrupar instrucciones de código que realizan una acción concreta.

Para insertar un módulo active opción del menú **Insertar/ Módulo**. Se activará una nueva ventana, si aparece demasiado pequeña, maximícela.

### ***Insertar un procedimiento***

Ya hemos dicho que un procedimiento es un bloque de instrucciones de código que sirven para llevar a cabo alguna tarea específica. Un procedimiento empieza siempre con la instrucción **Sub** y termina con las palabras clave **End Sub**; la sintaxis de un procedimiento se muestra a continuación:

**Sub** *Nombre\_Procedimiento*

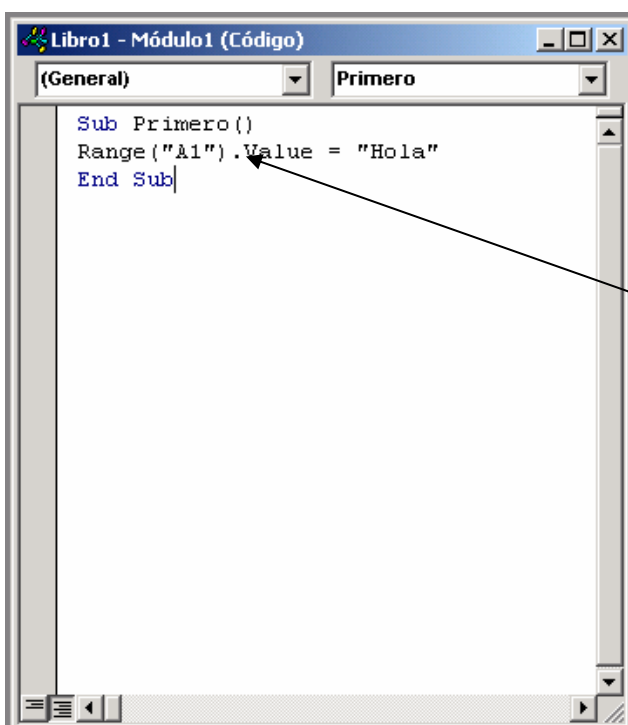
Sentencia1

Sentencia2

Sentencia3

**End Sub**

A continuación crearemos un procedimiento para poner el texto "Hola" en la casilla A1.



Para ello en la barra de menú del Visual Basic selecciona **“Insertar”** – **“Módulo”**, con ello se obtendrá la ventana de Módulo(1) código, en ella teclearemos el código de nuestro ejemplo.

Observe el código.

**Range("A1").Value="Hola"**

En esta línea estamos indicando que trabajamos con un objeto **Range**. Para indicarle que nos referimos a la casilla A1, encerramos entre paréntesis esta referencia (más adelante verá otra forma de referirnos a las casillas). De este objeto, indicamos que queremos establecer un nuevo valor para la propiedad **Value**, observe que para separar el objeto de su propiedad utilizamos la notación punto.

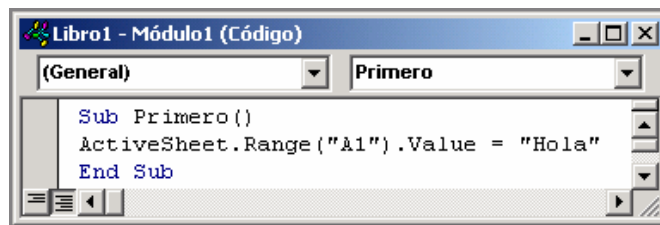
Recuerde que el conjunto **Range** es un objeto que depende del objeto **Worksheets**, así por ejemplo el siguiente código haría lo mismo que el anterior.

**Worksheets(1).Range("A1").Value = "Hola"**

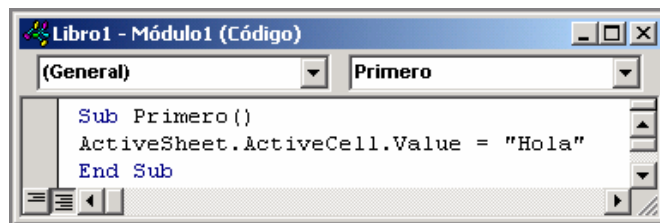
De hecho, estas 2 instrucciones no hacen lo mismo, en la primera instrucción, el texto "Hola" se pone dentro de la casilla A1 de la hoja activa, mientras que en la instrucción anterior ordena a Excel que en la casilla A1 de primera hoja (del conjunto de hojas) se ponga el texto "Hola".

La segunda notación es más larga, pero también más recomendable ya que se especifican todos los objetos. En muchas ocasiones se pueden omitir algunos objetos precedentes, no le aconsejamos hacerlo, sus programas perderán claridad y concisión.

Si desea hacer referencia a la hoja activa puede utilizar **ActiveSheet**, así, el primer ejemplo lo dejaremos de la manera siguiente:



Si desea poner "Hola" (o cualquier valor) en la casilla activa, puede utilizar la propiedad (objeto) **Activecell** de **Worksheets**. Así para poner "Hola" en la casilla activa de la hoja activa sería:



**Worksheets** están dentro del Objeto **WorkBooks** (libros de trabajo) y **WorkBooks** están dentro de **Application**. **Application** es el objeto superior, es el que representa la aplicación Excel. Así, el primer ejemplo, siguiendo toda la jerarquía de objetos quedaría de la forma siguiente:


**Sub**

**Application.WorkBooks(1).Worksheets(1).Range("A1").Value = "Hola"**

**End Sub**

Insistiendo con la nomenclatura, **Application** a veces es muy necesario especificarlo, pues a veces se hacen aplicaciones de office y por tanto, la aplicación puede ser Word,

PowerPoint, etc. **WorkBooks** es necesario implementarlo si en las macros se trabaja con diferentes libros de trabajo (diferentes archivos), a partir de **WorkSheets**, es aconsejable incluirlo en el código, sobre todo si se quiere trabajar con diferentes hojas, veré, sin embargo, que en muchas ocasiones no se aplica.

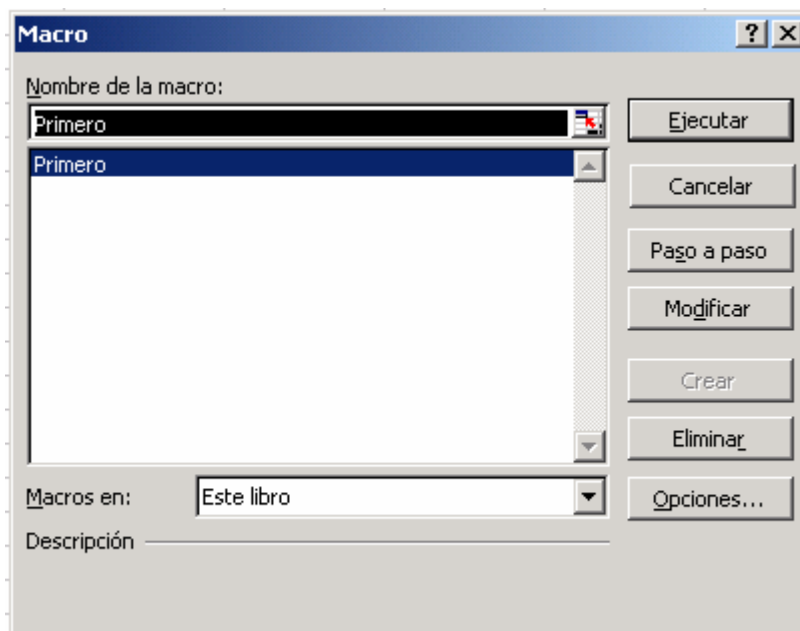
*Para ejecutar el programa anterior, presiona la tecla [F5], o haz clic en el botón izquierdo del ratón en .*

### Para ejecutar el procedimiento desde la hoja de cálculo.

Debe estar en una hoja, no en el editor de Visual Basic

Active opción de la barra de menús “**Herramientas**” – “**Macro**” – “**Macros**”.

A continuación aparecerá una ventana, en la cual se muestra una lista donde están todas las macros incluidas en el libro de trabajo.



Seleccione la macro de la lista y pulse sobre el botón **Ejecutar**.

Para dejar más en claro el funcionamiento de las macros, haremos este segundo ejemplo, el cual escribirá "Hola" en la celda A1, le pondremos en negrita y le daremos color al texto.

Para ello utilizaremos las propiedades **Bold** y **Color** del objeto **Font**.

#### Sub Segundo

```
ActiveSheet.Range("A1").Value = "Hola"  
ActiveSheet.Range("A1").Font.Bold = True
```

```
ActiveSheet.Range("A1").Font.Color = RGB(255,0,0)  
End Sub
```

## True y False

**True**, que traducido es verdadero, simplemente indica que la propiedad **Bold** está activada. Si se deseara desactivar, bastaría con igualarla al valor **False**.

## La función RGB

Observe que para establecer el color de la propiedad se utiliza la función **RGB**(Red, Green, Blue), los tres argumentos para esta función son valores del 0 a 255 que corresponden a la intensidad de los colores Rojo, Verde y Azul respectivamente.

## Seleccionar un rango de celdas (Referenciar)

Sólo tiene que cambiar a la forma **Casilla\_Inicial:Casilla\_Final**. Por ejemplo aplicar el último ejemplo al rango de casillas que va de la A1 a la A8, ponga.

```
Sub Segundo  
ActiveSheet.Range("A1:A8").Value = "Hola"  
ActiveSheet.Range("A1:A8").Font.Bold = True  
ActiveSheet.Range("A1:A8").Font.Color = RGB(255,0,0)  
End Sub.
```

## Variables

A continuación vamos a repetir el programa del primer ejemplo, pero en lugar de poner "Hola" en la casilla A1 de la hoja activa, dejaremos que el usuario entre un texto desde teclado y a continuación guardaremos ese valor en esa casilla. Observe que el valor que entre del usuario debe guardarse en algún lugar para poder ponerlo después en la casilla A1; pues bien, ese valor se guardará en una variable. Una variable es simplemente un trozo de memoria que la función o procedimiento se reserva para guardar datos, la forma general de declarar una variable es:

**DIM variable AS tipo.**

Siendo *variable* el nombre que se asigna a la misma y **Tipo** el tipo de datos que se guardarán (números, texto, fecha, booleanos,...). En nuestro ejemplo, declararemos la variable de tipo **String** (tipo texto), y lo haremos de la forma siguiente.

**Dim Texto As String**

Con esto estamos indicando que se reserve un trozo de memoria (el que sea), que se llama Texto y que el tipo de datos que se guardarán ahí serán caracteres.

## La Función *InputBox*

Esta función muestra una ventana para que el usuario pueda teclear datos. Cuando se pulsa sobre **Aceptar**, los datos entrados pasan a la variable a la que se ha igualado la función. Vea la línea siguiente.

Texto = **InputBox**("Introduzca el texto", "Entrada de datos").

Si en la ventana que muestra **InputBox** pulsa sobre el botón Aceptar, los datos tecleados se guardarán en la variable Texto.

## Sintaxis de InputBox

**InputBox**(Mensaje, Título, Valor por defecto, Posición horizontal, Posición Vertical, Archivo ayuda, Número de contexto para la ayuda).

La explicación de los parámetros utilizados para esta función se explican a continuación:

**Mensaje:** es el mensaje que se muestra en la ventana. Si desea poner más de una línea ponga Chr(13) para cada nueva línea, vea el ejemplo siguiente.

**Título:** es el título para la ventana **InputBox**. Es un parámetro opcional.

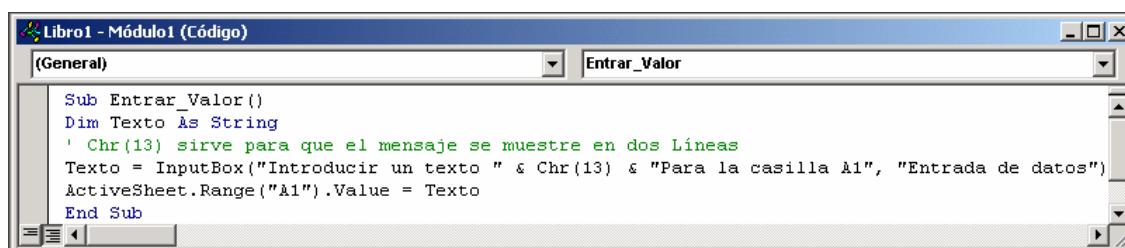
**Valor por defecto:** es el valor que mostrará por defecto el cuadro donde el usuario entra el valor. Parámetro opcional.

**Posición Horizontal:** la posición X de la pantalla donde se mostrará el cuadro, concretamente es la posición para la parte izquierda. Si se omite el cuadro se presenta horizontalmente centrado a la pantalla.

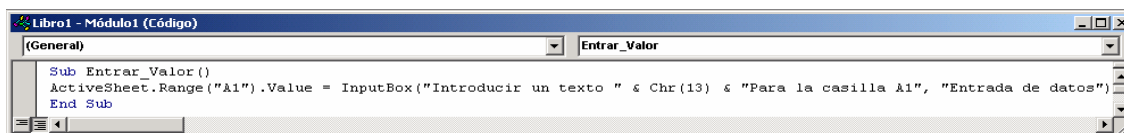
**Posición Vertical:** la posición Y de la pantalla donde se mostrará el cuadro, concretamente es la posición para la parte superior. Si se omite el cuadro se presenta verticalmente centrado a la pantalla.

**Archivo Ayuda:** es el archivo que contiene la ayuda para el cuadro. Parámetro opcional.

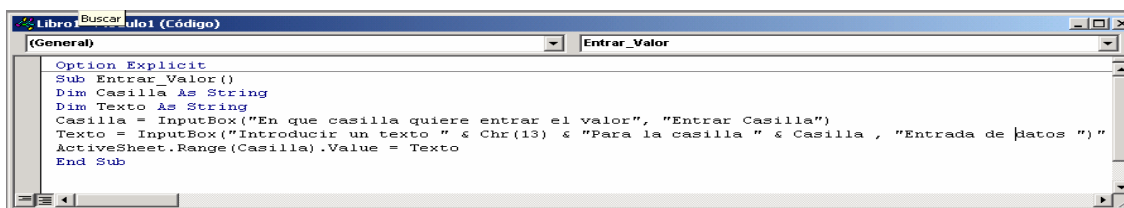
**Número de contexto para la ayuda:** número asignado que corresponde al identificador del archivo de ayuda, sirve para localizar el texto que se debe mostrar. Si se especifica este parámetro, debe especificarse obligatoriamente el parámetro Archivo Ayuda.



Este ejemplo también se puede hacer sin variables.



Ahora, haremos el ejemplo anterior, pero en lugar de entrar los valores sobre la casilla A1, dejaremos que el usuario elija en qué casilla quiere que los datos tecleados aparezcan, es decir, se le preguntará al usuario mediante un segundo Inputbox sobre o en qué casilla quiere entrar el valor del primer Inputbox. Serán necesarias dos variables, una para guardar la casilla que escoja el usuario y otra para guardar el valor.



### La sentencia *Option Explicit*

En visual basic no es necesario declarar las variables, por ejemplo, en el programa anterior se hubiera podido prescindir de las líneas

**Dim Casilla As String**

**Dim Texto As String**

A pesar de ello, recomiendo que siempre declare las variables que va a utilizar, de esta forma sabrá cuáles utiliza en el procedimiento y qué tipo de datos guarda cada una, piense que a medida que vaya aprendiendo, creará procedimientos cada vez más complicados y que requerirán el uso de más variables.

Si no declara las variables al principio del procedimiento ocurrirán dos cosas. Primero, las variables no declaradas son asumidas como tipo **Variant** (éste es un tipo de datos que puede almacenar cualquier valor, número, fechas, texto, etc. pero tenga en cuenta que ocupa 20 Bytes y para guardar una referencia a una casilla, la edad de alguien, etc. no son necesarios tantos bytes); segundo, reducirá considerablemente la legibilidad de sus procedimientos ya que las variables las irá colocando a medida que las necesite, esto, a la larga complicará la corrección o modificación del procedimiento.

La **Option Explicit** al principio del módulo fuerza a que se declaren todas las variables. Si al ejecutar el programa, se encuentra alguna variable sin declarar se producirá un error y no se podrá ejecutar el programa hasta que se declare.

Si todavía no se ha convencido sobre la conveniencia de declarar las variables y utilizar **Option Explicit**, pruebe el procedimiento siguiente, cópielo tal cual (Texto y Testo están puestos adrede simulando que nos hemos equivocado al teclear).

**Sub** Entrar\_Valor

Texto = InputBox("Introducir un texto " & Chr(13) & "Para la casilla A1", "Entrada de datos")

ActiveSheet.Range("A1").Value = Texto

**End Sub**

Observa que el programa no hace lo que se pretendía que hiciera. Efectivamente, Texto y Texto son dos variables diferentes, como no se ha declarado ninguna ni se ha utilizado **Option Explicit**. Visual Basic no da ningún tipo de error y ejecuta el programa. Pruebe el siguiente módulo e intente ejecutarlo.

**Option Explicit**

**Sub** Entrar\_Valor

**Dim** Texto **As String**

Texto = InputBox("Introducir un texto " & Chr(13) & "Para la casilla A1", "Entrada de datos")

ActiveSheet.Range("A1").Value = Texto

**End Sub**

Observe que el programa no se ejecuta, al poner **Option Explicit**, forzamos a que se declaren todas las variables. Visual Basic detecta que la variable *Texto* no ha sido declarada y así lo indica mostrando Error, entonces es más fácil darnos cuenta del error que hemos cometido al teclear y cambiamos Texto por Texto. Ahora imagine que el error se produce en un programa de cientos de líneas que necesita otras tantas variables.

## Tipos de datos en Visual Basic para Excel

Tipo de datos	Tamaño De almacenamiento	Intervalo
<b>Byte</b>	1 byte	0 a 255
<b>Boolean</b>	2 bytes	True o False
<b>Integer</b>	2 bytes	-32.768 a 32.767
<b>Long</b> (entero largo)	4 bytes	-2.147.483.648 a 2.147.483.647
<b>Single</b> (coma flotante/ precisión simple)	4 bytes	-3,402823E38 a -1,401298E-45 para valores negativos; 1,401298E-45 a 3,402823E38 para valores positivos
<b>Double</b> (coma flotante/ precisión doble)	8 bytes	-1,79769313486232E308 a -4,94065645841247E-324 para valores negativos; 4,94065645841247E-324 a 1,79769313486232E308 para valores positivos
<b>Currency</b> (entero a escala)	8 bytes	-922.337.203.685.477,5808 a 922.337.203.685.477,5807
<b>Decimal</b>	14 bytes	+/-79.228.162.514.264.337.593.543.950.335 sin punto decimal; +/-7,9228162514264337593543950335 con 28 posiciones a la derecha del signo decimal; el número más pequeño distinto de cero es +/- 0,000000000000000000000000000001
<b>Date</b>	8 bytes	1 de enero de 100 a 31 de diciembre de 9999
<b>Object</b>	4 bytes	Cualquier referencia a tipo Object
<b>String</b> (longitud variable)	10 bytes + longitud de la cadena	Desde 0 a 2.000 millones
<b>String</b> (longitud fija)	Longitud de la cadena	Desde 1 a 65.400 aproximadamente
<b>Variant</b> (con números)	16 bytes	Cualquier valor numérico hasta el intervalo de un tipo Double
<b>Variant</b> (con caracteres)	22 bytes + longitud de cadena	El mismo intervalo que para un tipo String de longitud variable
<b>Definido por el usuario</b> (utilizando Type)	Número requerido por los elementos	El intervalo de cada elemento es el mismo que el intervalo de su tipo de datos.



## Conversión de Tipos de datos

En muchas ocasiones nos encontramos números que necesitamos convertir en textos o viceversa, para ello es necesario utilizar funciones específicamente diseñadas, para dejar más en claro esto, teclea el siguiente código en el editor de macros de Visual Basic y ejecútalo.

### Option Explicit

**Sub** Sumar()

**Dim** Número1 **As Integer**

**Dim** Número2 **As Integer**

Número1 = InputBox("Entrar el primer valor", "Entrada de datos")

Número2 = InputBox("Entrar el primer valor", "Entrada de datos")

ActiveSheet.Range("A1").Value = Numero1 + Numero2

**End Sub**

Ejecute el procedimiento y ponga respectivamente los valores 25 y 25. Observe que se ha ejecutado correctamente y en la casilla A1 de la hoja activa aparece un 50.

Ahora, vuelva a ejecutar el programa y cuando se le pida el primer valor teclee "Hola". Observe que el programa se detiene indicando un error en el tipo de datos. Efectivamente, observe que la función InputBox devuelve siempre datos tipo **String**, en el primer ejemplo no ha habido ningún problema, al teclear caracteres numéricos, éstos pueden asignarse a variables tipo **Integer** porque Visual Basic hace automáticamente la conversión, pero al teclear texto e intentarlo asignar a una variable **Integer** Visual Basic muestra un error indicando que la variable no es adecuada para los datos que se desean guardar.

Para solucionar estos problemas se deben utilizar funciones de conversión de tipo. Estas funciones, como su nombre indica, convierten datos de un tipo a otro, de **String** a **Integer**, de **Integer** a **String**, de **Date** a **String**, etc. Así el procedimiento anterior quedaría.

### Option Explicit

**Sub** Sumar()

**Dim** Número1 **As Integer**

**Dim** Número2 **As Integer**

Número1 = Val(InputBox("Entrar el primer valor", "Entrada de datos"))

Número2 = Val(InputBox("Entrar el primer valor", "Entrada de datos"))

ActiveSheet.Range("A1").Value = Numero1 + Numero2

**End Sub**

La función **Val**(Dato String), convierte una cadena de texto que representa un número en un número. Si la cadena a convertir contiene algún carácter no numérico devuelve 0. Así, si al pedir un valor se teclea "Hola", la función Val, devolverá un cero.

Ten en cuenta que para las computadoras no es lo mismo el número 1 que el carácter "1". En los lenguajes de programación actuales la conversión de carácter "1" a número 1 se hace automáticamente en muchos casos, esto es bueno y es malo. Es bueno porque nos ahorra tener que hacer las conversiones y es malo porque es más difícil controlar ciertos casos.

Siga con los ejemplos y entenderá de lo que estamos hablando. Sólo para su información, la computadora guarda el número 1 de la siguiente manera 00000001, mientras que el carácter "1" se guarda como 00110000 (el número 48 del código ASCII).

## Funciones de conversión de tipos

**Val(Cadena).** Convierte la cadena a un valor numérico.

**Str(Número).** Convierte el número a una expresión cadena.

Las siguientes funciones tienen la forma **Función** (Expresión).

<b>Función</b>	<b>Tipo devuelto</b>	<b>Intervalo del argumento expresión</b>
<b><i>Cbool</i></b>	<b>Boolean</b>	Cualquier expresión de cadena o numérica válida.
<b><i>Cbyte</i></b>	<b>Byte</b>	0 a 255.
<b><i>Ccur</i></b>	<b>Currency</b>	-922.337.203.685.477,5808 a 922.337.203.685.477,5807.
<b><i>Cdate</i></b>	<b>Date</b>	Cualquier expresión de fecha.
<b><i>CDbl</i></b>	<b>Double</b>	-4,94065645841247E-324 para valores negativos; 4,94065645841247E-324 a 1,79769313486232E308 para valores positivos.
<b><i>Cdec</i></b>	<b>Decimal</b>	+/-7,9228162514264337593543950335. La menor posición para un número que no sea cero es 0,00000000000000000000000001.
<b><i>CInt</i></b>	<b>Integer</b>	-32.768 a 32.767; las fracciones se redondean.
<b><i>CLng</i></b>	<b>Long</b>	-2.147.483.648 a 2.147.483.647; las fracciones se redondean.
<b><i>CSng</i></b>	<b>Single</b>	-3,402823E38 a -1,401298E-45 para valores negativos; 1,401298E-45 a 3,402823E38 para valores positivos.
<b><i>CVar</i></b>	<b>Variant</b>	El mismo intervalo que Double para valores numéricos. El mismo intervalo que String para valores no numéricos.
<b><i>CStr</i></b>	<b>String</b>	El valor de retorno de CStr depende del argumento expresión.

A continuación veremos algunas funciones y objetos que se utilizan habitualmente en la programación de Excel con el lenguaje Visual Basic

### Objeto Cells(fila, columna)

Sirve, como el objeto range, para referenciar una casilla o rango de casillas, pero en lugar de utilizar la referencia de la forma A1, B1, X320,... utiliza la fila y la columna que ocupa la casilla dentro de la hoja (objeto WorkSheet). Por ejemplo, para poner hola en la casilla A1 de la hoja activa sería, `ActiveSheet.Cells(1,1).Value="Hola"`

## Utilizar Cells para referenciar un rango

Esto sería el equivalente a `Range("Casilla_Inicial:Casilla_Final")`. La forma que se obtiene utilizando `Cells` es un poco más larga, pero se verá que a veces resulta mucho más funcional que utilizando únicamente `range`. Para referirnos al rango A1:B8, pondremos, `Range(Cells(1, 1), Cells(8, 2)).Value = "Hola"`.

Otra forma interesante de Cells es la siguiente, Range("A5:B10").Cells(2, 1).Value = "Hola".

Pondrá en la celda A6 el valor "Hola", observe que en este ejemplo Cells comienza a contar filas y columnas a partir del rango especificado en el objeto Range.

### ***Variables de Objetos***

Una variable objeto sirve para hacer referencia a un objeto, esto significa que podremos acceder a las propiedades de un objeto e invocar a sus métodos a través de la variable en lugar de hacerlo directamente a través del objeto. Posiblemente no se utilice demasiado esta clase de variables (está claro que esto dependerá de las preferencias del programador), pero hay casos en los que no hay más remedio que utilizarlas, por ejemplo en estructuras **For Each ... Next** como veremos, o cuando sea necesario construir funciones que devuelvan rangos, referencias a hojas, etc.

Para declarar una variable objeto se utiliza también la palabra **Dim** de la forma siguiente:

**Dim** Var\_Objeto **As** Objeto

Por Ejemplo:

**Dim** R **As** Range  
**Dim** Hoja **As** WorkSheet

Para asignar un objeto a una variable debes utilizar la instrucción **Set**.

**Set** Variable\_Objeto = Objeto

Por Ejemplo:

**Set** R= ActiveSheet.Range("A1:B10")  
**Set** Hoja = ActiveSheet  
**Set** Hoja = WorkSheets(1)

Veamos a continuación un ejemplo de cómo utilizar este tipo de variables.

Algo muy simple, llenar el rango de celdas de la A1 a la B10 con la palabra "Hola" y después poner negrita, observe cómo se asigna una variable objeto al objeto y luego cómo se trabaja con esa variable de la misma forma que trabajaría directamente sobre el objeto.

**Sub** obj()  
**Dim** R **As** Range  
**Set** R = ActiveSheet.Range("A10:B15")  
R.Value = "Hola"

R.Font.Bold = True  
**End Sub**

## El valor **Nothing**

Algunas veces puede que sea necesario desasignar una variable del objeto al cual hace referencia, en este caso debe igualar la variable al valor **Nothing** de la forma siguiente.

**Set** Variable\_Objeto = **Nothing**

Habitualmente se utiliza **Nothing** en una estructura condicional para comprobar si la variable objeto está asignada. Observa que si se utiliza una variable objeto a la cual todavía no se le ha hecho ninguna asignación el programa dará error y detendrá su ejecución. Es buena práctica hacer este tipo de comprobaciones antes de trabajar con variables objeto. Veremos un ejemplo de esto en el tema siguiente.

## Estructuras condicionales

### Estructura **If**

Ahora que ya has experimentado con unos cuantos objetos y propiedades, nos detendremos a estudiar las estructuras condicionales. Las estructuras condicionales son instrucciones de programación que permiten controlar la ejecución de un fragmento de código en función de si se cumple o no una condición.

Estudiaremos en primer lugar la instrucción **if** Condición **then . else . End if** (**Si** Condición **Entonces...Si Fin**) La estructura condicional que se construye con la instrucción **Si** Condición **Entonces... Fin Si** tiene la forma siguiente:

**Si** Condición **Entonces**  
Sentencia1  
Sentencia2  
.  
.  
SentenciaN  
**Fin Si**

Cuando el programa llega a la instrucción **Si** Condición **Entonces**, se evalúa la condición, si ésta se cumple (es cierta), se ejecutan las sentencias que están encerradas en el bloque (Sentencia1, Sentencia2, ..., SentenciaN), en caso contrario se ejecutan las sentencias que están después del **else** (en este caso son Zentencia1, Zentencia2, ..., ZentenciaN que en este caso son de, si no se cumple la condición, se saltan estas sentencias. En el office 97, las instrucciones para Visual Basic pueden ser en inglés o en español, generalmente se puede cambiar de la sintaxis del lenguaje en español a inglés y viceversa, en el Office2000 y XP, la conversión es automática.

La estructura en Visual Basic tiene la sintaxis de la condicional If siguiente:

**If Condición Then**

Sentencia1

Sentencia2

.

SentenciaN

**else**

Zentencia1

Zentencia2

.

ZentenciaN

**End If**

Para clarificar esta instrucción, haremos un programa, el cual pedirá una cantidad que representa el precio de algo por el teclado con la instrucción **InputBox** y guardarlo en la celda A1 de la hoja activa. Si el valor entrado desde el teclado (y guardado en A1) es superior a 1000, pedir descuento con otro InputBox y guardarlo en la casilla A2 de la hoja activa. Calcular en A3, el precio de A1 menos el descuento de A2.

**Sub** Condicional()

ActiveSheet.Range("A1").Value = 0 ' Poner las casillas donde se guardan los valores 0.

ActiveSheet.Range("A2").Value = 0

ActiveSheet.Range("A3").Value = 0

ActiveSheet.Range("A1").Value = Val(InputBox("Entrar el precio", "Entrar"))

' Si el valor de la casilla A1 es mayor que 1000, entonces, pedir descuento

**If** ActiveSheet.Range("A1").Value > 1000 **Then**

ActiveSheet.Range("A2").Value = Val(InputBox("Entrar Descuento", "Entrar"))

**End If**

ActiveSheet.Range("A3").Value = ActiveSheet.Range("A1").Value - \_

ActiveSheet.Range("A2").Value

**End Sub.**

El mismo que el anterior pero utilizando variables.

**Option Explicit**

**Sub** Condicional()

**Dim** Precio **As Integer**

**Dim** Descuento **As Integer**

Precio = 0

Descuento = 0

Precio = Val(InputBox("Entrar el precio", "Entrar"))

' Si el valor de la variable precio es mayor que 1000, entonces, pedir descuento

**If** Precio > 1000 **Then**

Descuento = Val(InputBox("Entrar Descuento", "Entrar"))

**End If**

ActiveSheet.Range("A1").Value = Precio

ActiveSheet.Range("A2").Value = Descuento

ActiveSheet.Range("A3").Value = Precio - Descuento

**End Sub**

Viendo los dos programas anteriores puede que le surja la duda de si emplear variables o directamente valores almacenados en las celdas. La solución es fácil, lo que le parezca más conveniente en cada caso concreto que desee solucionar. Las variables, aunque muchas veces "innecesarias", quizás dejan los programas más legibles y claros, y la legibilidad de un programa es lo más valioso del mundo para un programador, sobre todo si se da el caso (inevitable el 99,999...% de las ocasiones) que se tenga que modificar un programa para dotarle de más funcionalidades, facilitar su manejo, etc.

En la mayoría de los ejemplos que encontrará en este manual verá que se utilizan variables preferentemente, pues como he programado en diferentes lenguajes de programación, se me hace más normal trabajar con variables que con celdas. Aunque muchas veces su función sea simplemente recoger datos de las celdas para operarlas y dejarlas en otras celdas y, consecuentemente, aumente el número de operaciones, creo que con ello se gana en legibilidad y flexibilidad.

A continuación haremos un programa que compara los valores de las casillas A1 y A2 de la hoja activa. Si son iguales pone el color de la fuente de ambas en azul.

```
Sub Condicional2()  
If ActiveSheet.Range("A1").Value = ActiveSheet.Range("A2").Value Then  
    ActiveSheet.Range("A1").Font.Color = RGB(0, 0, 255)  
    ActiveSheet.Range("A2").Font.Color = RGB(0, 0, 255)  
End If  
End Sub.
```

Hasta el momento, sólo hemos visto la Condicional If con la primera parte de las sentencias, es decir, ejecuta las sentencias cuando la pregunta de la condición es verdadera, si es falsa, no se ejecuta nada, ahora acabaremos de explicar la instrucción If para el caso en que sea verdadera la condición y también cuando sea falsa la condición que se está evaluando, pero escribiremos la sintaxis en español.

Observa que, si se cumple la condición (la condición es verdadera), se ejecuta el bloque de sentencias delimitado por **If** Condición **Then** y Si no se cumple la condición (la condición es falsa) se ejecuta el bloque delimitado por **Else** y **End If**.

Para clarificar el uso de la condicional If, haremos el siguiente ejemplo, en el cual introducimos una cantidad que representa el precio de algo por el teclado con la instrucción **InputBox** y guardarlo en la celda A1 de la hoja activa. Si el valor entrado desde el teclado (y guardado en A1) es superior a 1000, se aplica un descuento del 10% o si no, se aplica un descuento del 5%, el descuento se guarda en la casilla A2 de la hoja activa. Colocar en A3, el total descuento y en A4 el total menos el descuento.

```
Sub Condicional_Else()  
Dim Precio As Single  
Dim Descuento As Single  
Precio = 0
```

```
Precio = Val(InputBox("Entrar el precio", "Entrar"))
' Si el valor de la variable precio es mayor que 1000, entonces, aplicar descuento del 10%
If Precio > 1000 Then
Descuento = Precio * (10 / 100)
ActiveSheet.Range("A2").Value = 0,1
Else ' Si no aplicar descuento del 5%
Descuento = Precio * (5 / 100)
ActiveSheet.Range("A2").Value = 0,05
End If
ActiveSheet.Range("A1").Value = Precio
ActiveSheet.Range("A3").Value = Descuento
ActiveSheet.Range("A4").Value = Precio - Descuento
End Sub
```

Restar los valores de las casillas A1 y A2. Guardar el resultado en A3. Si el resultado es positivo o 0, poner la fuente de A3 en azul, si no, ponerla en rojo.

```
Sub Condicional_Else2()
ActiveSheet.Range("A3").Value = ActiveSheet.Range("A1").Value - _
ActiveSheet.Range("A2").Value
If ActiveSheet("A3").Value < 0 Then
ActiveSheet.Range("A3").Font.Color = RGB(255,0,0)
Else
ActiveSheet.Range("A3").Font.Color = RGB(0,0,255)
End If
End Sub
```

## Estructuras If anidadas

Dentro de una estructura If puede ir otra, y dentro de esta otra, y así sucesivamente; para mostrar esto, hagamos el siguiente ejercicio:

Comparar los valores de las casillas A1 y A2 de la hoja activa. Si son iguales, escribir en A3 "Los valores de A1 y A2 son iguales", si el valor de A1 es mayor que A2, escribir "A1 mayor que A2", si no, escribir "A2 mayor que A1" ..

```
Sub Condicional()
If ActiveSheet.Range("A1").Value = ActiveSheet.Range("A2").Value Then
ActiveSheet.Range("A3").Value = "Los Valores de A1 y A2 son iguales"
Else
If ActiveSheet.Range("A1").Value > ActiveSheet.Range("A2").Value Then
ActiveSheet.Range("A3").Value = "A1 mayor que A2"
Else
ActiveSheet.Range("A3").Value = "A2 mayor que A1"
End If
End If
End Sub
```

Observe que la segunda estructura **If..Else..End If** queda dentro del **Else** de la primera estructura. Esta es una regla general, cuando pone un **End If**, éste cierra siempre el último **If** ( o **Else**) abierto.

## Operadores lógicos

Cuando se necesitan evaluar dos o más condiciones se utilizan estos operadores, los cuales permiten evaluar toda la condición y así decidir si se ejecutan o no determinadas acciones. Para dejar en claro lo que son los operadores lógicos, debemos de saber qué son las tablas de verdad.

**Las Tablas de verdad se utilizan en lógica y nos permite deducir un razonamiento a partir de precisas que en computación llamaremos condiciones.**

**A continuación mostraremos las tablas de verdad para los operadores lógicos de Visual Basic.**

condición1	condición2	condición1 AND condición 2	condición1 OR condición 2	condición1 XOR condición 2	NOT (condición1)
FALSO	FALSO	FALSO	FALSO	FALSO	VERDADERO
FALSO	VERDADERO	FALSO	VERDADERO	VERDADERO	VERDADERO
VERDADERO	FALSO	FALSO	VERDADERO	VERDADERO	FALSO
VERDADERO	VERDADERO	VERDADERO	VERDADERO	FALSO	FALSO

## Operador Lógico And (Y)

Utilizaremos este operador cuando sea preciso que para ejecutar un bloque de instrucciones se cumpla más de una condición. Observa que deberán cumplirse todas las condiciones.

Para mostrar el operador lógico AND, supongamos que tecleamos el nombre, la cantidad y el precio de un producto desde el teclado y guardarlos respectivamente en A1, A2 y A3. Calcular el total y guardarlo en A4. Si el total es superior a 10,000 y el nombre del producto es "Papas", pediremos un descuento, calcular el total descuento y guardarlo en A5, luego restar el descuento del total y guardarlo en A6.

```

Sub Ejemplo_and()
Dim Producto As String
Dim Cantidad As Integer
Dim Precio As Single
Dim Total As Single
Dim Descuento As Single
Dim Total_Descuento As Single
Precio = 0
Producto = InputBox("Entrar Nombre del Producto", "Entrar")
Precio = Val(InputBox("Entrar el precio", "Entrar"))
Cantidad = Val(InputBox("Entrar la cantidad", "Entrar"))
Total = Precio * Cantidad
ActiveSheet.Range("A1").Value = Producto
ActiveSheet.Range("A2").Value = Precio
ActiveSheet.Range("A3").Value = Cantidad

```



```
ActiveSheet.Range("A4").Value = Total.  
' Si total mayor que 10,000 y el producto es Papas, aplicar descuento.  
If Total > 10000 And Producto = "Papas" Then  
Descuento = Val(InputBox("Entrar Descuento", "Entrar"))  
Total_Descuento = Total * (Descuento / 100)  
Total = Total - Total_Descuento  
ActiveSheet.Range("A5").Value = Total_Descuento  
ActiveSheet.Range("A6").Value = Total  
End If  
End Sub
```

Observe que para que se ejecute el bloque de instrucciones entre If End If deben cumplirse las dos condiciones que se evalúan, si falla cualquiera de las dos (o las dos a la vez), no se ejecuta dicho bloque.

### Operador Lógico Or (O)

Utilizaremos este operador cuando sea preciso que para ejecutar un bloque de instrucciones se cumpla alguna de una serie de condiciones. Observe que sólo es necesario que se cumpla alguna de las condiciones que se evalúan. Vea el ejemplo siguiente:

Introducir el Nombre, la cantidad y el precio de un producto desde el teclado y guardarlos respectivamente en A1, A2 y A3. Calcular el total y guardarlo en A4. Si el total es superior a 10.000 o el nombre del producto es "Papas", pedir un descuento, calcular el total del descuento y guardarlo en A5, luego restar el descuento del total y guardarlo en A6.

```
Sub Ejemplo_13()  
Dim Producto As String  
Dim Cantidad As Integer  
Dim Precio As Single  
Dim Total As Single  
Dim Descuento As Single  
Dim Total_Descuento As Single  
Precio = 0  
Producto = InputBox("Entrar Nombre del Producto", "Entrar")  
Precio = Val(InputBox("Entrar el precio", "Entrar"))  
Cantidad = Val(InputBox("Entrar la cantidad", "Entrar"))  
Total = Precio * Cantidad  
ActiveSheet.Range("A1").Value = Producto  
ActiveSheet.Range("A2").Value = Precio  
ActiveSheet.Range("A3").Value = Cantidad  
ActiveSheet.Range("A4").Value = Total  
' Si total mayor que 10.000 o el producto es Papas, aplicar descuento.  
If Total > 10000 Or Producto = "Papas" Then  
Descuento = Val(InputBox("Entrar Descuento", "Entrar"))  
Total_Descuento = Total * (Descuento / 100)  
Total = Total - Total_Descuento  
ActiveSheet.Range("A5").Value = Total_Descuento  
ActiveSheet.Range("A6").Value = Total  
End If  
End Sub.
```

Observe que para que se ejecute el bloque de instrucciones entre If.. End If sólo es necesario que se cumpla alguna de las dos condiciones que se evalúan (o las dos a la vez). Sólo cuando no se cumple ninguna de las dos no se ejecutan las instrucciones del bloque.

### Operador Lógico Not (no)

Este operador se utiliza para ver si NO se cumple una condición. El siguiente ejemplo hace lo mismo que el ejemplo en el que utiliza el operador and, pero utilizando el operador Not.

