ENUNCIADOS EJERCICIOS
CURSO MACROS Y PROGRAMACION VBA EXCEL

01.- GRABADORA DE MACROS

Ejercicio 01.01.-

Tenemos una **Tabla de Datos de Ventas** de **varios artículos**, con **Precio Unitario** y **Cantidades** y cuyo **Producto** nos da **SubTtotal**. Debajo aparece la **Sumatoria**, el **Impuesto (7%)** y, finalmente, el **Total del Importe de las Ventas**. Todo ello lo crearemos en **Hoja1 del Libro de Trabajo**. A continuación, copiaremos **Hoja1**, **con el nombre Hoja2**.

	A	B	C	D	E
1		Precio Unitario	Cantidad	SubTotal	
2	Camisetas				
3	Pantalones				
4	Zapatos				
5	Chaquetas				
6			Sumatoria		
7			IGIC (7%)		
8			TOTAL		

La Tabla no tiene Formato (tal y como se aprecia encima), **ni Fórmulas**. El **Ejercicio Propuesto** consiste en hacer una **Macro llamada Formato que cambie el aspecto de la Tabla** de **Hoja2**, quedando tal y como se ve debajo. Finalmente, guardaremos el Archivo como ***EJERCICIO0101.xlsm***.

	A	B	C	D	E
1		**Precio Unitario**	**Cantidad**	**SubTotal**	
2	**Camisetas**				
3	**Pantalones**				
4	**Zapatos**				
5	**Chaquetas**				
6			Sumatoria		
7			IGIC (7%)		
8			**TOTAL**		

NOTA. A lo largo del Curso y de los Ejercicios, se usa gran cantidad de **Constantes** en los diferente **Métodos de los Objetos de VBA Excel** con los que trabajamos. En el siguiente enlace puedes ver **las más usadas**, con su **Nombre**, **Valor** y **Descripción**.

https://learn.microsoft.com/es-es/office/vba/api/excel.constants

Ejercicio 01.02.-

Tomamos el ***EJERCICIO0101.xlsm*** y copiamos **Hoja2** (después de haber aplicado la **Macro Formato**) con el **nombre Hoja3**. Introduciremos los **Valores** de **Precio Unitario** y **Cantidad**, tal y como se aprecia debajo.

	A	B	C	D	E
1		**Precio Unitario**	**Cantidad**	**SubTotal**	
2	**Camisetas**	15,99 €	4,00		
3	**Pantalones**	25,95 €	3,00		
4	**Zapatos**	85,75 €	2,00		
5	**Chaquetas**	178,25 €	1,00		
6			Sumatoria		
7			IGIC (7%)		
8			**TOTAL**		

El **Ejercicio Propuesto** consiste en hacer varias **Macros** y guardar el Archivo como ***EJERCICIO0102.xlsm***.

1. Macro que Calcule el **SubTotal** por cada Artículo (Activar **Referencias Relativas**).
2. Macro que Calcule la **Sumatoria**.
3. Macro que Calcule el **Impuesto**.
4. Macro que Calcule el **Total**.

Una Vez aplicada cada Macro obtendremos lo siguiente.

	A	B	C	D	E
1		**Precio Unitario**	**Cantidad**	**SubTotal**	
2	**Camisetas**	15,99 €	4,00	63,96 €	
3	**Pantalones**	25,95 €	3,00	77,85 €	
4	**Zapatos**	85,75 €	2,00	171,50 €	
5	**Chaquetas**	178,25 €	1,00	178,25 €	
6			Sumatoria	491,56 €	
7			IGIC (7%)	34,41 €	
8			**TOTAL**	525,97 €	

Ejercicio 01.03.-
Partimos ahora del ***EJERCICIO0102.xlsm*** y el **Ejercicio Propuesto** consiste en crear la **Macro AutoAjustarFilaColumna** que se encarga de **Autoajustar el Alto de Fila** y el **Ancho de Columna del Rango Seleccionado** al contenido del mismo.

Entraremos en la **Hoja3** y aplicaremos un aspecto parecido al siguiente, donde **estrechamos el Ancho** de las Columnas y **reducimos el Alto** de las Filas para **comprobar el funcionamiento de la Nueva Macro**.

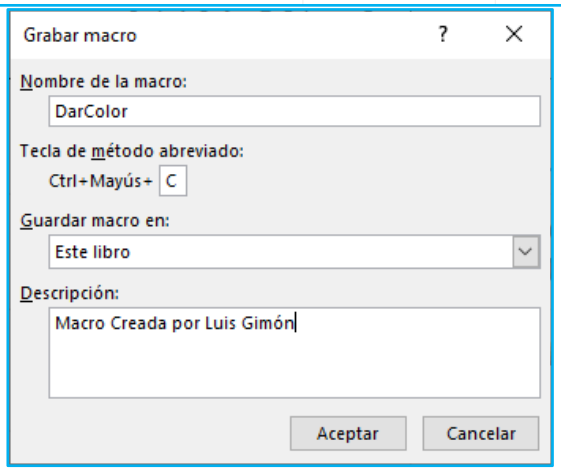
	A	B	C	D	E
1		Precio Unitar	antida	SubTotal	
2	Camis	15,99 €	4,00	63,96 €	
3	Pantal	25,95 €	3,00	77,85 €	
4	Zapato	85,75 €	2,00	171,50 €	
5	Chaqu	178,25 €	1,00	178,25 €	
6			Sumatoria	491,56 €	
7			IGIC (7%)	34,41 €	
8			TOTAL	#######	
9					

Una vez finalizado guardamos el Archivo con el **Nombre *EJERCICIO0103.xlsm***.

Ejercicio 01.04.-
Partimos ahora del ***EJERCICIO0103.xlsm*** y copiamos la **Hoja3** para crear la **Hoja4**.

El **Ejercicio Propuesto** consiste en crear la **Macro DarColor** que se encarga de dejar la Tabla con un aspecto parecido al que te muestro debajo. Esta vez, crearemos la Macro añadiendo la **Combinación de Teclas Ctrl+Mayús+C**.