Introducir una cantidad que representa el precio de algo por el teclado con la instrucción InputBox y guardarlo en la celda A1 de la hoja activa. Si el valor entrado desde el teclado (y guardado en A1) es superior a 1000, pedir descuento con otro InputBox y guardarlo en la casilla A2 de la hoja activa. Calcular en A3, el precio de A1 menos el descuento de A2.

```
Sub Ejemplo_not()  
Dim Precio As Integer  
Dim Descuento As Integer  
Precio = 0  
Descuento = 0  
Precio = Val(InputBox("Entrar el precio", "Entrar"))  
' Si el valor de la variable precio NO es menor igual 1000, es decir, el precio es mayor que 1000,  
entonces, pedir descuento  
If Not (Precio <= 1000) Then  
Descuento = Val(InputBox("Entrar Descuento", "Entrar"))  
End If  
ActiveSheet.Range("A1").Value = Precio  
ActiveSheet.Range("A2").Value = Descuento  
ActiveSheet.Range("A3").Value = Precio - Descuento  
End Sub
```

### Operador Lógico XOR

Este operador no existe como tal en la lógica matemática, pero es muy utilizado en computación, se utiliza mucho en gráficos, entre otros usos, y sólo es verdadero si las 2 condiciones son diferentes, es decir, una de las 2 debe ser falsa y la otra verdadera.

### Estructura Select Case

En ocasiones se dará el caso que se requiere hacer varias preguntas acerca de un valor que se tiene en una variable o en una casilla, y la forma de hacerlo es preguntar varias veces con la pregunta IF, dependiendo del valor que se tenga, se pueden tener varias opciones.

Por ejemplo hacer una Macro que suma, resta, multiplica o divide los valores de las casillas A1 y A2 dependiendo de si B1 contiene el signo +, -, x, :. El resultado lo deja en A3. Si en B1 no hay ninguno de los signos anteriores en A3 debe dejarse un 0.

```
Sub Ejemplo_select1()
```

```
Dim Signo As String
Dim Valor1 As Integer, Valor2 As Integer, Total As Integer
Valor1 = ActiveSheet.Range("A1").Value
Valor2 = ActiveSheet.Range("A2").Value
Signo = ActiveSheet.Range("B1").Value
Total=0
If Signo = "+" Then
    Total = Valor1 + Valor2
End if
If Signo = "-" Then
    Total = Valor1 - Valor2
End if
If Signo = "x" Then
    Total = Valor1 * Valor2
End if
If Signo = ":" Then
    Total = Valor1 / Valor2
End if
ActiveCell.Range("A3").Value = Total
End Sub
```

Observe que en el ejemplo anterior todas las instrucciones If evalúan la misma variable. El programa funciona correctamente pero para estos casos es mejor utilizar la instrucción Select Case, el motivo principal es por legibilidad y elegancia. Select Case tiene la sintaxis siguiente:

```
Select Case Expresión
Case valores:
    Instrucciones.
Case valores:
    Instrucciones.
.
.
Case valores:
    Instrucciones.
Case Else
```

Instrucciones en caso que no sean ninguno de los valores anteriores.

**End Select**

Este ejemplo es el mismo que el anterior, pero en vez de utilizar If, se utiliza la instrucción Select ... Case

```
Sub Ejemplo_select2()
Dim Signo As String
Dim Valor1 As Integer, Valor2 As Integer, Total As Integer
Valor1 = ActiveSheet.Range("A1").Value
Valor2 = ActiveSheet.Range("A2").Value
Signo = ActiveSheet.Range("A3").Value
```

```
Select Case signo
Case "+"
Total = Valor1 + Valor2
Case "-"
Total = Valor1 - Valor2
Case "x"
Total = Valor1 * Valor2
Case ":"
Total = Valor1 / Valor2
Case Else
Total = 0
End Select
ActiveCell.Range("A3").Value = Total
End Sub
```

Como vimos anteriormente, la instrucción Select...Case es una versión más fácil de usar el If...Then, por esto, el case puede evaluar igualmente 2 o más condiciones que se necesiten para ejecutar una parte del programa, a continuación haremos un programa que pida tres calificaciones de un alumno mediante la función InputBox. Las calificaciones aparecerán de la celda A1 a la A3 respectivamente. El programa calcula la media y la deja en A4. Si la media está entre 0 y 2 deja en A5 el mensaje "Muy deficiente", si la nota es 3 deja en A5 el mensaje "Deficiente", si la nota es 4 deja "Insuficiente", si es 5 "Suficiente", si es 6 "Bien", si está entre 7 y 8 deja "Muy Bien", si es mayor que 8 deja "Excelente".

```
Sub Ejemplo_select2()
Dim Nota1 As Integer, Nota2 As Integer, Nota3 As Integer, Dim Media As Single
Nota1 = Val(InputBox("Entrar Nota primera evaluación", "Nota"))
Nota2 = Val(InputBox("Entrar Nota Segunda evaluación", "Nota"))
Nota3 = Val(InputBox("Entrar Nota Tercera evaluación", "Nota"))
Media = (Nota1 + Nota2 + Nota3) / 3
ActiveSheet.Range("A1").Value = Nota1
ActiveSheet.Range("A2").Value = Nota2
ActiveSheet.Range("A3").Value = Nota3
ActiveSheet.Range("A4").Value = Media.
Select Case Media
Case 0 To 2
ActiveSheet.Range("A5").Value = "Muy deficiente"
Case 3
ActiveSheet.Range("A5").Value = "Deficiente"
Case 4
ActiveSheet.Range("A5").Value = "Insuficiente"
Case 5
ActiveSheet.Range("A5").Value = "Suficiente"
Case 6
ActiveSheet.Range("A5").Value = "Bien"
Case 7 To 8
ActiveSheet.Range("A5").Value = "Muy bien"
Case >8
ActiveSheet.Range("A5").Value = "Excelente"
End Select
End Sub
```

## Funciones de comprobación

Antes de terminar con el tema de condicionales veremos unas funciones que nos serán útiles a la hora de comprobar o validar el tipo de los datos introducidos desde el teclado o simplemente los datos contenidos en una casilla.

Volvamos al ejemplo que codificamos de la manera siguiente:

```
Sub Ejemplo_solver2()  
Dim Signo As String  
Dim Valor1 As Integer, Valor2 As Integer, Total As Integer  
Valor1 = ActiveSheet.Range("A1").Value  
Valor2 = ActiveSheet.Range("A2").Value  
Signo = ActiveSheet.Range("A3").Value  
Select Case signo  
Case "+"  
Total = Valor1 + Valor2  
Case "-"  
Total = Valor1 - Valor2  
Case "x"  
Total = Valor1 * Valor2  
Case ":"  
Total = Valor1 / Valor2  
Case Else  
Total = 0  
End Select  
ActiveCell.Range("A3").Value = Total  
End Sub.
```

Imagine que en alguna de las casillas que se deben operar no hubiera ningún valor o bien datos alfanuméricos. Al ejecutar la macro se producirá un error. Aunque con Visual Basic se puede controlar el flujo del programa cuando se produce un error imprevisto, para solucionar este problema utilizaremos una función de comprobación que nos diga si en las casillas A1 y A2 hay valores adecuados (numéricos) para proseguir con la ejecución de la macro, en caso contrario se mostrará un error y no se ejecutará ninguna de las operaciones.

La función que utilizaremos es **IsNumeric**(expresión), esta función devuelve un valor **True** si la expresión que se evalúa es un valor numérico, en caso contrario devuelve **False**. Observa cómo quedaría el programa. También se utiliza la función **IsEmpty** para comprobar si en B1 hay algo, **IsEmpty**(Expresión) evalúa si expresión está vacía, devuelve **True** si es así y **False** en caso contrario.

```
Sub Ejemplo_select3()  
Dim Signo As String  
Dim Valor1 As Integer, Valor2 As Integer, Total As Integer  
Dim Continuar As Boolean  
Valor1 = ActiveSheet.Range("A1").Value  
Valor2 = ActiveSheet.Range("A2").Value  
Signo = ActiveSheet.Range("B1").Value  
Continuar = True
```

```

' Si en la casilla A1 no hay un valor numérico
If Not IsNumeric(ActiveSheet.Range("A1")) Then
    MsgBox Prompt:="En la casilla A1 no hay ningún valor numérico", Title:="ERROR"
    Continuar= False
End If
' Si en la casilla A2 no hay un valor numérico
If not IsNumeric(ActiveSheet.Range("A2")) Then
    MsgBox Prompt:="En la casilla A2 no hay ningún valor numérico", Title:="ERROR"
    Continuar= False
End If
If IsEmpty(ActiveSheet.Range("B1")) Then
    MsgBox Prompt:="la casilla B1 está vacía", Title:="ERROR"
    Continuar= False
End If
If Continuar Then
    Select Case signo
        Case "+"
            Total = Valor1 + Valor2
        Case "-"
            Total = Valor1 - Valor2
        Case "x"
            Total = Valor1 * Valor2
        Case ":"
            Total = Valor1 / Valor2
        Case Else
            Total = 0
    End Select.
    ActiveCell.Range("A3").Value = Total
End if
End Sub

```

En lugar de los tres If de comprobación se hubiera podido utilizar el operador OR de la manera siguiente:

```

If not IsNumeric(ActiveSheet.Range("A1")) Or not IsNumeric(ActiveSheet.Range("A2")) _
Or IsEmpty(ActiveSheet.Range("B1")) Then
    MsgBox Prompt:="Debe entrar números en A1 y A2 y un signo (+,-,x, :) en B1,
    Title:="ERROR"
Else
    ' Instrucciones de las operaciones
    .....
End if

```

## Lista de Funciones de Comprobación

<b>IsNuméric</b> (Expresión).	Comprueba si expresión tiene un valor que se puede interpretar como numérico.
<b>IsDate</b> (Expresión).	Comprueba si expresión tiene un valor que se puede interpretar como tipo fecha.
<b>IsEmpty</b> (Expresión).	Comprueba que expresión tenga algún valor, que se haya inicializado.
<b>IsError</b> (Expresión).	Comprueba si expresión devuelve algún valor de error.

<b>IsArray</b> (Expresión).	Comprueba si expresión (una variable) es un array o no.
<b>IsObject</b> (Expresión).	Comprueba si expresión (una variable) representa una variable tipo objeto.
<b>IsNull</b> (Expresión).	Comprueba si expresión contiene un valor nulo debido a datos no válidos.
<b>Nothing</b> .	No es propiamente una función, sirve para comprobar si una variable objeto está asociada a un objeto antes de hacer cualquier operación con ella. Recuerde que para trabajar con una variable objeto antes debe asignarse a uno (mediante la instrucción <b>Set</b> ), en caso contrario se producirá un error en el programa cuando utilice el objeto y se detendrá su ejecución.

```

Sub Obj()
Dim R As Range
.
.
.
' Si la variable R es Nothing es que no ha sido asignada, no se puede trabajar con ella
If R Is Nothing Then
    MsgBox Prompt := "La variable Objeto no ha sido asignada", Buttons:=vbOk, _
    Title := "Error"
Else
    R.Value = "Hola"
End If
.
.
End Sub.

```

## La función MsgBox

Esta función muestra un mensaje en un cuadro de diálogo hasta que el usuario pulse un botón. La función devuelve un dato tipo Integer en función del botón pulsado por el usuario. A la hora de invocar esta función, se permiten diferentes tipos de botones.

## Sintaxis de MsgBox

**MsgBox**( Mensaje, Botones, Título, Archivo de ayuda, contexto)

- Mensaje:** Obligatorio, es el mensaje que se muestra dentro del cuadro de diálogo.
- Botones:** Opcional. Es un número o una suma de números o constantes (vea tabla Valores para botones e Iconos), que sirve para mostrar determinados botones e iconos dentro del cuadro de diálogo. Si se omite este argumento asume valor 0 que corresponde a un único Botón OK (ver la tabla que se muestra a continuación).
- Título:** Opcional. Es el texto que se mostrará en la barra del título del cuadro de diálogo.

**Archivo de Ayuda:** Opcional. Si ha asignado un texto de ayuda al cuadro de diálogo, aquí debe especificar el nombre del archivo de ayuda donde está el texto.

**Context:** Opcional. Es el número que sirve para identificar el texto al tema de ayuda correspondiente que estará contenido en el archivo especificado en el parámetro Archivo de Ayuda.

**Tabla de botones e iconos de la ventana de MsgBox**

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.
VbRetryCancel	5	Muestra los botones Reintentar y Cancelar.
VbCritical	16	Muestra el icono de mensaje crítico.
VbQuestion	32	Muestra el icono de pregunta de advertencia.
VbExclamation	48	Muestra el icono de mensaje de advertencia.
VbInformation	64	Muestra el icono de mensaje de información.
VbDefaultButton1	0	El primer botón es el predeterminado.
VbDefaultButton2	256	El segundo botón es el predeterminado.
VbDefaultButton3	512	El tercer botón es el predeterminado.
VbDefaultButton4	768	El cuarto botón es el predeterminado.
VbApplicationModal	0	Aplicación modal; el usuario debe responder al cuadro de mensajes antes de poder seguir trabajando en la aplicación actual.
VbSystemModal	4096	Sistema modal; se suspenden todas las aplicaciones hasta que el usuario responda al cuadro de mensajes.

- Primer grupo de valores (0 a 5) describe el número y el tipo de los botones mostrados en el cuadro de diálogo;
- Segundo grupo (16, 32, 48, 64) describe el estilo del icono;
- Tercer grupo (0, 256, 512) determina el botón predeterminado y el cuarto grupo (0, 4096) determina la modalidad del cuadro de mensajes. Cuando se suman números para obtener el valor final del argumento buttons, se utiliza solamente un número de cada grupo.

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.

Los valores que puede devolver la función msgbox en función del botón que pulse el usuario se muestran en la tabla siguiente:



Constante	Valor	Descripción
VbOK	1	Aceptar
VbCancel	2	Cancelar
VbAbort	3	Anular
VbRetry	4	Reintentar
VbIgnore	5	Ignorar
VbYes	6	Sí
VbNo	7	No

Para clarificar el uso de la función MsgBox, haremos el ejercicio que se muestra a continuación:

```

Sub mensaje()
.
.
' El cuadro Muestra los botones Si y No y un icono en forma de interrogante. Cuando se pulsa
' un botón, el valor lo recoge la variable X. En este caso los valores devueltos pueden ser 6 ó 7
' que corresponden respectivamente a las constantes VbYes y VbNo, observe la instrucción If de
' después.
X = MsgBox("Desea Continuar", vbYesNo + vbQuestion, "Opción",,)
' Se ha pulsado sobre botón Si
If X = vbYes Then
.
.
Else ' Se ha pulsado sobre botón No
.
.
End If
.
.
End Sub

```

Algunas veces puede que le interese simplemente desplegar un cuadro MsgBox para mostrar un mensaje al usuario sin que se requiera recoger ningún valor. En este caso puede optar por la forma siguiente:

**MsgBox Prompt:**"Hola usuaria, Ha acabado el proceso", **Buttons:**=VbOkOnLy, **Title:**"Mensaje"

Lo que no puede hacer porque Visual Basic daría error es poner la primera forma sin igualarla a ninguna variable. Por ejemplo, la expresión siguiente es incorrecta:

**MsgBox** ("Hola usuario, Ha acabado el proceso", VbOkOnly, "Mensaje")

Sería correcto poner

**X= MsgBox** ("Hola usuario, Ha acabado el proceso", VbOkOnly, "Mensaje")

En este caso, aunque X reciba un valor, luego no se utiliza para nada, es decir simplemente se pone para que Visual Basic dé error.

## La instrucción With

Suponemos que llegado a este punto le parecerá engorroso tener que referirse a los objetos siguiendo toda o casi toda la jerarquía. Ya hemos indicado que es mejor hacerlo de esta manera porque el programa gana en claridad y elegancia y, consecuentemente, el programador gana tiempo a la hora de hacer modificaciones o actualizaciones. La sentencia **With** te ayudará a tener que escribir menos código sin que por esto el programa pierda en claridad. Concretamente esta función sirve para ejecutar una serie de acciones sobre un mismo Objeto. Su sintaxis es la siguiente:

**With** Objeto  
Instrucciones  
**End With**

Para ver el uso del **With**, haremos el ejemplo que hicimos anteriormente en el cual introducimos el Nombre, la cantidad y el precio de un producto desde el teclado y guardarlos respectivamente en A1, A2 y A3, Calculamos el total y guardarlo en A4. Si el total es superior a 10,000 o el nombre del producto es "Papas", pedir un descuento, calcular el total descuento y guardarlo en A5, luego restar el descuento del total y guardarlo en A6. Observa cómo con **With** se hace referencia al objeto **ActiveSheet**.

```
Sub Ejemplo_with()  
Dim Producto As String  
Dim Cantidad As Integer  
Dim Precio As Single  
Dim Total As Single  
Dim Descuento As Single  
Dim Total_Descuento As Single  
Precio = 0  
Producto = InputBox("Entrar Nombre del Producto","Introducir")  
Precio = Val(InputBox("Entrar el precio", " Introducir "))  
Cantidad = Val(InputBox("Entrar la cantidad", " Introducir "))  
Total = Precio * Cantidad  
With ActiveSheet  
    .Range("A1").Value = Producto  
    .Range("A2").Value = Precio  
    .Range("A3").Value = Cantidad  
    .Range("A4").Value = Total  
End With  
' Si total mayor que 10.000 o el producto es Papas, aplicar descuento.  
If Total > 10000 Or Producto = "Papas" Then  
    Descuento = Val(InputBox("Entrar Descuento", "Entrar"))  
    Total_Descuento = Total * (Descuento / 100)  
    Total = Total - Total_Descuento  
    With ActiveSheet  
        .Range("A5").Value = Total_Descuento  
        .Range("A6").Value = Total  
    End With  
End If  
End Sub.
```

Como puedes ver, en vez de poner varias veces

```
ActiveSheet.Range("A1").Value = Producto
ActiveSheet.Range("A2").Value = Precio
ActiveSheet.Range("A3").Value = Cantidad
ActiveSheet.Range("A4").Value = Total
```

Se utiliza la estructura With como se ve observa a continuación:

```
With ActiveSheet
    .Range("A1").Value = Producto
    .Range("A2").Value = Precio
    .Range("A3").Value = Cantidad
    .Range("A4").Value = Total
End With
```

## Estructuras Repetitivas (ciclos)

Este tipo de estructuras permiten ejecutar más de una vez un mismo bloque de sentencias. Por ejemplo si quisiéramos hacer un programa para guardar las calificaciones de 5 alumnos en las celdas de A1 a A5 y calcular su media y el resultarlo ponerlo en A6, haremos el siguiente programa:

```
Sub Ejemplo_ciclo1()
Dim Nota As Integer
Dim Media As Single
Media = 0

Calificación = Val(InputBox("Entrar la 1 Calificación : ", "Entrar Calificación"))
ActiveSheet.Range("A1").Value = Calificación
Media = Media + Calificación

Calificación = Val(InputBox("Entrar la 1 Calificación : ", "Entrar Calificación"))
ActiveSheet.Range("A2").Value = Calificación
Media = Media + Calificación

Calificación = Val(InputBox("Entrar la 1 Calificación : ", "Entrar Calificación"))
ActiveSheet.Range("A3").Value = Calificación
Media = Media + Calificación

Calificación = Val(InputBox("Entrar la 1 Calificación : ", "Entrar Calificación"))
ActiveSheet.Range("A4").Value = Calificación
Media = Media + Calificación

Calificación = Val(InputBox("Entrar la 1 Calificación : ", "Entrar Calificación"))
ActiveSheet.Range("A5").Value = Calificación
Media = Media + Calificación

Media = Media / 5
ActiveSheet.Range("A6").Value = Media
End Sub
```

Observe que este programa repite el siguiente bloque de sentencias, 5 veces.

```
Calificación = Val(InputBox("Entrar la 1 Calificación : ","Entrar Calificación"))
ActiveSheet.Range("A5").Value = Calificación
Media = Media + Calificación
```

Para evitar este tipo de repeticiones de código, los lenguajes de programación incorporan instrucciones que permiten la repetición de bloques de código como las veremos a continuación.

### Estructura repetitiva Para (for)

Esta estructura sirve para repetir la ejecución de una sentencia o bloque de sentencias, un número definido de veces. La sintaxis del ciclo for en español es la siguiente:

```
Para var =Valor_Inicial Hasta Valor_Final Paso Incremento Hacer
Inicio
Sentencia 1
Sentencia 2
.
.
Sentencia N
Fin
```

**Var** es una variable que la primera vez que se entra en el ciclo se iguala a *Valor\_Inicial*, las sentencias del ciclo se ejecutan hasta que **Var** llega al *Valor\_Final*, cada vez que se ejecutan el bloque de instrucciones **Var** se incrementa según el valor de *Incremento*.

En Visual Basic para Excel la estructura Para se implementa con la instrucción **For ... Next**.

```
For Variable = Valor_Inicial To Valor_Final Step Incremento
    Sentencia 1
    Sentencia 2
    .
    .
    Sentencia N
Next Variable
```

*Si el incremento es 1, no hace falta poner Step 1.*

A continuación haremos un ejemplo utilizando la función InputBox, sumarlos y guardar el resultado en la casilla A1 de la hoja activa.

```
Sub Ejemplo_ciclofor()
Dim i As Integer
Dim Total As Integer
```

```
Dim Valor As Integer
For i=1 To 10
    Valor= Val(InputBox("Entrar un valor","Entrada"))
    Total = Total + Valor
Next i
ActiveCell.Range("A1").Value = Total
End Sub.
```

## Recorrer casillas de una hoja de cálculo

Una operación bastante habitual cuando se trabaja con Excel es el recorrido de rangos de casillas para llenarlas con valores, mirar su contenido, etc. Las estructuras repetitivas son imprescindibles para recorrer grupos de celdas o rangos. Vea los siguientes ejemplos de utilización de estructuras repetitivas para recorrer rangos de casillas, observe la utilización de las propiedades **Cells** y **Offset**.

### Propiedad Cells

Esta propiedad, sirve para referenciar una celda o un rango de celdas según coordenadas de fila y columna.

A continuación llenaremos el rango de las casillas A1..A5 con valores pares consecutivos empezando por el 2.

```
Sub Ejemplo_ciclofor2()
Dim Fila As Integer
Dim i As Integer
Fila = 1
For i=2 To 10 Step 2
    ActiveSheet.Cells(Fila,1).Value = i
    Fila = Fila+1
Next i
End Sub
```

Otro ejemplo sería llenar un rango de filas, empezando por una celda, que se debe especificar desde teclado, con una serie de 10 valores consecutivos (comenzando por el 1).

```
Sub Ejemplo_ciclofor3()
Dim Casilla_Inicial As String
Dim i As Integer
Dim Fila As Integer, Columna As Integer
Casilla_Inicial = InputBox("Introducir la casilla Inicial : ", "Casilla Inicial")
' recuerda que la casilla debe de teclearse como A1, pues la columna A y el renglón es 1
ActiveSheet.Range(Casilla_Inicial).Activate
' tomar el valor de fila de la celda activa sobre la variable Fila
Fila = ActiveCell.Row
' tomar el valor de columna de la celda activa sobre la variable Fila
Columna = ActiveCell.Column
For i = 1 To 10
    ActiveSheet.Cells(Fila, Columna).Value = i
    Fila = Fila + 1
Next i
End Sub
```

```
Next i
End Sub
```

## Propiedades ROW y COLUMN

Como habrás visto en el ejemplo anterior devuelven la fila y la columna de un objeto range. En el ejemplo anterior se utilizaban concretamente para obtener la fila y la columna de la casilla activa.

Otra forma de hacer el programa es:

```
Sub Ejemplo_ciclofor4()
Dim Casilla_Inicial As String
Dim i As Integer
Dim Fila As Integer, Columna As Integer
Casilla_Inicial = InputBox("Introducir la casilla Inicial : ", "Casilla Inicial")
ActiveSheet.Range(Casilla_Inicial).Activate
Fila = 1
For i = 1 To 10
    ActiveSheet.Range(Casilla_Inicial).Cells(Fila, 1).Value = i
    Fila = Fila + 1
Next i
End Sub
```

Recuerda que cuando utilizamos **Cells** como propiedad de un rango (Objeto Range), **Cells** empieza a contar a partir de la casilla referenciada por **Range**.

Ahora haremos el programa con el que iniciamos la sección de ciclos, pero utilizando el ciclo **for** y propiedad **Cells**

```
Sub Ejemplo_ciclofor5()
Dim Nota As Integer
Dim Media As Single
Dim Fila As Integer
Media = 0
For Fila = 1 To 5
    Nota = Val(InputBox("Entrar la " & Fila & " Nota : ", "Entrar Nota"))
    ActiveSheet.Cells(Fila, 1) = Nota
    Media = Media + Nota
Next Fila
Media = Media / 5
ActiveSheet.Cells(6, 1).Value = Media
End Sub
```

## Propiedad Offset

Esta propiedad es también muy útil a la hora de recorrer rango. **Offset**, que significa desplazamiento, es una propiedad del objeto **Range** y se utiliza para referenciar una casilla situada n Filas y n Columnas de una casilla dada. Veamos algunos ejemplos:

**ActiveSheet.Range("A1").Offset(2, 2).Value="Hola" ' Casilla C3=Hola, 2 filas y 2 columnas desde A1.**

**ActiveCell.Offset(5,1).Value="Hola"** ' 5 Filas por debajo de la casilla Activa=Hola

**ActiveCell.Offset(2,2).Activate** 'Activar la casilla que está 2 filas y 2 columnas de la activa

Ahora haremos el mismo programa del ciclo for, pero utilizando el **For** y propiedad **Offset**

```
Sub Ejemplo_ciclofor6()
Dim Nota As Integer
Dim Media As Single
Dim Fila As Integer
Media = 0
ActiveSheet.Range("A1").Activate
For Fila = 0 To 4
    Nota=Val(Input Box("Entrar la " & Fila+1 & " Nota : ", "Entrar Nota"))
    ActiveCell.Offset(Fila, 0).Value = Nota
    Media = Media + Nota
Next Fila
Media = Media / 5
ActiveCell.Offset(6, 1).Value = Media
End Sub
```

El mismo con el que introducíamos el tema, pero utilizando el **For** y propiedad **Offset**. Observe que ahora vamos cambiando de celda activa.

```
Sub Ejemplo_ciclofor7()
Dim Nota As Integer
Dim Media As Single
Dim i As Integer
Media = 0
ActiveSheet.Range("A1").Activate
For i = 1 To 5
    Nota=Val(InputBox("Entrar la " & i & " Nota : ", "Entrar Nota"))
    ActiveCell.Value = Nota.
    Media = Media + Nota
    ' Hacer activa la casilla situada una fila por debajo de la actual
    ActiveCell.Offset(1, 0).Activate
Next Fila
Media = Media / 5
ActiveCell.Value = Media
End Sub
```

Observe la diferencia entre los ejemplos llamados ciclofor7 y ciclofor8, ambos utilizan la propiedad **Offset** de diferente forma, en el primero la casilla activa siempre es la misma A1, **Offset** se utiliza para referenciar una casilla a partir de ésta. En el segundo se va cambiando de casilla activa cada vez de forma que, cuando termina la ejecución del programa la casilla activa es A6.

Cuándo utilizar cada método, como casi siempre depende de lo que se pretenda hacer. Aquí es bastante claro, si le interesa que no cambie la casilla utilice **Offset** como en el ejemplo 25, en caso que interese que vaya cambiando, haga como en el Ejemplo 6. Por supuesto hay

muchas variantes sobre el estilo de recorrer Celdas, tanto con **Cells** como con **Offset**, sólo tiene que ir probando y, como casi siempre, utilizar el que más le guste.

### **Ciclo Do While..Loop (Hacer Mientras)**

La estructura repetitiva **for** se adapta perfectamente a aquellas situaciones en que se sabe previamente el número de veces que se ha de repetir un proceso, entrar veinte valores, recorrer cincuenta celdas, etc. Pero hay ocasiones o casos en los que no se sabe previamente el número de veces que se debe repetir un proceso. Por ejemplo, suponga que ha de recorrer un rango de filas en los que no se sabe cuántos valores habrá (esto es, cuántas filas llenas habrá), en ocasiones puede que hayan veinte, en ocasiones treinta, en ocasiones ninguna, etc. Para estos casos la estructura **for** no es adecuada y deberemos recurrir a la sentencia **Do While..Loop** en alguna de sus formas.

**Hacer Mientras** (se cumpla la condición)

Sentencia1  
Sentencia2  
.  
.  
Sentencia N

**Fin Hacer Mientras**

La sintaxis en Visual Basic es la siguiente:

**Do While** (se cumpla la condición)

Sentencia1  
Sentencia2  
.  
.  
Sentencia N

**Loop**

Esta estructura repetitiva está controlada por una o varias condiciones, la repetición del bloque de sentencias dependerá de si se va cumpliendo la condición o condiciones. Esto no significa que el ciclo For se utiliza para ciertos programas y para otros el ciclo While, esto es incorrecto, pues se pueden utilizar ambos ciclos para ejecutar determinados procesos, pero la forma de pararlo es diferente. La estructura del ciclo While es la siguiente:

Los ejemplos que veremos a continuación sobre la instrucción **Do While..Loop** se harán sobre una base de datos. Una base de datos en Excel es simplemente un rango de celdas en que cada fila representa un registro y cada columna un campo de registro, la primera fila es la que da nombre a los campos. Para nuestra base de datos utilizaremos los campos siguientes, *Nombre*, *Ciudad*, *Edad*, *Fecha*. Ponga estos títulos en el rango A1:D1 de la Hoja1 (En A1 ponga Nombre, en B1 ponga Ciudad, en C1 ponga Edad y en D1 Fecha), observe que los datos se empezarán a entrar a partir de A2.

A continuación haremos un programa para capturar registros en la base de datos, cada campo se entra con InputBox. El programa pedirá datos mientras se teclea un valor en el



InputBox correspondiente al nombre, es decir cuando al preguntar el nombre no se entre ningún valor, terminará la ejecución del bloque encerrado entre **Do While...Loop**.

Observa la utilización de la propiedad **Offset** para colocar los datos en las celdas correspondientes.

```
Sub Ejemplo_dowhile1()  
Dim Nombre As String  
Dim Ciudad As String  
Dim Edad As Integer  
Dim fecha As Date  
    ' Activar hoja1  
Worksheets("Hoja1").Activate  
    ' Activar casilla A2  
ActiveSheet.Range("A2").Activate  
Nombre = InputBox("Entre el Nombre (Return para Terminar) : ", "Nombre")  
    ' Mientras la variable Nombre sea diferente a cadena vacía  
Do While Nombre <> ""  
    Ciudad = InputBox("Entre la Ciudad : ", "Ciudad")  
    Edad = Val(InputBox("Entre la Edad : ", "Edad"))  
    Fecha=Cdate(InputBox("Entra la Fecha : ", "Fecha"))  
    ' Copiar los datos en las casillas correspondientes  
    With ActiveCell  
        .Value = Nombre  
        .Offset(0,1).Value = Ciudad  
        .Offset(0,2).Value = Edad  
        .Offset(0,3).Value = fecha  
    End With  
    'Hacer activa la celda de la fila siguiente a la actual  
ActiveCell.Offset(1,0).Activate  
Nombre = InputBox("Entre el Nombre (Return para Terminar) : ", "Nombre")  
Loop  
End Sub
```

Ponga atención a este ejemplo, ya que el código que viene a continuación lo utilizará en muchas ocasiones. Antes que nada observe el ejemplo anterior, fíjese en que siempre empezamos a llenar el rango de la hoja a partir de la celda A2, esto borrará la información antes introducida y como consecuencia, la segunda vez que ejecute la macro no quedará nada de lo que anteriormente se tecleó, pues iniciará en A2:D2 y si continúa ejecutando borrará la información del siguiente renglón.

Una solución sería observar previamente cuál es el siguiente renglón vacío y cambiar en la instrucción **ActiveSheet.Range("A2").Activate**, la referencia **A2** por la que corresponde a la primera casilla vacía de la columna A. El código que le mostramos a continuación hará esto por nosotros, es decir recorrerá una fila de celdas a partir de A1 hasta encontrar una vacía y dejará a ésta como celda activa para que la entrada de datos comience a partir de ella.

```

Sub Ejemplo_dowhile2()
.
.
' Activar hoja1
WorkSheets("Hoja1").Activate
' Activar casilla A2
ActiveSheet.Range("A1").Activate
' Mientras la celda activa no esté vacía
Do While Not IsEmpty(ActiveCell)
' Hacer activa la celda situada una fila por debajo de la actual
ActiveCell.Offset(1,0).Activate
Loop
.
.
End Sub

```

Si anexamos la parte que falta del programa para que no sobrescriba la información que ya teníamos capturada, el programa buscará la primera casilla vacía de la base de datos y otro para pedir los valores de los campos hasta que se pulse Enter, quedaría de la forma siguiente:

```

Sub Ejemplo_dowhile3()
Dim Nombre As String, Dim Ciudad As String
Dim Edad As Integer
Dim fecha As Date
WorkSheets("Hoja1").Activate
ActiveSheet.Range("A1").Activate
' Buscar la primera celda vacía de la columna A y convertirla en activa
Do While Not IsEmpty(ActiveCell)
ActiveCell.Offset(1,0).Activate
Loop
Nombre = InputBox("Entre el Nombre (Return para Terminar) : ", "Nombre")
' Mientras la variable Nombre sea diferente a cadena vacía
Do While Nombre <> ""
Ciudad = InputBox("Entre la Ciudad : ", "Ciudad")
Edad = Val(InputBox("Entre la Edad : ", "Edad"))
Fecha=Cdate(InputBox("Entra la Fecha : ", "Fecha"))
With ActiveCell
.Value = Nombre
.Offset(0,1).Value = Ciudad
.Offset(0,2).Value = Edad
.Offset(0,3).value = fecha
End With
ActiveCell.Offset(1,0).Activate
Nombre = InputBox("Entre el Nombre (Return para Terminar) : ", "Nombre")
Loop
End Sub

```

Cuando se tiene que introducir la información desde el teclado conjuntos de valores, algunos programadores y usuarios prefieren la fórmula de que el programa pregunte si se desean entrar más datos, la típica pregunta ¿Desea introducir más datos?, si el usuario contesta Sí, el programa vuelve a ejecutar las instrucciones correspondientes a la entrada de

datos, si contesta que No se finaliza el proceso. Observe cómo quedaría nuestro ciclo de entrada de datos con este sistema.

```
Mas_datos = vbYes
Do While Mas_Datos = vbYes
    Nombre = InputBox("Entre el Nombre (Return para Terminar) : ", "Nombre")
    Ciudad = InputBox("Entre la Ciudad : ", "Ciudad")
    Edad = Val(InputBox("Entre la Edad : ", "Edad"))
    Fecha=Cdate(InputBox("Entra la Fecha : ", "Fecha"))
    With ActiveCell
        .Value = Nombre
        .Offset(0,1).Value = Ciudad
        .Offset(0,2).Value = Edad
        .Offset(0,3).value = fecha
    End With
    ActiveCell.Offset(1,0).Activate
    ' Preguntar al usuario si desea entrar otro registro.
    Mas_datos = MsgBox("Otro registro ?", vbYesNo+vbQuestion,"Entrada de datos")
Loop
```

Observe que es necesaria la línea anterior al ciclo **Mas\_datos = vbYes**, para que cuando se evalúe la condición por vez primera, ésta se cumpla y se ejecuten las sentencias de dentro del ciclo, Mas\_datos es una variable de tipo **Integer**.

### **Estructura Do..Loop While**

El funcionamiento de esta estructura repetitiva es similar a la anterior, salvo que la condición se evalúa al final, la inmediata consecuencia de esto es que las instrucciones del cuerpo del ciclo se ejecutarán al menos una vez. Observe que para nuestra estructura de entrada de datos vista en el último apartado de la sección anterior, esta estructura es más conveniente, al menos más elegante, si vamos a entrar datos, al menos uno entraremos, por tanto las instrucciones del cuerpo del ciclo se deben ejecutar al menos una vez, luego ya decidiremos si se repiten o no, la sintaxis es la siguiente:

```
Hacer
    Sentencia1
    Sentencia2
    .
    .
    Sentencia N
Mientras (se cumpla la condición)
```

La sintaxis en Visual Basic es la siguiente:

```
Do
    Sentencia1
    Sentencia2
    .
    .
    Sentencia N
While (se cumpla la condición)
```

A continuación haremos un ejemplo de el ciclo Do...While

#### Do

```
Nombre = InputBox("Entre el Nombre (Return para Terminar) : ", "Nombre")
```

```
Ciudad = InputBox("Entre la Ciudad : ", "Ciudad")
```

```
Edad = Val(InputBox("Entre la Edad : ", "Edad"))
```

```
Fecha=Cdate(InputBox("Entra la Fecha : ", "Fecha"))
```

#### With ActiveCell

```
.Value = Nombre
```

```
.Offset(0,1).Value = Ciudad
```

```
.Offset(0,2).Value = Edad
```

```
.Offset(0,3).value = fecha
```

#### End With

```
ActiveCell.Offset(1,0).Activate
```

```
Mas_datos = MsgBox("Otro registro ?", vbYesNo+vbQuestion,"Entrada de datos")
```

```
'Mientras Mas_Datos = vbYes
```

```
Loop While Mas_Datos = vbYes
```

Observe que en este caso no es necesaria la línea `Mas_Datos = vbYes` antes de **Do** para forzar la entrada en el ciclo, ya que la condición va al final.

### **Estructura Do..Loop Until (Hacer.. Hasta que se cumpla la condición).**

Es otra estructura que evalúa la condición al final, observe que la interpretación es distinta ya que el ciclo se va repitiendo **HASTA que se cumple la condición**, no MIENTRAS se cumple la condición. Como mencionamos anteriormente, puede utilizar el ciclo al que más se acostumbre.

La entrada de datos con este ciclo quedaría

#### Do

```
Nombre = InputBox("Entre el Nombre (Return para Terminar) : ", "Nombre")
```

```
Ciudad = InputBox("Entre la Ciudad : ", "Ciudad")
```

```
Edad = Val(InputBox("Ent re la Edad : ", "Edad"))
```

```
Fecha=Cdate("InputBox("Entra la Fecha : ", "Fecha")
```

#### With ActiveCell

```
.Value = Nombre
```

```
.Offset(0,1).Value = Ciudad
```

```
.Offset(0,2).Value = Edad
```

```
.Offset(0,3).value = fecha
```

#### End With

```
ActiveCell.Offset(1,0).Activate
```

```
Mas_datos = MsgBox("Otro registro ?", vbYesNo+vbQuestion,"Entrada de datos")
```

```
'Hasta que Mas_Datos sea igual a vbNo
```

```
Loop Until Mas_Datos= vbNo.
```

### **Estructura For Each**

Este ciclo se utiliza básicamente para ejecutar un grupo de sentencias con los elementos de una colección o una matriz (pronto veremos lo que es). Recuerde que una colección es un

conjunto de objetos, hojas, rangos, etc. Observe el ejemplo siguiente que se utiliza para cambiar los nombres de las hojas de un libro de trabajo.

Este programa que pregunta el nombre para cada hoja de un libro de trabajo, si no se pone nombre a la hoja, queda el que tiene.

```
Sub Ejemplo_foreach1()  
Dim Nuevo_Nombre As String  
Dim Hoja As Worksheet  
    ' Para cada hoja del conjunto Worksheets  
For Each Hoja In Worksheets  
    Nuevo_Nombre=InputBox("Nombre de la Hoja : " & Hoja.Name,"Nombrar Hojas")  
    If Nuevo_Nombre <> "" Then  
        Hoja.Name=Nuevo_nombre  
    End if  
Next  
End Sub
```

Este programa va referenciando cada una de las hojas del conjunto Worksheets a cada paso de ciclo.

Ahora vamos a hacer un programa para introducir valores en cada una de las celdas en el rango de rango A1:B10 de la hoja Activa.

```
Sub Ejemplo_foreach2()  
Dim R As Range  
    ' Para cada celda del rango A1:B10 de la hoja activa  
For Each R in ActiveSheet.Range("A1:B10")  
    R.Value = InputBox("Entrar valor para la celda " & R.Address, "Entrada de valores")  
Next  
End Sub
```

Observe que se ha declarado una variable tipo Range, este tipo de datos, como puede imaginar y ha visto en el ejemplo, sirve para guardar Rangos de una o más casillas, estas variables pueden luego utilizar todas las propiedades y métodos propios de los Objetos Range. Tenga en cuenta que la asignación de las variables que sirven para guardar o referenciar objetos (Range, Worksheet, etc.) deben inicializarse muchas veces a través de la instrucción SET, esto se estudiará en otro capítulo.

## Procedimientos y funciones

Se define como procedimiento y/o función a un bloque de código que realiza alguna tarea específica. Hasta ahora, hemos construido los programas utilizando un único procedimiento, pero a medida que los programas (y los problemas) crecen se va haciendo necesaria la inclusión de más procedimientos. Podría fácilmente caer en la tentación de utilizar, como hasta ahora, un único procedimiento por programa pero se dará cuenta rápidamente de que este método no es nada práctico ya que grandes bloques de código implican mayor complicación del mismo, repetición de sentencias y lo que es más grave,

mayores problemas de seguimiento a la hora de depurar errores, ampliar funcionalidades o incluir modificaciones.

La filosofía de utilizar procedimientos es la antigua fórmula del "divide y vencerás", es decir, con los procedimientos podremos tratar cada problema o tarea de forma más o menos aislada de forma que construiremos el programa paso a paso evitando tener que resolver o controlar múltiples cosas a la vez.

Cada tarea realizará un procedimiento, si esta tarea implica la ejecución de otras tareas, cada una se implementará y solucionará en su correspondiente procedimiento de manera que cada uno haga una cosa concreta. Así, los diferentes pasos que se deben ejecutar para que un programa haga algo, quedarán bien definidos en su correspondiente procedimiento, si el programa falla, fallará a partir de un procedimiento y de esta forma podremos localizar el error más rápidamente.

Los procedimientos son también un eficaz mecanismo para evitar la repetición de código en un mismo programa e incluso en diferentes programas. Suponemos que habrás intuido que hay muchas tareas que se repiten en casi todos los programas, veremos cómo los procedimientos que ejecutan estas tareas se pueden incluir en un módulo de forma que éste sea exportable a otros programas y de esta manera ganar tiempo que, como dice el tópico, es precioso.

## Definición de procedimientos

Como podrás ver, desde el principio hemos creado procedimientos sin darnos cuenta que ellos son tan poderosos, pues son los programas mismos que se ejecutan en la macro, definimos el programa de la siguiente manera:

```
Sub Nombre_Procedimiento
    Sentencias.
End Sub.
```

## Llamar a un procedimiento

Para llamar un procedimiento desde otro se utiliza la instrucción **Call** *Nombre\_Procedimiento*.

```
Sub P_Uno
    Sentencias.
.
Call P_Dos
.
    Sentencias
.
End Sub
```

Las secuencias del procedimiento *P\_Uno* se ejecutan hasta llegar a la línea **Call P\_Dos**, entonces se salta al procedimiento *P\_Dos*, se ejecutan todas las sentencias de este procedimiento y el programa continúa ejecutándose en el procedimiento *P\_Uno* a partir de la sentencia que sigue a **Call P\_Dos**.

El programa ejemplo\_dowhile3 que hicimos anteriormente, en el cual el usuario tecleaba los nombres de personas, ciudad, edad y fecha y se pasaban a una hoja de cálculo, esta información se ponía en la celda que estuviera vacía, pero ahora en vez de que se ponga todo el código en la misma macro, haremos un procedimiento llamado, *Saltar\_Celdas\_Llenas*. Observa que para entrar valores se ha sustituido Do While..Loop por Do.. Loop While.

```

Sub Ejemplo_proc1()
Dim Nombre As String
Dim Ciudad As String
Dim Edad As Integer
Dim fecha As Date
' Llamada a la función Saltar_Celdas_Llenas, el programa salta aquí a ejecutar las
'instrucciones de este procedimiento y luego vuelve para continuar la ejecución a partir de la
'instrucción Do
Call Saltar_Celdas_Llenas
Do
    Nombre = InputBox("Entre el Nombre (Return para Terminar) : ", "Nombre")
    Ciudad = InputBox("Entre la Ciudad : ", "Ciudad")
    Edad = Val(InputBox("Entre la Edad : ", "Edad"))
    Fecha=Cdate(InputBox("Entra la Fecha : ", "Fecha"))
    With ActiveCell
        .Value = Nombre
        .Offset(0,1).Value = Ciudad
        .Offset(0,2).Value = Edad
        .Offset(0,3).value = fecha
    End With
    ActiveCell.Offset(1,0).Activate
    Mas_datos = MsgBox("Otro registro ?", vbYesNo+vbQuestion,"Entrada de datos")
Loop While Mas_Datos = vbYes
End Sub

' Función que salta celdas de una misma columna. sirve para encontrar la primera
' celda vacía de la columna, utiliza Public al principio de la función para que cualquier
' procedimiento en cualquier parte del programa lo pueda llamar
Public Sub Saltar_Celdad_Llenas()
Worksheets("Hoja1").Activate
ActiveSheet.Range("A1").Activate
Do While not IsEmpty(ActiveCell)
    ActiveCell.Offset(1,0).Activate
Loop
End Sub.

```

## Generalizar una función

Observe que para saltar un rango de casillas llenas sólo necesitará llamar a la función *Saltar\_Celdas\_Llenas*, pero, siempre y cuando este rango esté en una hoja llamada "Hoja1"

y empiece en la casilla A1, el procedimiento es poco práctico ya que su ámbito de funcionamiento es limitado. En la siguiente sección modificaremos el procedimiento de manera que sirva para recorrer un rango que empiece en cualquier casilla de cualquier hoja.

## Parámetros

Los parámetros son el mecanismo por el cual un procedimiento puede pasarle valores a otro y de esta forma condicionar, moldear, etc. las acciones que ejecuta. El procedimiento llamado gana entonces en flexibilidad. La sintaxis de llamada de un procedimiento es la siguiente, **Call** Procedimiento(Parámetro1, Parámetro2,..., ParámetroN). Los parámetros pueden ser valores o variables. La sintaxis para el procedimiento llamado es la siguiente:

**Sub** Procedimiento(Parámetro1 as Tipo, Parámetro2 As Tipo,..., Parámetro3 As Tipo)

Observa que los parámetros son variables que recibirán los valores, evidentemente debe haber coincidencia de tipo. Por ejemplo, si el primer parámetro es una variable tipo Integer, el primer valor que se le debe pasar al procedimiento cuando se llama también ha de ser de tipo Integer (recuerde que puede ser un valor directamente o una variable).

Vamos a utilizar el programa anterior, pero ahora la función Saltar\_Celdas\_Llenas tiene dos parámetros Hoja y Casilla\_Inicial que reciben respectivamente la hoja donde está el rango a recorrer y la casilla inicial del rango.

```
Sub Ejemplo_proc2()
Dim Nombre As String
Dim Ciudad As String
Dim Edad As Integer
Dim fecha As Date
' Llamada a la función Saltar_Celdas_Llenas, observar que mediante dos parámetros se
' Al procedimiento en que hoja está el rango a saltar y en la casilla donde debe empezar.
Call Saltar_Celdas_Llenas("Hoja1", "A1")
Do
    Nombre = InputBox("Entre el Nombre (Return para Terminar) : ", "Nombre")
    Ciudad = InputBox("Entre la Ciudad : ", "Ciudad")
    Edad = Val(InputBox("Entre la Edad : ", "Edad"))
    Fecha=Cdate(InputBox("Entre la Fecha : ", "Fecha"))
    With ActiveCell
        .Value = Nombre
        .Offset(0,1).Value = Ciudad
        .Offset(0,2).Value = Edad
        .Offset(0,3).value = fecha
    End With
    ActiveCell.Offset(1,0).Activate
    Mas_datos = MsgBox("Otro registro ?", vbYesNo+vbQuestion, "Entrada de datos")
Loop While Mas_Datos = vbYes
End Sub

' Procedimiento Saltar_Celdas_Llenas. Sirve para Saltar celdas llenas de una columna
' hasta encontrar una vacía que se convierte en activa
' Parámetros :
```



*' Hoja : Hoja donde está el rango a saltar. ' Casilla\_Inicial : Casilla Inicial de la columna*

```
Sub Saltar_Celdas_Llenas(Hoja As String, Casilla_Inicial As String)
Worksheets(Hoja).Activate
ActiveSheet.Range(Casilla_Inicial).Activate
    Do While not IsEmpty(ActiveCell)
        ActiveCell.Offset(1,0).Activate
    Loop
End Sub
```

Observa que ahora el procedimiento Saltar\_Celdas\_Llenas sirve para recorrer cualquier rango en cualquier hoja y que al procedimiento se le pasan dos valores directamente, también pueden pasarse variables como lo podemos ver en el siguiente ejemplo:

```
Sub Ejemplo_proc3()
.
.
Dim Hoja As String
Dim Casilla_Inicial As String
Hoja = InputBox("En que hoja está la base de datos : ", "Entrar Nombre de Hoja")
Casilla_Inicial = InputBox("En que casilla comienza la base de datos","Casilla Inicial")
' Observe que los parámetros son dos variables cuyo valor se ha entrado desde teclado en
' las dos instrucciones InputBox anteriores.
Call Saltar_Celdas_Llenas(Hoja, Casilla_Inicial)
.
.
End Sub.
```

## Variables locales y variables Globales

El ámbito de una variable declarada dentro de una función es la propia función, es decir, no podrá utilizarse fuera de dicha función. Así, el siguiente programa que debería sumar las cinco filas siguientes a partir de la casilla activa y guardar el resultado en la sexta es incorrecto.

```
Sub Alguna_Cosa()
.
.
Call Sumar_Cinco_Siguientes
ActiveCell.Offset(6,0).Value = Suma
.
.
End Sub

Sub Sumar_Cinco_Siguientes()
Dim i As Integer
Dim Suma As Single
Suma=0
For i=1 To 5
Suma = Suma+ActiveCell.Offset(i,0).Value
Next i
End Sub
```

Es incorrecto porque tanto la variable *i* como la variable *Suma* están declaradas dentro del procedimiento *Sumar\_Cinco\_Siguientes* consecuentemente, su ámbito de acción es este procedimiento, es decir, sólo existen y pueden estar usadas en este procedimiento. Por tanto, la instrucción *ActiveCell.Offset(6,0).Value=Suma* del procedimiento *Alguna\_Cosa*, generaría un error (con Option Explicit activado) ya que la variable *Suma* no está declarado dentro de él. Si piensas en declarar la variable *Suma* dentro del procedimiento *Hacer\_Algo*, no solucionará nada porque ésta será local a dicho procedimiento, en este caso tendría dos variables llamadas *Suma* pero cada una de ellas local a su propio procedimiento y consecuentemente con el ámbito de acción restringido a ellos.

Una solución es declarar suma como variable global. Una variable global se declara fuera de todos los procedimientos pero es reconocida en todos los procedimientos del módulo, pero, si trabajáramos con programa que tuviera 10 mil líneas y si utilizo casualmente esa variable en otro procedimiento lo más seguro es que no devolviera los valores esperados.

#### Option Explicit

*' Suma es una variable global reconocida por todos los procedimientos del módulo.*

**Dim** Suma **As Single**

**Sub** Alguna\_Cosa()

**Call** Sumar\_Cinco\_Siguientes  
ActiveCell.Offset(6,0).Value = Suma

**End Sub**

**Sub** Sumar\_Cinco\_Siguientes()

**Dim** i **As Integer**

Suma=0

**For** i=1 **To** 5

Suma = Suma+ActiveCell.Offset(i,0).Value

**Next** i

**End Sub.**

Las variables globales son perfectas en cierta ocasiones, para este caso sería mejor declarar *Suma* en la función *Hacer\_Algo* y pasarla como parámetro al procedimiento *Sumar\_Cinco\_Siguientes*.

**Sub** Alguna\_Cosa()

**Dim** Suma **As Single**

.

.

*' Llamada a la función Sumar\_Cinco\_Siguientes pasándole la variable Suma*

**Call** Sumar\_Cinco\_Siguientes(Suma)

ActiveCell.Offset(6,0).Value = Suma

.

.

**End Sub**

**Sub** Sumar\_Cinco\_Siguientes(S **As Single**)

**Dim** i **As Integer**

Suma=0

```
For i=1 To 5
S = S+ActiveCell.Offset(i,0).Value
Next i
End Sub
```

Esto funcionaría porque la variable parámetro *S* (y se le ha cambiado el nombre adrede) de *Sumar\_Cinco\_Siguientes* es la variable *Suma* declarada en *Hacer\_Algo*. Funcionará porque en Visual Basic, a menos que se indique lo contrario, el paso de parámetros es por referencia, vea la siguiente sección.

### Paso por referencia y paso por valor

No entraremos en detalles sobre cómo funciona el paso de parámetros por valor y el paso de parámetros por referencia, sólo indicar que

- El **paso por valor** significa que la variable parámetro del procedimiento recibe el valor de la variable (o directamente el valor) de su parámetro correspondiente de la instrucción de llamada y;
- El **paso por referencia**, la variable parámetro del procedimiento es la misma que su parámetro correspondiente de la instrucción de llamada, es decir, la declarada en el procedimiento desde el que se hace la llamada.

Por defecto, y siempre que en la instrucción de llamada se utilicen variables, las llamadas son por referencia. Si desea que el paso de parámetros sea por valor, debe anteponer a la variable parámetro la palabra reservada **ByVal**, por ejemplo

```
Sub Saltar_Celdas_Llenas(ByVal Hoja As String, ByVal Casilla_Inicial As String)
```

Aunque lo elegante y efectivo por razones de memoria sería pasar siempre que sea posible por valor, es poco habitual que así se haga en Visual Basic, seguramente por comodidad. Como suponemos que hará como la mayoría, es decir, pasar por referencia, tenga cuidado con los (indeseables) efectos laterales. Para ver estos efectos escriba el siguiente programa:

### Ejemplo Efecto\_Lateral

Antes de copiar el programa, active una hoja en blanco y ponga los siguientes valores en la hoja de cálculo:

- En el rango A1:A5 valores del 1 al 5;
- En el rango B1:B5 valores del 6 al 10 y;
- En el rango C1:C5 valores del 11 al 15.

El siguiente programa debe recorrer cada una de tres las columnas de valores, sumarlos y poner el resultado en las filas 6 de cada columna. Entonces, según los valores que ha entrado en cada una de las columnas, cuando haya acabado la ejecución del programa debe haber los siguientes resultados, A6 = 15, B6=40, C6=65.

Para llevar a cabo la suma de los valores de cada columna se llama a la función *Recorrer\_Sumar* tres veces, una para cada columna, esta función recibe en el parámetro *F* el valor de la fila donde debe empezar a sumar, sobre el parámetro *C* el valor de la columna a sumar y sobre el parámetro *Q* la cantidad de filas que ha de recorrer.

El programa utiliza la propiedad **Cells** para referenciar las filas y columnas de los rangos. Observa atentamente los valores que irá cogiendo la variable *Fila* ya que ésta será la que sufra el efecto lateral.

```
Sub Efecto_Lateral()  
Dim Fila As Integer  
Fila = 1  
Call Recorrer_Sumar(Fila, 1,5) ' Columna A  
Call Recorrer_Sumar(Fila, 2,5) ' Columna B  
Call Recorrer_Sumar(Fila, 3,5) ' Columna C  
End Sub  
  
Sub Recorrer_Sumar(F As Integer, C As Integer, Q As Integer)  
Dim i As Integer  
Dim Total As Integer  
Total = 0  
For i = 1 To Q  
    Total = Total + ActiveSheet.Cells(F, C).Value  
    F = F + 1 ' OJO con esta asignación, recuerde que F es la variable Fila declarada en  
            ' el procedimiento Efecto_Lateral  
Next i  
ActiveSheet.Cells(F, C) = Total  
End Sub
```

Cuando ejecute el programa se producirá la salida siguiente, en A6 habrá un 15, hasta aquí todo correcto, pero observe que en la segunda columna aparece un 0 en B12 y en la tercera columna aparece un 0 en C18, veamos qué ha pasado. La primera vez que se llama la función, la variable *F* vale 1 ya que éste es el valor que tiene su parámetro correspondiente (*Fila*) en la instrucción **Call**. Observa que *F* se va incrementando una unidad a cada paso de ciclo **For**, RECUERDA que *F* es realmente la variable *Fila* declarada en el procedimiento *Efecto\_Lateral*, por tanto cuando finaliza el procedimiento *Recorrer\_Sumar* y vuelve el control al procedimiento *Efecto\_Lateral* *Fila* vale 6, y éste es el valor que se pasará a *Recorrer\_Suma* la segunda vez que se llama, a partir de ahí todo irá mal ya que se empezará el recorrido de filas por la 6.

Una de las soluciones a este problema para hacer que cada vez que se llame *Recorrer\_Sumar* la variable *F* reciba el valor 1, es utilizar un paso por valor, es decir que *F* reciba el valor de *Fila*, no que sea la variable *Fila*, observe que entonces, si *F* no es la variable *Fila*, cuando incremente *F* no se incrementará *Fila*, ésta siempre conservará el valor 1. Para hacer que *F* sea un parámetro por valor, simplemente ponga la palabra **ByVal** antes de *F* en la declaración del procedimiento. Vuelva a ejecutar el programa, verá cómo ahora funciona correctamente.

Es importante tener cuidado cuando se esté programando, con este tipo de errores de programación, pues se invierte mucho tiempo en resolverlos y no nos permite avanzar en el diseño del programa.

Para acabar, observa que en muchas ocasiones le hemos indicado que en el paso por referencia la variable del procedimiento llamado es la variable declarada en el procedimiento que llama. En este último ejemplo, le hemos dicho que *F* era la variable *Fila*, pues bien, esto no es cierto, *Fila* es una variable y *F* es otra variable, ahora es lógico que se pregunte por qué entonces *F* actúa como si fuera *Fila*, si alguna vez programa en C y llega al tema de los punteros entenderá qué es lo que sucede realmente en el paso por parámetro y en el paso por valor. Si ya conoce los punteros de C o Pascal entonces ya habrá intuido que el paso por valor en nuestro ejemplo equivaldría a:

```
Recorrer_Fila(F, C, Q);  
void Recorrer_Fila(int F, int C, int Q)
```

Y un paso por referencia a

```
Recorrer_Fila(&F, C, Q);  
Void Recorrer_Fila(int *F, int C, int Q).
```

## Funciones

Una función es lo mismo que un procedimiento con la salvedad que éste devuelve un valor al procedimiento o función que lo llama. Vea el siguiente ejemplo, es una función muy sencilla ya que simplemente suma dos números y devuelve el resultado.

La sintaxis es similar a la cabecera de un procedimiento, sólo que una función tiene tipo, esto tiene su lógica, ya que una función devuelve un valor, ese valor será de un tipo determinado, a continuación veremos la sintaxis en Visual Basic:

```
Function Nombre_Funcion(par1 As Tipo, par2 As Tipo,..., parN As Tipo) As Tipo.
```

```
.  
Conjunto de instrucciones de la función
```

```
.  
Nombre_Función=valor que se obtuvo y va a ser devuelto por la función  
End Function
```

A continuación haremos una función que devuelve la suma de dos valores que se le pasan como parámetros.

```
Sub Ejemplo_fun1()  
Dim x As Integer  
Dim n1 As Integer, n2 As Integer  
X = Suma(5, 5)  
n1= Val ( InputBox("Entrar un número : ", "Entrada"))  
n2= Val ( InputBox("Entrar otro número : ", "Entrada"))  
X= suma(n1,n2)
```

```
ActiveCell.Value = Suma(ActiveSheet.Range("A1").Value , ActiveSheet.Range("A2").Value)
X = Suma(5, 4) + Suma (n1, n2)
End Sub
```

```
Function Suma(V1 As Integer, V2 As Integer) As Integer
Dim Total As Integer
    Total = V1 + V2
    Suma = Total
End Function
```

Observe la sintaxis de la cabecera de función,

**Function Suma(V1 As Integer, V2 As Integer) As Integer**

El resultado que devuelve nuestra **Function Suma** es del tipo **Integer**, o dicho de otra manera, la función ejecuta sus sentencias y devuelve un valor hacia el procedimiento o la función que la llamó, el valor devuelto se establece igualando el nombre de la función a algo, Nombre\_Función = resultado.

En el ejemplo de **Function Suma**, Suma = Total, observa también la sintaxis de la llamada función, en el ejemplo hemos utilizado unas cuantas formas de llamarla, lo que debe tener siempre presente es que en cualquier expresión aritmética o de cálculo, la computadora realiza un mínimo de dos operaciones, una de cálculo y otra de asignación. Por ejemplo, A= B+C La computadora primero calcula el resultado de sumar B+C luego asigna ese resultado a la variable A. En cualquier llamada a una función, cojamos por caso, X= suma(n1,n2), primero se ejecutan todas las sentencias de la función Suma, luego se asigna el cálculo de la función a la variable X. Da otro vistazo a la función de ejemplo y vea lo que realiza cada sentencia en la que se llama a la función *Suma*.

Veamos a continuación unos cuantos ejemplos de funciones. Para las funciones se utilizan los mismos conceptos de parámetros por valor y referencia, variables locales y globales, etc. que vimos en los procedimientos.

Función que devuelve la dirección de la primera celda vacía de un rango. La función es de tipo **String** ya que devuelve la casilla en la forma "FilaColumna ", por ejemplo "A10". Utilizaremos la propiedad **Address** del objeto range, esta propiedad devuelve un string que contiene la referencia "FilaColumna" de una casilla o rango de casillas. En el caso de un rango devuelve, "FilaColumna\_Inicial:FilaColumna\_Final", por ejemplo "A1:C10"

```
Sub Ejemplo_fun2()
Dim Casilla As String
Casilla = Casilla_Vacia("A1")
.....
End Sub.
```

```
' Función Casilla_Vacia de Tipo String
' Sirve para Recorrer las filas de una columna hasta encontrar una vacía.
' Parámetros :
' Casilla_Inicio : Casilla donde debe empezar a buscar.
```

*' Devuelve Un string que contiene la referencia de la primera casilla*

**Function** Casilla\_Vacia(Casilla\_Inicio **As String**) **As String**

ActiveSheet.Range(Casilla\_Inicio).Activate

**Do While Not** IsEmpty(ActiveCell)

ActiveCell.Offset(1, 0).Activate

**Loop**

Casilla\_Vacia = ActiveCell.Address

**End Function**

Similar al anterior. Es la típica búsqueda secuencial de un valor en un rango de casillas, en esta función sólo se avanzará a través de una fila. La función devuelve la dirección (address) de la casilla donde está el valor buscado, en caso que el valor no esté en el rango de filas, devuelve una cadena vacía ("").

**Sub** Ejemplo\_fun3()

**Dim** Casilla **As** String

Casilla = Buscar\_Valores("A1", 25)

*' Si valor no encontrado*

**If** Casilla = "" **Then**

.....

**Else** *'Valor encontrado*

....

**End if**

**End Sub**

*' Función Buscar de Tipo String*

*' Sirve para, Recorrer las filas de una columna hasta encontrar el valor buscado*

*' o una de vacía.*

*' Parámetros:*

*' Casilla\_Inicial : Casilla donde debe empezar a buscar.*

*' Valor\_Buscado : Valor que se debe encontrar*

*' Devuelve, Un string que contiene la referencia de la casilla donde se ha encontrado el valor.*

*' También puede devolver "" en caso que el valor buscado no esté.*

**Function** Buscar(Casilla\_Inicial **As String**, Valor\_Buscado **As Integer**) **As String**

ActiveSheet.Range(Casilla\_Inicial).Activate

*' Mientras casilla no vacía Y valor de casilla diferente al buscado*

**Do While Not** IsEmpty(ActiveCell) **And** ActiveCell.Value <> Valor\_Buscado

ActiveCell.Offset(1, 0).Activate

**Loop**

*' Si la casilla donde se ha detenido la búsqueda NO ESTÁ VACÍA es que se ha encontrado el valor.*

**If Not** IsEmpty(ActiveCell) **Then**

Buscar = ActiveCell.Address *' Devolver la casilla donde se ha encontrado el valor*

**Else** *' La casilla está vacía, NO se ha encontrado el valor buscado*

Buscar="" *' Devolver una cadena vacía*

**End if**

**End Function.**

Similar al anterior. Búsqueda secuencial de un valor en un rango de casillas, en esta función se avanzará a través de filas y columnas. La función devuelve la dirección (address) de la

casilla donde está el valor buscado, en caso que el valor no esté en el rango, devuelve una cadena vacía ("").

```

Sub Ejemplo_fun4()
Dim Casilla As String
Casilla = Buscar_Valor("A1", 25)
If Casilla = "" Then
    ....
Else
    ....
End if
End Sub

Function Buscar(Casilla_Inicial As String, Valor_Buscado As Integer) As String
Dim Incremento_Columna As Integer
Dim Continuar As Boolean
ActiveSheet.Range(Casilla_Inicial).Activate
Continuar = True
Do While Continuar
    Incremento_Columna = 0
    ' Buscar el valor por las columnas hasta encontrarlo o encontrar una celda vacía.
    Do While Not IsEmpty(ActiveCell.Offset(0, Incremento_Columna) And
        ActiveCell.Offset(0, Incremento_Columna).Value <> Valor_Buscado
        ' Siguiente columna
        Incremento_Columna = Incremento_Columna + 1
    Loop
    ' Si no está vacía la casilla entonces parar la búsqueda, se ha encontrado el valor
    If Not IsEmpty(ActiveCell.Offset(0, Incremento_Columna)) Then
        Continuar=False
    Else ' La casilla está vacía, no se ha encontrado el valor
        ActiveCell.Offset(1, 0).Activate ' Saltar a una nueva fila
        If IsEmpty(ActiveCell) Then ' Si la casilla de la nueva fila está vacía
            Continuar=False ' Parar la búsqueda, no hay más casilla a recorrer
        End if
    End if
Loop
    ' Si la casilla donde se ha detenido la búsqueda NO ESTÁ VACÍA es que se ha encontrado
    ' el valor.
    If Not IsEmpty(ActiveCell) Then
        Buscar = ActiveCell(0, Incremento_Columna).Address ' Devolver la casilla donde se
        ' ha encontrado el valor
    Else ' La casilla está vacía, NO se ha encontrado el valor buscado
        Buscar="" ' Devolver una cadena vacía
    End if
End Function.

```

## La cláusula Private

Puede anteponer la cláusula private a todos los procedimientos y funciones que sean llamados sólo desde el mismo módulo, es una forma de ahorrar memoria y hacer que el programa corra un poco más rápido. Si necesita llamar un procedimiento o función desde otro módulo, nunca debe precederlo por la cláusula private, recuerde que esta cláusula



restringe el ámbito de utilización de un procedimiento a su propio módulo. Veamos el ejemplo siguiente.

```
' Módulo 1
Sub General
....
End Sub

Private Sub Privado
....
End Sub

' Módulo 2
Sub Procedimiento_de_modulo2
' Esto es correcto. Llama al procedimiento General definido en Módulo1
Call General
' Esto no es correcto. Llama al procedimiento Privado definido en Módulo 1, este
' procedimiento va precedido por la cláusula Private, por tanto sólo puede ser llamado
' desde procedimientos de su propio módulo
Call Privado
End Sub.
```

A continuación veremos más ejemplos sobre funciones. Es importante que los teclee en un libro de trabajo nuevo y los ponga en un mismo módulo, más adelante veremos cómo utilizar las opciones de exportar e importar módulos de procedimientos y funciones. En todos los ejemplos verá el Procedimiento\_Llamador, es para mostrar de qué forma se debe llamar al procedimiento o función. Los procedimientos implementados son, por llamarlo de alguna manera, de tipo general, es decir, son procedimientos que podrá utilizar en muchas aplicaciones.

Procedimiento que abre un cuadro MsgBox y muestra el texto que se le pasó como parámetro.

```
Sub Procedimiento_Llamador()
.
.
Call mAviso("Esto es el mensaje de aviso", "Esto es el Título")
.
.
End Sub

' Procedimiento mAviso
' Función Mostrar el cuadro de función MsgBox, con el icono información y
' el botón OK (Aceptar). Se utiliza para mostrar avisos.
' Parámetros:
' Texto = Texto que muestra el cuadro
' Titulo = Título del cuadro
'

Sub mAviso(Texto As String, Titulo As String)
MsgBox Prompt:=Texto, Buttons:=vbOKOnly + vbInformation, Title:=Titulo
End Sub
```

Función tipo range que devuelve un rango. Observe cómo la función se iguala a una variable tipo Range, recuerde que con esta variable podrá acceder a todas las propiedades e invocar todos los métodos propios de los objetos Range. En este ejemplo en concreto se utilizan las variables para Copiar un grupo de celdas de un rango hacia otro, se utilizan los métodos Copy y Paste del objeto Range.

```
Sub Procedimiento_Llamador()  
Dim Rango_Origen As Range  
Dim Rango_Destino As Range  
Set Rango_Origen=Coger_Rango(A1,5,5)  
Rango_Origen.Copy  
Set Rango_Destino=Coger_Rango(G1,5,5)  
Rango_Destino.Paste PasteSpecial:=xlPasteAll  
End Sub.  
' Función que devuelve un rango a una variable de este tipo  
' Parámetros  
' Casilla = casilla inicial del rango  
' Filas = número' de filas  
' Columnas = número de columnas del rango  
  
Function Coger_Rango(Casilla As String, Filas As Integer, Columnas As Integer) As Range  
Dim Casilla_Final As String  
ActiveSheet.Range(Casilla).Activate  
ActiveCell.Cells(Filas, Columnas).Activate  
Casilla_Final = ActiveCell.Address  
ActiveSheet.Range(Casilla & ":" & Casilla_Final).Select  
Set Coger_Rango = ActiveSheet.Range(Casilla & ":" & Casilla_Final)  
End Function
```

Función para comprobar el tipo de datos. Es una función de comprobación que se puede utilizar para validar los datos que se entran desde un cuadro InputBox o desde los cuadros de texto de formularios. La función es de tipo Booleano, devuelve True (cierto) o False en función de si el dato pasado es correcto.

En esta función se evalúan sólo datos numéricos y datos tipo Fecha, puede ampliarla para que se comprueben más tipos.

```
Sub Procedimiento_Llamador()  
Dim Cantidad As Integer  
Dim Fecha As Date  
Dim Datos As String  
  
.  
.  
Datos = InputBox("Entrar una Cantidad : ", "Entrar")  
If Not Comprobar_Tipo(Datos,"N") Then  
mAviso("Los datos introducido no son numéricos", "Error")  
Else  
Cantidad = Val(Datos)  
  
.  
.  
End If  
.
```

```

.
Datos=InputBox("Entrar Fecha","Entrar")
If Not Comprobar_Tipo(Datos,"F") Then
mAviso("Los fecha introducida no es correcta", "Error")
Else
Fecha = Val(Datos)
.
.
End If
.
.
End Sub.
' Función que evalúa si el tipo de datos que se le pasan son correctos o no.
'Si son correctos devuelve TRUE , en caso contrario devuelve FALSE
' Parámetros
' Valor =valor que se debe comprobar, de tipo String
' Tipo = tipo a comprobar, "N" --> Numérico, "F", tipo fecha

Function Comprobar_Tipo(Valor As String, Tipo As String) As Boolean
Dim Valido As Boolean
Valido = True
Select Case Tipo
' Comprueba si es un valor numérico válido
Case "N"
If Not IsNumeric(Valor) Then
Valido = False
End If
' Comprueba si es un valor fecha válido
Case "F"
If Not IsDate(Valor) Then
Valido = False
End If
End Select
Comprobar_Tipo = Valido
End Function

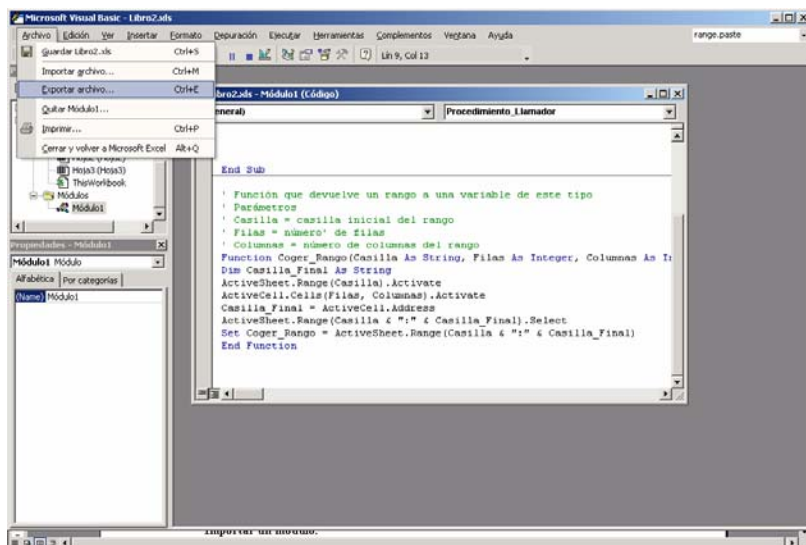
```

## Importar y Exportar módulos

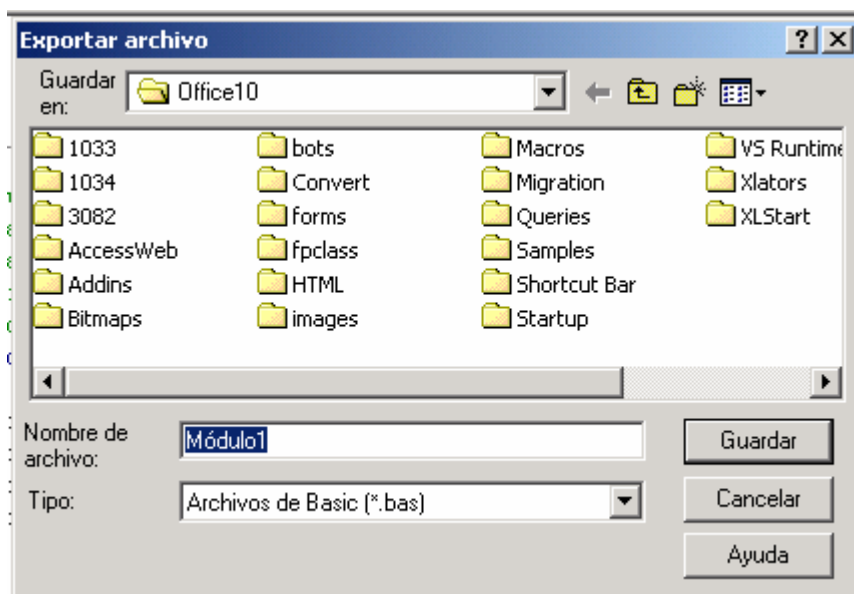
Los últimos tres ejemplos que hemos visto, como le hemos indicado, son procedimientos que pueden ser utilizados en multitud de ocasiones o situaciones, sería interesante tenerlos disponibles en cualquiera de las hojas que confeccionemos. Podría pensar en utilizar las opciones de copiar y pegar para pasar procedimientos de una hoja a otra, es un método totalmente válido y efectivo, pero le proponemos a continuación otro método más "profesional", por llamarlo de otra manera, e igual de efectivo. Este método consiste en guardar los procedimientos de un módulo en un archivo aparte, es decir, independiente de cualquier hoja de cálculo, luego, cuando en una nueva hoja necesite estas funciones, sólo deberá importar este archivo para incorporarlo.

### Exportar un módulo. Guardar un módulo en un archivo

Como ejemplo, abra la hoja donde puso los tres últimos procedimientos.



1. Pase al editor de Visual Basic y active el módulo a exportar.
2. Active opción de la barra de menús **Archivo/ Exportar archivo**. Aparece un cuadro de diálogo.



3. En cuadro de edición **Nombre de Archivo**, teclee el nombre del archivo donde se guardará el módulo, por ejemplo "General.Bas", observe que .BAS es la extensión de estos archivos.
4. Pulse sobre el botón **Guardar**.

## Importar un módulo

Si está siguiendo el ejemplo, cierre todos los archivos de Excel y abra uno nuevo.

1. Active el editor Visual Basic.
2. Active opción de la barra de menús **Archivo/ Importar Archivo**. Aparece un cuadro de diálogo.
3. Seleccione en la lista **Buscar en:** la carpeta donde tiene ubicado el archivo a importar (la carpeta donde está General.Bas si está siguiendo el ejemplo).
4. Una vez localizada la carpeta, seleccione el archivo a importar (General.Bas en el ejemplo) y pulsa sobre **Abrir**.

Observe cómo en la ventana de proyecto se ha incorporado un nuevo módulo que contiene todos los procedimientos y funciones del archivo importado.

Terminamos aquí el tema de procedimientos y funciones. Insistiremos de nuevo en que es muy importante que construya sus programas utilizando todas las ventajas que le ofrece la programación modular. Como último consejo, agrupe todas las funciones que usted considere de utilización general en uno o dos módulos y luego utilice las opciones de importación y exportación para incorporarlos a sus programas.

Un ejemplo muy utilizado es la conversión de una cantidad numérica a letras, a continuación haremos un ejemplo en el cual mostramos cómo hacer esto, obviamente con el uso de la información que ya hemos visto anteriormente.

#### Function letras(n) As String

```
' {1-} begin {programa principal}
c = 0
d = 0
u = 0
st = "("
For I = 1 To 3
    If I = 1 Then
        fc = 100000000
        fd = 10000000
        fu = 100000
    End If

    If I = 2 Then
        fc = 100000
        fd = 10000
        fu = 1000
    End If

    If I = 3 Then
        fc = 100
        fd = 10
        fu = 1
    End If
    c = Int(n / fc)
    n = n - c * fc
    d = Int(n / fd)
    n = n - d * fd
    u = Int(n / fu)
    n = n - u * fu
```



' 6

```

If (((u + d + c) <> 0) And (c = 1) And ((u + d) = 0)) Then st = st + "CIEN "
If (((u + d + c) <> 0) And (c = 1) And ((u + d) <> 0)) Then st = st + "CIENTO "
If (u + d + c) <> 0 Then
    If c = 2 Then st = st + "DOSCIENTOS "
    If c = 3 Then st = st + "TRESCIENTOS "
    If c = 4 Then st = st + "CUATROCIENTOS "
    If c = 5 Then st = st + "QUINIENTOS "
    If c = 6 Then st = st + "SEISCIENTOS "
    If c = 7 Then st = st + "SETECIENTOS "
    If c = 8 Then st = st + "OCHOCIENTOS "
    If c = 9 Then st = st + "NOVECIENTOS "
End If

```

' 6

```

If ((d = 1) And (u = 0)) Then st = st + "DIEZ "
If ((d = 1) And (u <> 0)) Then
    If u = 1 Then st = st + "ONCE "
    If u = 2 Then st = st + "DOCE "
    If u = 3 Then st = st + "TRECE "
    If u = 4 Then st = st + "CATORCE "
    If u = 5 Then st = st + "QUINCE "
    If u = 6 Then st = st + "DIEZ Y SEIS "
    If u = 7 Then st = st + "DIEZ Y SIETE "
    If u = 8 Then st = st + "DIEZ Y OCHO "
    If u = 9 Then st = st + "DIEZ Y NUEVE "

```

' {8-} end; {fin del else}

End If

' {7-} end; {fin del d=1}

```

If ((d = 2) And (u = 0)) Then st = st + "VEINTE "
If ((d = 2) And (u <> 0)) Then st = st + "VEINTI "

```

```

If d = 3 Then st = st + "TREINTA "
If d = 4 Then st = st + "CUARENTA "
If d = 5 Then st = st + "CINCUENTA "
If d = 6 Then st = st + "SESENTA "
If d = 7 Then st = st + "SETENTA "
If d = 8 Then st = st + "OCHENTA "
If d = 9 Then st = st + "NOVENTA "

```

```

If ((d <> 0) And (d <> 2) And (d <> 1) And (u <> 0)) Then st = st + "Y "

```

```

If ((u = 1) And (d <> 1)) Then st = st + "UN "
If ((u = 2) And (d <> 1)) Then st = st + "DOS "
If ((u = 3) And (d <> 1)) Then st = st + "TRES "
If ((u = 4) And (d <> 1)) Then st = st + "CUATRO "
If ((u = 5) And (d <> 1)) Then st = st + "CINCO "
If ((u = 6) And (d <> 1)) Then st = st + "SEIS "
If ((u = 7) And (d <> 1)) Then st = st + "SIETE "
If ((u = 8) And (d <> 1)) Then st = st + "OCHO "
If ((u = 9) And (d <> 1)) Then st = st + "NUEVE "

```

```

If ((I = 1) And (u + d + c <> 0) And ((d + c) = 0) And (u = 1)) Then st = st + "MILLON "

```

```

If ((I = 1) And (u + d + c <> 0) And ((d + c) <> 0) And (u <> 1)) Then st = st +
"MILLONES "
If ((I = 2) And ((u + d + c) <> 0)) Then st = st + "MIL "

```

Next I

```

n = n * 100
If n <> 0 Then
    n = n * 100 + 1

    n = Int(n / 100)
    nstring = Str(n)
    st = st + "PESOS " + nstring + "/100 M.N.)"
Else
    st = st + "PESOS 00/100 M.N.)"
End If

```

letras = st

End Function

' {1-} end; {end del procedimiento}

' Fin Proced

## La grabadora de macros

Microsoft Excel lleva incluida una utilidad que sirve para registrar acciones que se llevan a cabo en un libro de trabajo y registrarlas en forma de macro. Podemos aprovechar esta utilidad para generar código engorroso por su sintaxis un tanto complicada de recordar, además de ahorrar tiempo. Casi siempre después deberemos modificarlo para adaptarlo a nuestros programas, sin embargo eso resultará sumamente sencillo. Vea el ejemplo siguiente que sirve para poner bordes al rango de celdas de A1 a G6, observe los comentarios para saber qué bordes se ponen y dónde se ponen.