	A	B	C	D	E
1		**Precio Unitario**	**Cantidad**	**SubTotal**	
2	**Camisetas**	15,99 €	4,00	63,96 €	
3	**Pantalones**	25,95 €	3,00	77,85 €	
4	**Zapatos**	85,75 €	2,00	171,50 €	
5	**Chaquetas**	178,25 €	1,00	178,25 €	
6			Sumatoria	491,56 €	
7			IGIC (7%)	34,41 €	
8			**TOTAL**	525,97 €	
9					

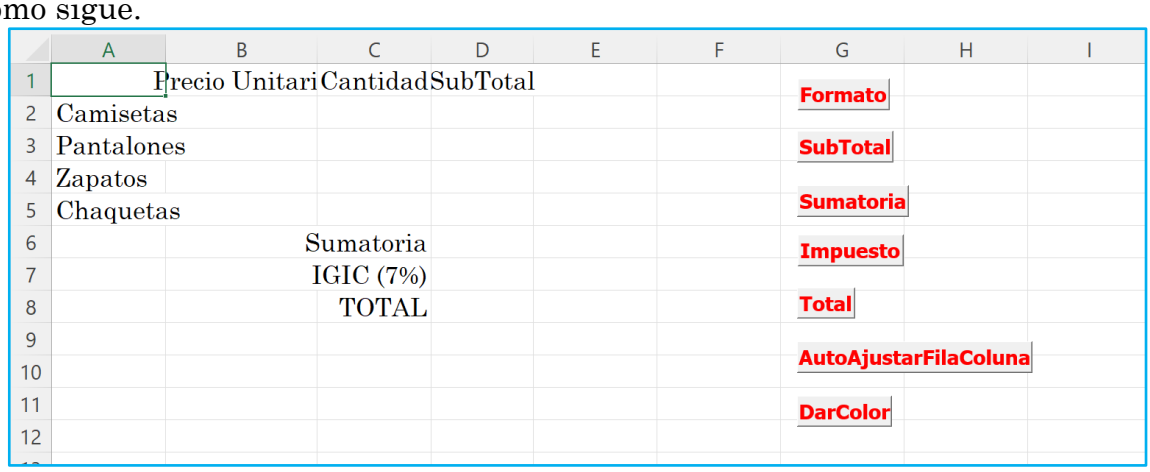


Ejercicio 01.05.-
Partimos ahora del ***EJERCICIO0104.xlsm*** y copiamos la **Hoja1** para crear la **Hoja5 al final**.

El **Ejercicio Propuesto** consiste en crear **Varios Botones de Formulario** a los que asignaremos las Macros que ya tenemos creadas.

1. **Formato**
2. **SubTotal**
3. **Sumatoria**
4. **Impuesto**
5. **Total**
6. **AutoAjustarFilaColumna**
7. **DarColor**

Aplicaremos cambios en el Formato de Cada Control para que el aspecto Final sea como sigue.



02.- CELDAS\RANGOS

Ejercicio 02.01.-

Crearemos un Archivo vacío llamado ***EJERCICIO0201.xlsm***. El **Ejercicio Propuesto** consiste en hacer el Código VBA Excel que genere la **Tabla de Datos de Ventas** que aparece debajo, dentro del **Módulo *TABLA_DATOS***.

	A	B	C	D	E
1		**Precio Unitario**	**Cantidad**	**SubTotal**	
2	**Camisetas**	15,99 €	4,00	63,96 €	
3	**Pantalones**	25,95 €	3,00	77,85 €	
4	**Zapatos**	85,75 €	2,00	171,50 €	
5	**Chaquetas**	178,25 €	1,00	178,25 €	
6			Sumatoria	491,56 €	
7			IGIC (7%)	34,41 €	
8			**TOTAL**	525,97 €	
9					

Crearemos los siguientes **Procedimientos** usando la **Jerarquía Completa para definir cada una de las Instrucciones** que usemos. Utiliza **Range** y **Cells**, indistintamente, cuando lo necesites.

1. **CrearArticulos**
2. **CrearEncabezados**
3. **CrearTotal**
4. **CrearFormatoNumerico**
5. **EjecutarCalculos**
6. **DarColorAjustesFilasColumnas**
7. **Limpiar**

Finalmente, insertaremos una serie de **Botones de Formulario** que, al pulsarlo, ejecute cada Procedimiento.

A continuación, te añado los enlaces de **Microsoft** donde leer cómo **Alinear en Horizontal\Vertical un Rango en VBA Excel**.
https://learn.microsoft.com/es-es/office/vba/api/excel.cellformat.horizontalalignment
https://learn.microsoft.com/es-es/office/vba/api/excel.xlhalign
https://learn.microsoft.com/es-es/office/vba/api/excel.cellformat.verticalalignment
https://learn.microsoft.com/es-es/office/vba/api/excel.xlvalign

En VBA Excel, el **Objeto Font** contiene los **Atributos de Fuente** (nombre, tamaño, color, etc.) del (https://learn.microsoft.com/es-es/office/vba/api/excel.font(object)).

En VBA Excel el **Método *Range.ClearContents* borra** las **Fórmulas** y los **Valores del Rango**, dejando el **Formato intacto** (https://learn.microsoft.com/es-es/office/vba/api/excel.range.clearcontents).