Poner bordes al rango que va de A1 a G6.

```

Sub Poner_Bordes()
    ' Seleccionar el rango A1:G6
    Range("A1:G6").Select
    ' No hay borde diagonal hacia abajo
    Selection.Borders(xlDiagonalDown).LineStyle = xlNone
    ' No hay borde diagonal hacia arriba
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    ' Borde izquierdo de la selección
    With Selection.Borders(xlEdgeLeft)
        .LineStyle = xlContinuous 'Estilo de línea continuo
        .Weight = xlMedium ' Ancho de línea Medio
        .ColorIndex = xlAutomatic ' Color de línea automático (negro)
    End With
    ' Borde superior de la selección

```

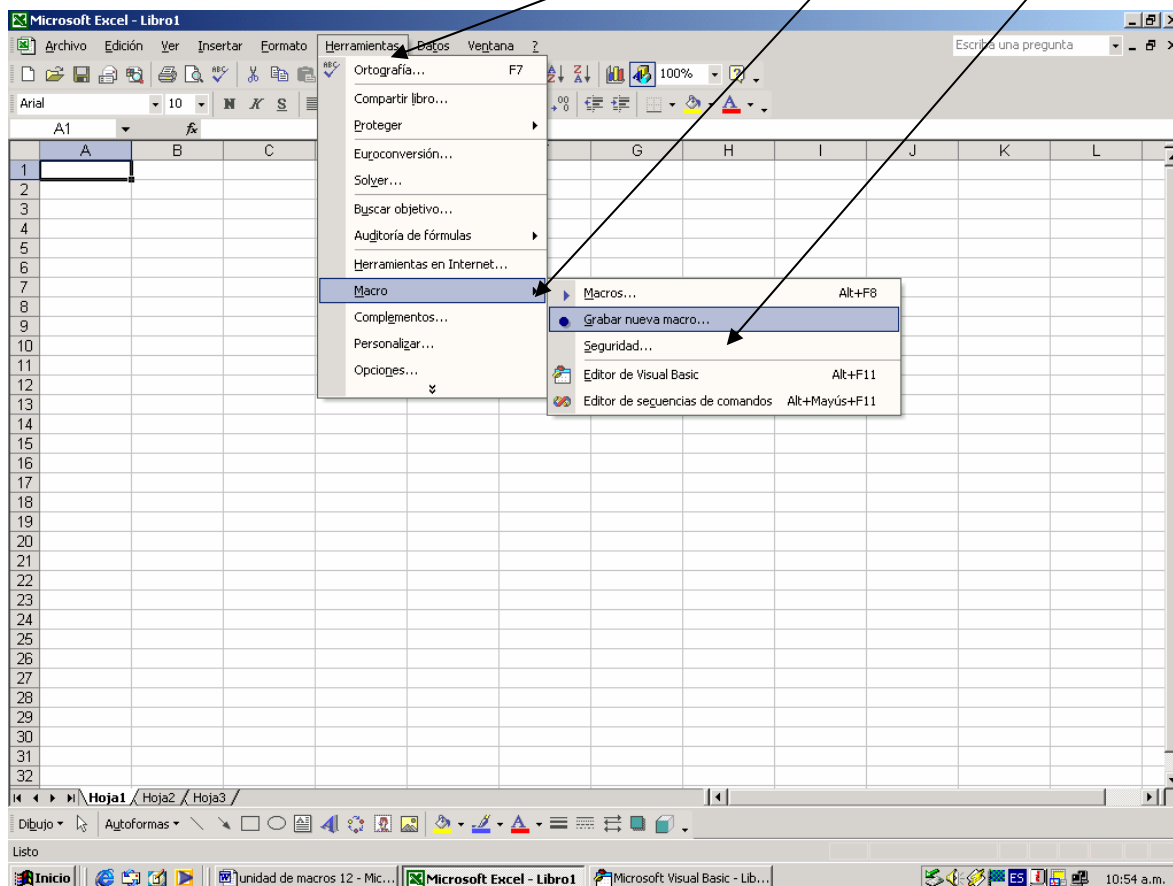
```
With Selection.Borders(xlEdgeTop)
.LineStyle = xlContinuous
.Weight = xlMedium
.ColorIndex = xlAutomatic
End With
' Borde inferior de la selección
With Selection.Borders(xlEdgeBottom)
.LineStyle = xlContinuous
.Weight = xlMedium
.ColorIndex = xlAutomatic
End With
' Borde derecho de la selección
With Selection.Borders(xlEdgeRight)
.LineStyle = xlContinuous
.Weight = xlMedium
.ColorIndex = xlAutomatic
End With
' Bordas verticales interiores de la selección
With Selection.Borders(xlInsideVertical)
.LineStyle = xlContinuous
.Weight = xlThin ' Ancho Simple.
.ColorIndex = xlAutomatic
End With
' No hay bordes horizontales interiores en la selección
Selection.Borders(xlInsideHorizontal).LineStyle = xlNone
' Seleccionar rango A1:G1
Range("A1:G1").Select
' No hay borde diagonal hacia arriba
Selection.Borders(xlDiagonalDown).LineStyle = xlNone
' No hay borde diagonal hacia arriba
Selection.Borders(xlDiagonalUp).LineStyle = xlNone
' Borde izquierdo de la selección
With Selection.Borders(xlEdgeLeft)
.LineStyle = xlContinuous
.Weight = xlMedium
.ColorIndex = xlAutomatic
End With
' Borde superior de la selección
With Selection.Borders(xlEdgeTop)
.LineStyle = xlContinuous
.Weight = xlMedium
.ColorIndex = xlAutomatic
End With
' Borde inferior de la selección
With Selection.Borders(xlEdgeBottom) ' Doble línea
.LineStyle = xlDouble
.Weight = xlThick
.ColorIndex = xlAutomatic
End With
' Borde derecho de la selección
With Selection.Borders(xlEdgeRight)
.LineStyle = xlContinuous
.Weight = xlMedium
.ColorIndex = xlAutomatic
End With
```



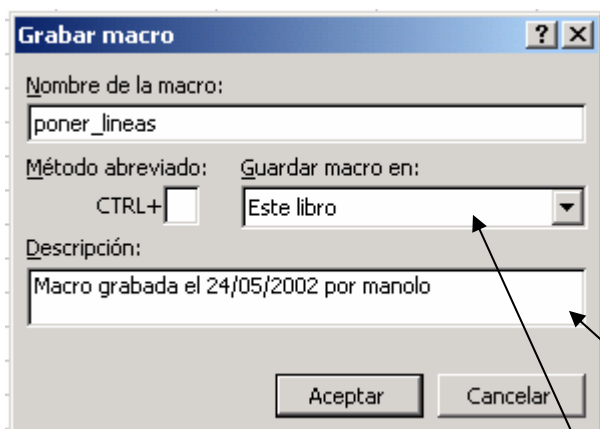
## End Sub

Suponemos que el procedimiento anterior le parecerá muy complicado, no se preocupe, a continuación explicaremos lo que hemos hecho y verá que lo único que debe hacer son los pasos sobre la hoja de cálculo, el grabador de macros se ocupa del resto. A continuación haremos la macro, activando la grabadora de macros que viene en Excel, siguiendo los pasos descritos a continuación:

Seleccione de la barra de menús “**Herramientas**” – “**Macro**” – “**Grabar nueva macro**”



En **Nombre de Macro**, pon *Poner\_Líneas*.



En **Guardar Macro en**, deje la opción **Libro Activo**.

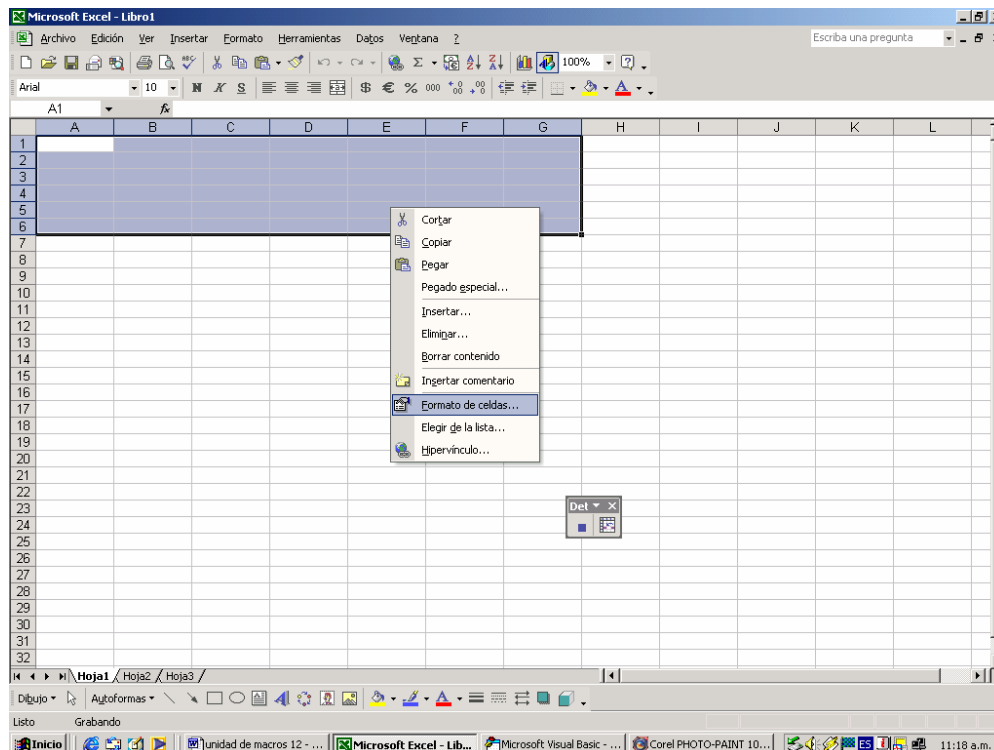
En **Descripción**, ponga, *macro para bordes a un rango de celdas*.

Pulse sobre el botón **Aceptar**.

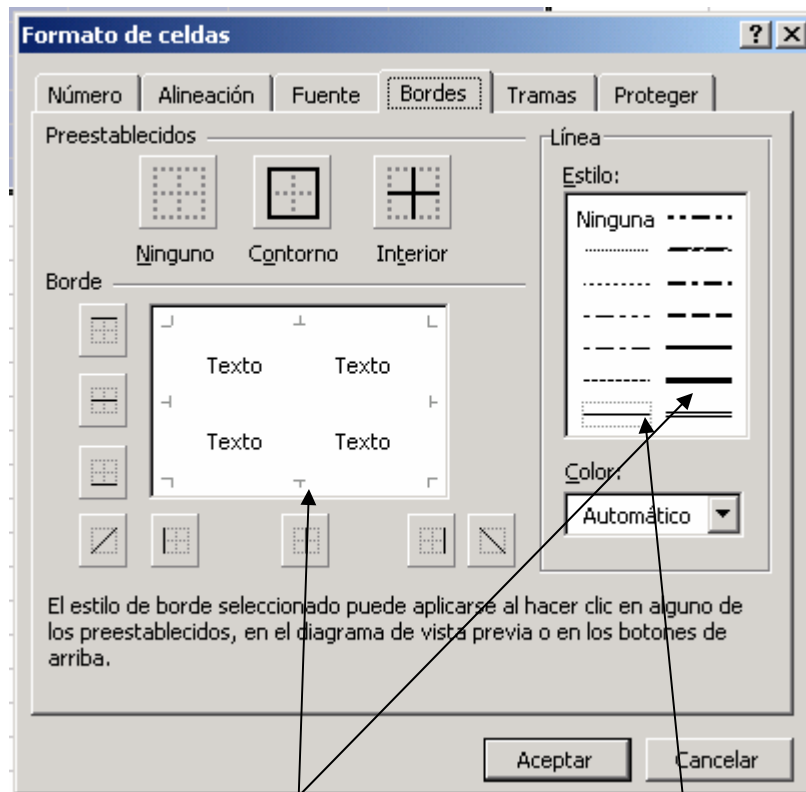
Ejecute los pasos siguientes, los cuales son para poner los bordes

Seleccione el rango A1:G6

Active el menú contextual pulsando el botón derecho del ratón sobre la parte seleccionada y seleccione la opción de **Bordes**.



En la ventana de Formato de celdas, seleccione la pestaña de bordes.



Ponga línea alrededor de la selección

En cuadro **Estilos**, seleccione la penúltima o antepenúltima línea.

Pulse sobre el botón que representa un cuadrado

Ponga las líneas verticales

Seleccione del cuadro estilo la última línea de la primera columna

Pulse sobre el botón que representa las líneas interiores de la selección (el situado en el centro de la línea inferior de botones)

Pulse sobre el botón **Aceptar**

Seleccione el rango **A1:G1**

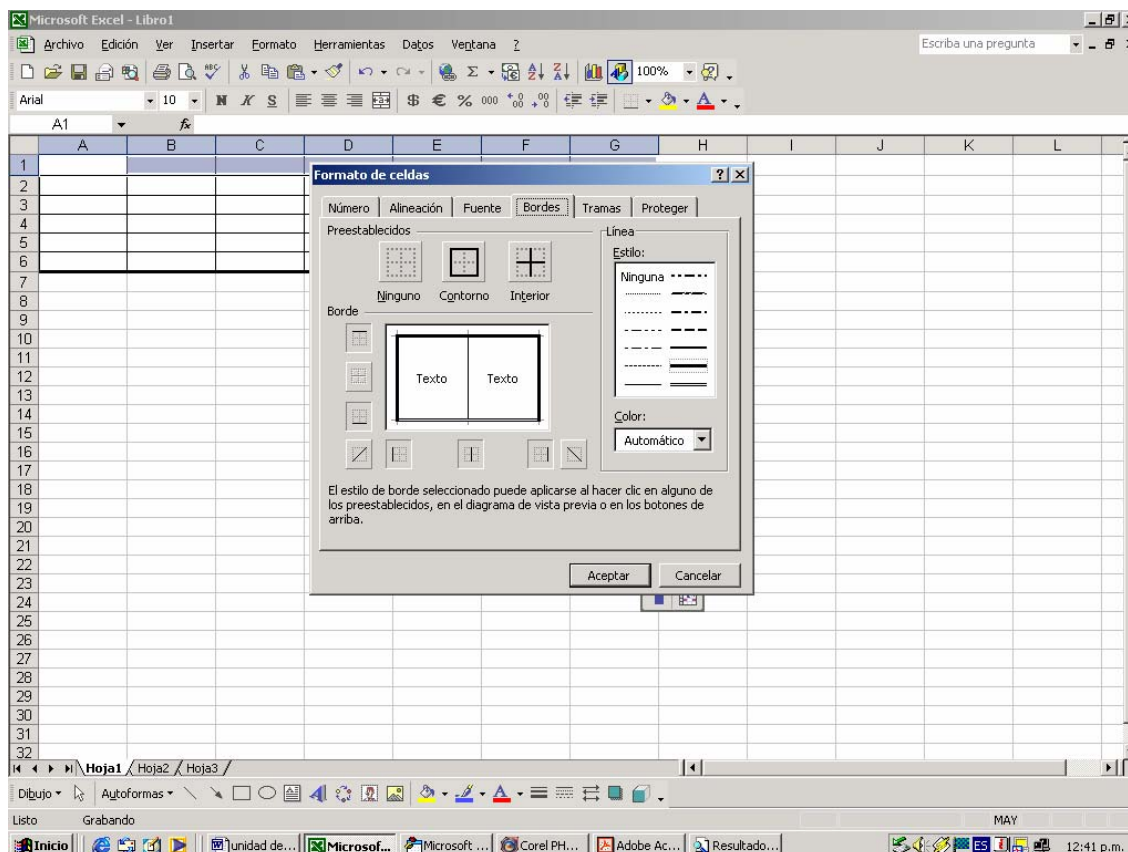
Active de nuevo la opción “**Formato**” –“**Celdas**” y la pestaña **Bordes**


Ponga línea inferior doble.

Seleccione del cuadro Estilo la última línea de la segunda columna.

Pulse sobre el botón que representa la línea inferior de la selección.

Pulse sobre el botón **Aceptar**.



Detén la grabadora de Macros pulsando sobre el botón  o bien activa de la barra de menús “**Herramientas**” – “**Macros**” – “**Detener Grabación**”

Y ya está, ahora abre el editor de Visual Basic y comprueba cómo la grabadora ha hecho todo el trabajo por tí. Ya sólo nos queda modificar la macro de forma que este formato de bordes sea aplicable a cualquier rango como lo veremos en el ejemplo siguiente, en el cual modificaremos la macro de manera que sirva para dar ese formato a cualquier rango de la hoja. Observa que en este ejemplo simplemente pasamos como parámetro el rango a formatear de manera que la cabecera del procedimiento, esto quedaría como sigue:

*' Procedimiento Poner\_Bordes.*

*' Procedimiento que pone bordes a un rango. Concretamente lo encierra en un cuadrado,  
' le pone líneas verticales y pone una doble línea inferior en la primera fila.*

*' Parámetros:*

*' Nombre\_Hoja: Nombre de la hoja donde está el rango.*

*' Rango\_Total : Rango al que se le aplica el formato.*

*' Rango\_Primer\_Fila : Rango de la primera fila al que se debe aplicar doble línea inferior.*

**Sub** Poner\_Bordes (Nombre\_Hoja **As String**, Rango\_Total **As String**, Rango\_Primer\_Fila **As String**)

*' Seleccionamos la hoja en la que se encuentra en rango*

WorkSheets(Nombre\_Hoja).Activate

*' Seleccionamos en rango*

ActiveSheet.Range(Rango\_Total).Select

*' Hacemos cuadro y líneas verticales.*

.....

.....

*' Selección de la primera fila*

ActiveSheet.Range(Primera\_Fila).Select

*' Hacemos línea inferior doble*

.....

**End Sub.**

Haciendo referencia a la macro del ejemplo anterior, la perfeccionaremos aún, sobre todo en lo referente a los parámetros. Básicamente cambiaremos el tipo de parámetros, así en lugar del nombre de la hoja pasaremos el número y en lugar de los parámetros Rango\_Total y Rango\_Primer\_Fila, simplemente pasaremos el rango de la casilla inicial, la macro se encargará de seleccionar todas las casillas adyacentes y de buscar la primera fila. En esta macro además se han incluido funcionalidades como borrar los formatos antes de aplicar las líneas, ajustar el ancho de las columnas, etc. Lea los comentarios para ver qué hace cada sección de la macro. Observe la propiedad **CurrentRegion** del objeto **Range**, esta propiedad devuelve el rango de las casillas llenas adyacentes a una dada. Por ejemplo imagine una hoja con el rango A1:B10 lleno de valores, la instrucción.

ActiveSheet.Range("A1").CurrentRegion.Select  
Seleccionaria el rango correspondiente a A1:B10.

*' Función Poner\_Líneas\_Selección.*

*' Esta función sirve para poner bordes a un rango.*

*' Pone cuadro al rango, separa las columnas con*

*' una línea y pone doble línea inferior en la ' primera fila.*

,

*' Parámetros:*

*' Num\_Hoja = Número de hoja donde está el rango a formatear.*

*' Casilla = Casilla inicial (superior izquierda) del rango a formatear*

**Sub** Poner\_Líneas\_Seleccion(Numero\_Hoja **As Integer**, Casilla **As String**)

**Dim** x **As Integer**

**Dim** Casilla\_Inicial **As String**

**Dim** Casilla\_Final **As String**

Worksheets(Numero\_Hoja).Activate

ActiveSheet.Range(Casilla).Activate

*' Seleccionar el rango de celdas con valores adyacentes a la activa*

ActiveCell.CurrentRegion.Select

*' Borrar los bordes actuales de la selección actuales*

**With** Selection

.Borders(xlDiagonalDown).LineStyle = xlNone

.Borders(xlDiagonalUp).LineStyle = xlNone

```
.Borders(xlEdgeLeft).LineStyle = xlNone
.Borders(xlEdgeTop).LineStyle = xlNone
.Borders(xlEdgeBottom).LineStyle = xlNone
.Borders(xlEdgeRight).LineStyle = xlNone
.Borders(xlInsideVertical).LineStyle = xlNone
.Borders(xlInsideHorizontal).LineStyle = xlNone
```

**End With**

*' Líneas del recuadro*

**With** Selection.Borders(xlEdgeLeft)

```
.LineStyle = xlContinuous
```

```
.Weight = xlMedium
```

```
.ColorIndex = xlAutomatic
```

**End With.**

**With** Selection.Borders(xlEdgeTop)

```
.LineStyle = xlContinuous
```

```
.Weight = xlMedium
```

```
.ColorIndex = xlAutomatic
```

**End With**

**With** Selection.Borders(xlEdgeBottom)

```
.LineStyle = xlContinuous
```

```
.Weight = xlMedium
```

```
.ColorIndex = xlAutomatic
```

**End With**

**With** Selection.Borders(xlEdgeRight)

```
.LineStyle = xlContinuous
```

```
.Weight = xlMedium
```

```
.ColorIndex = xlAutomatic
```

**End With**

*' Líneas verticales interiores, si en el*

*' rango hay más de una columna.*

**If** Selection.Columns.Count > 1 **Then**

**With** Selection.Borders(xlInsideVertical)

```
.LineStyle = xlContinuous
```

```
.Weight = xlThin
```

```
.ColorIndex = xlAutomatic
```

**End With**

**End If**

*' Ajustar ancho de columnas*

```
Selection.Columns.AutoFit
```

*' Línea doble inferior de primera fila*

*' Para este proceso se selecciona la casilla inicial pasada a la macro, luego se busca*

*' la última casilla con datos. Se construye un rango combinando las direcciones de ambas*

*' casillas y se utiliza el objeto Range junto con el método Select para hacer la selección*

*,*

*' RANGE(Inicial:Final).Select*

*' Seleccionar primera casilla*

```
ActiveSheet.Range(Casilla).Select
```

*' Buscar primera casilla vacía de misma fila recorriendo las columnas*

```
x = 0
```

**Do While Not** IsEmpty(ActiveCell.Offset(0, x))

```
x = x + 1
```

**Loop**

*' Recoger las direcciones de la casilla inicial y la casilla final, es decir las referencias*

*' FilaColumna (A1, A2,...)*

```

Casilla_Inicial = ActiveCell.Address
Casilla_Final = ActiveCell.Offset(0, x - 1).Address.
' Seleccionar el rango de la fila. Observar la concatenación de las cadenas de
' Casilla_Inicial y casilla_final que representan las direcciones del rango a seleccionar
ActiveSheet.Range(Casilla_Inicial & ":" & Casilla_Final).Select
' Poner doble línea
With Selection.Borders(xlEdgeBottom)
.LineStyle = xlDouble
.Weight = xlThick
.ColorIndex = xlAutomatic
End With
ActiveSheet.Range(Casilla).Activate
End Sub

```

Bueno, este segundo ejemplo, un poco más sofisticado es sólo para que vea hasta qué punto se puede refinar una macro. Primero construimos una macro con la grabadora que ponía formato a un rango de casillas que siempre era el mismo, después modificamos esta macro de manera que el rango de casillas que pudiera formatear fuera cualquiera, como tercer paso, además de conseguir lo mismo que la segunda versión pero con menos parámetros le hemos añadido nuevas funcionalidades como ajustar el ancho de las celdas, borrar los formatos previos, etc. y esta macro todavía podría refinarse más, poner color de fondo a la primera fila, poner color a los datos numéricos, etc. lo que intentamos decirle es que aproveche al máximo la utilidad de la grabadora de macros, pero tenga en cuenta que casi siempre tendrá que añadir código usted mismo, si quiere ampliar la funcionalidad de la macro, sobre todo si quiere aplicarle cierta generalidad.

Vea el siguiente ejemplo, sirve para representar gráficamente un rango de valores. La macro se ha hecho de forma similar a la anterior, es decir, una vez activada la grabadora de macros se han seguido todos los pasos necesarios para diseñar el gráfico, luego se han cambiado las referencias a hojas y rangos por variables que se colocan como parámetros del procedimiento para así dotarle de generalidad.

```

' Procedimiento Gráfico.
' Procedimiento que representa gráficamente los valores de un rango.
' Parámetros:
' Hoja_Datos : Hoja donde está el rango de valores.
' Rango_Datos : Rango de valores a representar gráficamente.
' Titulo : Título para el gráfico.
' Eje_X : Título para el eje X
' Eje_Y : Título para el eje Y
Sub Grafico(Hoja_Datos As String, Rango_Datos As String, Titulo As String, Eje_X As String,
Eje_Y
As String)
' Añadir el gráfico
Charts.Add
' Tipo de gráfico-> xlColumnClustered (Columnas agrupadas)
ActiveChart.ChartType = xlColumnClustered
' Definir el origen de datos y representar las series(PlotBy) por filas (xlRows)
ActiveChart.SetSourceData Source:=Sheets(Hoja_Datos).Range(Rango_Datos), PlotBy:= _
xlRows
' El gráfico debe ponerse en una hoja nueva

```

```
ActiveChart.Location Where:=xlLocationAsNewSheet
With ActiveChart
    ' Tiene título
    .HasTitle = True
    ' Poner título
    .ChartTitle.Characters.Text = "Ventas de Frutas"
    ' Tiene título para el eje X
    .Axes(xlCategory, xlPrimary).HasTitle = True
    ' Título para el eje X
    .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "Meses"
    ' Tiene título para el eje Y principal
    .Axes(xlValue, xlPrimary).HasTitle = True
    ' Título para el eje Y principal
    .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Cantidades"
End With
' Poner líneas de división para el eje de categorías X (horizontales)
With ActiveChart.Axes(xlCategory)
    ' Poner Líneas de división primarias
    .HasMajorGridlines = True
    ' No poner líneas de división secundarias
    .HasMinorGridlines = False
End With

' Poner líneas de división para el eje Y (verticales)
With ActiveChart.Axes(xlValue)
    .HasMajorGridlines = True
    .HasMinorGridlines = False
End With
' Tiene leyenda
ActiveChart.HasLegend = True
' Seleccionar leyenda
ActiveChart.Legend.Select
' Situar la leyenda en la parte inferior
Selection.Position = xlBottom
ActiveChart.ApplyDataLabels Type:=xlDataLabelsShowNone, LegendKey:=False
End Sub
```

Para finalizar, seguro que cuando utilice la grabadora aparecerán muchas propiedades y métodos que desconoce, sólo debe hacer clic sobre estas propiedades o métodos y pulsar F1, automáticamente se activará la ayuda sobre esa propiedad o método concreto.

## Insertar funciones de Microsoft Excel desde Visual Basic

Copie el siguiente procedimiento y ejecútelo. Es un procedimiento que sencillamente va pidiendo números y los va colocando en las celdas de la columna A a partir de A1, al final coloca la función =SUMA para sumar los valores introducidos y la función =PROMEDIO para hacer el promedio de los mismos valores.

```
Sub Sumar()
Dim Valor As Integer
Dim Casilla_Inicial As String
```



**Dim** Casilla\_Final **As String***' Hacer activa la casilla A1 de la hoja activa*

ActiveSheet.Range("A1").Activate

**Do***' Entrar un valor y convertirlo a numérico*

Valor = Val(InputBox("Entrar un valor", "Entrada"))

*' Si el valor es distinto de 0***If** Valor <> 0 **Then***' Guardar el valor en la casilla activa*

ActiveCell.Value = Valor

*' Hacer activa la casilla de la fila siguiente*

ActiveCell.Offset(1, 0).Activate

**End If****Loop Until** Valor = 0.*' Establecer la casilla inicial del rango a sumar*

Casilla\_Inicial = "A1"

*' Establecer la casilla final del rango a sumar.**' Cogér la dirección de la casilla activa, la última*

Casilla\_Final = ActiveCell.Address

ActiveCell.Offset(1, 0).Activate

*' Poner en la casilla activa la función SUMA*

ActiveCell.Formula = "=Suma(" &amp; Casilla\_Inicial &amp; ":" &amp; Casilla\_Final &amp; ")"

ActiveCell.Offset(1, 0).Activate

*' Poner en la casilla activa la función promedio*

ActiveCell.Formula = "=Promedio(" &amp; Casilla\_Inicial &amp; ":" &amp; Casilla\_Final &amp; ")"

**End Sub**

Una vez que haya ejecutado la macro, observe que en las celdas donde se han colocado respectivamente las funciones =SUMA, =PROMEDIO aparece **¿NOMBRE?** (es posible que aparezca #####, en ese caso amplíe la columna), esto significa que Excel no reconoce el nombre de la función, que no existe. Sin embargo, estas funciones sí existen y funcionan perfectamente cuando se teclean directamente sobre la hoja de cálculo, se preguntará el por qué cuando se colocan desde una macro no funcionan. Pues resulta que para que cualquier función de Excel insertada desde una macro NO dé error, debe ponerse con su nombre en Inglés, la traducción se hace luego de forma automática. Es decir en la macro debe ponerla en inglés y luego cuando ésta se inserte en la hoja aparecerá con su nomenclatura en el idioma que corresponda.

Modifica el procedimiento del ejemplo y en lugar de poner

ActiveCell.Formula = "=Suma(" &amp; Casilla\_Inicial &amp; ":" &amp; Casilla\_Final &amp; ")"

Pon

ActiveCell.Formula = "=Sum(" &amp; Casilla\_Inicial &amp; ":" &amp; Casilla\_Final &amp; ")"

Y ahora, en lugar de

ActiveCell.Formula = "=Promedio(" &amp; Casilla\_Inicial &amp; ":" &amp; Casilla\_Final &amp; ")"

Pon

ActiveCell.Formula = "=Average(" & Casilla\_Inicial & ":" & Casilla\_Final & ")"

Ejecute la macro y compruebe que ahora todo funciona correctamente. Observe que en la hoja, las funciones se han insertado con su nombre correcto según el idioma, es decir SUMA y PROMEDIO.

A continuación le explicaremos cómo puede averiguar el nombre de cualquier función en inglés.

Utilizaremos la grabadora de macros. Como ejemplo obtendremos el nombre de la función =PROMEDIO. Deberá seguir los mismos pasos para obtener el nombre de cualquier función.

Active la grabadora de macros.

Vaya a una casilla cualquiera, C1 por ejemplo y teclee la función tal como lo haría en su idioma. Por ejemplo, ponga =PROMEDIO(A1:A10) o el nombre de cualquier otra función Excel.

Detenga la ejecución de la macro.

Edite la macro y observe el nombre que se ha puesto, ya sólo debe apuntárselo y pasarlo a su procedimiento. Es posible que la función que vea tenga una nomenclatura que le suene rara y es que la grabadora de macros utiliza referencias tipo RC (row column).

## **Detección de errores y depuración de programas**

A medida que los programas van creciendo la probabilidad de cometer errores también va creciendo. Los errores se clasifican normalmente en tres categorías.

### **Errores en tiempo de compilación**

Son los típicos errores que impiden hacer funcionar el programa debido, por ejemplo, a errores de sintaxis en las instrucciones, llamadas a funciones que no existen o llamadas con el tipo o el número de parámetros incorrectos, etc. Este tipo de errores no dan demasiados problemas, primero porque el compilador avisa de dónde se han producido y luego porque simplemente revisando la sintaxis se solucionan rápidamente.

## Errores en tiempo de ejecución

Estos errores se producen por una mala programación del código al no haber previsto determinados casos concretos o especiales, como por ejemplo intentar abrir un archivo que no existe, imprimir sin comprobar que la impresora está conectada, definir mal la dimensión de un array e intentar acceder a miembros que no existen, etc. Cuando se produce este tipo de errores se detiene la ejecución del programa y normalmente se informa del tipo de error que se ha producido. Muchos de estos errores se pueden solucionar mediante rutinas o funciones de tratamiento de errores, estudiaremos este tipo de rutinas un poco más adelante.

## Errores de función

Son los más complicados de detectar ya que ni se detectan en la fase de ejecución, ni provocan la detención del programa, son debidos a la incorrecta programación de algún proceso y como resultado se obtienen datos erróneos. Errores de este tipo son cálculos mal hechos, ciclos infinitos, devolución de valores incorrectos, etc. Como ni los detecta el compilador, ni provocan la interrupción del programa deben revisarse a mano, y claro, si el programa es extenso y trata muchos datos, su detección puede resultar dificultosa. Visual Basic, y todos los entornos de programación incorporan herramientas para facilitar la detección de este tipo de errores, son las herramientas de depuración. Antes de comenzar a describir cómo funcionan estas herramientas, le recomendamos, una vez más, que modularice su programa utilizando procedimientos cortos que realicen trabajos concretos y precisos, de esta forma conseguirá, además de que el programa quede más elegante y en un futuro sea más sencillo modificarlo, evitar el tener que revisar grandes bloques de código para detectar errores de este tipo.

## Herramientas de depuración

Como se acaba de indicar, estas herramientas son muy útiles a la hora de probar paso a paso el funcionamiento del programa y detectar los procesos que provocan un mal funcionamiento del mismo.

Copie los datos siguientes en la primera hoja de un libro de trabajo, estos datos serán utilizados por las funciones que utilizaremos para explicar el funcionamiento de las herramientas de depuración.

	A	B	C	D	E	F	G
1		Enero	Febrero	Marzo	Abril	Mayo	Junio
2	1	3406	3336	3135	2402	4345	4891
3	2	2754	2807	3945	4780	3352	3946
4	3	3646	3704	3140	3236	2640	2052
5	4	2546	4275	4370	3193	2710	2670
6	5	3805	3533	4409	3227	3458	4917
7	6	2709	4509	3153	4894	4801	3454
8	7	2248	4293	3171	3834	3596	3258
9	8	2906	4530	3336	4770	2212	4141
10	9	3827	3538	3748	4800	3869	4896
11	10	3897	4052	4189	3132	4016	3593
12							


Copie el código siguiente, es el que utilizaremos para estudiar y ver ejemplos sobre las herramientas de depuración. La primera (*Sub Prueba*) recorre los datos de las columnas hasta encontrar una vacía, esta función va llamando a *Recorrer\_Columna*, sirve para recorrer las filas de una columna hasta encontrar una vacía, va sumando los valores que encuentra en las filas y va contando cuántas hay de llenas, al final llama a la función *Cálculos*, esta función coloca en respectivas casillas, la suma de los valores de la columna, la cantidad de celdas llenas que ha encontrado, y la media. Una vez que haya copiado las funciones ejecútelas para comprobar su correcto funcionamiento antes de proceder al estudio de las herramientas de depuración.

```
Sub Prueba()  
Worksheets(1).Range("B2").Activate  
' Recorrer las casillas de una fila hasta que se encuentre una vacía  
Do While Not IsEmpty(ActiveCell)  
    Call Recorrer_Columna  
    ActiveCell.Offset(0, 1).Activate  
Loop  
End Sub  
  
Private Sub Recorrer_Columna()  
Dim Suma_Columna As Long 'Suma de los valores de la columna  
Dim Mayor_Que_Cero As Integer ' Contar casillas con valores mayores que cero  
Dim Desp_Fila As Integer ' Incremento de Fila  
Suma_Columna = 0  
Mayor_Que_Cero = 0  
Desp_Fila = 0  
' Recorrer las filas de una columna hasta que se encuentre una vacía  
Do While Not IsEmpty(ActiveCell.Offset(Desp_Fila, 0))  
    If ActiveCell.Offset(Desp_Fila, 0).Value > 0 Then  
        Suma_Columna = Suma_Columna + ActiveCell.Offset(Desp_Fila, 0).Value  
        Mayor_Que_Cero = Mayor_Que_Cero + 1  
    End If  
    Desp_Fila = Desp_Fila + 1  
Loop  
Call Calcular(Suma_Columna, Mayor_Que_Cero, Desp_Fila)  
End Sub  
  
Private Sub Calcular(Suma As Long, Q As Integer, F As Integer)  
ActiveCell.Offset(F + 2, 0).Value = Suma  
ActiveCell.Offset(F + 3, 0).Value = Q  
ActiveCell.Offset(F + 4, 0).Value = Suma / Q  
End Sub
```

Activa la barra de depuración para ver los botones que se utilizarán en las secciones que se explican a continuación, para ello selecciona de la barra de menú “Ver” – “Barras de Herramientas” – “Depuración”.

### Modo Ejecución paso a paso por instrucciones

El modo paso a paso permite la ejecución del programa instrucción por instrucción, de esta forma es posible ver que el funcionamiento del programa es el correcto, es decir que la

ejecución de instrucciones sigue los pasos que se habían previsto. Para ejecutar un procedimiento paso a paso, sólo debes ir pulsando sobre el botón , activar la opción de menú **Depuración /Paso a Paso por Instrucciones** o ir pulsando la tecla **F8**, que seguramente es lo más cómodo.

## Ejemplo

Sitúe el cursor dentro del procedimiento Prueba.

Vaya pulsando la tecla **F8** y verá cómo se ejecuta una sola instrucción por cada pulsación.

Puede ir alternando con la hoja de cálculo para ver qué es lo que ocurre cada vez que se ejecuta una instrucción. Cuando esté ejecutando paso a paso puede utilizar los botones siguientes para llevar a cabo determinadas acciones.



Sirve para detener la ejecución del programa.



Sirve para ejecutar el resto del programa.



Sirve para ejecutar todo un procedimiento.

Cuando en la ejecución de un procedimiento, se llega a una línea que llama a otro procedimiento o función, pulsando este botón se puede provocar la ejecución de todo el código de esta función, para luego continuar con el modo paso a paso.

Para ver más claro, hagamos el siguiente ejemplo:

Sitúe el cursor dentro del procedimiento *Prueba*.

Vaya ejecutando paso a paso hasta la línea

*Call Recorrer\_Columna*



En este punto pulse el botón , observe cómo se ejecuta todo el procedimiento

*Recorrer\_Columna* para luego continuar con la ejecución normal de Paso a Paso. Para activar esta opción, también puede activar la opción **Depuración/ Paso a paso por procedimientos**, o bien pulsar la combinación **MAY+F8**.



Sirve para ejecutar todas las instrucciones del procedimiento activo y volver (o terminar).


## Ejemplo

Sitúe el cursor dentro del procedimiento *Prueba*.

Vaya ejecutando paso a paso hasta la instrucción.

*Mayor\_Que\_Cero = Mayor\_Que\_Cero + 1*

Ya dentro del procedimiento *Recorrer\_Columna*.

Pulse sobre el botón  verá cómo se termina la ejecución de este procedimiento y se vuelve al procedimiento *Prueba* para continuar con la ejecución paso a paso.


Para activar esta opción, también puede la opción *Depuración/ Paso a paso para salir*, o bien pulsar la combinación [ctrl.]+May]+[F8]

## El modo Interrupción

En programas largos resulta fastidioso tener que ejecutarlos paso a paso, sobre todo si sabemos que el error se produce en una parte avanzada del programa. El modo interrupción, permite la ejecución del programa hasta una instrucción determinada para, a partir de ésta, ejecutar paso a paso y así poder detectar el error.

### Definir puntos de interrupción

Sitúe el cursor sobre la instrucción en la cual debe detenerse el programa para continuar paso a paso.

Pulse sobre el botón . También puede activar la opción “**Depuración**” – “**Alternar punto de interrupción**”, pulsar la tecla F9 o bien hacer un clic en la parte izquierda de la ventana del módulo (la franja vertical en color gris).


*Para desactivar un punto de interrupción siga los mismos pasos.*

## Solucionar los errores

Todo lo dicho anteriormente no serviría de gran cosa si no fuera posible revisar los valores que las variables van cogiendo a medida que vamos avanzando, o si no tuviéramos ocasión de evaluar las expresiones del programa. En las siguientes secciones veremos cómo llevar a cabo estas acciones.


## Inspecciones rápidas de variables

Estas opciones sirven para revisar el valor de las variables a medida que se va ejecutando el programa. Para ver los valores que van tomando las variables es conveniente tener visible la **Ventana de inspección**, para activarla seleccione en la barra de menú “Ver” –

“**Ventana de Inspección**” o pulse sobre el botón .

## Añadir una variable a la ventana de inspección

Aunque no es necesario estar ejecutando el programa en modo paso a paso, es conveniente.

1. Seleccionar la variable que desea añadir a la ventana haciendo un clic sobre ella.
2. Pulse sobre el botón , también puede activar **Depuración/ Inspección rápida** o pulsar la combinación [May]+[F9]. Aparece un cuadro de diálogo donde se muestra el valor actual de la variable. Si no está ejecutando el programa paso a paso, aparecerá el valor **Fuera de Contexto**.
3. Pulse sobre el botón **Agregar** para añadir la variable a la ventana de inspección.

Debe tener en cuenta que para revisar las variables las expresiones que les asignan valores deben de ejecutarse al menos una vez.

A continuación haremos un ejemplo para dejar más en claro la interrupción de un programa.

Sitúa un punto de interrupción en la línea.

$$Mayor\_Que\_Cero = Mayor\_Que\_Cero + 1$$

Ejecuta el programa, cuando éste se detenga en el punto de interrupción, sitúa el cursor sobre la variable *Suma\_Columna* (puedes ponerlo en cualquier parte).

Pulsa sobre el botón .

Pulsa sobre el botón **Agregar** para que la variable se inserte en la ventana Inspecciones.

Repite los pasos anteriores para las variables *Mayor\_Que\_Cero* y *Desp\_Fila*

Ve ejecutando el programa paso a paso y observa cómo va cambiando el valor de las variables en la ventana de inspección.

Recuerda que puedes agregar una variable a la ventana de inspección aunque no esté ejecutando el programa.

*Como ya te habrás dado cuenta, cuando ejecutas el programa paso a paso, si*

*sitúas el puntero de ratón sobre una variable, se muestra el valor de la misma.*

### **Borrar una variable de la ventana de Inspección**

Sólo debes seleccionarla en la ventana de inspección y pulsar sobre la tecla **SUPR**.

### **Modificar el valor de una variable en tiempo de ejecución**

A veces resulta interesante cambiar el valor de alguna variable cuando se está ejecutando el programa, para ver qué ocurre si coge determinados valores, para terminar un ciclo, etc., para ello sigamos el ejemplo siguiente:

Sitúe un punto de interrupción en la línea.

*Mayor\_Que\_Cero = Mayor\_Que\_Cero + 1*

Agregue a la ventana de inspección (si no está) la variable *Suma\_Columna*.

Ejecute el programa, al detenerse, observe en la **Ventana de Inspección** que la variable *Suma\_Columna* tiene un valor que ahora cambiaremos.

Haga doble clic sobre el valor de *Suma\_Columna* dentro de la ventana de inspección.

Borre el valor que tiene, cámbielo por otro y pulse **ENTER**.

Ahora puede continuar con la ejecución normal del programa.

### **Expresiones de Revisión**

Además de permitir añadir una variable o expresión dentro de la **Ventana Inmediato**, una **Expresión de Revisión** permite interrumpir la ejecución del programa cuando una variable coge determinado valor.

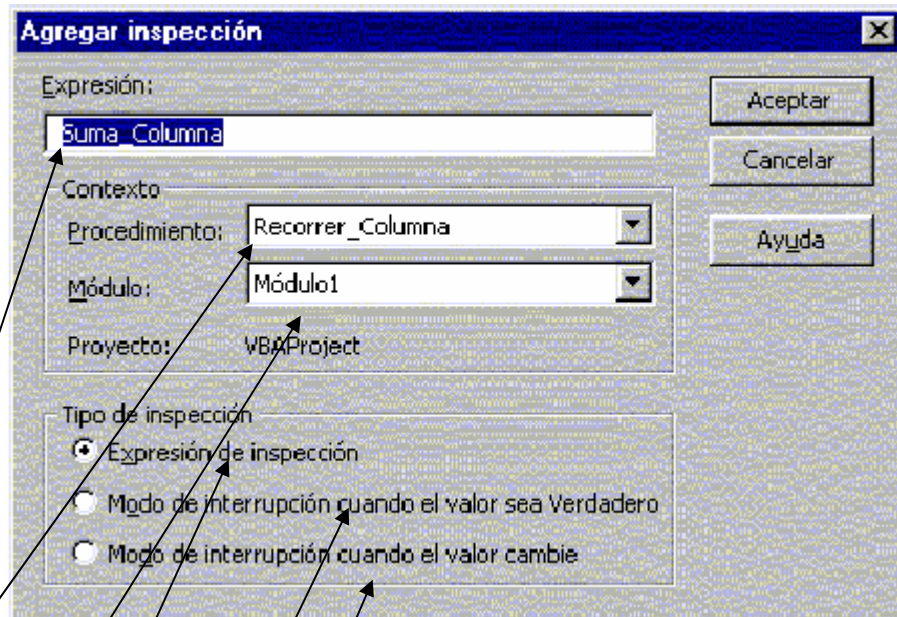
Piensa que muchas veces un programa deja de funcionar, o funciona mal cuando una variable toma determinados valores. Con una expresión de revisión, podremos detener la ejecución del programa cuando una variable contiene determinado valor (a partir de determinado valor), luego, podremos continuar con la ejecución paso a paso para ver qué ocurre a partir de este punto.

En el ejemplo que veremos a continuación, haremos que el programa se interrumpa cuando la variable *Suma\_Columna* alcance un valor superior a 30000.

Sitúa el cursor sobre el nombre de la variable *Suma\_Columna*, puede escoger cualquier posición donde aparece la variable.

Activa la opción **Depuración/ Agregar Inspección**. Aparece el siguiente cuadro de diálogo.





**Expresión :** Variable o expresión que se desea evaluar.

**Procedimiento:** Aquí se describe el procedimiento donde debe evaluarse la variable, esto significa que el ámbito de comprobación de la variable es sólo dentro de este procedimiento. Imagine que tiene dos o más procedimientos donde utiliza una variable con el mismo nombre, o bien que tiene una variable global, aquí se indica en qué procedimiento debe hacerse la evaluación.

**Módulo.** Lo mismo que en apartado procedimiento, pero a nivel módulo.

**Expresión de inspección.** Activando esta opción, indicamos que lo único que deseamos hacer es incluir la variable o expresión dentro de la ventana de expresión.

**Modo de interrupción cuando el valor sea verdadero.** Indicamos que cuando la expresión indicada en cuadro de texto Expresión sea cierta, debe detenerse el programa.

**Modo de interrupción cuando el valor cambie.** El programa se detendrá cuando la expresión cambie de valor.

Para nuestro ejemplo rellene con los datos siguientes:

**Expresión :** Suma\_Columna > 30000

**Procedimiento:** Recorrer\_Columna

**Módulo:** Módulo1 (o como se llame su módulo).

Active **Modo de interrupción cuando el valor sea verdadero.**

Pulse sobre botón **Aceptar.**

Ejecute el programa. Observe que el programa se interrumpe cuando *Suma\_Columna* coge un valor superior a 30000, a partir de éste podríamos continuar con la ejecución paso a paso y ver cómo evoluciona el valor de la variable o variables.

### La Ventana Inmediato

Es otra forma de inspeccionar variables cuando el programa está en modo interrupción (ejecutándose paso a paso), pero además, ofrece la posibilidad de cambiar valores de las variables e incluso ejecutar o evaluar expresiones. Para ver el valor de una variable en la ventana inmediato debe anteponerle un ? y luego pulsar **Enter**.

Para activar la ventana Inmediato, active opción “**Ver**” – “**Inmediato**”, o pulse la combinación [Ctrl]+[G].

En el siguiente ejemplo, utilizaremos la ventana Inmediato para ver el valor de las variables *Suma\_Columna*, *Mayor\_Que\_Cero* y *Desp\_Fila*. También cambiaremos el valor de una de ellas y comprobaremos una expresión.

Activa la ventana de Inmediato. “**Ver**” – “**Ventana Inmediato**”

Sitúa un punto de interrupción en la instrucción

*Call Calcular(Suma\_Columna, Mayor\_Que\_Cero, Desp\_Fila)*

Ejecuta el programa, éste debe detenerse cuando llegue a la instrucción indicada en paso 2.

Teclee en la ventana inmediato, haga las pruebas siguientes.

Escriba  
? *Suma\_Columna*  
? *Mayor\_Que\_Cero*  
? *Desp\_Fila*

Pruebe la expresión siguiente. En este caso concreto sólo sirve para ver que es una posibilidad.

$X = Suma\_Columna / Mayor\_Que\_Cero$   
? *X*

Para terminar, cambiaremos el valor de la variable *Suma\_Columna* y continuaremos la ejecución normal del programa.

*Suma\_Columna = -2350000*

Quite el punto de interrupción y termine la ejecución del programa.

## La instrucción Debug.Print

Esta instrucción, que se utiliza directamente sobre el código del programa, permite ver todos los valores que ha ido cogiendo una variable o expresión durante la ejecución del programa. Los valores se mostrarán en la ventana Inmediato una vez finalizado el programa. Esta expresión resulta útil en una fase avanzada de depuración, ya que permite ir viendo la evolución de una variable o expresión sin necesidad de poner puntos de interrupción. Evidentemente cuando el programa esté listo deben de sacarse.

A continuación usaremos la instrucción **Debug.Print** veremos la evolución de la variable *Suma\_Columna*.

Sitúe el cursor después de la instrucción

```
Suma_Columna = Suma_Columna + ActiveCell.Offset(Desp_Fila, 0).Value  
Escriba: Debug.Print "Dentro del Ciclo : " & Suma_Columna..
```

Sitúe el cursor después de la instrucción Loop y escriba

```
Debug.Print " Fuera del Ciclo : " & Suma_Columna.
```

Ejecute el programa (sin puntos de interrupción).

Una vez terminada la ejecución, observe lo que hay escrito en la ventana inmediato. A veces, resulta interesante controlar en qué pasos la variable ha ido cogiendo determinados valores, para hacer esto deberán declararse las correspondientes variables que hagan las funciones de contador.

Haremos lo mismo que en el ejemplo anterior pero indicando los pasos de ciclo y las llamadas al procedimiento *Recorrer\_Columna*.

- Declara a nivel global la variable *Contar\_Llamadas* de tipo Integer.
- Declara dentro del Procedimiento *Recorrer\_Columna* la variable *Pasos\_De\_Ciclo* de tipo Integer.
- Inicialice a 1 la variable *Contar\_Llamadas* dentro de la función *Prueba*, hágalo antes de la instrucción *Do While*.
- Debajo de la instrucción *ActiveCell.Offset(0, 1).Activate* ponga *Contar\_Llamadas = Contar\_Llamadas+1*
- Inicialice a 1 la variable *Pasos\_De\_Ciclo* dentro del procedimiento *Recorrer\_Columna*, hágalo antes de la instrucción *Do While*.
- Debajo de la instrucción *Desp\_Fila = Desp\_Fila + 1*, ponga *Pasos\_De\_Ciclo = Pasos\_De\_Ciclo+1*
- Cambie las expresiones *Debug.Print* por, *Debug.Print "Paso de Ciclo: " & Paso\_De\_Ciclo & " Suma\_Columna = " & Suma\_Columna Debug.Print " Llamada : " & Contar\_Llamadas & " Suma\_Columna = " & Suma\_Columna*

- Ejecute el programa y observe la salida en la ventana Inmediato.

Por supuesto cuando el programa esté terminado y comprobado, deberá quitar todas las instrucciones Debug.Print.

Es muy importante utilizar las herramientas de depuración, no sólo a la hora de localizar errores de programación sino también a la hora de comprobar la evolución del programa cuando se dan determinadas condiciones. Déjenos acabar el tema insistiendo de nuevo en la importancia que modularice sus programas, observe que si lo hace, también le resultará mucho más sencillo detectar errores de programación.

## Errores de Ejecución

Es imposible excluir del todo los errores en los programas. Si además, y como es de desear, el programa será utilizado por usuarios que no han tenido nada que ver en el proceso de desarrollo e implementación posiblemente (seguramente) se producirán errores debido a su incorrecta utilización. Estos errores son los que se deben prevenir. Errores de este tipo son, por ejemplo, intentar acceder a un archivo inexistente, entrar valor es incorrecto a través de un cuadro de diálogo o formulario (datos tipo String cuando se requieren números,...). También entrarían en este tipo de errores aquellos casos excepcionales pero que deben ser previstos por el programador, como por ejemplo que se llene la unidad de disco, que la impresora se quede sin papel (éste ya no es tan excepcional), etc.

Visual Basic y la mayoría de los lenguajes de programación permiten implementar rutinas de tratamiento de errores cuya finalidad es interceptar determinados tipos de errores que se producen en tiempo de ejecución.

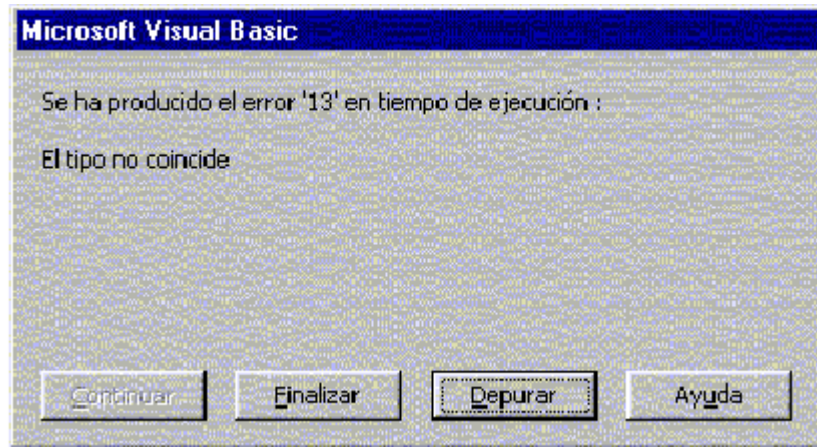
La finalidad de estas rutinas es que el programa no se detenga, o al menos si se detiene, informar sobre la posible causa del error e intentar controlarlo de alguna forma. Estudiaremos a continuación cómo se realiza esto en Visual Basic.

Copia el módulo siguiente, será el que utilizaremos en los ejemplos. Es un simple procedimiento que pide dos valores al usuario, los suma y los guarda en la celda A1 de Hoja2.

```
Option Explicit  
Sub Prueba()  
  Dim n1 As Integer  
  Dim n2 As Integer  
  Dim total As Integer  
  n1 = InputBox("Entrar un valor", "Entrada")  
  n2 = InputBox("Entrar otro valor ", "Entrada")  
  total = n1 + n2  
  Worksheets("Hoja2").Range("A1").Value = total  
End Sub.
```

## Rutinas de tratamiento de errores

Generalmente una rutina de tratamiento de errores reacciona ante casos esperados por el programador. En el ejemplo que nos ocupa, podría ser que el usuario entrará a caracteres en lugar de números por lo que el programa generaría el error siguiente.



Mediante una rutina de tratamiento de errores informaremos del error que se ha producido y direccionalaremos la ejecución del programa hacia donde interese. En Visual Basic el tratamiento de errores es una parte normal de la ejecución del programa. La instrucción para el tratamiento de errores es **ON ERROR GOTO línea**, línea es una etiqueta o marca de línea que indica hacia dónde debe dirigirse el programa en caso de error. El formato general de un procedimiento o función donde se implementa una rutina de tratamiento de errores es la siguiente:

```
Sub prueba()
On Error GOTO Tratar_errores
' Instrucciones del procedimiento
Exit Sub ' Salir del procedimiento
Tratar_Errores:
' Instrucciones de tratamiento de error
End Sub
```

Con **On Error GOTO Tratar\_Errores**, indicamos al programa que en caso que se produzca un error en tiempo de ejecución vaya a ejecutar las líneas que siguen a la etiqueta o marca de línea *Tratar\_Errores*.

Observe que antes de la etiqueta *Tratar\_Errores* hay la instrucción **Exit Sub**, sirve para que si el procedimiento se ha desarrollado correctamente, salga en ese punto, tenga en cuenta que si no se pusiera esta línea la ejecución continuaría secuencialmente y se ejecutarían las líneas de la rutina *Tratar\_Errores*.

Si la rutina de tratamiento de errores se implementa en una función debe poner **Exit Function** en lugar de **Exit Sub**.

## Escribir código de tratamiento de errores

En este primer ejemplo haremos algo muy simple, mostrar información sobre el tipo de error que se ha producido y detener la ejecución del programa. Vale, esto es lo que hace ya Visual Basic sin necesidad de implementar nada, pero nos servirá para introducir el objeto **Err**.

### *El objeto Err*

Siempre que se produce un error en tiempo de ejecución Visual Basic genera (o dispara como dicen algunos autores) un objeto tipo **Err**, estudiaremos dos propiedades de este objeto **Number** y **Description**.

**Number** es un número indicativo sobre el tipo de error que se ha producido, dicho de otra forma, cuando Visual Basic dispara un objeto **Err** en tiempo de ejecución, podemos revisar su propiedad **Number** para saber qué es lo que ha causado el error. **Description** es el mensaje asociado al número de error y es una cadena de texto que describe brevemente el error. Ahora en nuestro ejemplo, implementaremos una rutina de error que muestre el número de error y su descripción, insistimos que es lo que hace Visual Basic por sí solo, pero luego veremos cómo gracias al número de error podremos determinar la causa del error y controlar la ejecución del programa como más convenga. El procedimiento quedaría de la siguiente manera:

```
Sub Prueba()  
On Error GoTo Tratar_Errores  
Dim n1 As Integer  
Dim n2 As Integer  
Dim total As Integer  
n1 = InputBox("Entrar un valor", "Entrada")  
n2 = InputBox("Entrar otro valor ", "Entrada")  
total = n1 + n2  
Worksheets("Hoja2").Range("A1").Value = total  
Exit Sub  
Tratar_Errores:  
MsgBox ("Número de Error : " & Err.Number & Chr(13) & "Descripción : " & Err.Description)  
End Sub
```

Ejecuta el procedimiento anterior y cuando se le pida el primer número entre el Texto "Hola". Observe que como se ha producido un error, el programa salta hacia la rutina de tratamiento de errores y muestra el número de error y su descripción asociada.

En la mayoría de los casos, las rutinas de tratamiento de errores son para que el programa no se detenga. El proceso habitual cuando se produce el error es tratar de corregirlo y continuar la ejecución del programa. La continuación del programa se consigue mediante la instrucción **Resume**. Esta instrucción da tres posibilidades, ejecutar otra vez la instrucción que ha causado el error, continuar por la instrucción siguiente a la que ha causado el error o dirigir la ejecución hacia una línea marcada con una etiqueta.

**Resume.** Ejecutar de nuevo la instrucción que ha causado el error.

Esto, en nuestro caso, sería volver a preguntar de nuevo el valor. La rutina de tratamiento de errores quedaría.

**Tratar\_Errores:**

MsgBox ("Número de Error : " & Err.Number & Chr(13) & "Descripción : " & Err.Description)

**Resume**

Podríamos cambiar el mensaje que ve el usuario para hacerlo más comprensivo. La rutina quedaría:

**Tratar\_Errores:**

MsgBox ("Debe entrar un valor numérico")

**Resume**

**Resume Next.** Continuar el programa por la instrucción siguiente a la que ha causado el error.

**Tratar\_Errores:**

MsgBox ("Debe entrar un valor numérico")

**Resume Next**

En este caso el programa informaría de la causa del error pero continuaría la ejecución por la línea siguiente a la que ha causado el error. No es una buena solución para este caso, pero los hay en los que es la mejor.

**Resume ETIQUETA.** Continuar por la instrucción que sigue a la *ETIQUETA*.

Haremos que el programa vuelva al principio cuando se produzca un error. Para ello debemos poner una etiqueta o marca de línea mediante la cual indicaremos en qué punto debe continuar la ejecución del programa. Observe que hemos puesto la etiqueta Inicio: al principio del procedimiento. Evidentemente no siempre tiene que empezar desde el principio, puede dirigirse la ejecución hacia donde más convenga.

**Sub** Prueba()

**On Error GoTo** Tratar\_Errores

**Dim** n1 **As Integer**

**Dim** n2 **As Integer**

**Dim** total **As Integer**

**Inicio:** *'Aquí vuelve el programa si se produce un error*

n1 = InputBox("Entrar un valor", "Entrada")

n2 = InputBox("Entrar otro valor ", "Entrada")

total = n1 + n2

Worksheets("Hoja2").Range("A1").Value = total

**Exit Sub.**

Tratar\_Errores:

MsgBox ("Debe entrar un valor numérico")

**Resume** Inicio

**End Sub**

Llegados a este punto ya sabemos cómo controlar errores simples y cómo redirigir la ejecución del programa. Ahora, pruebe lo siguiente, ejecute el programa y cuando se le pida el primer valor, ponga 50000.



Efectivamente, se produce un error, 50000 no es un valor válido para datos tipo Integer, recuerde que el rango para los datos tipo Integer es de -32768 a 32767. Observe que la rutina continúa tratando el error y pidiendo de nuevo los datos, pero el mensaje que verá el usuario continuará siendo *"Debe entrar un valor numérico"*, cosa que seguramente le dejará perplejo. Con esto hemos pretendido que vea que puede haber más de una causa que provoque error, consecuentemente las rutinas de tratamiento de errores deberán prevenir más de un error. En este nuevo caso el error lo produce un desbordamiento, cuyo número de error es el 6. Suponemos que ahora ya se habrá dado cuenta de la importancia del número de error ya que éste nos permitirá tratar dentro de la rutina diferentes casos de error con la instrucción **Select Case**. Nuestro ejemplo quedaría:

```
Sub Prueba()  
On Error GoTo Tratar_Errores  
Dim n1 As Integer  
Dim n2 As Integer  
Dim total As Integer  
n1 = InputBox("Entrar el primer valor", "Entrada")  
n2 = InputBox("Entrar el segundo valor ", "Entrada")  
total = n1 + n2  
Worksheets("Hoja2").Range("A1").Value = total  
Exit Sub  
Tratar_Errores:  
Select Case Err.Number  
Case 13:  
MsgBox ("Debe introducir valores numéricos")  
Case 6:  
MsgBox ("El valor es demasiado grande o demasiado pequeño. " & _  
"El número "debe estar comprendido entre -32768 y 32767")  
End Select  
Resume  
End Sub
```

Te estarás preguntando si todavía se pueden producir más errores. La respuesta es que por supuesto que sí, por ejemplo, vaya al procedimiento y pruebe poner Hoja22 en lugar de Hoja2 en *WorkSheets(..)*.

Recomendamos que termine siempre la rutina de errores con un case else de la forma siguiente:

```
Tratar_Errores:  
Select Case Err.Number  
Case 13:  
MsgBox ("Debe introducir valores numéricos")  
Case 6:  
MsgBox ("El valor es demasiado grande o demasiado pequeño. " & _  
"El número debe estar comprendido entre -32768 y 32767")  
Case Else  
MsgBox ("Error no previsto. Llame al responsable facilitando la " & _  
"información que sigue " & Chr(13) & "Número de Error : " & _  
Err.Number & Chr(13) & "Descripción : " & Err.Description)
```



```
Exit Sub ' O lo que sea
End Select
Resume ' O lo que sea
```

Es decir, si se produce cualquier otro error no previsto, se informa al usuario que avise a quien proceda, informando del número de error y su descripción.

La cosa se complica cuando en un programa se definen múltiples procedimientos y funciones, cada una de ellas puede tratar sus propios errores, pero ¿Qué pasa si el error se provoca debido a los valores que pasa la función que llama? o ¿Cómo continuar la ejecución fuera del procedimiento?, es decir en el procedimiento que llama al que ha provocado el error. Copia los procedimientos siguientes mediante los cuales estudiaremos diferentes casos.

```
Sub Prueba()
On Error GoTo Tratar_Errores
Dim n1 As Integer
Dim n2 As Integer
Dim Total As Integer
n1 = InputBox("Entrar el primer valor", "Entrada")
n2 = InputBox("Entrar el segundo valor ", "Entrada")
Call Poner_Cálculo(n1, n2, 2)
Exit Sub
Tratar_Errores:
Select Case Err.Number
Case 13:
MsgBox ("Debe introducir valores numéricos")
Case 6:
MsgBox ("El valor es demasiado grande o demasiado pequeño. " & _
El número debe estar comprendido entre -32768 y 32767")
Case Else
MsgBox ("Error no previsto. Llame al responsable facilitando la " & _
"información que sigue " & Chr(13) & "Número de Error : " & _
Err.Number & Chr(13) & "Descripción : " & Err.Description)
Exit Sub ' O lo que sea
End Select
Resume ' O lo que sea
End Sub.

Private Sub Poner_Cálculo(n1 As Integer, n2 As Integer, Numero_Hoja As Integer)
Dim Total As Integer
Total = n1 + n2
Worksheets(Numero_Hoja).Range("A1").Value = Total
End Sub
```

Una vez entrados los valores en el procedimiento *Prueba* se llama a *Poner\_Cálculo*, simplemente suma los dos valores y los asigna a una hoja del libro de trabajo (observe que la hoja viene indicada por el parámetro *Numero\_Hoja*).

Observe que el procedimiento *Poner\_Cálculo* no tiene rutina de tratamiento de errores, puede pensar que si se produce un error dentro de él se detendrá el programa, pero no es

así. Si se produce un error dentro del procedimiento *Poner\_Cálculo* el programa pasa a ejecutar la rutina de tratamiento de errores definida dentro de *Prueba*, esto algunas veces (pocas) puede ser lo que conviene, pero compruebe el efecto indeseable que ocurre aquí.

Ejecute el programa e introduzca los valores siguientes  $n1=25000$  y  $n2=20000$ , los dos valores entran dentro del rango de los datos Integer, pero su suma 45000 no, se producirá un error de desbordamiento en la instrucción  $Total = n1 + n2$  del procedimiento *Poner\_Cálculo* y se procederá a ejecutar las instrucciones de la rutina de error definida en el procedimiento *Prueba*; aquí, después de mostrar el error, se vuelve a ejecutar la instrucción  $Total = n1 + n2$  debido a la instrucción *Resume* y el programa entra en un ciclo infinito, ya que no es posible cambiar los valores de  $n1$  y  $n2$ . La solución aquí es sencilla, sólo debe cambiar *Resume* por *Resume Next* o *Resume Etiqueta*. Otra posibilidad es implementar una rutina de tratamiento de errores dentro del procedimiento *Poner\_Cálculo*, o simplemente desactivar la rutina de tratamiento de errores antes de llamar al procedimiento mediante la instrucción **On Error Goto 0**.

```
On Error Goto 0 ' Desactivar el tratamiento de errores
Call Poner_Cálculo(n1, n2, 2)
```

Esto último conlleva la consecuencia que si se produce un error dentro del procedimiento *Poner\_Cálculo* el programa se detendrá.

Muchas veces este tipo de errores se pueden controlar mejor utilizando una variable global. Esta variable permitirá recoger el error que se ha producido en la función para luego tratarlo cuando la ejecución del programa vuelva a la función llamadora. Observe cómo quedaría nuestro ejemplo.

```
Option Explicit
Dim gError As Integer 'Variable que recogerá números de error
Sub Prueba()
On Error GoTo Tratar_Errores
Dim n1 As Integer
Dim n2 As Integer
Dim Total As Integer
n1 = InputBox("Entrar el primer valor", "Entrada")
n2 = InputBox("Entrar el segundo valor ", "Entrada").
Call Poner_Cálculo(n1, n2, 2)
If gError <> 0 Then ' Si al volver de la función gError <> 0 --> Error dentro de la función
MsgBox ("Error en la Función Poner Cálculo." & Chr(13) & gError)
Exit Sub ' O lo que sea
End If
Exit Sub
Tratar_Errores:
Select Case Err.Number
Case 13:
MsgBox ("Debe introducir valores numéricos")
Case 6:
MsgBox ("El valor es demasiado grande o demasiado pequeño. " & _
El número debe estar comprendido entre -32768 y 32767")
Case Else
```

```
MsgBox ("Error no previsto. Llame al responsable facilitando la " & _  
"información que sigue " & Chr(13) & "Número de Error : " & _  
Err.Number & Chr(13) & "Descripción : " & Err.Description)  
Exit Sub ' O lo que sea  
End Select  
Resume ' O lo que sea  
End Sub  
Private Sub Poner_Cálculo(n1 As Integer, n2 As Integer, Número_Hoja As Integer)  
On Error GoTo Tratar_Errores2  
Dim Total As Integer  
gError = 0  
Total = n1 + n2  
Worksheets(Número_Hoja).Range("A1").Value = Total  
Exit Sub  
Tratar_Errores2:  
Error = Err.Number ' Si hay error gError recoge el número de error  
End Sub
```

El funcionamiento aquí es bastante simple. Se construye una rutina de tratamiento de errores dentro de la función que simplemente asigna el número de error a la variable *gError*, cuando se devuelve el control al procedimiento llamador se inspecciona esta variable y si su valor es distinto de 0 se debe tratar el error (en el ejemplo simplemente se termina el programa, por supuesto el tratamiento podría ser diferente).

### Definir errores propios

Visual Basic incorpora la instrucción **Error** con la que es posible definir o generar errores propios. Puede que se esté preguntando para qué quiere definirse errores propios, dicho de otra forma, para qué provocar errores propios. Bueno, pues por ejemplo, para un mayor control en programas largos y complejos, para validar datos, y algunos procesos más.

Para provocar un error simplemente tiene que utilizar la instrucción **Error(Número\_De\_Error)**, *Número\_De\_Error* debe ser un valor comprendido entre 0 y 65535, es decir debe estar dentro del rango de errores definidos por Visual Basic, ahora bien, utilice siempre valores altos ya que si no, corre el riesgo de modificar un valor ya establecido por Visual Basic. Le recomendamos que empiece a partir del valor 65535 y vaya bajando, estos valores tan elevados nunca serán utilizados por Visual Basic.

Vea el siguiente ejemplo, es un caso en el que puede resultar interesante provocar un error. Es el ejemplo que hemos utilizado en la sección, el programa pide tres valores, luego llama a la función *Poner\_Cálculo* donde se suman *n1* y *n2* y el resultado se divide por *n3*. Observe que se ha definido una rutina de tratamiento de errores que según el error que se haya producido dentro de la función, asigna un valor a la variable global *gError* (ahora definida como tipo Long). Cuando el programa vuelve al procedimiento *Prueba* se comprueba el valor de *gError*, si es distinto de 0, se provoca un error **Error(gError)**, el programa salta entonces a la rutina de tratamiento de errores definida dentro del procedimiento donde gracias a los valores de error asignados, sabemos qué tipo de error se ha producido y en qué función o procedimiento se produjo.

### Option Explicit

**Dim** gError **As Long** *'Variable que recogerá números de error*

**Sub** Prueba()

**On Error GoTo** Tratar\_Errores

**Dim** n1 **As Integer**

**Dim** n2 **As Integer**

**Dim** n3 **As Integer**

**Dim** Total **As Integer**

n1 = InputBox("Entrar el primer valor", "Entrada")

n2 = InputBox("Entrar el segundo valor ", "Entrada")

n3 = InputBox("Entrar el tercer valor ", "Entrada")

**Call** Poner\_Cálculo(n1, n2, n3, 25)

**If** gError <> 0 **Then** ' Si al volver de la función gError <> 0 --> Error dentro de la función

Error (gError) ' **Generar Error**

**End If**

**Exit Sub.**

Tratar\_Errores:

**Select Case** Err.Number

**Case** 13:

MsgBox ("Debe introducir valores numéricos")

**Case** 6:

MsgBox ("El valor es demasiado grande o demasiado pequeño. " & \_

"El número debe estar comprendido entre -32768 y 32767")

**Case** 65535:

MsgBox ("Se produjo un error en la función Poner Cálculo. " & \_

"Número demasiado grande para un entero")

**Case** 65534:

MsgBox ("Se produjo un error en la función Poner Cálculo. " & \_

"División por cero")

**Case** 65533:

MsgBox ("Se produjo un error en la función Poner Cálculo. " & \_

"Número de Hoja no existe")

**Case Else**

MsgBox ("Error no previsto. Llame al responsable facilitando " & \_

la información que sigue " & Chr(13) & "Número de Error : " & \_

Err.Number & Chr(13) & "Descripción : " & Err.Description)

Exit Sub ' O lo que sea

**End Select**

**Resume Next** ' O lo que sea

**End Sub**

**Private Sub** Poner\_Cálculo(n1 **As Integer**, n2 **As Integer**, n3 **As Integer**, Número\_Hoja **As Integer**)

**On Error GoTo** Tratar\_Errores2

**Dim** Total **As Integer**

gError = 0

Total = n1 + n2

Total = Total / n3

Worksheets(Número\_Hoja).Range("A1").Value = Total

**Exit Sub**

Tratar\_Errores2:

**Select Case** Err.Number

' Valor demasiado grande

**Case** 6:

gError = 65535

```
' División por cero
Case 11:
gError = 65534
' Subíndice fuera del intervalo (Número de hoja no existe)
Case 9:
gError = 65533.
' Otro Error
Case Else
gError = Err.Number
End Select
End Sub
```

Y con esto terminamos el capítulo de depuración de programas y tratamiento de errores. Utilice estos mecanismos siempre que sus programas vayan a ser usados por otras personas. Con la depuración podrá, además de detectar errores más fácilmente, comprobar cómo el programa en situaciones extremas (y créanos, si lo han de utilizar diferentes usuarios), se producirán. Con las rutinas de tratamiento de errores podrá controlar muchos de ellos, y si se produce uno de imprevisto, al menos le será más fácil localizarlo o saber las causas que lo han provocado.

Finalmente y como opinión personal, si conoce lenguajes como Delphi o Java o algún día trabaja con ellos comprobará que el control de errores está mucho mejor acabado que con Visual Basic. Con Visual, a uno siempre le queda la sensación de que algo queda suelto, y cosas como los saltos de Resume Etiqueta dejan siempre cierta intranquilidad.

## Controles de formulario en la hoja de cálculo

En este tema estudiaremos cómo insertar controles (botones, cuadros de texto, cuadros de lista, botones de radio, etc. ) dentro de una hoja de cálculo. Seguramente es más habitual utilizar este tipo de controles dentro de formularios y a través de ellos gestionar datos de una o varias hojas, sin embargo resulta conveniente muchas veces incluir directamente estos controles dentro de una misma hoja, sobre todo cuando sólo se requiere procesos simples como elegir datos de una lista o activar una macro desde un botón, etc.

No estudiaremos en profundidad estos controles, simplemente utilizaremos las propiedades más habituales (concretamente las que necesitemos), dejaremos un estudio más completo para el tema de los formularios.

### Aplicación de ejemplo

Para ver el funcionamiento de los distintos controles, construiremos una pequeña aplicación que nos sirva para gestionar una pequeña tabla de registros, básicamente extraer datos que cumplan una determinada condición. La mayoría de las funciones que aplicaremos pueden hacerse directamente desde las utilidades del menú **Datos/ Filtro avanzado** que lleva incorporado el propio Excel pero creemos que será un buen ejemplo para ver las posibilidades de los controles.

Abra su aplicación Excel y active la hoja de cálculo *Lista7.xls* que debió bajar junto a este documento, en la primera hoja está la lista de registros que utilizaremos en los ejemplos que veremos a continuación. En la Hoja2 se encuentran algunos datos necesarios para los controles.

## Mostrar la barra de herramientas para cuadros de control


Obviamente para insertar controles en la hoja deberá tener activada la barra de menús, seleccione “Ver” – “Barras de Herramientas” – “Cuadro de Controles”, deberá activarse una barra como la siguiente:



## Cuadro de texto y Botón


Lo primero que haremos es algo muy sencillo, simplemente copiaremos en Hoja2, los datos correspondientes a los registros de Hoja1 cuyos datos correspondientes a la columna *Nombre* coincidan con el que teclearemos en una cuadro de texto que insertaremos en Hoja2. Los datos se copiarán a partir de la celda A16. El botón simplemente servirá para invocar la macro que copiará los datos.

## Insertar el cuadro de texto


Sólo tiene que seleccionar el elemento  de la barra de controles y dibujarlo sobre la hoja (Hoja 2 en nuestro ejemplo, procure que coja más o menos el rango correspondiente a las celdas C2 y D2).

## Insertar una etiqueta


Las etiquetas sirven básicamente para acompañar los controles con texto descriptivo.

Seleccione el botón  y dibuje en la hoja el rectángulo para insertar la etiqueta, póngalo al lado del control cuadro de texto.


## Insertar un Botón

Los botones se utilizan básicamente para invocar las macros que realizarán las acciones . No es el único control que puede invocar macros, cualquiera de los controles puede invocarla, pero es el más habitual.

## Cambiar las propiedades de los objetos

A continuación desplegaremos la ventana de propiedades para cambiar algunas de los objetos acabados de incrustar. Debe estar en modo diseño, el botón  debe estar activado.

### Cambiar el texto del control Label. Propiedad Caption

1. Seleccione el control **Etiqueta**.
2. Pulse sobre el botón  de la barra de controles, se activa la ventana de **Propiedades**.
3. En la propiedad **Caption**, cambie el texto **Label1** por **Datos a Buscar**.
4. Ajuste la posición y el tamaño del control.

### Cambiar el nombre del control Cuadro de Texto. Propiedad Name

No es necesario cambiar el nombre de los controles pero sí muy conveniente, tenga en cuenta que a través de los nombres de un control será como se refiera a ellos a través de las macros. Siempre es mejor llamar a un control por un nombre descriptivo que por Text1 o Command1, etc. Al control cuadro de texto le pondremos el nombre **Datos\_Buscar**.

1. Seleccione el control Cuadro de Texto.
2. Si no tiene activada la ventana de propiedades, actívela.
3. En la propiedad **Name**, cambie el text1 por **Datos\_Buscar**.

Cambie la propiedad **Caption** del Botón por **Copiar Datos** y su propiedad **Name** por **Copiar\_Datos** (debe poner el guión bajo ya que la propiedad **Name** no permite espacios en blanco).

### Establecer la acción de copiar datos cuando se pulse el botón

A continuación crearemos la macro que será invocada cuando se pulse el botón. La macro simplemente debe buscar en la columna A de la lista de Hoja1, el nombre que coincida con el teclado en el cuadro de texto y luego copiarlo hacia Hoja2 a partir de la casilla A16. La macro controlará que haya algo en el cuadro de texto. Se copiarán todas las coincidencias, es decir si hay dos nombres Ramón se copiarán los dos. Si no hay ninguna coincidencia se mostrará un mensaje avisando de ello.

### Los eventos

Cuando se programan controles bien sea directamente en la hoja como estamos haciendo ahora o desde un formulario, debe tener en cuenta los eventos. Un evento es cuando ocurre algo sobre un objeto, en entornos Windows constantemente se están produciendo eventos. Clics con el ratón sobre un control, teclear sobre un cuadro de texto, etc., provocan eventos



que son recogidos por el sistema. Programar un evento significa hacer que se ejecuten determinadas instrucciones cuando ocurra dicho evento. En el caso que nos ocupa ahora, haremos que las acciones necesarias para copiar los datos se ejecuten cuando se haga un clic sobre el botón **Copiar\_Datos**. En general, todos los controles son capaces de capturar diferentes eventos. El sistema de eventos es bastante más complejo de lo que estudiaremos aquí, nosotros simplemente tendremos en cuenta qué evento debemos elegir para lanzar la ejecución de determinado código. Veamos en la siguiente sección cómo asociar el código necesario para copiar datos cuando ocurre el evento clic (pulsar el botón y soltarlo) sobre el botón **Copiar\_Datos**.

### Escribir código para el evento Clic del Botón

Deberá estar en modo **Diseño**, asegúrese que el botón  esté pulsado.

Haga doble clic sobre el botón, observe que se activa automáticamente la ventana de Visual Basic y aparece un esqueleto de función.

```
Sub Copiar_Datos_Click()
```

```
End Sub
```

Es lo que se llama procedimiento de evento, es decir, este procedimiento está asociado al evento Click del Botón **Copiar\_Datos**, observe que el procedimiento lleva un nombre compuesto por el nombre del control "Copiar\_Datos", un guión bajo y el nombre del evento "Click", en general todos los procedimientos de evento se nombran de esta forma:

### NombreDeControl\_NombreDeEvento

Observe la lista de la parte superior derecha, la que tiene el elemento **Click**. Es la lista de eventos, si la despliega verá que además del elemento Click aparecen unos cuantos más **DbClick** (Doble Clic) **Gotfocus** (Coger el foco), etc., todos ellos son eventos programables del control botón, es decir, podemos incluir el código que se ejecutará cuando ocurran dichos eventos. Por otra parte, todos los controles tienen un evento "*por defecto*", dicho de otra forma, cuando se programa un evento del control casi siempre será ése. En el caso de nuestro botón (y de todos los botones), el evento por defecto es **Click**, observe que lo habitual es que queramos que el código se ejecute cuando se hace clic sobre el botón, no cuando éste coge el foco o cuando el puntero de ratón pasa sobre él, etc. El evento por defecto de un control es el que aparece cuando, en modo diseño, se hace doble clic sobre él, obviamente éste se puede cambiar, por el que más nos convenga.

Teclee el código para llevar a cabo las acciones. Recuerde que lo que se desea hacer es copiar hacia hoja2 todos los nombres que coincidan con el que está en el cuadro de texto. El código será el que sigue, observe los comentarios.



### Option Explicit

*' Numero de columnas(campos) de las que consta cada registro*

**Const** Num\_Columnas = 6

**Private Sub** Copiar\_Datos\_Click()

**Dim** r1 **As Range**, r2 **As Range**

**Dim** encontrado **As Boolean**

*' Si el cuadro de texto está vacío, no se busca nada*

**If** Len(Datos\_Buscar.Value) = 0 **Then**

MsgBox ("No hay datos que buscar")

**Else**

*' Borrar los datos actuales*

**Call** borrar\_datos

*' Activar Casilla A16 de Hoja2 y referenciarla con r2,*

*' Es la casilla donde se copiarán*

*' los datos en caso que se encuentren*

Worksheets(2).Range("A16").Activate

**Set** r2 = ActiveCell

*' Activar casilla A2 de Hoja1 y referenciarla con r1*

Worksheets(1).Activate

Worksheets(1).Range("A2").Activate

*' Recorrer todo el rango de datos de Hoja1*

encontrado = False

**Do While Not** IsEmpty(ActiveCell)

*' Si la casilla activa = Datos\_Buscados*

**If** ActiveCell.Value = Datos\_Buscar.Text **Then**

encontrado = True

*' Referenciar con r1 la celda donde están los datos*

**Set** r1 = ActiveCell

*' Copiar los datos*

Call Copiar\_Datos\_Hojas(r1, r2)

*' Referenciar con r2 la casilla donde se copiaran los próximos datos*

Set r2 = r2.Offset(1, 0)

**End If**

ActiveCell.Offset(1, 0).Activate

**Loop**

Worksheets(2).Activate

**If** encontrado **Then**

MsgBox ("Datos Copiados")

**Else**

MsgBox ("Ninguna coincidencia")

**End If**

**End If**

**End Sub**

*' Procedimiento para borrar los datos de Hoja2 se llama antes de proceder a la nueva copia*

**Private Sub** borrar\_datos()

**Dim** i **As Integer**

Worksheets(2).Range("A16").Activate

**Do While Not** IsEmpty(ActiveCell)

**For** i = 0 **To** Num\_Columnas - 1

ActiveCell.Offset(0, i).Value = ""

**Next** i

```
ActiveCell.Offset(1, 0).Activate
Loop
End Sub
' Procedimiento para copiar los datos de Hoja1 a Hoja3
' Parámetros.
' r1 = Celda Origen
' r2 = Celda Destino
Private Sub Copiar_Datos_Hojas(r1 As Range, r2 As Range)
Dim i As Integer
Dim Datos As Variant
' Recorrer las columnas del registro y copiar celda a celda
For i = 0 To Num_Columnas - 1
    Datos = r1.Offset(0, i).Value
    r2.Offset(0, i).Value = Datos
Next i
End Sub
```

## Cuadros Combinados (ComboBox)



Con lo hecho hasta ahora podemos extraer de la tabla los registros cuyo nombre coincida con el tecleado en el cuadro de texto. A continuación haremos que se pueda escoger el campo, es decir, podremos extraer coincidencias del *Nombre*, los *Apellidos*, la *Ciudad*, etc. Para ello incluiremos un cuadro combinado que permita escoger en qué campo o columna tiene que buscarse la coincidencia. La lista, por supuesto, mostrará los nombres de las columnas.