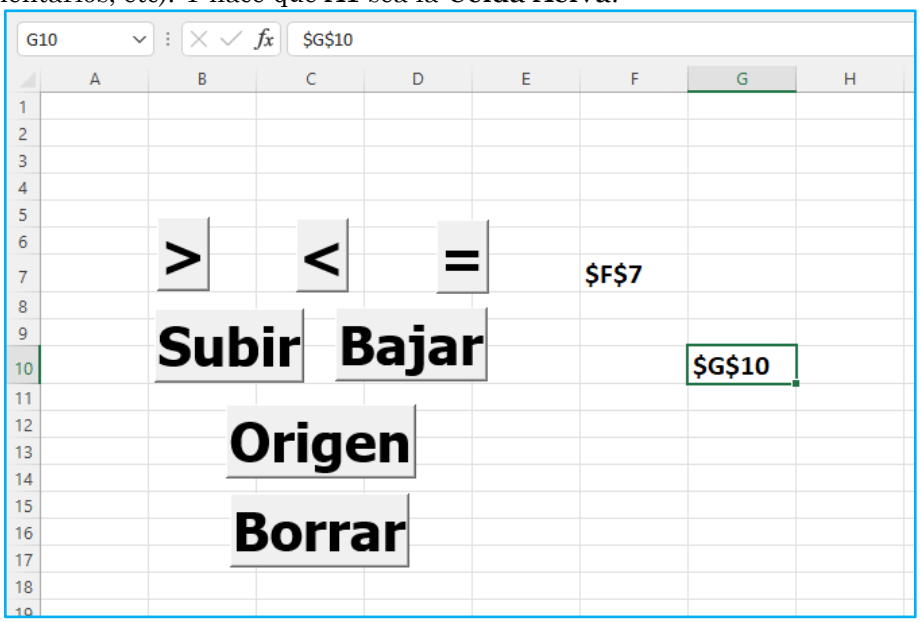
NOTA. Deberemos pulsar los Botones de Formulario en el Orden numerado para obtener el resultado esperado. Obviamente, se podría crear un Botón que ejecutara todos los Procedimientos automáticamente, una tras otro.

NOTA. Aprovecha las Macros de Ejemplos anteriores para Copiar y adaptar el Código VBA Excel.

Ejercicio 02.02.-

Crearemos un Archivo vacío llamado ***EJERCICIO0202.xlsm***. El **Ejercicio Propuesto** consiste en crear varios **Botones de Formulario** con diferentes acciones, cuyo Código VBA Excel irá dentro de la **Hoja1**.

1. Permite **desplazar** la **Celda Activa**, de una en una, en la **dirección** que se señale.
2. **Escribe** en la **Celda Activa** su Propia **Dirección**, usando **Fuente de Letra "Tahoma"**, **Negrita** y **Tamaño de Fuente de Letra = 16**.
3. La **Celda Activa** pasa a ser la **A1**.
4. **Borra completamente todas las Celdas** de la Hoja (incluyendo Valores, Formato, Comentarios, etc). Y hace que **A1** sea la **Celda Aciva**.

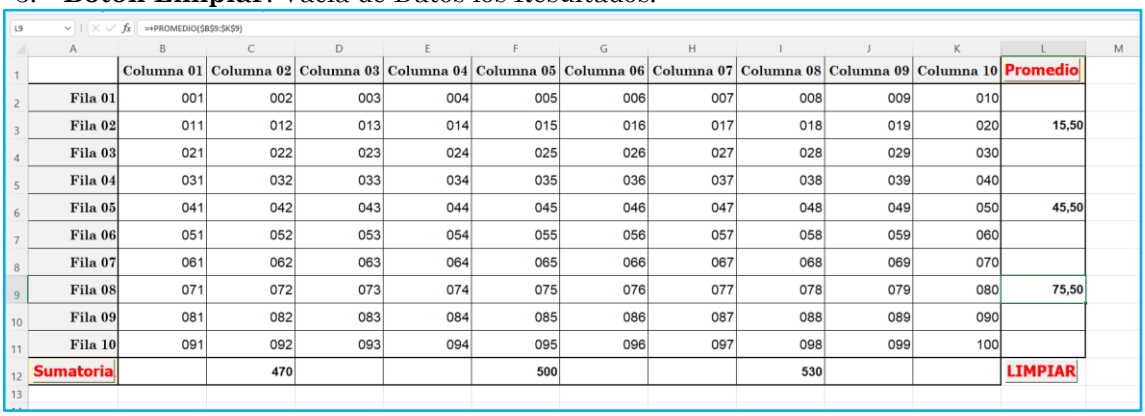


NOTA. Deberemos tener en cuenta que **no podemos Subir** estando en la **Fila 1** y **no podemos desplazarnos a la Izquierda**, estando en la **Columna A**, porque el programa devuelve **Error** y todavía no tenemos Control de Errores.

Ejercicio 02.03.-

Crearemos un Archivo vacío llamado ***EJERCICIO0203.xlsm***. El **Ejercicio Propuesto** consiste en crear **3 Botones de Formulario** con diferentes acciones. El Código VBA Excel irá dentro del **Módulo *Operaciones*** y realizará las siguientes operaciones.

1. **Botón Sumatoria** de la Columna seleccionada.
   a. ***Application.ActiveCell.Value = "=+SUM(" & Selection.Address & ")"***
2. **Botón Promedio** de la Fila seleccionada.
   a. ***Application.ActiveCell.Value = "=+AVERAGE(" & Selection.Address & ")"***
3. **Botón Limpiar**. Vacía de Datos los Resultados.



	Columna 01	Columna 02	Columna 03	Columna 04	Columna 05	Columna 06	Columna 07	Columna 08	Columna 09	Columna 10	Promedio
Fila 01	001	002	003	004	005	006	007	008	009	010	
Fila 02	011	012	013	014	015	016	017	018	019	020	15,50
Fila 03	021	022	023	024	025	026	027	028	029	030	
Fila 04	031	032	033	034	035	036	037	038	039	040	
Fila 05	041	042	043	044	045	046	047	048	049	050	45,50
Fila 06	051	052	053	054	055	056	057	058	059	060	
Fila 07	061	062	063	064	065	066	067	068	069	070	
Fila 08	071	072	073	074	075	076	077	078	079	080	75,50
Fila 09	081	082	083	084	085	086	087	088	089	090	
Fila 10	091	092	093	094	095	096	097	098	099	100	
Sumatoria		470			500			530			LIMPIAR

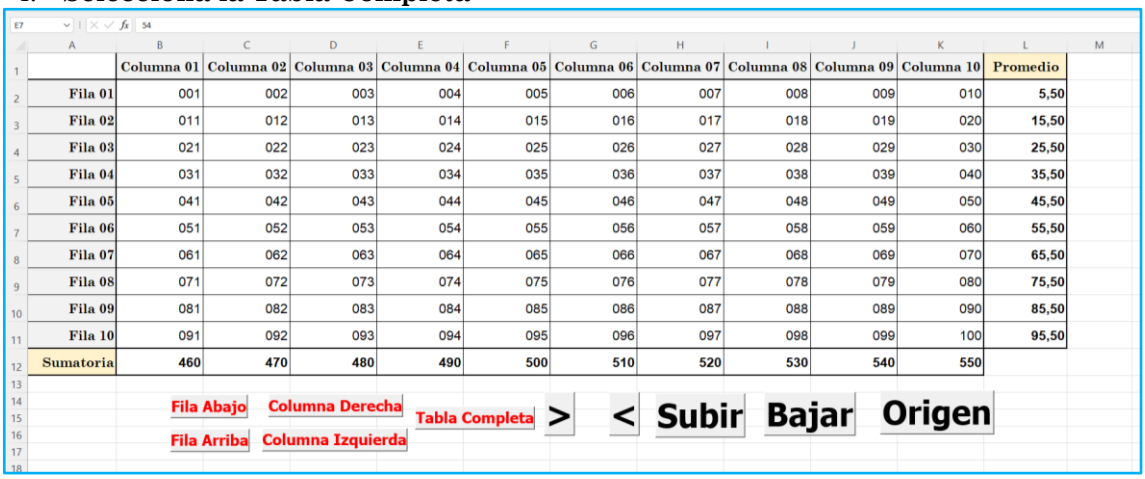
Tendremos varias cosas en cuenta porque **el programa no impide escribir en cualquier Celda**.

1. Antes de pulsar alguno de los Botones, habrá que seleccionar la Celda donde irá el **Resultado**.
2. En caso de realizar la **Sumatoria**, se deberá seleccionar alguna **Celda entre B12 y K12**.
3. En caso de realizar el **Promedio**, se deberá seleccionar alguna **Celda entre L2 y L11**.

Ejercicio 02.04.-

Crearemos un Archivo vacío llamado ***EJERCICIO0204.xlsm***. El **Ejercicio Propuesto** consiste en crear varios **Botones de Formulario** con diferentes acciones. El Código VBA Excel irá dentro del **Módulo *SeleccionEnd*** y realizará las siguientes operaciones.

1. **Avanza una Celda en la Dirección Indicada.**
2. **Avanza hasta la Ultima Celda con Datos (Filas-Abajo\Columnas-Derecha), al tiempo que realiza la Selección.**
3. **Avanza hasta la Primera Celda con Datos (Filas-Arriba\Columnas-Izquierda), al tiempo que realiza la Selección.**
4. **Selecciona la Tabla Completa**



	Columna 01	Columna 02	Columna 03	Columna 04	Columna 05	Columna 06	Columna 07	Columna 08	Columna 09	Columna 10	Promedio
Fila 01	001	002	003	004	005	006	007	008	009	010	5,50
Fila 02	011	012	013	014	015	016	017	018	019	020	15,50
Fila 03	021	022	023	024	025	026	027	028	029	030	25,50
Fila 04	031	032	033	034	035	036	037	038	039	040	35,50
Fila 05	041	042	043	044	045	046	047	048	049	050	45,50
Fila 06	051	052	053	054	055	056	057	058	059	060	55,50
Fila 07	061	062	063	064	065	066	067	068	069	070	65,50
Fila 08	071	072	073	074	075	076	077	078	079	080	75,50
Fila 09	081	082	083	084	085	086	087	088	089	090	85,50
Fila 10	091	092	093	094	095	096	097	098	099	100	95,50
Sumatoria	460	470	480	490	500	510	520	530	540	550	

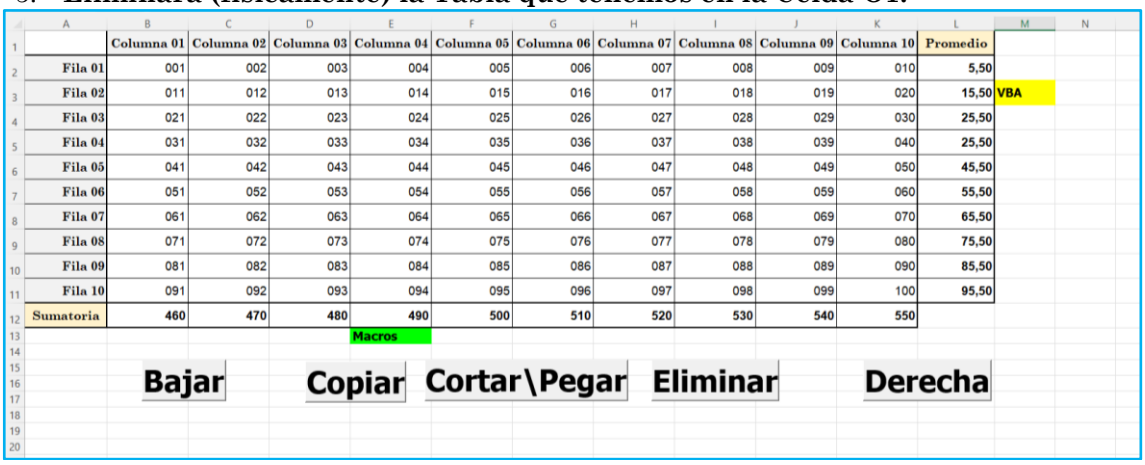
NOTA. Deberemos tener en cuenta que **no podemos Subir** estando en la **Fila 1** y **no podemos desplazarnos a la Izquierda**, estando en la **Columna A**, porque el programa devuelve **Error** y no tenemos **Control de Errores**.

NOTA. Utiliza la Tabla del ***Ejercicio 02.03***.

Ejercicio 02.05.-

Crearemos un Archivo vacío llamado ***EJERCICIO0205.xlsm***. El **Ejercicio Propuesto** consiste en crear varios **Botones de Formulario** con diferentes acciones. El Código VBA Excel irá dentro del **Módulo *SeleccionEnd*** y realizará las siguientes operaciones.