Incluya un cuadro combinado en Hoja2 y póngale por nombre (propiedad **Name**).

### Lista\_Campos Propiedad ListFillRange

Con esta propiedad deberemos definir los elementos que debe mostrar la lista, debe especificarse el rango que contiene los elementos a mostrar, el rango debe ser una columna (o dos, o tres, etc.). En nuestro caso el rango será J1:J6 de Hoja2 (Observe que en este rango están especificados los nombres de las columnas).

### Propiedad LinKedCell

En esta propiedad debe especificar en qué celda debe copiarse el elemento seleccionado de la lista. En esta lista no utilizaremos esta propiedad. *Cuidado con esta propiedad, tenga en cuenta que los elementos de la lista son tratados como datos de tipo **String** aunque contenga números o fechas, por lo que en estos casos, a veces será necesario aplicar funciones de conversión de datos antes que el dato se copie en la hoja. Por ejemplo, si alguna vez construye una lista con números verá que el dato seleccionado se alinea a la derecha, si son fechas, no se muestra con el formato correspondiente.*

### Propiedad ListIndex

Mediante esta propiedad podremos saber qué elemento de la lista es el seleccionado por su número de orden. Es decir, si está seleccionado el primero, ListIndex valdrá 0, si está seleccionado el segundo valdrá 1, etc. Si no hay ningún elemento seleccionado valdrá -1. Tenga en cuenta que esta propiedad sólo está disponible en tiempo de ejecución, es decir la podremos leer mientras esté funcionando el programa, no se puede establecer en modo diseño, observe que no aparece en la ventana propiedades del cuadro combinado.

Bien, ya sabemos cómo funcionan las propiedades que utilizaremos para hacer que se extraigan de la tabla los elementos que coincidan con el valor del cuadro de texto y cuya columna o campo sea el seleccionado de la lista, veamos cómo quedará la macro.

En primer lugar cree un procedimiento llamado **Proceder** donde deberá copiar todo el código que ahora está en **Copiar\_Datos**. Debemos hacer esto porque antes de proceder se deben hacer ciertas comprobaciones que ya iremos viendo conforme avanzamos, por el momento la comprobación a hacer es la de ver sobre qué campo o columna se deben buscar las coincidencias con los datos tecleados en el cuadro de texto. La función **Copiar\_Datos** quedará de la forma siguiente:

```
Private Sub Copiar_Datos_Click()
Dim i As Integer
' Recoger el elemento seleccionado de la lista
i = Lista_Campos.ListIndex
' Si i < 0 es que no hay ningún elemento seleccionado.
If i < 0 Then
MsgBox ("Debe Seleccionar un campo de la lista")
Else
' Llamar a proceder para iniciar la copia.
Call Proceder(i)
End If
End Sub
```

La cabecera de la función proceder quedará de la forma siguiente:

```
' Procedimiento Proceder
' Inicia la copia de los datos coincidentes
' Parámetros:
' Columna = Elementos seleccionado de la lista que coincidirá con la columna sobre la que se
' debe buscar
Private Sub Proceder(Columna As Integer)
.....
```

Ahora, dentro del procedimiento Proceder cambie la línea

```
If ActiveCell.Value = Datos_Buscar.Text Then
```

Por

```
If ActiveCell.Offset(0, Columna).Value = Datos_Buscar.Text Then
```

**Explicación del proceso.** Cuando se selecciona un elemento de la lista, su propiedad **ListIndex** es igual al orden que ocupa dicho elemento en la lista, supongamos que escogemos **Ciudad**, **ListIndex** valdrá 2. Este valor se pasa a **Proceder** y es recogido por el parámetro **Columna**. Ahora observe la línea

**If** ActiveCell.Offset(0, Columna).Value = Datos\_Buscar.Text **Then**

Es decir la coincidencia con el valor del cuadro de texto **Datos\_Buscar** se busca respecto a la casilla que está a la derecha ( offset ) de la activa, tantas columnas como las expresadas por el valor de la variable.

**Columna.** Observe que en este caso la casilla activa siempre corresponde a una de la columna A, si la variable **Columna** vale 2 la coincidencia se buscará respecto al valor de la columna C (2 más hacia la derecha) y que coincide con la columna correspondiente a la **Ciudad**.

## Segunda Lista

Ahora crearemos una lista donde sea posible escoger la relación de comparación. Hasta ahora la extracción se realizaba con aquellos elementos iguales al valor entrado en el cuadro de texto **Datos\_Buscar**, esta segunda lista permitirá escoger si los elementos a extraer deben ser **Iguales**, **Menores**, **Mayores**, **Menores Iguales** o **Mayores Iguales** que el valor de **Datos\_Buscar**. Para ello debe construir una segunda lista con las propiedades siguientes:

**Name** = Lista\_Comparación.

**ListFillRange** = L1:L5 Observe que en este rango están los valores correspondientes a la operación relacional que se desea realizar (Igual, Menor, etc.)

Obviamente deberemos modificar las funciones para realizar las operaciones con respecto al elemento seleccionado en el cuadro de lista **Lista\_Comparación**. Dejaremos el procedimiento **Proceder** de la forma siguiente y crearemos una función que haga las comparaciones, esta función a la que hemos llamado **Comparar** devuelva el valor True si el resultado de la comparación es Cierto y False si es falso.

*' Procedimiento Proceder Inicia la copia de los datos coincidentes*

*' Parámetros:*

*' Columna = Elementos seleccionado de la lista que coincidirá*

*' con la columna sobre la que se debe buscar*

**Private Sub** Proceder(Columna As Integer)

**Dim** r1 **As Range**, r2 **As Range**

**Dim** encontrado **As Boolean**

**Dim** Valor\_Comparacion **As Boolean**

*' Si el cuadro de texto está vacío, no se busca nada*

**If** Len(Datos\_Buscar.Value) = 0 **Then**

MsgBox ("No hay datos que buscar")

**Else**

*' Borrar los datos actuales*

**Call** borrar\_datos

```

' Activar Casilla A16 de Hoja2 y referenciarla con r2' Es la casilla donde se copiarán
' los datos en caso que' se encuentren
Worksheets(2).Range("A16").Activate
Set r2 = ActiveCell
' Activar casilla A2 de Hoja1 y referenciarla con r1
Worksheets(1).Activate
Worksheets(1).Range("A2").Activate
encontrado = False
' Recorrer todo el rango de datos de Hoja1
Do While Not IsEmpty(ActiveCell)
    Valor_Comparacion = Comparar(ActiveCell.Offset(0, Columna).Value, _
    Datos_Buscar.Value, Lista_Comparacion.ListIndex)
    If Valor_Comparacion = True Then
        encontrado = True
        ' Referenciar con r1 la celda donde están los datos
        Set r1 = ActiveCell
        ' Copiar los datos
        Call Copiar_Datos_Hojas(r1, r2)
        ' Referenciar con r2 la casilla donde se copiaran los próximos datos.
        Set r2 = r2.Offset(1, 0)
    End If
    ActiveCell.Offset(1, 0).Activate
Loop
Worksheets(2).Activate
If encontrado Then
    MsgBox ("Datos Copiados")
Else
    MsgBox ("Ninguna coincidencia")
End If
End If
End Sub

' Función que compara dos valores con un operador relacional =, >, <, etc.
' La función devuelve True o False en función de la comparación.
' Parámetros.
' Valor1 y Valor2 = Valores que se comparan
' Signo = variable que sirve para escoger el operador relacional
' en función de su valor, ver estructura Select Case
Private Function Comparar(Valor1 As Variant, Valor2 As Variant, Operador As Integer) As
Boolean
Dim q As Boolean
Select Case Operador
    Case 0:
        q = Valor1 = Valor2
    Case 1:
        q = Valor1 > Valor2
    Case 2:
        q = Valor1 < Valor2
    Case 3:
        q = Valor1 >= Valor2
    Case 4:
        q = Valor1 <= Valor2
End Select
Comparar = q
End Function

```

Observe la línea que llama a la función **Comparar**:

```
Valor_Comparacion = Comparar(ActiveCell.Offset(0, Columna).Value, _  
Datos_Buscar.Value, Lista_Comparacion.ListIndex)
```

*ActiveCell.Offset(0,Columna)* serán los valores que se compararán con el valor del cuadro de texto.

*Datos\_Buscar.value* es el valor del cuadro de texto.

**Lista\_Comparación.ListIndex** devuelve el índice del elemento seleccionado de la lista, observe cómo se utiliza este valor en la estructura **Select Case** de **Comparar** para determinar que operador utilizar.

### **Pero todo esto no funcionará**

Prueba lo siguiente:

- En el cuadro de texto escriba México ( o simplemente M).
- Seleccione de la lista de Campos *Ciudad*.
- Seleccione de la lista de operadores *Mayor*.
- Pulse sobre el botón y observe que se copian todos los registros cuyo campo Ciudad sea superior
- a México (o a la M). Hasta aquí todo correcto.

Ahora pruebe lo siguiente.

- En el cuadro de texto escriba 100000
- Seleccione de la lista de Campos *Cantidad*.
- Seleccione de la lista de operadores *Mayor*.
- Pulse sobre el botón y observe que no se copia nada a pesar que en cantidad hay registros con el valor superior a 100000.

Recuerda que los valores de un cuadro de texto son siempre datos tipo String, entonces en este caso estarán comparándose valores String (los del cuadro de texto) con valores numéricos, (los recogidos de la columna *Cantidad*). Tenga en cuenta siempre esta circunstancia cuando trabaje con elementos de formulario. Vea la sección siguiente donde se solucionará este problema y de paso se verá cómo construir Lista con más de una columna.

### **Listas con más de una columna**

Para solucionar el problema del apartado anterior utilizaremos una segunda columna cuyos elementos indicarán el tipo de datos del campo. Observe el rango K1:K6 de Hoja2, las letras significan lo siguiente:

*T* campo tipo texto o string, *N* campo Numérico, *F* campo fecha. Para incluir esta segunda columna en la lista deberá cambiar las propiedades siguientes:

**ListFillRange**, J1:K6

**ColumnCount**, 2 (Indicamos el número de columnas de la lista).

Además especificaremos el ancho de cada columna mediante la propiedad,

**ColumnWidths**, 4pt;0pt Debe separar el ancho de cada columna mediante un punto y coma.

Observe que la segunda columna no se mostrará debido a que hemos especificado el ancho a 0.

**ColumnBound**, 1 significa que el dato que recogerá la propiedad Value corresponde al elemento seleccionado de la primera columna.

Si desea recoger datos de la segunda columna deberá utilizar la propiedad

**Column**(Número de Columna, Índice del elemento seleccionado).

Las columnas empiezan a numerarse a partir de la 0.

La función **Comparar** y su correspondiente llamada quedarán de la forma siguiente. Observe que se han incluido variables que recogen los valores de **Lista\_Comparación** y **Lista\_Campos**, simplemente lo hemos hecho para que quede más claro.

```
.....
Do While Not IsEmpty(ActiveCell)
' Recoger el Signo de comparación
Signo = Lista_Comparacion.ListIndex
' Recoger el tipo de datos
Tipo_Datos = Lista_Campos.Column(1, Columna)
Valor_Comparacion = Comparar(ActiveCell.Offset(0, Columna).Value, _
Datos_Buscar.Value, Signo, Tipo_Datos)
La función Comparar.
Private Function Comparar(Valor1 As Variant, Valor2 As Variant, Operador As Integer, Tipo As
String As Boolean
Dim q As Boolean
' Conversión del tipo de datos de las variables
Select Case Tipo
Case "N": ' Convertir a número
Valor2 = Val(Valor2)
Case "F": ' Convertir a Fecha
Valor2 = CDate(Valor2)
End Select
Select Case Operador
Case 0:
q = Valor1 = Valor2
Case 1:
q = Valor1 > Valor2
```

**Case 2:**

q = Valor1 < Valor2

**Case 3:**

q = Valor1 >= Valor2

**Case 4:**

q = Valor1 <= Valor2

**End Select**

Comparar = q

**End Function.**

## Control Numérico



Inserte un control de número y póngale por nombre (propiedad **Name**) *Número*.

Establezca su propiedad **Orientation** a *fmOrientationVertical* para que se alinee verticalmente.

Este control se utiliza normalmente para aumentar y disminuir valores numéricos de un cuadro de texto, aunque por supuesto puede utilizarse para otras funciones. Utilizaremos un control de este tipo para aumentar los valores del Cuadro de Texto *Datos\_Buscar* pero sólo si el campo seleccionado de *Lista\_Campos* es de tipo numérico. Para ello activaremos este control únicamente cuando el campo seleccionado sea de este tipo. Para activar o desactivar un control se utiliza la propiedad **Enabled**, si está a true el control estará activado y sino estará desactivado.

Observe que la acción de activar o desactivar el control de número deberemos hacerlo cuando se seleccione un elemento de *Lista\_Campos*. Es decir el código deberemos insertarlo en el evento **Change** (cambio) de *Lista\_Campos*. Haga doble clic sobre el elemento *Lista\_Campos* para desplegar su procedimiento de evento. El código para controlar la activación del control es el que sigue:

```
Private Sub Lista_Campos_Change()
Dim i As Integer
Dim Tipo_Datos As String
i = Lista_Campos.ListIndex
If i >= 0 Then
    Tipo_Datos = Lista_Campos.Column(1, i)
    If Tipo_Datos = "N" Then
        Número.Enabled = True
    Else
        Número.Enabled = False
    End If
End If
End Sub
```

## Establecer los valores del control de número

Para establecer los valores que puede devolver un control de número se deben modificar las propiedades siguientes.



**Max**, establece el valor máximo que puede tener el control.

**Min**, establece el valor mínimo que puede tener el control.

**Smallchange**, establece el incremento o decremento para la propiedad value cada vez que se pulse sobre alguno de los dos botones.

Estos valores se pueden establecer en tiempo de diseño, pero lo que haremos a continuación será establecerlos en tiempo de ejecución dependiendo del campo que se seleccione en *Lista\_Campos*.

Estableceremos los valores siguientes:

Para campo *Edad*.

**Max** = 99

**Min** = 18

**SmallChange** = 1.

**Página 98**

Para campo cantidad.

**Max** = 500.000

**Min** = 10.000

**SmallChange** = 1.000

Debemos modificar el código del procedimiento de evento *Lista\_Campos\_Change* de la forma siguiente:

```
Private Sub Lista_Campos_Change()  
    Dim i As Integer  
    Dim Tipo_Datos As String  
    i = Lista_Campos.ListIndex  
    If i >= 0 Then  
        Tipo_Datos = Lista_Campos.Column(1, i)  
        If Tipo_Datos = "N" Then  
            Número.Enabled = True  
            If Lista_Campos.Value = "Edad" Then  
                Número.Min = 18  
                Número.Max = 99  
                Número.SmallChange = 1  
                Datos_Buscar.Value = 0  
                Número.Value = 0  
            End If  
            If Lista_Campos.Value = "Cantidad" Then  
                Número.Min = 10000  
                Número.Max = 500000  
                Número.SmallChange = 1000  
                Datos_Buscar.Value = 0  
                Número.Value = 0  
            End If  
        Else  
            Número.Enabled = False  
        End If  
    End If  
End Sub
```



Y para terminar ya sólo debemos codificar el evento **Change** del control de número para que el Cuadro de texto vaya incrementando o decrementando su valor cada vez que se haga clic sobre el control.

```
Private Sub Número_Change()  
    Datos_Buscar.Value = Número.Value  
End Sub.
```

Pero además debemos hacer que cuando se cambie el valor del cuadro de texto también se cambie el del control de número de forma que cuando se pulse sobre él, incremente o decremente a partir del valor que hay en el cuadro de texto. Si no se hace así, el incremento o decremento se hará en función del valor que tenga el control de número, no el que está en el cuadro de texto. Modificaremos el evento **Change** del cuadro de texto. Observe que se controla que sólo se ejecute la acción si el control de número está activado, además se debe controlar el valor máximo y mínimo que puede contener el cuadro de texto, si no se hiciera así se generaría un error al poner un valor mayor o menor que los definidos en las propiedades Max y Min del control de número.

```
Private Sub Datos_Buscar_Change()  
    ' Si el número de control está activado  
    If Número.Enabled Then  
        ' No permite coger valores superiores a la propiedad Max  
        If Val(Datos_Buscar.Value) > Número.Max Then  
            MsgBox ("Valor demasiado grande")  
            Datos_Buscar.Value = Número.Max  
        Else  
            ' No permite coger valores inferiores a la propiedad Min  
            If Val(Datos_Buscar.Value) < Número.Min Then  
                MsgBox ("Valor demasiado pequeño")  
                Datos_Buscar.Value = Número.Min  
            Else  
                Número.Value = Val(Datos_Buscar.Value)  
            End If  
        End If  
    End If  
End Sub
```

Antes de proceder con el siguiente control permítenos remarcar que la programación de controles implica que muchas veces unos dependan de otros por lo que debe ser extremadamente cuidadoso en la elaboración del código de los eventos. Observe las últimas operaciones que hemos realizado debido a la interdependencia de dos controles.

### Casillas de verificación (CheckBox)



Estos controles se suelen utilizar para activar o desactivar la ejecución de determinadas acciones. Casi siempre implican una estructura condicional a la hora de hacer alguna cosa.

Si la casilla está activada **Entonces**

Acciones

....

**Fin Si**

A continuación utilizaremos una casilla de verificación que si está activada, provocará que los datos tipo texto se conviertan a mayúscula antes de copiarse, se utilizará la función **Ucase**. Simplemente se deberá comprobar que la casilla esté activada y si lo está proceder a la conversión (sólo si el dato es tipo texto).

Inserte un control casilla de verificación. Establezca las siguientes propiedades.

**Name**, Mayúsculas.

**Caption**, Mayúsculas.

Para comprobar si la casilla está activada o no simplemente debe mirarse su propiedad **Value**. Si value es true es que está activada, sino valdrá False.

Vea cómo quedará el procedimiento **Copiar\_Datos\_Hojas**, observe que además de comprobar que la casilla esté activada se utiliza la función **TypeName** para comprobar si los datos que se van a copiar son del tipo String, si no lo fueran, la función **Ucase** provocaría un error. **TypeName(Expresión)** devuelve una cadena que indica el tipo de datos de la expresión.

```
Private Sub Copiar_Datos_Hojas(r1 As Range, r2 As Range)
Dim i As Integer
Dim Datos As Variant
' recorrer las columnas del registro y copiar celda a celda
For i = 0 To Num_Columnas - 1
' Si la casilla Mayúsculas está activada y el tipo de datos es String
If Mayusculas.Value = True And TypeName(r1.Offset(0, i).Value) = "String" Then
Datos = UCase(r1.Offset(0, i).Value)
Else
Datos = r1.Offset(0, i).Value
End If
r2.Offset(0, i).Value = Datos
Next i
End Sub
```

## Botones de Opción (Option Button)



Los botones de opción se utilizan para elegir una única opción entre una serie de ellas, es decir, de un grupo de opciones sólo permitirán que se escoja una. De la misma forma que las casillas de verificación, casi siempre implican una estructura condicional para comprobar cuál de ellas está activada. El botón activado tendrá su propiedad **Value** igual a true.

Como ejemplo de su utilización crearemos dos botones de opción que sirvan para que a la hora de copiar datos hacia la hoja, se copien sólo los valores de *Nombre* y *Apellidos* o todos como hasta ahora.

Incluya dos botones de opción y establezca las siguientes propiedades:

Botón1.  
**Name**, Todo.  
**Caption**, Todo.  
 Botón2.  
**Name**, Solo\_Nombre.  
**Caption**, Nombre y Apellidos.

Si está activado el primer botón deberán copiarse todos los datos mientras que si está activado el segundo, sólo se copiarán el *Nombre* y los *Apellidos*. El procedimiento *Copiar\_Datos\_Hojas* quedará de la forma siguiente:

```
Private Sub Copiar_Datos_Hojas(r1 As Range, r2 As Range)
Dim i As Integer
Dim Datos As Variant
Dim Final As Integer
' Si Botón Todo Activado, se copian todas las columnas
If Todo.Value = True Then
    Final = Num_Columnas - 1
Else ' Sólo se copian las dos primera columnas
    Final = 1
End If
' recorrer las columnas del registro y copiar celda a celda
For i = 0 To Final
    ' Si la casilla Mayúsculas está activada y el tipo de datos es String
    If Mayusculas.Value = True And TypeName(r1.Offset(0, i).Value) = "String" Then
        Datos = UCase(r1.Offset(0, i).Value)
    Else
        Datos = r1.Offset(0, i).Value
    End If
    r2.Offset(0, i).Value = Datos
Next i
End Sub
```

Y aquí terminamos el estudio de cómo se pueden utilizar los controles de formulario dentro de una hoja de cálculo. Recordarle para terminar que debe ser extremadamente cuidadoso con el código que utilice en los procedimientos de evento, sobre todo si los controles se interrelacionan entre ellos.

El siguiente Listado incluye todo el código que se ha utilizado.

```
Option Explicit
' Numero de columnas(campos) de las que consta cada registro
Const Num_Columnas = 6
Private Sub Copiar_Datos_Click()
```

**Dim i As Integer**

**Dim x As Integer**

*' Recoger el elemento seleccionado de la lista*

i = Lista\_Campos.ListIndex

*' Si i < 0 no está seleccionado ningún elemento*

**If i < 0 Then**

MsgBox ("Debe Seleccionar un campo de la lista")

**Else**

x = Lista\_Comparacion.ListIndex

**If x < 0 Then**

MsgBox ("Debe Seleccionar uno operador de Comparación")

**Else**

*' llamar a proceder*

**Call** Proceder(i)

**End If**

**End If**

**End Sub**

*' Procedimiento Proceder*

*' Inicia la copia de los datos coincidentes*

*' Parámetros:*

*' Columna = Elementos seleccionado de la lista que coincidirá*

*' con la columna sobre la que se debe buscar*

**Private Sub** Proceder(Columna As Integer)

**Dim** r1 **As Range**, r2 **As Range**

**Dim** encontrado **As Boolean**

**Dim** Valor\_Comparacion **As Boolean**

**Dim** Signo **As Integer**

**Dim** Tipo\_Datos **As String**

*' Si el cuadro de texto está vacío, no se busca nada*

**If** Len(Datos\_Buscar.Value) = 0 **Then**

MsgBox ("No hay datos que buscar")

**Else**

*' Borrar los datos actuales*

**Call** borrar\_datos

*' Activar Casilla A16 de Hoja2 y referenciarla con r2*

*' Es la casilla donde se copiarán los datos en caso que se encuentren*

Worksheets(2).Range("A16").Activate

**Set** r2 = ActiveCell

*' Activar casilla A2 de Hoja1 y referenciarla con r1*

Worksheets(1).Activate

Worksheets(1).Range("A2").Activate

*' Recorrer todo el rango de datos de Hoja1*

encontrado = False

Do While Not IsEmpty(ActiveCell).

*' Recoger el Signo de comparación*

Signo = Lista\_Comparacion.ListIndex

*' recoger el tipo de datos*

Tipo\_Datos = Lista\_Campos.Column(1, Columna)

Valor\_Comparacion = Comparar(ActiveCell.Offset(0, Columna).Value, \_

Datos\_Buscar.Value, Signo, Tipo\_Datos)

**If** Valor\_Comparacion = True **Then**

encontrado = True

*' Referenciar con r1 la celda donde están los datos*

**Set** r1 = ActiveCell

```

' Copiar los datos
Call Copiar_Datos_Hojas(r1, r2)
' Referenciar con r2 la casilla donde se copiaran los próximos datos
Set r2 = r2.Offset(1, 0)
End If
ActiveCell.Offset(1, 0).Activate
Loop
Worksheets(2).Activate
If encontrado Then
MsgBox ("Datos Copiados")
Else
MsgBox ("Ninguna coincidencia")
End If
End If
End Sub

' Función que compara dos valores con un operador relacional =, >, <, etc.
' La función devuelve True o False en función de la comparación.
' Parámetros.
' Valor1 y Valor2 = Valores que se comparan
' Signo = variable que sirve para escoger el operador relacional
' en función de su valor, ver estructura Select Case
Private Function Comparar(Valor1 As Variant, Valor2 As Variant, Operador As Integer, Tipo As String) As Boolean
Dim q As Boolean
Select Case Tipo
Case "N": ' Convertir a número
Valor2 = Val(Valor2)
Case "F": ' Convertir a Fecha
Valor2 = CDate(Valor2)
End Select
Select Case Operador
Case 0:
q = Valor1 = Valor2
Case 1:
q = Valor1 > Valor2
Case 2:
q = Valor1 < Valor2.
Case 3:
q = Valor1 >= Valor2
Case 4:
q = Valor1 <= Valor2
End Select
Comparar = q
End Function

' Procedimiento para borrar los datos de Hoja2 se llama antes de proceder a la nueva copia
Private Sub borrar_datos()
Dim i As Integer
Worksheets(2).Range("A16").Activate
Do While Not IsEmpty(ActiveCell)
For i = 0 To Num_Columnas - 1
ActiveCell.Offset(0, i).Value = ""
Next i
ActiveCell.Offset(1, 0).Activate
Loop

```

### End Sub

*' Procedimiento para copiar los datos de Hoja1 a Hoja3*

*' Parámetros.*

*' r1 = Celda Origen*

*' r2 = Celda Destino*

**Private Sub** Copiar\_Datos\_Hojas(r1 **As Range**, r2 **As Range**)

**Dim** i **As Integer**

**Dim** Datos **As Variant**

**Dim** Final **As Integer**

*' Si Botón Todo Activado, se copian todas las columnas*

**If** Todo.Value = True **Then**

Final = Num\_Columnas - 1

**Else** *' Sólo se copian las dos primera columnas*

Final = 1

**End If**

*' recorrer las columnas del registro y copiar celda a celda*

**For** i = 0 To **Final**

*' Si la casilla Mayúsculas está activada y el tipo de datos es String*

**If** Mayúsculas.Value = True **And** TypeName(r1.Offset(0, i).Value) = "String" **Then**

Datos = UCase(r1.Offset(0, i).Value)

**Else**

Datos = r1.Offset(0, i).Value

**End If**

r2.Offset(0, i).Value = Datos

**Next** i

**End Sub.**

### Página 105

**Private Sub** Datos\_Buscar\_Change()

*' Si el número de control está activado*

**If** Número.Enabled **Then**

*' No permite coger valores superiores a la propiedad Max*

**If** Val(Datos\_Buscar.Value) > Numero.Max **Then**

MsgBox ("Valor demasiado grande")

Datos\_Buscar.Value = Número.Max

**Else**

*' No permite coger valores inferiores a la propiedad Min*

**If** Val(Datos\_Buscar.Value) < Numero.Min **Then**

MsgBox ("Valor demasiado pequeño")

Datos\_Buscar.Value = Número.Min

**Else**

Número.Value = Val(Datos\_Buscar.Value)

**End If**

**End If**

**End If**

**End Sub**

**Private Sub** Lista\_Campos\_Change()

**Dim** i **As Integer**

**Dim** Tipo\_Datos **As String**

i = Lista\_Campos.ListIndex

**If** i >= 0 **Then**

Tipo\_Datos = Lista\_Campos.Column(1, i)

**If** Tipo\_Datos = "N" **Then**

Número.Enabled = True

**If** Lista\_Campos.Value = "Edad" **Then**

Número.Min = 18



```
Número.Max = 99
Número.SmallChange = 1
Datos_Buscar.Value = 0
Número.Value=0
End If
If Lista_Campos.Value = "Cantidad" Then
Número.Min = 10000
Número.Max = 500000
Número.SmallChange = 1000
Datos_Buscar .Value= 0
Número.Value=0
End If
Else
Número.Enabled = False
End If
End If.
End Sub
Private Sub Número_Change()
Datos_Buscar.Value = Número.Value
End Sub
```





## UNIDAD 11

### Funciones más comunes en Excel

Una función es una fórmula ya preparada por **Excel**, que permite ahorrar tiempo y cálculos, y que produce un resultado. Por ejemplo, imaginemos que tenemos que sumar una columna de datos numéricos:

	A	B	C	D
1				
2		<b>VENTAS</b>		
3		Enero	120	
4		Febrero	342	
5		Marzo	124	
6		Abril	654	
7		Mayo	123	
8				
9			1363	
10				

En el ejemplo de la izquierda podríamos colocar en la celda **C10** la fórmula: **=C3+C4+C5+C6+C7**, pero esto mismo resultaría muy complicado si en lugar de 5 celdas hubiese que sumar 100.

En lugar de esa fórmula, utilizamos la función **=SUMA(C3:C8)** que realizará exactamente la misma operación; sumar el rango de celdas C3:C8.

Las funciones aceptan unos valores (en este caso el rango de celdas) llamados **argumentos**.

Las funciones las podemos introducir de dos formas:

- **Mediante teclado.**
- **Mediante el asistente para funciones**

### FUNCIONES MÁS UTILIZADAS EN EXCEL

#### BUSCARV

La **BUSCARV()** nos sirve para buscar información dentro de nuestra hoja de cálculo.

Sintaxis

**=BUSCARV(Celda;Rango;Columna)**

**=BUSCARV(valor\_buscado;matriz\_buscar\_en;indicador\_columnas;ordenado)**

Una de las funciones más útiles que existen, ésta buscará el valor de una celda en un rango de celdas y retornará el contenido de **n** columnas a su derecha.

Ejemplo

Tenemos una lista muy extensa de artículos en nuestro almacén, y requerimos buscar las existencias de un producto, para ello se han puesto los primeros 3 renglones para poner la

información deseada, y en la celda C2 se introducirá la clave del producto que estamos buscando.

Antes de comenzar a buscar los valores, debemos de ordenar la tabla o matriz en donde vamos a buscar la información, de lo contrario, se mostrará el error #N/A el cual no indica que la matriz no está ordenada.

Sitúese en la celda C2

Teclee la siguiente fórmula **=BUSCARV(C1;A7:C15;2)**

C2      =BUSCARV(C1,A9:D16,2)				
	A	B	C	D
1		CODIGO A BUSCAR	AMIBACS	
2		DESCRIPCIÓN	AMIBAC JABON	
3		CANTIDAD EN EL ALMACEN	78	
4				
5				
6		VENTAS		
7				
8	CLAVE	DESCRIPCION	EXISTENCIAS	PRECIO
9	AMIBACJ	AMIBAC SOLUCION	10	143
10	AMIBACS	AMIBAC JABON	78	123
11	ESPECTRUMA	ESPECTRUM AMBIENTAL	45	52
12	ESPECTRUME	ESPECTRUM ENZIMATICO	88	156
13	ESPECTRUMV	ESPECTRUM DESINFECTANTE	12	52
14	MICRO1	MICRO KILL JABON DESINFECTANTE	24	91
15	PURIBAC2	PURIBAC 5%	100	52
16	PURIBAC3	PURIBAC 7%	65	52

Para obtener el nombre del producto que estamos buscando se escribe el código del artículo que estamos buscando en la celda C1 y Excel hará que aparezca automáticamente la **descripción** y la **cantidad** disponible en las dos celdas inferiores.

Este tipo de hojas ayuda a hacer una consulta a un listado. La fórmula mirará lo que hay en la celda **C1**, y lo buscará en el rango **A7:C15**. Una vez que lo encuentre, (lo encontrará en la 1ª columna), mostrará lo que hay 2 columnas a su derecha (contándose ella), es decir, la descripción del producto.

Observa detenidamente los tres argumentos que nos pide la función **=BUSCARV**, primero la celda donde estará lo que intentamos buscar (el código), luego el rango donde ha de buscarlo, y por último el número de columna que queremos mostrar.

Para escribir la fórmula que nos falta en C3

Sitúese en la celda C3

Teclee la siguiente fórmula **=BUSCARV(C1;A7:C15;3)**

Ahora sólo faltará comprobar las dos fórmulas escribiendo cualquier código de la lista de artículos en la celda **C1**.

Un detalle importante de la función **=BUSCARV( )** es que si la lista o rango donde hay que buscar está desordenada, tendremos que añadir la palabra **FALSO** al final de la fórmula.

## SI

La función **=SI( )** es una de las más potentes que tiene Excel. Esta función comprueba si se cumple una condición. Si ésta se cumple, da como resultado **VERDADERO**. Si la condición no se cumple, da como resultado **FALSO**.

Sintaxis

**=SI(Condición;Verdadero;Falso)**

**SI(prueba\_lógica;valor\_si\_verdadero;valor\_si\_falso)**

Ejemplo

Observa la hoja de la derecha, se trata de un ejemplo en el cual utilizaremos la función **SI**, la cual permitirá hacer un descuento del 10% al cliente si su compra es de contado, de otra manera no hará ningún descuento. La fórmula se introducirá en la celda **H27**, la fórmula, que se introducirá es la que se señala.

H27		=SI(H10="CONTADO",H26*10%,0)					
	A	B	C	D	E	F	G
1			<b>CALIDAD SANITARIA S.A. DE C.V.</b>				
2			Sucursal Durango Castaña #217 Fracc. Nogales				
3			Tel. 811.69.04 826.2548 cel. 044(618)808.4941 808.1659				
4			email:elizabethleyva2000@yahoo.com.mx www.calidadsanitaria.com				
5							
6			<b>LISTA DE PRODUCTOS QUIMICOS 2002</b>				
7							
8							
9	VENDIDO A:		FACTURA No.				129
10			CONDICIONES DE VTA.				CONTADO
11							
12							
13							
14							
15							
16							
17	CLAVE	DESCRIPCIÓN	CANT.	PRECIO UNIT.	IMPORTE		
18	AMIBACJ	AMIBAC JABON	\$ 10.00	\$ 143.00	\$ 1,430.00		
19							
20							
21							
22							
23							
24							
25							
26					SUBTOTAL \$ 1,430.00		
27					DESCUENTO \$ 143.00		
28					IVA \$ 193.05		
29					TOTAL \$ 1,480.05		

El desglose de la fórmula es el siguiente:

**=SI(H10="CONTADO",H26\*10%,0)**



**=SI(Y((H10="CONTADO"),(H26>1999.99)),H26\*10%,0)**

**=SI(Y** La letra **Y** controla que se cumpla una de las dos condiciones

**(H10="CONTADO"):** Primera condición, en la cual se pide que H10 tenga el valor de contado, lo que significa que la compra se hace de contado.

**(H26>1999.99)** Separada por coma, la segunda condición controla que la compra del total de la factura sea mayor de 1999.99, lo que significa que para hacer el descuento la compra mínima tendrá que ser de 2,000 pesos.

**H26\*10%,0)** el resto de la fórmula ya se había comentado anteriormente, en donde se obtiene el descuento si la compra es mayor de 2,000, o no se hace descuento si la compra es menor a 2,000 pesos.

## FUNCIONES ESTADÍSTICAS

Existen en Excel algunas funciones estadísticas de gran ayuda, pero antes de comenzar a utilizarlas daré un breve repaso a algunos conceptos estadísticos que son importante recordarlos o en su caso conocerlos para entender qué es lo que se está obteniendo al aplicar las funciones.

La estadística estudia las técnicas de ordenación, recuento, clasificación y presentación de datos numéricos en forma de tablas, gráficos y valores que resuman de forma precisa y coherente la información elegida. Algunas definiciones básicas de la estadística son:

**Frecuencia absoluta:** La *frecuencia absoluta* de un dato es el número de veces que aparece éste en una sucesión de datos. Se representa por **n1**

**Frecuencia relativa:** La *frecuencia relativa* de un resultado es el cociente entre la frecuencia absoluta y el total de resultados observados. Se representa por **f1**. Podemos calcular la frecuencia relativa con la fórmula: **f1 = n1/N** donde **N** es el número total de veces.

La suma de frecuencias relativas será siempre igual a **1**, independientemente del número total de observaciones. Veamos un ejemplo:

### Ejemplo 1

Un aula está formada por 10 alumnos, y en la primera evaluación del curso, se obtuvieron las siguientes notas:

5 3 8 9 10 2 5 8 1 6

¿Cuántos alumnos han obtenido un 5 de nota?

Respuesta: 2

¿Cuál es la frecuencia absoluta de esta puntuación?

Respuesta: **2**

¿Cuál es su frecuencia relativa?

Respuesta: **2/10**

¿Cuál es su frecuencia relativa en porcentaje?

Respuesta: **2/10 = 0.2 (el 20%)**

Observa la siguiente tabla: en la parte izquierda se representan las puntuaciones, y en la parte derecha las frecuencias absolutas de cada una de ellas:

Puntuaciones	Frecuencia abs.	Frecuencia relat.
1	1	1/10
2	1	1/10
3	1	1/10
4	0	0/10
5	2	2/10
6	1	1/10
7	0	0/10
8	2	2/10
9	1	1/10
10	1	1/10

Si sumamos las frecuencias relativas veremos que la suma total nos da el resultado de **1**, tal y como indicábamos al principio de esta parte teórica.

En Excel existen varias funciones que permiten trabajar con los datos que estamos estudiando:

**Media aritmética** (función **PROMEDIO**) la media aritmética es el cociente de la suma de valores de un conjunto de **n** elementos por el número **n**. Por ejemplo; en el caso anterior, si queremos determinar la nota media de los 10 alumnos:

$$(5 + 3 + 8 + 9 + 10 + 2 + 5 + 8 + 1 + 6) / 10 = 5.7$$

Si en Excel escribimos la misma lista en la columna 1, podemos escribir la siguiente fórmula para calcular la media:

$$=PROMEDIO(A1:A10)$$

**Mediana** (función **MEDIANA**) es el valor en medio de un conjunto de números, es decir, que la mitad de los números es mayor que la mediana y la otra mitad es menor. Por ejemplo, dada la siguiente sucesión de números:

1 2 3 4 5      La mediana sería el número **3**

y para la siguiente sucesión sería:

1 2 3 4 5 6      La mediana sería el número **3.5** (promedio de 3 y 4)

En nuestro ejemplo, la fórmula para calcular la mediana sería: **=MEDIANA(A1:A10)**, dando casualmente como resultado **5.7**

**Moda** (función **MODA**): Esta función es el valor de la variable que representa la máxima frecuencia. En otras palabras; es el valor que más se repite en un rango de datos.

En nuestro ejemplo, la función se escribiría: **=MODA(A1:A10)** y daría como resultado: **5**

**Desviación media** (función **DESVPROM**): es la media aritmética de los valores absolutos de las desviaciones de todos los valores del conjunto. Se utiliza para medir la dispersión de los valores en un conjunto de datos (ver el grado de desviación de la media). Su utilización en Excel sería:

**=DESVPROM(A1:A10)**

En nuestro ejemplo, la función se escribiría: **=DESVPROM(A1:A10)** y daría como resultado: **2.5**

**Desviación Estándar** (función **DESVEST**) Calcula la desviación estándar. La desviación estándar es la medida de la dispersión de los valores respecto a la media (valor promedio).

En nuestro ejemplo, la fórmula para calcular la desviación estándar es: **=DESVEST(A1:A10)**, dando como resultado: **3.05**

A continuación veremos la aplicación de la teoría mostrada anteriormente, utilizando las funciones estadísticas, para lo cual utilizaremos la siguiente tabla de alumnos, la cual contiene su peso y su estatura. A partir de ahí, realizaremos una serie de cálculos utilizando las funciones que vamos a estudiar. Vamos primero a ver sus sintaxis, y a continuación su aplicación en el ejemplo:

	A	B	C	D
1				
2				
3		<b>NOMBRE</b>	<b>PESO</b>	<b>ESTATURA</b>
4		ADAN EDUARDO GARCIA IBARRA	70	1.81
5		BLANCA ORALIA MORALES REYES	76	1.55
6		ALMA ELIZABETH LARA OROZCO	50	1.5
7		RUBEN AGUIRRE ELIZALDE	78	1.6
8		URIEL HERRERA GARCIA	77	1.7
9		FERNANDO MORA MUÑOZ	80	1.82
10		FRANCISCO LEYVA SOTO	68	1.6
11		ADRIANA GABRIELA GARCIA ANTUNEZ	40	1.4
12		ROSA ISELA GARCIA MERAZ	45	1.7
13		JUAN CARLOS MELENDEZ MELENDEZ	50	1.65
14		EFRAIN IBARRA JIMENEZ	85	1.82
15		ROMERO LUJAN GUILLERMO ANTONIO	70	1.7
16		HERNANDEZ CASAS JORGE ARMANDO	70	1.75
17		IBÁÑEZ RODRIGUEZ YADIRA	50	1.5



## PROMEDIO

Función que nos devolverá la media aritmética de los números o el rango encerrado entre paréntesis. También es conocida como media.

Sintaxis

**=PROMEDIO(Número1;Número2;.....)**

En donde **Número1;Número2, etc**, son celdas, números o un rango de números.

Ejemplo

En nuestro ejemplo, obtendremos la media de los pesos y las estaturas de 15 alumnos de la Escuela de Matemáticas. La fórmula para obtener la media de los pesos de los alumnos es: **=PROMEDIO(C4:C17)**. Para obtener la media de la estatura lo único que se cambiará es la columna C por la D.