1. **Buscará la Primera Fila (hacia Abajo) cuya Celda no contenga Datos y escribirá la Palabra "Macros", en Negrita, Tamaño 12, Fuente de Letra "Tahoma" y Relleno de Celda Verde.**
2. **Buscará la Primera Columna (hacia la Derecha) cuya Celda no contenga Datos y escribirá la Palabra "VBA", en Negrita, Tamaño 12, Fuente de Letra "Tahoma" y Relleno de Celda Amarillo.**
3. **Copiará la Tabla Completa en la Celda A20.**
4. **Cortará y Pegará en la Celda O1 la Tabla que pegamos en A20.**
5. **Eliminará (físicamente) la Tabla que tenemos en la Celda O1.**



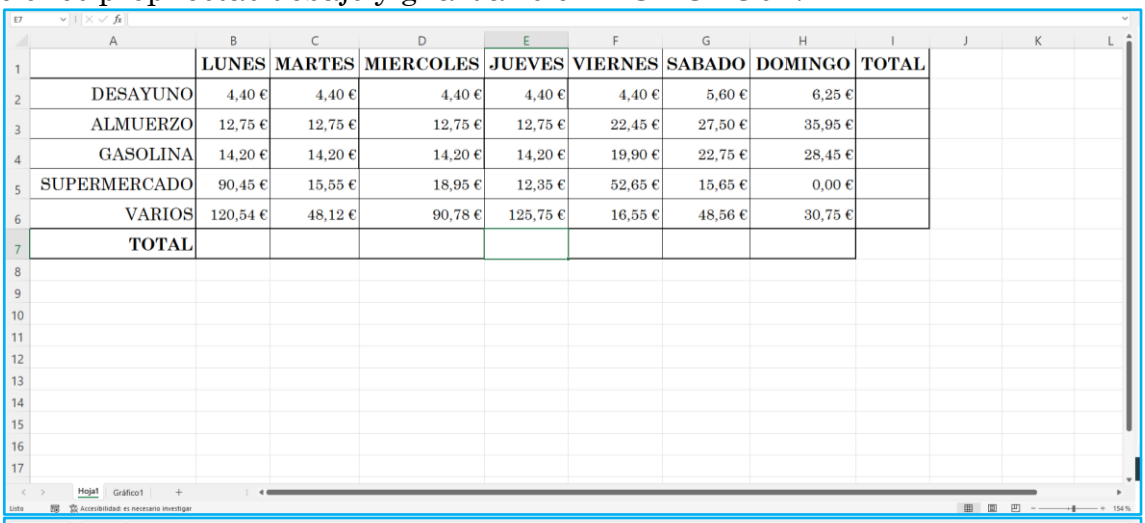
NOTA. Utiliza la Tabla del ***Ejercicio 02.03***.

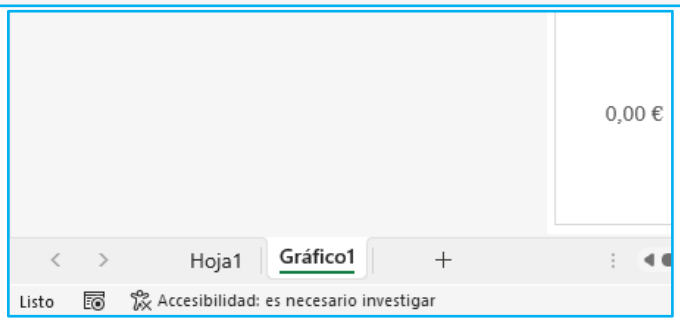
03.- HOJAS\LIBROS DE TRABAJO

Ejercicio 03.01.-

Crearemos un Archivo llamado ***EJERCICIO0301.xlsm*** en el que haremos la **Tabla** que aparece debajo (**Hoja1**) y el **Gráfico** correspondiente (**Gráfico1**). **Ejercicio Propuesto** consiste en hacer el Código VBA Excel que realice las acciones propuestas debajo y guardarlo en ***MODULO01***.

	LUNES	MARTES	MIERCOLES	JUEVES	VIERNES	SABADO	DOMINGO	TOTAL
DESAYUNO	4,40 €	4,40 €	4,40 €	4,40 €	4,40 €	5,60 €	6,25 €	
ALMUERZO	12,75 €	12,75 €	12,75 €	12,75 €	22,45 €	27,50 €	35,95 €	
GASOLINA	14,20 €	14,20 €	14,20 €	14,20 €	19,90 €	22,75 €	28,45 €	
SUPERMERCADO	90,45 €	15,55 €	18,95 €	12,35 €	52,65 €	15,65 €	0,00 €	
VARIOS	120,54 €	48,12 €	90,78 €	125,75 €	16,55 €	48,56 €	30,75 €	
TOTAL								



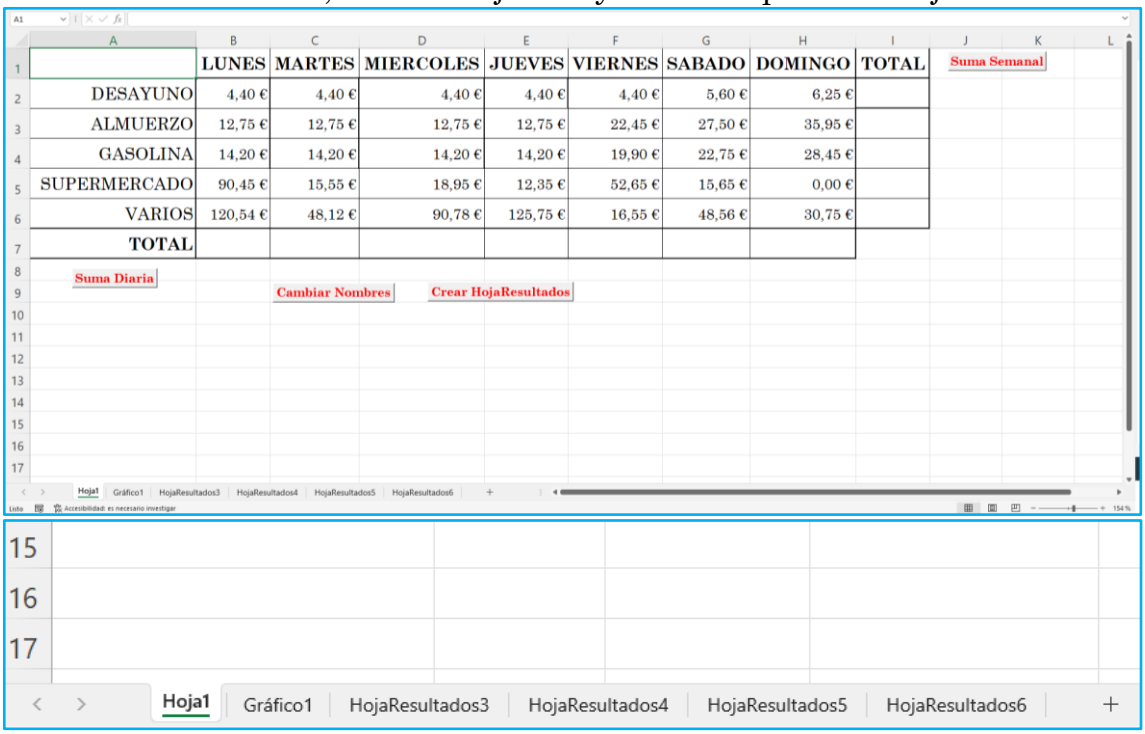


1. **Crear Botones de Formulario que ejecuten Sumatoria Diaria y Semanal (*Sub SumarDiario* y *Sub SumarSemanal*).**
2. **Cada vez que abra el Archivo, se borrará todas las Sumas. Probarlo antes de seguir.**
3. **Crear Botón de Formulario que cambie el Nombre *Hoja1* por *HojaDatos* y *Gráfico1* por *GráficoDatos*.**

4. **Volver a ejecutar las Sumas. Comprobar si aparece Error y el por qué.**
5. **Crear Botón de Formulario que cree la Hoja de Datos llamada *HojaResultados*, seguido del Número de Hojas que tiene el Libro de Trabajo actual, cada vez que la pulse. Las nuevas Hojas irán posicionadas al Final de todas (incluidas las Hojas Gráficas).**

Ejercicio 03.02.-

Crearemos un Archivo llamado ***EJERCICIO0302.xlsm*** que partirá del Archivo ***EJERCICIO0301.xlsm***, con las Hojas tal y como se aprecia debajo.



	LUNES	MARTES	MIERCOLES	JUEVES	VIERNES	SABADO	DOMINGO	TOTAL
DESAYUNO	4,40 €	4,40 €	4,40 €	4,40 €	4,40 €	5,60 €	6,25 €	
ALMUERZO	12,75 €	12,75 €	12,75 €	12,75 €	22,45 €	27,50 €	35,95 €	
GASOLINA	14,20 €	14,20 €	14,20 €	14,20 €	19,90 €	22,75 €	28,45 €	
SUPERMERCADO	90,45 €	15,55 €	18,95 €	12,35 €	52,65 €	15,65 €	0,00 €	
VARIOS	120,54 €	48,12 €	90,78 €	125,75 €	16,55 €	48,56 €	30,75 €	
TOTAL								

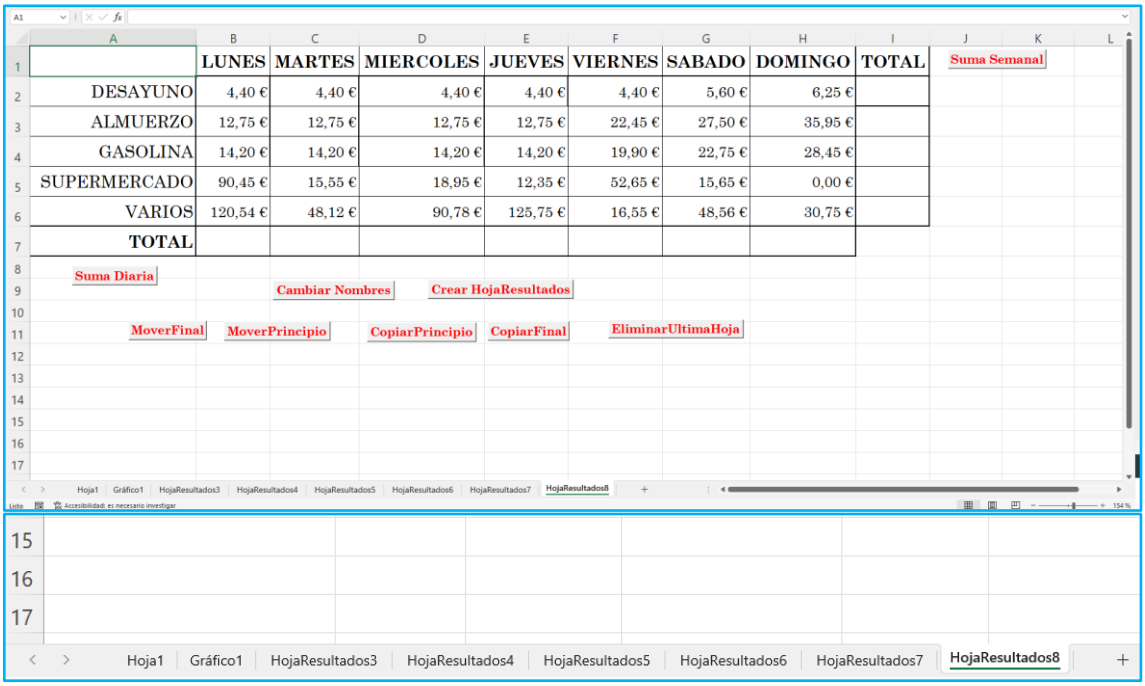
Suma Semanal

Suma Diaria | Cambiar Nombres | Crear HojaResultados

1. **Crear Botón de Formulario que Elimine la Ultima Hoja cada vez que se pulse. Ojo con no eliminar ni Hoja1, ni Gráfico1.**
2. **Crea un Botón que, al pulsarlo, Mueva la Hoja Seleccionada al Final y otro que Mueva la Hoja Seleccionada al Principio de todas.**
3. **Crea un Botón que, al pulsarlo, Copie la Hoja Seleccionada al Final y otro que Copie la Hoja Seleccionada al Principio. Tras Copiar la Hoja, en cada caso, deberá cambiarle el Nombre por el de *HojaResultados*, seguido del Número de Hojas que tiene el Libro de Trabajo actual.**

Partiendo del Archivo, el **Ejercicio Propuesto** consiste en hacer la siguiente secuencia, pulsando Botones de Formulario, cuyo Resultado sea el que se aprecia más abajo.