C19		fx =PROMEDIO(C4:C17)		
	A	B	C	D
1				
2				
3		<b>NOMBRE</b>	<b>PESO</b>	<b>ESTATURA</b>
4	1	ADAN EDUARDO GARCIA IBARRA	70	1.81
5	2	BLANCA ORALIA MORALES REYES	76	1.55
6	3	ALMA ELIZABETH LARA OROZCO	50	1.50
7	4	RUBEN AGUIRRE ELIZALDE	78	1.60
8	5	URIEL HERRERA GARCIA	77	1.70
9	6	FERNANDO MORA MUÑOZ	80	1.82
10	7	FRANCISCO LEYVA SOTO	68	1.60
11	8	ADRIANA GABRIELA GARCIA ANTUNEZ	40	1.40
12	9	ROSA ISELA GARCIA MERAZ	45	1.70
13	10	JUAN CARLOS MELENDEZ MELENDEZ	50	1.65
14	11	EFRAIN IBARRA JIMENEZ	85	1.82
15	12	ROMERO LUJAN GUILLERMO ANTONIO	70	1.70
16	13	HERNANDEZ CASAS JORGE ARMANDO	70	1.75
17	14	IBAÑEZ RODRIGUEZ YADIRA	50	1.50
18				
19		PROMEDIO	64.92857143	1.65

## MAX y MIN

Estas funciones devuelven los valores máximo y mínimo respectivamente de una lista de números. En este ejemplo, el valor máximo de los pesos de los alumnos es 85 y el mínimo es el 40.

### Sintaxis

**=MAX(número1;número2; ...)**      o      **=MAX(RANGO)**

**=MIN(número1;número2; ...)**      o      **=MIN(RANGO)**

C20		fx =MAX(C4:C17)		
	A	B	C	D
1				
2				
3		<b>NOMBRE</b>	<b>PESO</b>	<b>ESTATURA</b>
4	1	ADAN EDUARDO GARCIA IBARRA	70	1.81
5	2	BLANCA ORALIA MORALES REYES	76	1.55
6	3	ALMA ELIZABETH LARA OROZCO	50	1.50
7	4	RUBEN AGUIRRE ELIZALDE	78	1.60
8	5	URIEL HERRERA GARCIA	77	1.70
9	6	FERNANDO MORA MUÑOZ	80	1.82
10	7	FRANCISCO LEYVA SOTO	68	1.60
11	8	ADRIANA GABRIELA GARCIA ANTUNEZ	40	1.40
12	9	ROSA ISELA GARCIA MERAZ	45	1.70
13	10	JUAN CARLOS MELENDEZ MELENDEZ	50	1.65
14	11	EFRAIN IBARRA JIMENEZ	85	1.82
15	12	ROMERO LUJAN GUILLERMO ANTONIO	70	1.70
16	13	HERNANDEZ CASAS JORGE ARMANDO	70	1.75
17	14	IBAÑEZ RODRIGUEZ YADIRA	50	1.50
18				
19		PROMEDIO O MEDIA	64.92857143	1.65
20		MÁXIMO	85	1.82
21		MÍNIMO	40	1.40
22				

### Ejemplo

Para obtener a la persona que tiene mayor peso del grupo de alumnos se introducirá en la celda C20 la función **=MAX(C4:C17)** , el resultado es 85

## MODA

Obtiene el valor que más se repite en un rango de celdas.

Sintaxis

**=MODA(número1;número2; ...) o =MODA(RANGO)**

C22		fx =MODA(C4:C17)		
	A	B	C	D
1				
2				
3		<b>NOMBRE</b>	<b>PESO</b>	<b>ESTATURA</b>
4	1	ADAN EDUARDO GARCIA IBARRA	70	1.81
5	2	BLANCA ORALIA MORALES REYES	76	1.55
6	3	ALMA ELIZABETH LARA OROZCO	50	1.50
7	4	RUBEN AGUIRRE ELIZALDE	78	1.60
8	5	URIEL HERRERA GARCIA	77	1.70
9	6	FERNANDO MORA MUÑOZ	80	1.82
10	7	FRANCISCO LEYVA SOTO	68	1.60
11	8	ADRIANA GABRIELA GARCIA ANTUNEZ	40	1.40
12	9	ROSA ISELA GARCIA MERAZ	45	1.70
13	10	JUAN CARLOS MELENDEZ MELENDEZ	50	1.65
14	11	EFRAIN IBARRA JIMENEZ	85	1.82
15	12	ROMERO LUJAN GUILLERMO ANTONIO	70	1.70
16	13	HERNANDEZ CASAS JORGE ARMANDO	70	1.75
17	14	IBAÑEZ RODRIGUEZ YADIRA	50	1.50
18				
19		PROMEDIO O MEDIA	64.92857143	1.65
20		MÁXIMO	85	1.82
21		MÍNIMO	40	1.40
22		VALORA QUE MAS SE REPITE (moda)	70	1.70

## Ejemplo

Para obtener la moda (el valor que más se repite) nos posicionaremos en la celda C22 y a continuación introduciremos la fórmula **=MAX(C4:C17)** y presionamos enter. El resultado será 70 para los pesos y 1.70 para la altura de los alumnos.

## CONTAR y CONTARA

La primera función devuelve cuántas veces aparece un elemento numérico en una lista; mientras que la segunda cuenta las celdas no vacías en el rango dado, el uso de esta función es importante para ver el número de datos válidos que tenemos en una encuesta.

Sintaxis

**CONTAR(ref1;ref2;...) o CONTAR(Rango)**

**CONTAR(ref1;ref2;...) o CONTAR(Rango)**

## Ejemplo

Queremos saber cuántos valores numéricos existen en nuestro ejemplo, para ello nos posicionamos en la celda C23 e introduciremos la fórmula **=CONTAR(B4:D17)** y obtendremos como resultado 28, que es la suma de las 14 estaturas y los 14 pesos de los alumnos.

C23		=CONTAR(B4:D17)		
	A	B	C	D
1				
2				
3		<b>NOMBRE</b>	<b>PESO</b>	<b>ESTATURA</b>
4	1	ADAN EDUARDO GARCIA IBARRA	70	1.81
5	2	BLANCA ORALIA MORALES REYES	76	1.55
6	3	ALMA ELIZABETH LARA OROZCO	50	1.50
7	4	RUBEN AGUIRRE ELIZALDE	78	1.60
8	5	URIEL HERRERA GARCIA	77	1.70
9	6	FERNANDO MORA MUÑOZ	80	1.82
10	7	FRANCISCO LEYVA SOTO	68	1.60
11	8	ADRIANA GABRIELA GARCIA ANTUNEZ	40	1.40
12	9	ROSA ISELA GARCIA MERAZ	45	1.70
13	10	JUAN CARLOS MELENDEZ MELENDEZ		
14	11	EFRAIN IBARRA JIMENEZ	85	1.82
15	12	ROMERO LUJAN GUILLERMO ANTONIO	70	1.70
16	13	HERNANDEZ CASAS JORGE ARMANDO	70	1.75
17	14	IBÁÑEZ RODRIGUEZ YADIRA	50	1.50
18				
19		PROMEDIO O MEDIA	66.07692308	1.65
20		MÁXIMO	85	1.82
21		MÍNIMO	40	1.40
22		VALORA QUE MAS SE REPITE (moda)	70	1.70
23		CANTIDAD DE VALORES numericos y texto	28	40
24				

Ahora contaremos la cantidad de datos que existen en la lista, para ello nos colocamos en la celda D23 y tecleamos la siguiente fórmula **=CONTARA(B4:D17)**, el resultado que obtenemos es 40, debido a que 2 celdas no tienen valores.

## CONTAR.SI

Cuenta las celdas no vacías de un rango, y que cumplen con un criterio especificado.

### Sintaxis

**=CONTAR.SI(Rango)**

**=CONTAR.SI(rango;criterio)**

El rango se refiere al rango de celdas en las cuales quiero buscar los valores, y el criterio es la pregunta que estoy tratando de responderme.

## Ejemplo

Se requiere saber cuál es la cantidad de alumnos que pesan 70 kilos o más, para ello nos posicionamos en la celda C24 y escribimos la fórmula **=CONTAR.SI(C4:D17,">69.9970")**, el C4:D17 se refiere al rango de celdas en donde voy a buscar el valor, y el criterio es que la persona pese 69.99 kilos o más ">69.99".

C24		=CONTAR.SI(C4:D17,">69.9970")		
	A	B	C	D
1				
2				
3		<b>NOMBRE</b>	<b>PESO</b>	<b>ESTATURA</b>
4	1	ADAN EDUARDO GARCIA IBARRA	70	1.81
5	2	BLANCA ORALIA MORALES REYES	76	1.55
6	3	ALMA ELIZABETH LARA OROZCO	50	1.50
7	4	RUBEN AGUIRRE ELIZALDE	78	1.60
8	5	URIEL HERRERA GARCIA	77	1.70
9	6	FERNANDO MORA MUÑOZ	80	1.82
10	7	FRANCISCO LEYVA SOTO	68	1.60
11	8	ADRIANA GABRIELA GARCIA ANTUNEZ	40	1.40
12	9	ROSA ISELA GARCIA MERAZ	45	1.70
13	10	JUAN CARLOS MELENDEZ MELENDEZ		
14	11	EFRAIN IBARRA JIMENEZ	85	1.82
15	12	ROMERO LUJAN GUILLERMO ANTONIO	70	1.70
16	13	HERNANDEZ CASAS JORGE ARMANDO	70	1.75
17	14	IBAÑEZ RODRIGUEZ YADIRA	50	1.50
18				
19		PROMEDIO O MEDIA	66.07692308	1.65
20		MÁXIMO	85	1.82
21		MÍNIMO	40	1.40
22		VALOR QUE MAS SE REPITE (moda)	70	1.70
23		CANTIDAD DE VALORES numericos y texto	26	40
24		CONTAR.SI (# de personas con peso mayor de 70)	8	
25				

## MEDIANA(Números)

La función MEDIANA devuelve la mediana de los números. La mediana es el número que se encuentra en medio de un conjunto de números, es decir, la mitad de los números es mayor que la mediana y la otra mitad es menor.

### Sintaxis

**=MEDIANA(número1;número2; ...)**    o    **=MEDIAN(RANGO)**

## Ejemplo

Para obtener la mediana de la estatura de los alumnos, nos posicionamos en la celda D25 y tecleamos la fórmula **=MEDIANA(D4:D17)**, y el resultado es 1.68.

D25		fx =MEDIANA(D4:D17)		
	A	B	C	D
1				
2				
3		<b>NOMBRE</b>	<b>PESO</b>	<b>ESTATURA</b>
4	1	ADAN EDUARDO GARCIA IBARRA	70	1.81
5	2	BLANCA ORALIA MORALES REYES	76	1.55
6	3	ALMA ELIZABETH LARA OROZCO	50	1.50
7	4	RUBEN AGUIRRE ELIZALDE	78	1.60
8	5	URIEL HERRERA GARCIA	77	1.70
9	6	FERNANDO MORA MUÑOZ	80	1.82
10	7	FRANCISCO LEYVA SOTO	68	1.60
11	8	ADRIANA GABRIELA GARCIA ANTUNEZ	40	1.40
12	9	ROSA ISELA GARCIA MERAZ	45	1.70
13	10	JUAN CARLOS MELENDEZ MELENDEZ	69	1.65
14	11	EFRAIN IBARRA JIMENEZ	85	1.82
15	12	ROMERO LUJAN GUILLERMO ANTONIO	70	1.70
16	13	HERNANDEZ CASAS JORGE ARMANDO	70	1.75
17	14	IBAÑEZ RODRIGUEZ YADIRA	50	1.50
18				
19		PROMEDIO O MEDIA	66.28571429	1.65
20		MÁXIMO	85	1.82
21		MÍNIMO	40	1.40
22		VALORA QUE MAS SE REPITE (moda)	70	1.70
23		CANTIDAD DE VALORES numericos y texto	28	42
24		CONTAR.SI (# de personas con peso mayor de 70)	8	
25		MEDIANA	70	1.68
26				

Para la mediana seguimos el mismo procedimiento, sólo que en la celda C25.

## DESVEST

La desviación estándar es la medida de la dispersión de los valores respecto a la media (valor promedio).

### Sintaxis

**=DESVEST(número1;número2; ...)**    o    **=DESVEST(RANGO)**

## Ejemplo

Para obtener la desviación estándar de los pesos de los alumnos, nos posicionamos en la celda C26 y tecleamos la fórmula **=DESVEST(D4:D17)**, y el resultado es 14.7880632.

Para Obtener la desviación de las estaturas, seguimos el mismo proceso, solamente que la fórmula la introduciremos en la celda D27.

C26		=DESVEST(C4:C17)		
	A	B	C	D
1				
2				
3		<b>NOMBRE</b>	<b>PESO</b>	<b>ESTATURA</b>
4	1	ADAN EDUARDO GARCIA IBARRA	70	1.81
5	2	BLANCA ORALIA MORALES REYES	76	1.55
6	3	ALMA ELIZABETH LARA OROZCO	50	1.50
7	4	RUBEN AGUIRRE ELIZALDE	78	1.60
8	5	URIEL HERRERA GARCIA	77	1.70
9	6	FERNANDO MORA MUÑOZ	80	1.82
10	7	FRANCISCO LEYVA SOTO	68	1.60
11	8	ADRIANA GABRIELA GARCIA ANTUNEZ	40	1.40
12	9	ROSA ISELA GARCIA MERAZ	45	1.70
13	10	JUAN CARLOS MELENDEZ MELENDEZ	50	1.65
14	11	EFRAIN IBARRA JIMENEZ	85	1.82
15	12	ROMERO LUJAN GUILLERMO ANTONIO	70	1.70
16	13	HERNANDEZ CASAS JORGE ARMANDO	70	1.75
17	14	IBAÑEZ RODRIGUEZ YADIRA	50	1.50
18				
19		PROMEDIO O MEDIA	66.28571	1.65
20		MÁXIMO	85	1.82
21		MÍNIMO	40	1.40
22		VALORES QUE MA SE REPITEN (moda)	70	1.70
23		CANTIDAD DE VALORES numericos y texto	28	42.00
24		CONTAR SI (# personas con peso mayor de 70)	8	
25		MEDIANA	70	1.68
26		DESVIACION ESTANDAR	14.7880632	0.131090104
27				

## MATRICES

El concepto de **Matriz** viene de los lenguajes de programación y de la necesidad de trabajar con varios elementos de forma rápida y cómoda. Podríamos decir que una matriz es una serie de elementos formando filas (matriz bidimensional) o filas y columnas (matriz tridimensional).

La siguiente tabla representa una matriz unidimensional (vector)

1	2	3	4	5
---	---	---	---	---

...ahora una matriz bidimensional:

1,1	1,2	1,3	1,4	1,5
2,1	2,2	2,3	2,4	2,5
3,1	3,2	3,3	3,4	3,4

Observa por ejemplo el nombre del elemento **3,4** que significa que está en la posición de fila 3, columna 4. En Excel, podemos tener un grupo de celdas en forma de matriz y aplicar una fórmula determinada en ellas de forma que tendremos un ahorro del tiempo de escritura de fórmulas.

En Excel, las fórmulas que hacen referencia a matrices se encierran entre corchetes **{}**. Hay que tener en cuenta al trabajar con matrices lo siguiente:

- No se puede cambiar el contenido de las celdas que componen la matriz
- No se pueden eliminar o mover celdas que componen la matriz
- No se pueden insertar nuevas celdas en el rango que compone la matriz

*Crea la siguiente hoja:*

	A	B	C	D	E
1					
2		<b>Art.1</b>	<b>Art.2</b>	<b>Art.3</b>	<b>Art.4</b>
3	Unidades	12	15	17	13
4	Precio	45	69	45	33
5	Total Unidad	540	1035	765	429
6					
7	<b>TOTAL</b>	<b>2769</b>			

Si te sitúas en la celda **B5**, observarás que hemos hecho una simple multiplicación para calcular el precio total de las unidades. Lo mismo pasa con las demás fórmulas.

En vez de esto, podríamos haber combinado todos los cálculos posibles en uno solo utilizando una fórmula matricial.

Una fórmula matricial se tiene que aceptar utilizando la combinación de teclas **CTRL+MYSC+Intro** y Excel colocará los corchetes automáticamente.

*Borra las celdas adecuadas para que quede la hoja de la siguiente forma:*



	A	B	C	D	E
1					
2		<b>Art.1</b>	<b>Art.2</b>	<b>Art.3</b>	<b>Art.4</b>
3	Unidades	12	15	17	13
4	Precio	45	69	45	33
5	Total Unidad				
6					
7	<b>TOTAL</b>				

Sitúa el cursor en la celda **B7** e introduce la fórmula:

**=SUMAPRODUCTO(B3:E3\*B4:E4)**

Acepta la fórmula usando la combinación de teclas adecuada.

Observa cómo hemos obtenido el mismo resultado tan sólo con introducir una fórmula.

Observa la misma en la barra de fórmulas. Ahora hay que tener cuidado en editar celdas que pertenezcan a una matriz, ya que no se pueden efectuar operaciones que afecten sólo a un rango de datos. Cuando editamos una matriz, editamos todo el rango como si de una sola celda se tratase.

### Constantes matriciales

Al igual que en las fórmulas normales podemos incluir referencias a datos fijos o constantes, en las fórmulas matriciales también podemos incluir datos constantes. A estos datos se les llama **constantes matriciales** y se debe incluir un separador de columnas (símbolo ;) y un separador de filas (símbolo \).

Por ejemplo, para incluir una matriz como constante matricial:

1. 25
2. 18

Debemos escribir: **{30;25\31;18}**

	A	B	C	D
1	5	5		
2	5	5		
3				
4	1	2		



1. *Escribe estas celdas en la hoja2*
2. *Selecciona el rango C1:D2*
3. *Escribe la fórmula: =A1:B2\*{10;20\30;40}*
4. *Acepta la fórmula con la combinación de teclas adecuada.*

Observa que Excel ha ido multiplicando los valores de la matriz por los números introducidos en la fórmula:

Cuando trabajamos por fórmulas matriciales, cada uno de los elementos de la misma, debe tener idéntico número de filas y columnas, porque de lo contrario, Excel expandiría las fórmulas matriciales. Por ejemplo:

$=\{1;2;3\}*\{2\3\}$  se convertiría en  $=\{1;2;3\1;2;3\}*\{2;2;2\3;3;3\}$

5. *Selecciona el rango C4:E5*
6. *Introduce la fórmula: =A4:B4+{2;5;0\3;9;5} y acéptala.*

Observemos que Excel devuelve un mensaje de error diciendo que el rango seleccionado es diferente al de la matriz original.

Otros usos importantes de las matrices, son para la obtención del determinante, la inversa de una matriz y la multiplicación de matrices, para ello, el Excel cuenta con funciones específicamente diseñadas para cada una.

## MDETERM

Devuelve la matriz determinante de una matriz. Para poder obtener el determinante de una matriz, ésta debe tener el mismo número de renglones y de columnas.

### Sintaxis

**MDETERM(matriz)**

Matriz es una matriz numérica con el mismo número de filas y de columnas.

### Ejemplo

Deseamos obtener el determinante de la siguiente matriz, observemos que en

A1 se tiene el valor de 1,

A2=3,

B1=2

y B2=4,

	A	B	C
1	1	2	
2	3	4	
3			
4			
5			

a continuación nos posicionamos en la celda en donde se va a obtener el resultado del determinante, que en este caso es A4 como se muestra a en la imagen.

	A	B
1	1	2
2	3	4
3		
4		
5		

Después, tecleamos la fórmula para obtener el determinante de una matriz, observemos que la tecla seleccionada es la **A4**, y a continuación se teclea la siguiente función **=MDETERM(A1:B2)**, y a continuación se obtendrá el valor de **-2**

SUMA				
	A	B	C	
1	1	2		
2	3	4		
3				
4	=MDETERM(A1:B2)			
5				
6				

A5			
	A	B	
1	1	2	
2	3	4	
3			
4	-2		
5			
6			

## TRANSPONER

Devuelve un rango de celdas vertical como un rango horizontal o viceversa. TRANSPONER debe introducirse como una fórmula matricial en un rango que tenga el mismo número de filas y columnas, respectivamente, que el número de columnas y filas en una matriz. Utilice TRANSPONER para cambiar la orientación vertical y horizontal de una matriz en una hoja de cálculo.

Sintaxis

**TRANSPONER(matriz)**

Matriz es una matriz numérica con el mismo número de filas y de columnas.

Ejemplo

Deseamos obtener la matriz transpuesta, observemos que en **A1** se tiene el valor de **1**, **A2=3**, **B1=2** y **B2=4**, a continuación seleccionamos un rango del mismo tamaño, que en donde se encuentra la matriz, en este caso seleccionaremos el siguiente rango de celdas **A4:B5**, como se muestra a continuación:

	A	B	C
1	1	2	
2	3	4	
3			
4			
5			

	A4		f <sub>x</sub>
	A	B	
1	1	2	
2	3	4	
3			
4			
5			
6			

Una vez seleccionado el rango y posicionado en la celda A4, se introduce la siguiente fórmula: =TRANSPONER(A1:B2), pero tenga cuidado, **no presione enter para introducir la fórmula**, la forma de introducir fórmulas matriciales, es la siguiente: mantenga presionada la tecla [CTRL], a continuación presione la tecla [MAYÚS] y manténgala presionada y a continuación presione la tecla [ENTER], podrá observar

que la fórmula cambió y ahora estará entre llaves (la forma condensada es la siguiente [CTRL]+ [MAYÚS]+ [ENTER]).

	A4		f <sub>x</sub>		
	A	B	C	D	
1	1	2			
2	3	4			
3					
4	1	3			
5	2	4			
6					
7					

La matriz transpuesta, es la que se obtiene en el rango de celdas de A4:B5.

## MINVERSA

Devuelve la matriz inversa de la matriz almacenada en una matriz.

Sintaxis

**MINVERSA(matriz)**

Matriz. Es una matriz numérica con el mismo número de filas y de columnas.

Ejemplo

Deseamos obtener la matriz inversa de la siguiente matriz, observemos que en

A1 se tiene el valor de 1,

A2=3,

B1=2

y B2=4,

	A	B	C
1	1	2	
2	3	4	
3			
4			
5			

en seguida seleccionamos un rango del mismo tamaño, en donde se encuentra la matriz, en este caso seleccionaremos el siguiente rango de celdas A4:B5, como se muestra a continuación:

	A4		f <sub>x</sub>
	A	B	
1	1	2	
2	3	4	
3			
4			
5			
6			

Una vez seleccionado el rango y posicionado en la celda A4, se introduce la siguiente fórmula: =MINVERSA(A1:B2), pero tenga cuidado, **no presione enter para introducir la fórmula**, la forma de introducir fórmulas matriciales, es la siguiente: mantenga presionado la tecla [CTRL], a continuación presione la tecla [MAYÚS] y manténgala presionada y a continuación presione la tecla [ENTER], podrá observar que la fórmula cambió y ahora estará entre llaves (la forma condensada es la siguiente [CTRL]+ [MAYÚS]+ [ENTER]).

	SUMA			f <sub>x</sub>	=minversa(A1:B2)
	A	B	C		
1	1	2			
2	3	4			
3					
4					=minversa(A1:B2)
5					MINVERSA(matriz) 4
6					
7					

La matriz transpuesta, es la que se obtiene en el rango de celdas de A4:B5.

	A4		f <sub>x</sub>	{=MINVERSA(A1:B2)}
	A	B	C	D
1	1	2		
2	3	4		
3				
4	-2	1		
5	1.5	-0.5		
6				
7				

## MMULT

Devuelve la matriz producto de dos matrices. El resultado es una matriz con el mismo número de filas que matriz1 y el mismo número de columnas que matriz2.

Sintaxis

**MMULT(matriz)**

Matriz. Es una matriz numérica con el mismo número de filas y de columnas.

## Ejemplo

Deseamos

obtener

la

multiplicación de

2 matrices, la

primera se

encuentra en el

rango de **A1:B2** y

la segunda matriz

se encuentra en el

rango de **D1:E2**

como se muestra a continuación.

	A	B	C	D	E
1	1	2		5	6
2	3	4		7	8
3					
4	=mmult(A1:B2,D1:E2 )				
5	MMULT(matriz1, matriz2)				
6					
7					

Una vez seleccionado el rango y posicionado en la celda A4, se introduce la siguiente fórmula: =MMULT(A1:B2), pero tenga cuidado, **no presione enter para introducir la fórmula**, la forma de introducir fórmulas matriciales, es la siguiente: mantenga presionado la tecla [CTRL], a continuación presiona la tecla [MAYÚS] y manténgala presionada y a continuación presione la tecla [ENTER], podrás observar que la fórmula cambió y ahora estará entre llaves (la forma condensada es la siguiente [CTRL]+ [MAYÚS]+ [ENTER]).

EL resultado de la multiplicación de las 2 matrices es el que se muestra a continuación, obsérvese que además la fórmula se encuentra entre llaves, esto se hace automáticamente al presionar las teclas que se mencionaron anteriormente.

	A	B	C	D	E	F
1	1	2		5	6	
2	3	4		7	8	
3						
4	19	22				
5	43	50				
6						
7						

## La función =SI()

Una de las funciones más potentes que se utilizan en Excel es la función =SI(). Esta función tiene la siguiente estructura:

=SI(condición;verdadero;falso)

Donde la condición se tiene que cumplir. Si ésta se cumple, se ejecutará verdadero, o en caso contrario, se ejecutará falso.

Por ejemplo:

=SI(A3>B12;"Correcto";"Incorrecto")

Si la celda A3 es mayor que la celda B12, aparecerá la palabra Correcto, en caso contrario, aparecerá la palabra Incorrecto.

=SI(A1="Bajo mínimos";"Quiebra";"Normal")

Si la celda A1 contiene la palabra Bajo mínimos, en la celda actual aparecerá la palabra Quiebra, en caso contrario, aparecerá la palabra Normal.

=SI(O(A1=B1;C1=D1);"Bien";"Mal")

Aquí ha de cumplirse una de las dos condiciones. Nótese la utilización del operador O, es decir, que se tiene que cumplir una de las dos condiciones.

## Sumar.si

La función que vamos a ver a continuación es una función avanzada de Excel que nos permite hacer sumas o promedios selectivos. Para comprender el funcionamiento de esta función, escribe la siguiente tabla en una hoja vacía de Excel:

	A	B	C	D
1	Producto	Precio	Cantidad	
2	TV	25,000	1	
3	PC	15,000	2	
4	TV	5,000	2	
5				
6	Total TV			
7	Total PC			

En esta tabla aparecen reflejadas una serie de ventas de una tienda de electrodomésticos. Solamente hemos indicado tres ventas (dos televisores y un PC), aunque en la realidad, esta lista sería mucho más larga.

En la parte inferior aparecen dos títulos 'Total TV' y 'Total PC', que tienen que reflejar cuántos televisores y cuántos PCs se han vendido.

En este caso no podemos utilizar el botón de autosuma (que suma una columna entera de números), ya que aparecerían sumados los precios de los televisores y los PCs juntos (25,000 + 15,000 + 5,000), así que tenemos que solucionarlo de otra forma.

Lo ideal en este caso es utilizar la función `sumar.si()`:

Sitúa el cursor al lado del rótulo 'Total TV', es decir, en la celda B6. Aquí es donde vamos a sumar los precios de todos los televisores vendidos.

Pulsa el botón de funciones (fx). Busca la función `sumar.si()`, y selecciónala. Verás que aparece una nueva ventana con tres casillas. En esta ventana vamos a definir los parámetros de la función.

La primera casilla es la titulada: 'Rango'. En esta casilla tenemos que indicar a Excel dónde está la tabla que vamos a sumar. En este caso, nuestra tabla comienza en la celda A1 y termina en la C4 (donde está la última cantidad) así que puedes indicar 'A1:C4' (sin las comillas). Esto indica la función que debe utilizar la tabla que comienza en la celda A1 y termina en la C4. Es importante que escribas dos puntos (:) entre ambas direcciones de celda.

La segunda casilla es la titulada: 'Criterio'. En esta casilla tenemos que indicarle a Excel qué es lo que queremos sumar, como estamos realizando el 'Total de TV', tendremos que indicarle que queremos sumar los televisores. Para ello tienes que escribir "TV" (INCLUYENDO las comillas), es importante que escribas las comillas dobles antes de TV y después de TV. También es importante que escribas "TV" y no "televisor", por ejemplo, ya que en la tabla aparece así.

La tercera casilla es la titulada 'Rango\_Suma'. Aquí tenemos que indicar qué columna queremos que se sume. Estamos calculando el precio, y el precio está en la columna B, más concretamente, los precios comienzan en la celda B1 y terminan en la B4 (donde aparece el último precio), así que tendremos que escribir B1:B4

Pulsa el botón Aceptar. Si has realizado bien los pasos, debe aparecer el resultado de la suma de los televisores ( $25,000 + 25,000 = 50,000$ )

Puedes intentar hacer la suma de los PCs de una forma muy parecida: sitúa el cursor en la celda B7, que es donde tiene que aparecer la suma de los PCs vendidos. Pulsa el botón del asistente de funciones (fx) y busca la función `sumar.si()`

En la primera casilla: 'Rango' tienes que indicar dónde está la tabla en nuestra hoja. Al igual que en el caso anterior, nuestra tabla comienza en la celda A1 y termina en la C4, así que debemos indicar A1:C4 (con los dos puntos en medio)

En la segunda casilla 'Criterio' tienes que indicarle a Excel qué es lo que quieres sumar, como vamos a sumar los PCs, tienes que escribir 'PC' (sin las comillas) En la tercera casilla 'Rango\_suma' hay que indicar dónde están los números que queremos sumar (en este caso B1:B4)



Pulsa el botón Aceptar y, si has realizado bien los pasos anteriores, debería aparecer el total de PCs (como solo hay uno y vale 150.000, debería aparecer esta cantidad).

Prueba ahora tú a calcular la cantidad de televisores vendidos. Es prácticamente igual que en el caso anterior, pero los números que se van a sumar están en la columna C en lugar de la B (C1:C4 en lugar de B1:B4)

Por último calcula la cantidad de PCs vendidos

Bdsuma(), Bdpromedio(), Bdproducto(), Bdmin(), Bdmax()

Las funciones que vamos a ver trabajan de una forma muy parecida a otra función que quizás ya conozcas: sumar.si(). La diferencia es que no solo podemos hacer sumas, sino multiplicar, calcular promedios y el número más grande o más pequeño, según utilicemos una función u otra. Tienes que tener claro que todas las funciones que vamos a ver se definen exactamente igual, si aprendes a manejar una de ellas, ya las has aprendido todas. Vamos a verlo en el siguiente ejemplo:

	A	B	C	D
1	Corredor	Tiempo (en minutos)		
2	Fast Slim	50		
3	Aitor Tuga	55		
4	Fast Slim	57		
5	Aitor Tuga	60		
6	Fast Slim	62		
7	Aitor Tuga	61		
8		Corredor	Corredor	
9		Fast Slim	Aitor Tuga	
10	Media			
11	Mejor tiempo			
12	Peor tiempo			
13	Suma			

En esta tabla están representados los tiempos (en minutos, para no complicarlo mucho) que tardan dos corredores de maratón en completar un recorrido. El más rápido, evidentemente, será el que tarde menos minutos.

Para calcular las estadísticas de la parte inferior de la hoja (suma, media, mejor tiempo y peor tiempo) vamos a utilizar las funciones BD:

Sitúa el cursor en la celda B10, que es donde se va a calcular la media de tiempo de Fast Slim.

Pulsa el botón del asistente para funciones (fx) y selecciona la función Bdpromedio(). La función Bdpromedio() calcula la media de una serie de números, verás que aparece una nueva ventana con tres casillas:

En la primera casilla “Base\_de\_datos” tenemos que indicar dónde está nuestra tabla. La tabla comienza en la celda A1 y termina en la celda B7 (la tabla donde están los tiempos de los corredores, que son los que vamos a promediar, por eso no se indica hasta la fila 13). Así que tienes que escribir A1:B7

En la segunda casilla “Nombre\_de\_campo” tenemos que indicar qué columna se va a promediar. Nosotros podemos verlo claro, entre otras cosas, porque sólo hay una columna que se puede promediar (la del tiempo), pero Excel no lo ve tan fácil, así que escribe en esta casilla B1. ¿Por qué B1? porque es la primera celda de la columna que queremos calcular.

Ya le hemos dicho a Excel que queremos calcular la media de la columna B, pero si no le indicamos nada, calculará la media de toda la columna. Como nosotros estamos intentando calcular la media del corredor Fast Slim, tenemos que indicarlo de alguna forma. Aquí es donde entra en juego la tercera casilla. ¿Cómo le podemos indicar a Excel que queremos calcular sólo la media de Fast Slim? Fíjate que en las celdas B8 y B9 aparecen las palabras “corredor” y “Fast Slim”. Esto será suficiente para que Excel comprenda qué es lo que queremos, así que escribe en la tercera casilla B8:B9.

Pulsa el botón Aceptar y, si has realizado bien los pasos, debería aparecer la media de tiempos de Fast Slim.

Si es la primera vez que utilizas la función Bdpromedio() tienes aproximadamente un 40% de posibilidades de que te haya salido bien el resultado. Si te ha salido mal o te ha salido un error mira a continuación algunos de los errores más frecuentes que se cometen:

- El error más común es el de no escribir los valores correctamente dentro de la hoja. Por ejemplo: Para Excel no es lo mismo Fast Slim que FastSlim o Fast Slim (con dos espacios en medio en lugar de uno). De igual forma, las cabeceras de las columnas deben estar escritas igual en todos sitios, por ejemplo: si en la celda A1 has escrito la palabra corredor y en la B8 corredores o al contrario.
- Otro error muy común es el de dar espacios en blanco detrás de una palabra o frase por ejemplo: Podemos escribir en una celda Aitor Tuga con un espacio en blanco entre ambas palabras, esto es correcto. Pero no podemos dar un espacio en blanco al final, es decir, detrás de Tuga, ya que Excel lo tiene en cuenta y nosotros no vamos a verlo.
- También puedes obtener errores si no has escrito la tabla que está al principio de esta hoja exactamente igual en Excel. Ten en cuenta que los valores para las

fórmulas que se han definido aquí están expresados según la tabla de arriba. Si en lugar de empezar a escribir en la celda A1, lo has hecho en la C4, no coincidirán las filas ni las columnas.

- Otros errores pueden ocurrir al teclear los valores en la ventana de la función. Por ejemplo, si tienes que escribir B8:B9 y, por error, tecleas B8:B8, por ejemplo.

Así que ya sabes, si no te ha salido a la primera, comprueba los cuatro puntos anteriores y vuelve a intentarlo.

La forma de calcular el resto de las funciones es prácticamente igual. Para calcular la suma de tiempo de Fast Slim:

Sitúa el cursor en la celda B13, que es donde debe aparecer el resultado. Pulsa el botón del asistente para funciones (fx) y busca Bdsuma(). Verás que aparecen las mismas tres casillas que cuando utilizaste Bdpromedio(). Rellénalas de la siguiente forma:

Casilla 1 : 'Base\_de\_datos'. A1:B7 (donde está la tabla con los datos)

Casilla 2 : 'Nombre\_de\_campo'. B1 (esto indica a Excel que lo que se van a sumar son los números de abajo)

Casilla 3 : 'Criterios'. B8:B9 (esto indica a Excel que sólo queremos sumar los tiempos de Fast Slim)

Pulsa el botón Aceptar. Si no obtienes el resultado correcto comprueba la lista de posibles errores del apartado anterior.

Para calcular el mejor tiempo, vamos a utilizar la función Bdmin(). Esta función nos devuelve el número más pequeño de la lista. Sitúa el cursor en la celda B11 y busca en el asistente de funciones Bdmin(). Sigue los mismos pasos que realizaste para las funciones Bdsuma() y Bdpromedio().

Para calcular el peor tiempo, utiliza la función Bdmax() de la misma forma que utilizaste Bdmin() o cualquiera de las otras.

Calcula los resultados para Aitor Tuga

## **=PAGO()**

Esta función calcula los pagos periódicos que tendremos que realizar sobre un préstamo, a un interés determinado, y en un tiempo x. Podremos ver cuánto tendremos que pagar mensualmente, o cuánto nos cobran los bancos de intereses. Nos permitirá jugar con diferentes capitales, años o tipos de interés.

## Sintaxis

### **=PAGO(Interés;Tiempo;Capital)**

Esta fórmula nos calculará el pago anualmente. Si queremos saber los pagos mensuales tendremos que dividir el interés por 12 y multiplicar el tiempo por 12. Observa:

### **=PAGO(Interés/12;Tiempo\*12;Capital)**

## Ejemplo:

Supongamos que hemos de calcular los pagos mensuales y anuales periódicos del siguiente supuesto:

Celda B5: **=PAGO(B2;B3;B1)**

Celda B6: **=PAGO(B2/12;B3\*12;B1)**

Observa que la fórmula PAGO ofrece un resultado en negativo (rojo). Si queremos convertir el resultado en un número positivo, debemos encerrar la función dentro de otra función: **=ABS()**. La función ABS significa absoluto. Un número absoluto de otro número, siempre será positivo. La fórmula en ese caso sería: **=ABS(PAGO(B2/12;B3\*12;B1))**.

	A	B
1	<b>Capital</b>	1,000
2	<b>Interés</b>	6.50%
3	<b>Período</b>	1
4		
5	Anual	<b>-\$1,065.00</b>
6	Mensual	<b>-\$86.30</b>
7		

Como ya hemos dicho, en este tipo de hojas podemos probar a cambiar cantidades de las celdas B1,B2 y B3 y comprobar los distintos resultados. A continuación tienes un completo e interesante ejemplo de un supuesto de crédito desglosado mes a mes. En este ejemplo se utiliza una función nueva: **=PAGOINT()**, que desglosa el interés que pagamos de la cantidad mensual. La función **=PAGO()** nos muestra lo que debemos pagar, pero no nos dice cuánto pagamos de capital real y de intereses. La función **=PAGOINT()** realiza esto último.

Colocaremos y comentaremos las fórmulas de las dos primeras filas. A partir de la segunda fila, sólo restará copiar las fórmulas hacia abajo. Supongamos un crédito de 10,000 pesos con un interés del 24% en un plazo de 2 años, es decir, 24 meses.

Observa la primera línea de fórmulas:

	A	B	C	D	E	F
1	Capital	10,000				
2	Interés	24.00%				
3	Período	2				
4						
5	No. de pago	Cantidad	Capital	Interés	Acumulado	Pendiente
6	1	\$528.71	\$328.71	\$200.00	\$328.71	\$9,671.29

**A6** Número de mes que se paga

**B6** Cálculo del pago mensual con la función **=ABS(PAGO(\$B\$2/12;\$B\$3\*12;\$B\$1))**

**C6** Restamos la cantidad pagada de los intereses y tenemos el capital real que pagamos **=B6-D6**

**D6** Desglose del interés con la función **=ABS(PAGOINT(B2/12;1;B3\*12;B1))**

**E6** El primer mes tenemos acumulado el único pago de capital real **=C6**

**F6** Pendiente nos queda el capital inicial menos el que hemos pagado en el primer pago **=B1-E6**

Bien, ahora hemos de calcular el segundo mes. A partir de ahí, sólo habrá que copiar la fórmula hacia abajo.

	A	B	C	D	E	F
1	Capital	10,000				
2	Interés	24.00%				
3	Período	2				
4						
5	No. de pago	Cantidad	Capital	Interés	Acumulado	Pendiente
6	1	\$528.71	\$328.71	\$200.00	\$328.71	\$9,671.29
7	1	\$528.71	\$335.29	\$193.43	\$664.00	\$9,336.00

Las celdas que cambian en el segundo mes son:

**D7** **=ABS(PAGOINT(\$B\$2/12;1;\$B\$3\*12;F6))** Calculamos el pago sobre el capital pendiente (F6) en vez de sobre el capital inicial como en el primer mes (B1). Convertimos las celdas B2 y B3 en absolutas, ya que copiaremos la función hacia abajo y queremos que se actualice sólo la celda F6 a medida que se copia la fórmula.

**E7** El acumulado del mes será igual al acumulado del mes anterior más el capital del presente mes. **=E6+C7**

**F7** Nos queda pendiente el capital del mes anterior menos el capital que pagamos el presente mes. **=F6-C7**

Ahora sólo nos queda seleccionar toda la segunda fila y copiarla hacia abajo, hasta la fila 29, donde tenemos la fila del último mes de pago.