1. **Selecciona *Hoja1* y pulsa el Botón *CopiarFinal*.**
2. **Pulsa Botón *MoverPrincipio*.**
3. **Selecciona *Hoja1* y pulsa Botón *MoverFinal*.**
4. **Pulsa Botón *CopiarPrincipio*.**
5. **Pulsa Botón *MoverFinal*.**
6. **Selecciona *HojaResultados7* y pulsa Botón *MoverFinal*.**
7. **Selecciona *Hoja1* y pulsa Botón *MoverPrincipio*.**
8. **Selecciona *HojaResultados8* y pulsa Botón *MoverFinal*.**
9. **Pulsa 3 veces el Botón Crear *HojaResultados*.**
10. **Pulsa 3 veces el Botón *EliminarUltimaHoja*.**



	LUNES	MARTES	MIERCOLES	JUEVES	VIERNES	SABADO	DOMINGO	TOTAL
DESAYUNO	4,40 €	4,40 €	4,40 €	4,40 €	4,40 €	5,60 €	6,25 €	
ALMUERZO	12,75 €	12,75 €	12,75 €	12,75 €	22,45 €	27,50 €	35,95 €	
GASOLINA	14,20 €	14,20 €	14,20 €	14,20 €	19,90 €	22,75 €	28,45 €	
SUPERMERCADO	90,45 €	15,55 €	18,95 €	12,35 €	52,65 €	15,65 €	0,00 €	
VARIOS	120,54 €	48,12 €	90,78 €	125,75 €	16,55 €	48,56 €	30,75 €	
TOTAL								