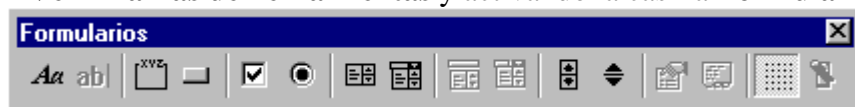
## Resultado completo de la hoja

Observa cómo a medida que vamos pagando religiosamente nuestro préstamo, los intereses se reducen, hasta que el último mes no pagamos prácticamente nada de intereses. Observa las sumas al final de la hoja que nos informan del total de intereses que hemos "pagado": al final del préstamo, hemos pagado 2,689.06 pesos de intereses:

	A	B	C	D	E	F
1	Capital	10,000				
2	Interés	24.00%				
3	Período	2				
4						
5	No. de pago	Cantidad	Capital	Interés	Acumulado	Pendiente
6	1	\$528.71	\$328.71	\$200.00	\$328.71	\$9,671.29
7	1	\$528.71	\$335.29	\$193.43	\$664.00	\$9,336.00
8	1	\$528.71	\$341.99	\$186.72	\$1,005.99	\$8,994.01
9	1	\$528.71	\$348.83	\$179.88	\$1,354.82	\$8,645.18
10	1	\$528.71	\$355.81	\$172.90	\$1,710.63	\$8,289.37
11	1	\$528.71	\$362.92	\$165.79	\$2,073.55	\$7,926.45
12	1	\$528.71	\$370.18	\$158.53	\$2,443.73	\$7,556.27
13	1	\$528.71	\$377.59	\$151.13	\$2,821.32	\$7,178.68
14	1	\$528.71	\$385.14	\$143.57	\$3,206.45	\$6,793.55
15	1	\$528.71	\$392.84	\$135.87	\$3,599.29	\$6,400.71
16	1	\$528.71	\$400.70	\$128.01	\$3,999.99	\$6,000.01
17	1	\$528.71	\$408.71	\$120.00	\$4,408.70	\$5,591.30
18	1	\$528.71	\$416.88	\$111.83	\$4,825.59	\$5,174.41
19	1	\$528.71	\$425.22	\$103.49	\$5,250.81	\$4,749.19
20	1	\$528.71	\$433.73	\$94.98	\$5,684.54	\$4,315.46
21	1	\$528.71	\$442.40	\$86.31	\$6,126.94	\$3,873.06
22	1	\$528.71	\$451.25	\$77.46	\$6,578.19	\$3,421.81
23	1	\$528.71	\$460.27	\$68.44	\$7,038.46	\$2,961.54
24	1	\$528.71	\$469.48	\$59.23	\$7,507.94	\$2,492.06
25	1	\$528.71	\$478.87	\$49.84	\$7,986.81	\$2,013.19
26	1	\$528.71	\$488.45	\$40.26	\$8,475.26	\$1,524.74
27	1	\$528.71	\$498.22	\$30.49	\$8,973.48	\$1,026.52
28	1	\$528.71	\$508.18	\$20.53	\$9,481.66	\$518.34
29	1	\$528.71	\$518.34	\$10.37	\$10,000.00	-\$0.00
30		\$12,689.06	\$10,000.00	\$2,689.06		

## UTILIZACIÓN DE BOTONES DE CONTROL

La utilización de los controles en forma de botón agilizan el manejo de las hojas de cálculo. Antes que nada debemos activar la barra de botones (si no lo está ya). La barra se activa con la opción **Ver - Barras de herramientas** y activando la casilla **Formularios**.



Vamos a diseñar una hoja de cálculo de préstamo para un coche. Supongamos que tenemos la siguiente hoja de cálculo con las fórmulas preparadas.

	A	B
1	Precio	70000
2	Reducción	20%
3	Préstamo	56000
4	Interés	0.00%
5	Años	1
6	Pagos	\$4,666.67

Comentario de las celdas:

**B1:** Aquí introducimos manualmente el precio del coche

- B2:** La reducción puede ser un adelanto en pesos del precio total del coche. Se refleja en porcentaje.
- B3:** Fórmula  $=B1-(B1*B2)$ , es decir, lo que queda del precio menos el adelanto. Ese será el precio.
- B4 y B5:** El interés y el número de años a calcular.
- B6:** Fórmula  $=ABS(PAGO(B4/12;B5*12;B3))$ . Calcula el pago mensual tal y como vimos en la lección anterior.

Esta hoja sería válida y podría calcular los pagos periódicos mensuales. Tan sólo tendríamos que introducir o variar las cantidades del precio, reducción, interés o años. El problema viene cuando en esta misma hoja podemos:

- Introducir cantidades desorbitantes como 1.500.000.000.000.000
- Borrar sin querer alguna celda que contenga fórmulas
- Introducir palabras como "Perro" en celdas numéricas
- Otras cantidades que se nos ocurran

Lo que vamos a hacer es crear la misma hoja, pero de una forma más "amigable", sobre todo para los que no dominan mucho esto del Excel. La hoja será más atractiva a la vista, más cómoda de manejar, y además no nos permitirá introducir barbaridades como las anteriormente expuestas. Para ello utilizaremos los controles de diálogo.


Bien, supongamos que hemos creado una lista de coches con sus correspondientes precios, tal que así:

K	L
Coche	Precio
Chevy	70000
Atos	65000
Ikon	85000
Pointer	95000
Civic	175000

Fíjate que hemos colocado el rango a partir de la columna K. Esto se debe a que cuando tengamos la hoja preparada, este rango "no nos moleste" y no se vea. Este rango de celdas comienza a la misma altura que el anterior, es decir, en la fila 1. Ahora haremos lo siguiente:

1. Selecciona el rango entero (desde K1 hasta L6)
2. Accede al menú **Insertar - Nombre - Crear** y desactiva la casilla **Columna izquierda** del cuadro de diálogo que aparece.
3. Acepta el cuadro de diálogo.

Con esto le damos el nombre **Coche** a la lista de coches y el de **Precio** a la lista de precios. Estos nombres nos servirán más adelante para incluirlos en fórmulas, de forma que no utilicemos rangos como D1:D6, sino el nombre del mismo (Coche). Vamos ahora a crear una barra deslizable que nos servirá para escoger un coche de la lista.

1. Pulsa un clic en el botón  (Cuadro combinado)
2. Traza un rectángulo desde la celda D2 hasta la celda E2
3. Coloca un título en D1: **Coche**

Observa más o menos el resultado hasta ahora:

	A	B	C	D	E
1	Precio	70000		Coche	
2	Reducción	20%		Chevy	
3	Préstamo	56000			
4	Interés	0.00%			
5	Años	1			
6	Pagos	\$4,666.67			
7					

Es muy importante resaltar el hecho de que en este cuadro de diálogo, si pulsamos un clic fuera, al volver a colocar el ratón sobre el mismo, aparecerá una mano para posteriormente utilizarlo. Si queremos editarlo para modificarlo, hemos de pulsar un clic **manteniendo la tecla de Control** del teclado pulsada. Una vez seleccionado, pulsaremos doble clic para acceder a sus propiedades.

- Pulsa doble Clic (manteniendo **Control** pulsada) sobre el cuadro que acabamos de crear y rellena el cuadro de diálogo que aparece con las siguientes opciones:

- Rango de entrada: **Coche**
- Vincular con la celda: **H2**
- Líneas de unión verticales: **8**

¿Qué hemos hecho? En la opción Rango de entrada le estamos diciendo a este cuadro de diálogo que "mire" en el rango que hemos definido como **Coche**, es decir: K2:K6 o lo que es lo mismo, **los precios**. De esta forma, cuando abramos esta lista que estamos creando y escojamos un coche, aparecerá un número en la celda H2. Este número será la posición en la lista que se encuentra el coche que hayamos escogido. Por ejemplo, si desplegamos la lista y escogemos el coche **Ford**, aparecerá en la celda H2 el número 2. Puedes probarlo. Pulsa un clic fuera del cuadro de lista para poder utilizarlo. Cuando salga el dedito, abre la lista y escoge cualquier coche. Su posición en la lista aparecerá en la celda H2. Esta celda servirá como celda de control para hacer otro cálculo más adelante. De igual forma, si escribiéramos un número en la celda H2, el nombre del coche aparecería en la lista desplegable.

### Recuperación del precio de la lista

- Selecciona la celda B2 y escribe: **=INDICE(Precio;H2)**

Observa que en la celda aparece el precio del coche escogido en la lista desplegable. Esto es gracias a la función **=INDICE**. Esta función busca el número que haya en la celda **H2** en el rango **Precio** y nos devuelve el contenido de ese mismo rango. De esta forma sólo encontraremos coches de una lista definida con unos precios fijos. Así no hay posibles equivocaciones.

### Limitación de la reducción para validar valores

Por desgracia aún podemos introducir un porcentaje inadecuado para la reducción del precio.





- Pulsa un clic en la herramienta Control de número y crea un control más o menos como éste:

D	E
Coche	
Ford	
Reducción	

- Con la tecla de control pulsada, haz doble clic sobre el control recién creado para acceder a sus propiedades.
- Rellena las casillas con los siguientes datos:

Valor actual: **20**  
 Valor mínimo: **0**  
 Valor máximo: **20**  
 Incremento: **1**  
 Vincular con la celda: **H3**

- Acepta el cuadro y pulsa **Esc** para quitar la selección del control y poder utilizarlo
- Pulsa sobre las flechas del control recién creado y observa cómo cambia el valor de la celda **H3**
- Sitúate en la celda **B3** y escribe: **=H3/100** Esto convierte en porcentaje el valor de H3

El control se incrementa sólo con números enteros pero es preciso que la reducción se introduzca como un porcentaje. La división entre 100 de la celda H3 permite que el control use números enteros y a nosotros nos permite especificar la reducción como un porcentaje.

### Creación de un control que incremente de cinco en cinco

Si queremos introducir reducciones por ejemplo del 80%, deberíamos ir pulsando la flecha arriba bastantes veces.

- Accede a las propiedades del control recién creado
- Escribe **100** en el cuadro **Valor máximo**, un **5** en el cuadro **Incremento**, y acepta.
- Pulsa **Esc** para desactivar el control

Observa que ahora la celda B3 va cambiando de 5 en 5. Ya puedes probar una amplia variedad de combinaciones de modelos y de porcentajes de reducción.

### Limitación del rédito para validar sus valores

El rédito es el tanto por ciento de la reducción. Nos van a interesar porcentajes que vayan variando de cuarto en cuarto y dentro de un rango del 0% al 20%. Ya que posibilitan porcentajes decimales, vamos a necesitar más pasos que los que precisamos con el pago de la reducción, y es por eso que vamos a usar una barra de desplazamiento en vez de un control como el anterior.

- Crea una **Barra de desplazamiento** más o menos así:

D	E
Coche	
Ford	
Reducción	
Rédito	



- Accede a sus propiedades y modifícalas de la siguiente forma:

Valor mínimo: **0**  
 Valor máximo: **2000**  
 Incremento: **25**  
 Vincular con celda: **H5**

- Acepta el cuadro de diálogo y pulsa **Esc** para quitar la selección
- Selecciona la celda **B4** y escribe en ella: **=H5/10000**
- Con el botón **Aumentar decimales**, auméntala en 2 decimales

Prueba ahora la barra de desplazamiento. La celda B4 divide por 100 para cambiar el número a un porcentaje y por otro 100 para poder aproximar a las centésimas. Ahora sólo nos falta el control para los años.

- Crea un nuevo Control numérico y colócalo más o menos así:

<b>Rédito</b>	
<b>Años</b>	

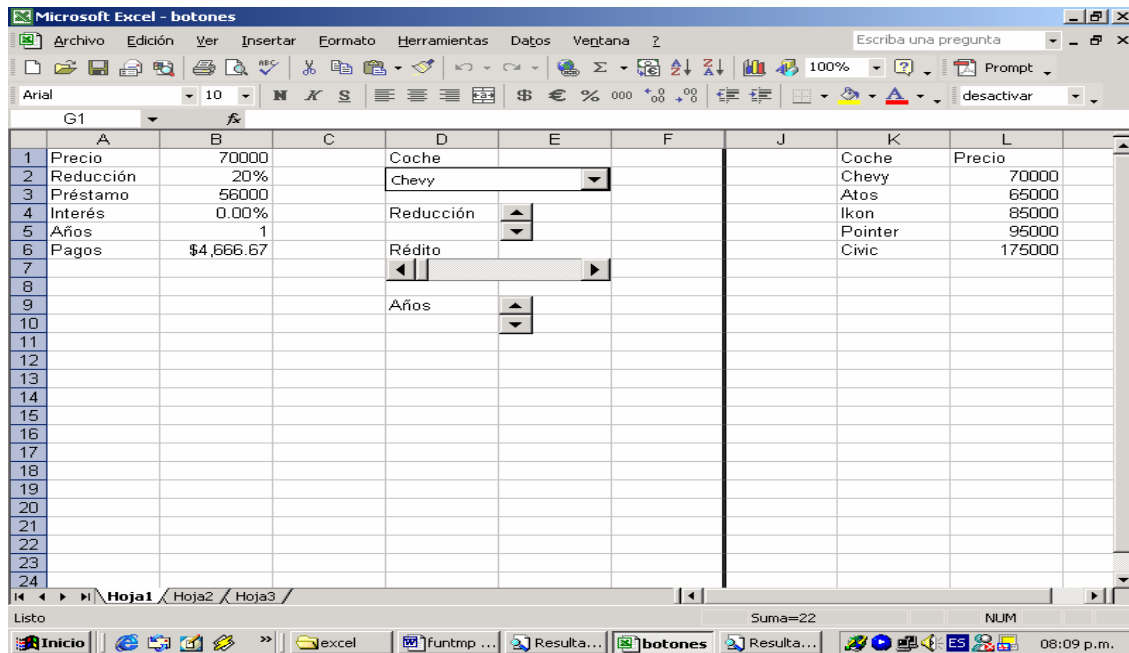
- Accede a sus propiedades y cámbialas de la siguiente forma:

Valor mínimo: **1**  
 Valor máximo: **6**  
 Incremento: **1**  
 Vincular con la celda: **H6**

- Prueba este último control y verifica que los años cambian de uno en uno.

Muy bien, el modelo ya está completo. Ya podemos experimentar con varios modelos sin tener que preocuparnos de que podamos escribir entradas que no sean válidas. De hecho, sin tener que escribir nada en el modelo. Una de las ventajas de una interfaz gráfica de usuario es la posibilidad de reducir las opciones para validar valores. Vamos ahora a darle un último toque:

- Selecciona las columnas desde la **G** hasta la **J** y ocúltalas. El aspecto final será el siguiente:



### Ejemplo de factura

En este ejemplo vamos a crear dos hojas y utilizarlas conjuntamente.

	A	B	C	D
1	Código	Descripción	Unidades	Precio/Unit
2	C-1	Tornillo	1,000.00	\$2.00
3	C-2	Tuerca	2,300.00	\$2.00
4	C-3	Arandela	2,600.00	\$1.00
5	C-4	Desarmador	1,500.00	\$200.00
6	C-5	Tenazas	5,000.00	\$250.00

**Artículos:** esta hoja contendrá un listado de artículos

**Factura:** modelo de factura con fórmulas que buscarán artículos en la anterior hoja

La primera hoja tiene la siguiente lista de artículos:

La segunda hoja tendrá el siguiente aspecto:

	A	B	C	D	E
1	Factura N°				
2	Fecha				
3	Cliente				
4					
5	<b>Código</b>	<b>Descripción</b>	<b>Unidades</b>	<b>Precio/Unit</b>	<b>TOTAL</b>
6					
7					
8					
9					
10					
11				Suma	
12	Condición			Descuento	
13	de pago			IVA	
14					
15				TOTAL	

Evidentemente, cada uno diseñará su formato de factura como mejor le convenga. En este caso, el diseño deja mucho que desear, pero lo importante son las fórmulas que vamos a utilizar. Observa la hoja: las fórmulas las introduciremos en las celdas azules. Escribiremos un código de artículo de la hoja anterior y nos aparecerá su descripción automáticamente en la celda de la derecha. También nos aparecerá el precio unitario. Luego introduciremos la cantidad deseada y Excel nos calculará el total de cada fila, y el total de toda la columna. En la celda E12 puede aparecernos un descuento de la factura sólo si en la celda B13 escribimos la palabra CONTADO. En ese caso, la fórmula de la celda E12 efectuará un 5% de descuento del total de la celda E11.

Veamos:

Celda	Fórmula	Comentario
B6	<b>=BUSCARV(A6;Hoja1!A2:B6;2)</b>	Buscamos el código en la Hoja 1 y nos sale su descripción. Esta función se estudió en la lección 1 del curso.
D6	<b>=BUSCARV(A6;Hoja1!A2:D6;4)</b>	Igual para que aparezca el precio unitario.
E6	<b>=C6*D6</b>	Calculamos el precio del artículo según la cantidad.
E11	<b>=SUMA(E6:E10)</b>	Sumamos la columna de los datos inmediatamente superiores.
B13	<b>(Escribir o no)</b>	Aquí podemos escribir la palabra CONTADO o no (opcional).
E12	<b>=SI(B13="CONTADO";E11*5%;0)</b>	En caso de que en la celda B13 exista la palabra CONTADO, se calcula el 5% de lo que hay en E11. En caso de que en B13 no esté la palabra CONTADO, en la celda actual aparecerá un cero.
E13	<b>=(E11-E12)*16%</b>	Se calcula el 16% de IVA de la diferencia del precio menos el descuento.
E15	<b>=E11-E12+E13</b>	Cálculo del precio final.



## CONFIGURACION Y ATAJO PARA EXCEL

---

### Guardado de archivo en formatos anteriores

Si comparte ficheros con personas que trabajan con otras versiones de Microsoft Excel, en Microsoft Excel 97 puede guardar los libros de trabajo en un formato que permiten ser utilizados en una versión anterior sin restringirle a usted las nuevas características. Cuando esté preparado para salvar su trabajo realice los siguientes pasos:

- En el menú Archivo, elija Guardar Como.
- En la lista Guardar Como Tipo, clic en Libros de Microsoft Excel 97 y 5.0/95 (\*.xls).
- Además puede poner este tipo de formato por defecto de la siguiente forma:
- En el menú Herramientas, elija Opciones.
- En la etiqueta de transición, en el cuadro desplegable de Guardar Como elija Libros de Microsoft Excel 97 y 5.0/95 (\*.xls).

Para las versiones de Excel 2000 y Excel XP, existe una compatibilidad al 100 por ciento, por lo que no es necesario seguir los pasos anteriores.

### Cambiar el desplazamiento del cursor en Excel

Por default, Excel mueve el cursor una celda abajo cuando se pulsa la tecla ENTER. Para cambiar la dirección del desplazamiento selecciona en la barra de menús la opción **"Herramientas" - "Opciones" - "Modificar"**. Una vez ahí cambiar la opción **"Mover la selección después de Entrar"** una vez hecho esto, seleccione la dirección que desea que se siga después de haber oprimido enter en la hoja de cálculo y pulsar el botón **"Aceptar"** para guardar la nueva configuración.



### Añadir una imagen de fondo a una hoja de cálculo

Para añadir una imagen de fondo, seleccione **"Formato"** - **"Hoja"** - **"Fondo"** y después buscar la imagen y pulsar en el botón "Aceptar".

### Convertir hojas de cálculo en páginas web

Para convertir una hoja de cálculo en una página web, selecciona de la barra de menús, **"Archivo"** - **"Guardar como"** y en la opción guardar como tipo elegir **"Como página web"**. Así podrás convertir rangos en tablas HTML y gráficos en archivos gif.

### Capturar una pantalla fácilmente

Para capturar una pantalla simplemente presiona la tecla "ImpPant" cuando estés viendo la imagen que deseas capturar. Una vez capturada la imagen, puedes ir a cualquier programa, ya sea Word, PowerPoint, Excel, etc., y selecciona **"Edición"** – **"Pegar"**, si escoges esta opción, la incrustación del objeto es la que se escoge por omisión, aunque se puede escoger la opción de **"Pegado Especial"** y aquí es donde se puede escoger el tipo de vínculo que se va a pegar, según sea el tipo de archivo, como objeto, como meta archivo, como gráfico independiente del dispositivo, entre varios tipos de archivo pegados especiales.

### Crear etiquetas de envío con Microsoft Access

En Microsoft Access, Selecciona la pestaña de **"Informes"**, pulsa en **"Nuevo"** y selecciona la opción **"Asistente para etiquetas"**. Selecciona el tipo de papel y los campos que deben aparecer en tus etiquetas y haz una vista preliminar para ver cómo quedan antes de imprimirlas.



### **Abrir ficheros y carpetas con sólo un clic**

Lo único que debes hacer para abrir los ficheros y carpetas con un solo clic, lo que se requiere es configurar Windows, para ello, selecciona en la barra de menús en cualquier ventana del explorador de Windows, "Ver" – "como estilo Web" (para una mayor configuración de Windows, vea la sección de Windows en el Cuaderno Pedagógico 1 PC y WINDOWS)

### **Modificar rápidamente celdas en Microsoft Excel**

Para modificar celdas de Excel fácilmente, sin tener que hacer doble clic ni ir a la barra de edición, sitúate en la celda a modificar y pulsa la tecla F2.

### **Abrir varios archivos simultáneamente**

Marca los archivos que deseas abrir, para ello mantén pulsada la tecla [CTRL] mientras haces clic en el botón izquierdo del ratón y a continuación presiona la tecla [INTRO] o haz clic en el botón "Aceptar".

### **Mover objetos menos espacio en Microsoft Office**

En cualquier aplicación de Microsoft Office, si deseas mover los objetos, ya sean dibujos, imágenes o cuadros de texto, o menos espacio para conseguir una alineación perfecta, márcalos y mantén pulsada la tecla CTRL mientras los mueves con los cursores de desplazamiento.

### **Seleccionar varios objetos en Microsoft Office**

Para seleccionar varios objetos insertados en cualquier aplicación de Microsoft Office y poder moverlos simultáneamente, presiona la tecla [SHIFT] y sin dejar de presionar esta tecla, haz clic en el botón izquierdo del ratón a todos los objetos que quieras seleccionar.





## Repetir lo mismo en varias hojas

Para escribir un mismo texto o fórmula en varias hojas, para no tener que copiar y pegar, pulsa la tecla **[Ctrl]** sin dejar de presionarla, haz clic en el botón izquierdo del ratón en todas las pestañas de las hojas en donde lo quieres copiar (por ejemplo Hoja 1 y Hoja 2), de esta forma se van seleccionando todas las hojas.

Ahora escribe lo que quieras en la celda A1, por ejemplo.

Pásate a las demás hojas y verás como se ha escrito lo mismo en todas las celdas.

## Insertar filas o columnas rápidamente

En cualquier hoja de Excel que ya tengas con datos, vete al principio de lo que tengas escrito. Pon el ratón sobre el control de relleno (es el puntito negro que tiene el cursor abajo a la derecha). Sabes que estás encima del control de relleno, porque la cruz blanca que es normalmente el ratón, se cambia a color negro.

Ahora pulsa la tecla de **[Shift]** (no la de bloq. mayús) y sin dejar de presionarla haz clic en el botón izquierdo del ratón y sin dejar de presionar, arrástralo hacia abajo o hacia la derecha, verás que se insertan filas o columnas.

## Activar la copia de seguridad de Excel

La opción que autoguarda nuestra hoja en Excel está algo escondida para activarla, selecciona en la barra de menú **“Herramientas” – “Complementos”** aquí tienes una casilla que dice Autoguardar, actívala y ya está, lo que no sé es cómo especificar cada cuanto tiempo quiero que se guarde y presiona el botón de aceptar.

Una vez seleccionada esta opción, en la barra de menú, selecciona la opción de **“Herramientas” – “Autoguardar”**, después aparecerá una ventana de diálogo, selecciona las opciones que requieras, así como el tiempo de guardado automático y después presiona el botón aceptar.



### **Escritura de varias líneas en la misma celda**

Muchas veces se requiere que tenga dos o más líneas en la misma celda, para ello, escribe lo que desees y cuando quieras pasar al segundo renglón en la misma celda presiona la tecla **[Alt]** y sin dejar de presionar esta tecla presiona la tecla **[Enter]** (**[Alt]**+**[Enter]**).

### **Ver las fórmulas en la celda en vez del resultado**

Para ver las fórmulas en vez del resultado en la hoja, lo puedes hacer seleccionando de la barra de menú “**Herramientas**” – “**Opciones**” – “**Ver**” y a continuación aparecerá una ventana, en ella activa la casilla y selecciona “Fórmulas”.

Para volver a ver los datos puedes hacer los mismos pasos pero desactivando la casilla “Fórmulas”.

### **Añadir una serie a un gráfico que ya está hecho**

Para añadir una serie más a un gráfico que ya tienes creado, puedes seleccionarlo en la hoja los datos del nuevo rango, luego le das a copiar, seleccionas el gráfico con el botón derecho y le das a pegar.

### **Hacer que una macro se ejecute cuando se abre un libro**

Para que esto sea posible, crea una nueva macro y dale el nombre `auto_open()`.

Para que se autoejecute una macro cuando salgas de un libro, dale el nombre `auto_close`.



## Proteger las celdas y esconder las fórmulas

Puedes proteger las celdas de tus hojas de cálculo de forma que no puedan ser modificadas, así como ocultar las fórmulas escritas para que nadie las vea, para ello realiza los siguientes pasos:

Selecciona la celda (o celdas) que quieras con el botón derecho, posíciónate en la que seleccionaste y haz clic en el botón derecho del ratón para obtener el menú contextual y selecciona **“Formato de celdas”**.

Haz clic en la pestaña de proteger, si activas la casilla Bloqueada harás que el contenido de la celda no pueda ser modificado (como podrás ver, esta casilla está siempre activada, o sea que como mucho puedes desactivarla). Si activas la casilla Oculta, harás que las fórmulas no se puedan ver. Activa las casillas que quieras y dale a aceptar.

Para que funcione tienes que proteger la hoja: selecciona en la barra de menú **“Herramientas” – “Proteger” – “Proteger hoja”**.

Si quieres puedes escribir una contraseña para que nadie pueda desproteger la hoja.

Pulsa el botón de aceptar.

## Ponerle una contraseña al libro

Para ponerle una contraseña a tu libro y que nadie pueda verlo ni modificarlo selecciona en la barra de menú **“Archivo” – “Guardar como”**, luego pulsa el botón opciones y escribe la contraseña que quieras. Cuando pulses aceptar, te pedirá que repitas la contraseña para ver que no te has equivocado la primera vez.



## **Aplicar distintos formatos a una frase que está en la misma celda**

Primero escribir la frase normal.

Luego poner el cursor encima de la frase (La frase aparece entonces arriba, en la barra de fórmulas).

Selecciona la palabra o palabras a las que quieres cambiar el formato, esto lo puedes hacer con toda la frase o por palabras).

Si no tienes la barra de fórmulas arriba también puedes pulsar F2 y trabajar directamente en la hoja.

## **Añadir un comentario en una celda**

Selecciona la celda deseada.

Haz clic con el botón derecho de ratón.

Selecciona Insertar comentario.

Después aparecerá un nuevo cuadro, escribe lo que quieras en él.

Las celdas que tienen un comentario aparecen con un triangulito rojo en la esquina. Cuando posicionas el cursor sobre la celda durante un segundo, aparece automáticamente el comentario.

Para quitarlo dale otra vez con el botón derecho a la celda y selecciona Eliminar comentario.

Para modificar un comentario ya escrito le das con el botón derecho a la celda y seleccionas Modificar comentario.

## **Captura de fechas**

Cuando se teclean sólo dos dígitos del año Microsoft Excel asume lo siguiente:

Si la fecha tecleada está en el rango 1/1/00 - 31/12/29, Microsoft Excel asume que la fecha es del año 2000. Si quiere introducir en una celda la fecha 17/5/23, debería teclear los cuatros dígitos del año, por ejemplo 17/5/1923, si no lo hace así Excel asumirá que es 17/5/2023.



Si la fecha tecleada está en el rango 1/1/30 - 31/12/99, Microsoft Excel asume que la fecha pertenece a los años 1900. Si quiere introducir la fecha 6/9/57, pero refiriéndose al año 2057, deberá introducir los cuatro dígitos del año, esto es 6/9/2057 en otro caso Microsoft Excel asumirá 6/9/1957.

### **Atajo para introducir la fecha y hora actual rápidamente**

En Microsoft Excel 97, se pueden actualizar rápidamente los datos que son devueltos por Microsoft Query. Pulsando F9 refresca la consulta en la ventana actual. Se puede utilizar esta tecla en lugar de hacer clic en la opción refrescar datos del menú.

Microsoft Excel 97 permite utilizar una combinación de teclas para insertar rápidamente la fecha y hora actual en una celda.

Para hacerlo, utilice la siguiente combinación de teclas:

Fecha actual: **[Ctrl]+[+]**;

Hora Actual: **[Ctrl]+[Shift]+[+]**;

La fecha y la hora se basan en el reloj interno del ordenador.

### **Copia y pegado de texto en un solo paso**

Para copiar los datos de la celda superior a la celda activa y pegarlos en la misma celda activa. Utilice la siguiente combinación de teclas:

**[Ctrl]+[Shift]+[,]**

Para copiar los datos de la celda superior a la celda activa, utilice el comando Pegado Especial para pegar sólo valores en la celda activa, este comando es la combinación de teclas **[Ctrl]+[Shift]+[‘]**

Si desea obtener un listado más completo de las teclas de acceso rápido, haga clic en el asistente de Office, y en el cuadro de diálogo del asistente, teclear teclas rápidas y haga clic en Buscar.



## Movimiento rápido en una hoja de cálculo

Puede utilizar **[Ctrl]+[Av Pág]** para moverse a las hojas a la derecha y **[Ctrl]+ [Re Pág]** para moverse a las hojas a la izquierda.

## Cambio del salto de página en una hoja de cálculo

Si quiere cambiar el salto de página mientras está previsualizando su archivo, utilice la Vista Previa de Salto de Página. Con esta característica puede ver el documento y mover el salto de página con un simple clic sobre el salto de página y arrastrándolo hasta la posición deseada. Para utilizar esta característica siga los pasos que a continuación se detallan:

Haga clic en la barra de menú en **“Ver” – “Vista previa del salto de página”**.

Haga clic en el salto de página para seleccionar y arrastrar a la nueva posición.

## Vinculación de cuadros de texto a una celda

Para vincular los cuadros de texto a una celda, realice los siguientes pasos:

- Dibuje un cuadro de texto en su hoja de cálculo.
- Compruebe que el cuadro de texto es el objeto seleccionado en su hoja de cálculo. Para hacer esto haga clic sobre el cuadro de texto una vez.
- Pulse **[F2]**. Esto situará el cursor en la barra de fórmulas.
- Teclear **=<referencia a la celda>**, donde la **<referencia a la celda>** es la celda a la que quiere vincular el cuadro de texto, por ejemplo **A1**.

## Cambio de mayúsculas a minúsculas en una cadena de texto

Si desea cambiar una celda a mayúsculas, no se puede hacer en la misma celda, lo que se puede hacer es utilizar una de las siguientes funciones y el resultado guardarlo en otra celda, las funciones son las siguientes:

- Para cambiar una cadena entera a minúsculas utilizar la función **MINUSC**.
- Para cambiar una cadena entera a mayúsculas utilizar la función **MAYUSC**.



- Para cambiar la primera letra de cada palabra a mayúsculas utilizar la función **NOMPROPIO**.

Por ejemplo para cambiar el nombre de una persona, el cual está en minúsculas y se va a cambiar a mayúsculas sólo la primera letra de cada nombre y apellido, este nombre se encuentra en la celda A1, para ello, siga estos 2 pasos:

En la celda A1, teclee su nombre y apellido todo en minúsculas.

En la celda B1, teclee =NOMPROPIO(A1).

### **Copia del formato de una celda a otra**

Una manera rápida y fácil de copiar y pegar formatos de celdas es la siguiente:

Seleccionar una celda que contenga los formatos que usted desee seleccionar.

- Hacer clic en el botón Copiar formato, con lo que el cursor se convierte en una brocha.
- Seleccionar las celdas a las que quiere pegar el formato. Cuando suelte el botón del ratón, se aplicará el formato a la selección.

### **Formato de celdas en las tablas dinámicas**

Se puede dar formato a las celdas de una tabla dinámica. Este formato se guardará cuando se actualice dicha tabla. Antes de comenzar a dar formato, asegúrese de que la opción Habilitar Selección del menú Seleccionar del menú de tablas dinámicas está activada. En caso de no estarlo, actívela.

Si aplica formatos condicionales a las celdas de tablas dinámicas, este formato no se mantendrá después de actualizar la tabla dinámica.

### Resultado de valores negativos de una fórmula

En ocasiones resulta muy útil poder ver rápidamente cuando los resultados de una fórmula son negativos. Para resaltar las celdas que tengan valores negativos, por ejemplo, que aparezcan de color rojo, debe dar un formato condicional a estas celdas. Para ello, seleccione las celdas en que desee aplicar el formato, y en la barra de menú selecciona **“Formato” – “Formato Condicional” – “Valor de la celda”**, seleccione la frase de comparación **“menor que”**, introduzca el valor cero, haga clic en el botón izquierdo del ratón en **“Formato”** y seleccione el color rojo.

### Cambio de color de las líneas de división

Para cambiar el color de las líneas de división de una hoja, selecciónela en la barra de menú **“Herramientas” – “Opciones” – “Ver”** seleccione el color que desee. Si no desea ver las líneas de división, desmarque en esta misma ficha la casilla **“Líneas de División”**.

### Apertura automática de un libro al iniciar Excel

En el Explorador de Windows, mueva el icono del libro que desee abrir a la carpeta Iniciar que se encuentra en la carpeta:

- C:\Archivos de Programa\Microsoft Office\Office
- Si no existe, cree una carpeta que tenga el nombre de Iniciar.

Si no desea mover el libro de su ubicación actual, puede crear un acceso directo al libro utilizando el comando Crear acceso directo en el menú Archivo en el Explorador de Windows. A continuación, mueva el acceso directo a la carpeta Iniciar.

### Cambiar la fuente en el texto del encabezado y del pie de página

Puede personalizar el texto incluido en un encabezado o pie de página cambiando la fuente utilizada, su estilo, tamaño o subrayado. No puede cambiar el color del texto.





Para personalizarlo, siga estos pasos:

- En la barra de menú seleccione **“Ver” – “Encabezado y pie de página” – “Mostrar”**.
- Haga clic en el botón izquierdo del ratón y seleccione el botón **“Personalizar encabezado”** o **“Personalizar pie de página”**.
- Seleccione el texto en las casillas Sección izquierda, Sección central o Sección derecha y haga clic en el botón **“Fuente”**.
- Seleccione las opciones de fuente, tamaño, estilo y subrayado que desea aplicar.

### **Cambiar el número de página de la primera página**

Para personalizar la numeración de las páginas, eligiendo el número de página por el que la primera página empiece a numerar, siga estos pasos:

- En el barra de menú seleccione **“Archivo” – “Configurar página” – “Página”**.
- En el cuadro de texto **“Primer número de página”**, escriba el número que debe aparecer en la primera página de la hoja de cálculo.

### **Corrección automática de datos y palabras**

Cuando estamos introduciendo información en las celdas de Excel, podemos tener errores, por ejemplo, si escribimos **“camion”** que es incorrecto, se debe reemplazar automáticamente por **“camión”**, esta opción existe en Excel y se llama corrección automática.

Para tener acceso a esta característica, debe seleccionar la opción de la barra de menús **“Herramientas” – “Autocorrección”**. Esta acción mostrará una ventana de diálogo que se llama Autocorrección.

El punto de inserción se colocará en el cuadro Reemplazar.



Sólo tiene que escribir (como siempre) la palabra incorrecta en esta sección. Por ejemplo, puede escribir "**camion**" (sin las comillas) en este cuadro. A continuación, presione la tecla [Tab] para ir al cuadro Con y, en este cuadro, escriba "camión". En cuanto escriba datos en el cuadro Con, observará que el botón "**Agregar**" está disponible. Haga clic en este botón. Ya está. Puede agregar todos los errores más habituales para que se corrijan automáticamente. Si comete un error al escribir los datos en el cuadro Reemplazar o Con, seleccione la entrada en la lista y haga clic en el botón izquierdo del ratón en Eliminar. Cuando termine de realizar cambios en el cuadro de diálogo Autocorrección, haga clic en el botón de "**Aceptar**" para salir.

### **Acceso rápido a las opciones de barras de herramientas**

Si desea agregar o quitar determinadas barras de herramientas de la sesión de trabajo en Excel, mostrar en pantalla las opciones elegidas, para ello coloca el puntero sobre cualquier barra de herramientas abierta y haga clic en el botón derecho del ratón, a continuación aparecerá un menú contextual, ahí podrá activar (o desactivar, si ya estaba activada) cualquier opción de barra de herramientas.

### **Mover o copiar el contenido de una celda**

Al pasar el puntero del ratón sobre la esquina inferior derecha de una celda, observará que el puntero cambia de forma. Cuando el puntero cambie a una flecha en blanco, puede hacer clic en el botón izquierdo del ratón y sin dejar de presionar el botón izquierdo, arrastra y coloca el contenido de la celda o rango de celdas en otro lugar. Si el puntero cambia a un signo más en negrita (+), puede copiar la información, ya sea de un lado a otro, o de arriba a abajo y viceversa; para ello, arrastre el puntero sobre el número de celda al que desee copiar el contenido y, a continuación, suelte el botón del ratón.



## Ordenamiento de información en tablas dinámicas

Cuando necesitamos ordenar datos para poder determinar el número de veces que un elemento específico aparecía en una columna, se realizan los siguientes pasos:

Supongamos que tiene una columna de nombres de personas (título Nombres), que han respondido Sí o No a una pregunta y que los datos de las respuestas se encuentran en una segunda columna (título respuestas). Resulta que desea saber cuántas personas respondieron Sí y cuántas No. Antes de contar manualmente la información, debe saber que Excel cuenta con una característica pensada para usted: la Tabla dinámica. Sólo tiene que hacer clic en la barra de menú en “Datos” – “Asistente para tablas dinámicas” y permitir que Excel le guíe por los cuatro cuadros de diálogo de instalación de Tabla dinámica. Cuando llegue al tercer cuadro de diálogo, asegúrese de que escribe el número de respuestas en el campo de datos. Cuando termine, los resultados se mostrarán de manera similar a los siguientes, excepto que los datos tendrán formato de tabla.

Número de RESPUESTAS

-----

Total de RESPUESTAS

-----

No	3
----	---

Sí	5
----	---

-----

Total general	8
---------------	---

## Agregar comandos a menús

Excel permite agregar fácilmente comandos a los menús. Primero, seleccione la barra de herramientas que contiene el menú al que desea agregar un comando. A continuación, haga clic en la barra de menú “**Herramientas**” - “**Personalizar**” – “**Comandos**”. Después, en



el cuadro de diálogo Categorías, haga clic en la categoría del comando. Ahora arrastre el comando que desea desde el cuadro Comandos sobre el menú de la barra de herramientas. Cuando el menú muestre una lista de comandos, elija el lugar del menú de la barra de herramientas en el que desea que aparezca el comando y, a continuación, suelte el botón del ratón.

### **Quitar saltos de página en hojas de cálculo**

Los saltos de página pueden resultar muy útiles para organizar los datos, pero puede ser algo complicado quitarlos cuando no se sabe cómo, pero existe una manera sencilla de eliminarlos.

Para quitar un salto de página manual horizontal o vertical, comience haciendo clic con el botón derecho del ratón en una celda debajo del salto de página horizontal, o a la derecha del salto de página vertical, cuando aparezca el menú contextual, seleccione Quitar salto de página. Si desea quitar todos los saltos de página manuales, haga clic con el botón secundario en cualquier celda de la hoja de cálculo y, a continuación, haga clic en Restablecer todos los saltos de página en el menú contextual. También puede quitar saltos de página en la vista previa de saltos de página; para ello, arrastre el salto de página fuera del área de impresión.

### **Cómo mostrar rápidamente tablas de datos en gráficos**

Excel 97 le permite ahora mostrar tablas de datos en gráficos. Primero, utilice un gráfico existente o cree uno mediante el Asistente para gráficos. Una vez creado, haga clic en el gráfico y seleccione el menú Gráfico en el menú principal. A continuación, haga clic en el comando Opciones de gráfico. Cuando se muestre el cuadro de diálogo Opciones de gráfico, haga clic en el botón izquierdo del ratón en la pestaña Tabla de datos. Finalmente, active la casilla de verificación Mostrar tabla de datos y haga clic en Aceptar. Ahora la tabla de datos aparecerá debajo del gráfico.



## Creación de una plantilla para libros en Excel

Cree una plantilla llamada LIBRO.XLT con formatos, tipos de letra, bordes, márgenes por defecto etc. y guárdela en el subdirectorío INICIAR del subdirectorío donde instaló Excel. Por defecto, Microsoft Excel utilizará este documento cuando cree un nuevo libro.

## Ajuste de las impresiones de Excel 97 al ancho de página

Cuando imprime una hoja de cálculo, ¿desea que se ajuste al ancho del papel y que utilice tantas hojas de papel como sea necesario para imprimir todos los datos? Cuando haga clic en el botón izquierdo del ratón en la barra de menú en **“Archivo” – “Configurar página”** haga clic en la ficha **“Página”** en **“Escala”**, haga clic en **“Ajustar a”** y seleccione **“1 página de ancho”**. En el segundo cuadro **“Ajustar a”** que indica la altura deseada de los datos, elimine el número para que el cuadro quede vacío.

## Cálculo de días entre dos fechas

Para calcular el número de días entre dos fechas, sólo hay que restar las dos fechas. Por ejemplo, si la celda A1 contiene la fecha 6/8/2000 y la A2 contiene 6/20/2000, la fórmula =A2-A1 calcula el número de días entre estas fechas (12).

## Funciones Escondidas en Excel

### Excel 2000

1. Abre un nuevo cuaderno en Excel 2000
2. Pulse [F5]
3. Escriba X2000:L2000 en el campo de "Referencia" y pulse Aceptar.
4. Pulse Tab.
5. Pulse [Ctrl] + [Shift] + [Mayús] y sin soltarlas pulse el icono del asistente de Gráficos Chart Wizard) en la barra de herramientas:



6. Se sorprenderá de lo que verá.



## Excel 97

1. Abrir un nuevo libro de trabajo y presionar F5.
2. Escribir "X97:L97" y pulsar Enter.
3. Pulsar Tab y después, manteniendo presionadas [Ctrl] + [Shift] pulsar el icono del Asistente para gráficos.