Suma Semanal

Suma Diaria

Cambiar Nombres

Crear HojaResultados

MoverFinal

MoverPrincipio

CopiarPrincipio

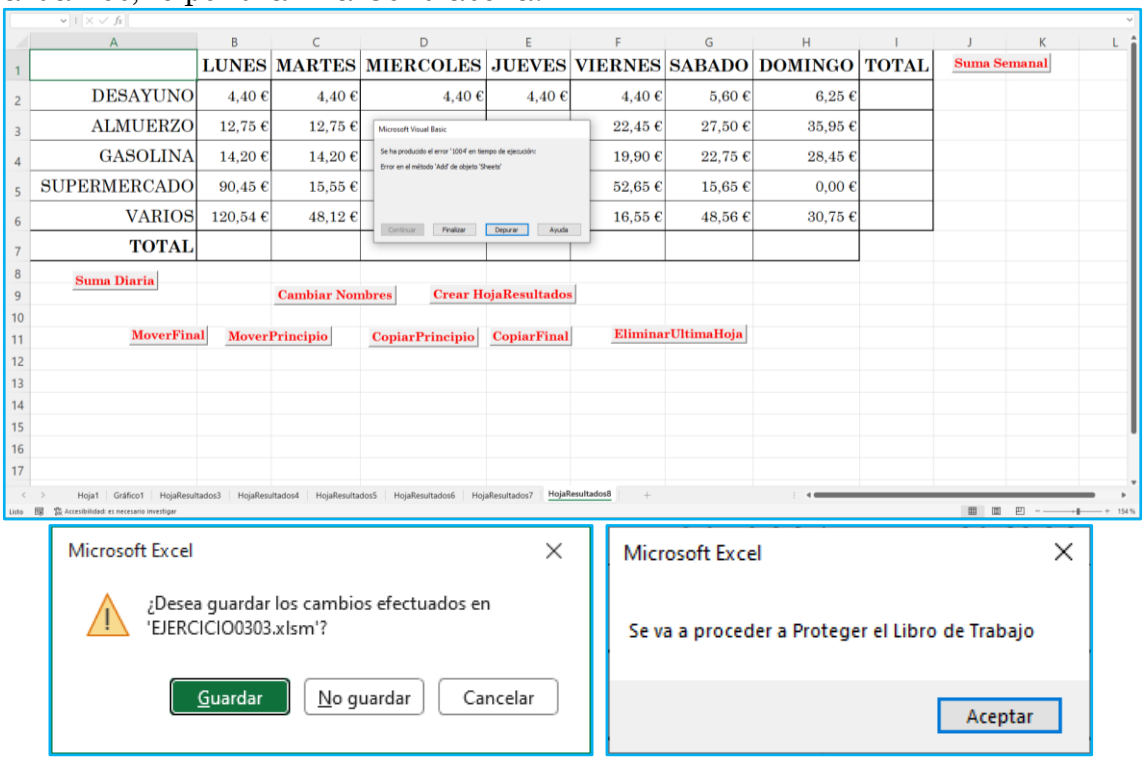
CopiarFinal

EliminarUltimaHoja

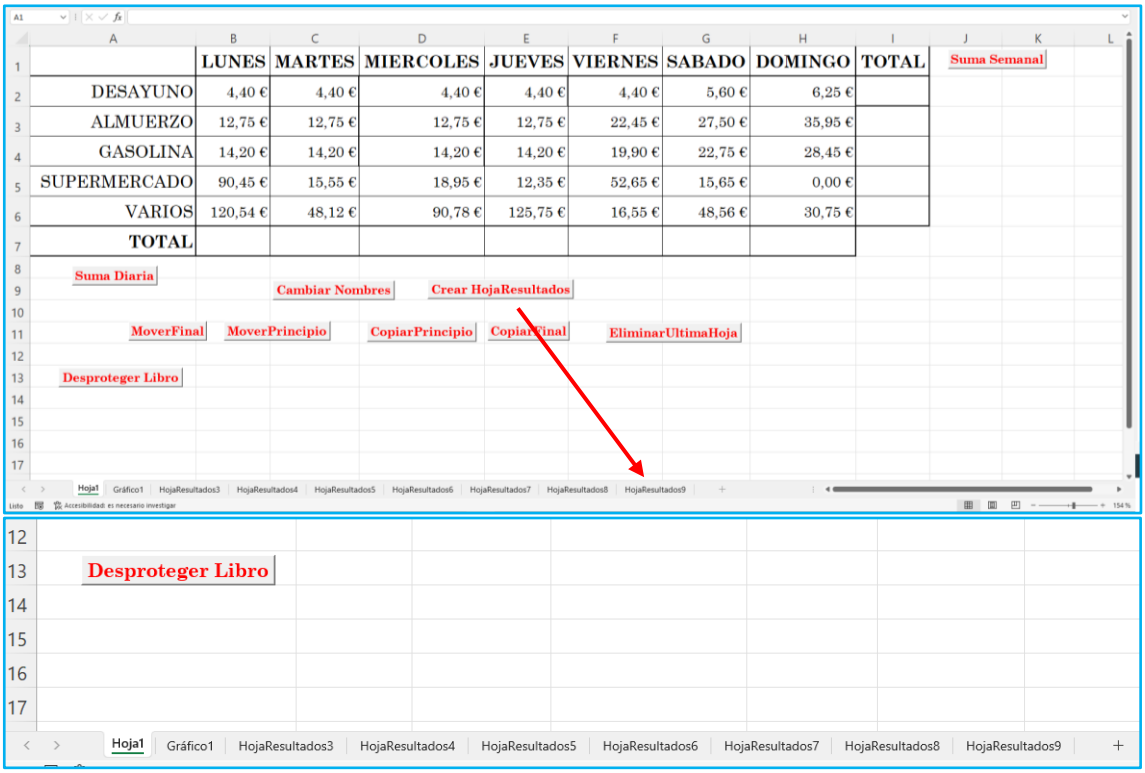
Ejercicio 03.03.-
Crearemos una Copia del Archivo ***EJERCICIO0302.xlsm*** con el Nombre ***EJERCICIO0303.xlsm*** y trabajaremos con este último.

A continuación, escribiremos el Código dentro del **Objeto *ThisWorkbook*** para que el **Evento *Application.Workbook_BeforeSave*** nos permita **Proteger el Libro de Trabajo** con contraseña (nuestro nombre) cada vez que **guardemos** el Archivo.

Al Abrir el Archivo y Pulsar el **Botón de Formulario *Crear HojaResultados***, obtendremos el siguiente **Error**, ya que **no se puede añadir** ninguna Hoja porque el **Libro de Trabajo está Protegido**. Efectivamente, al cerrar el Archivo, si lo guardamos, le pondrá una Contraseña.



El **Ejercicio Propuesto** consiste en crear un **Botón de Comando** en ***Hoja1*** que **Desproteja** el Libro de Trabajo. Una vez Desprotegido, al pulsar el **Botón de Formulario *Crear HojaResultados*** comprobamos que podemos crear nuevas Hojas.


LUNES MARTES MIERCOLES JUEVES VIERNES SABADO DOMINGO TOTAL
Suma Semanal
DESAYUNO 4,40 € 4,40 € 4,40 € 4,40 € 4,40 € 5,60 € 6,25 €
ALMUERZO 12,75 € 12,75 € 12,75 € 12,75 € 22,45 € 27,50 € 35,95 €
GASOLINA 14,20 € 14,20 € 14,20 € 14,20 € 19,90 € 22,75 € 28,45 €
SUPERMERCADO 90,45 € 15,55 € 18,95 € 12,35 € 52,65 € 15,65 € 0,00 €
VARIOS 120,54 € 48,12 € 90,78 € 125,75 € 16,55 € 48,56 € 30,75 €
TOTAL
Suma Diaria
Cambiar Nombres
Crear HojaResultados
MoverFinal
MoverPrincipio
CopiarPrincipio
CopiarFinal
EliminarUltimaHoja
Desproteger Libro
Hoja1 Gráfico1 HojaResultados3 HojaResultados4 HojaResultados5 HojaResultados6 HojaResultados7 HojaResultados8 HojaResultados9

Desproteger Libro
Hoja1 Gráfico1 HojaResultados3 HojaResultados4 HojaResultados5 HojaResultados6 HojaResultados7 HojaResultados8 HojaResultados9

Ejercicio 03.04.-
Crearemos una Copia del Archivo ***EJERCICIO0303.xlsm*** con el Nombre ***EJERCICIO0304.xlsm*** y trabajaremos con este último.

Vamos a trabajar con la **Propiedad *Visible*** de las **2 últimas Hojas de Libro de Trabajo**, de tal modo que haremos que se **Oculte (*Visible* = *False*)** la **Penúltima Hoja** y que se haga **VeryHidden**, la **última Hoja**.

El **Ejercicio Propuesto** consiste en crear **3 Botones de Formulario** que realicen cada acción sobre la Hoja Correspondiente. Los Botones de Formulario que permiten Ocultar, deberán estar en las Hojas que vayamos a **Ocultar** para que el Ejemplo se pueda llevar a cabo. El Tercer Botón de Formulario (***Hojas Visibles***), podrán estar en cualquier Hoja que **NO** vayamos a Ocultar.

El funcionamiento es sencillo. Al **seleccionar una Hoja**, podremos **Ocultarla** o hacerla **Muy Oculta**. Y al pulsar el **Botón de Formulario *Hojas Visibles*** haremos **Visibles las 2 últimas Hojas**. **Así es que sólo probaremos el ejemplo en las 2 últimas Hojas**.

Ejercicio 03.05.-

Crearemos un Archivo que guardaremos con el Nombre ***EJERCICIO0305.xlsm***, el cual será el **Libro Activo** en este momento. Dentro de ***Hoja1***, crearemos un Procedimiento que genere **2 Nuevos Libros de Trabajo**. Usaremos el siguiente Código y tendremos que analizar las diferencias que vemos entre usar ambos comandos, cuando lo ejecutemos. Nos preguntaremos **¿Cuál de los 2 Nuevos Libros de Trabajo es el Libro Activo ahora? ¿Cuántas Hojas tiene cada Nuevo Libro de Trabajo? ¿Cómo se llama cada Nuevo Libro de Trabajo?**

```
Sub CrearLibro()

    Application.SheetsInNewWorkbook = 3 'Cantidad de Hojas en Blanco
    MsgBox "Microsoft Excel añadirá " & Application.SheetsInNewWorkbook & " Hojas en cada Libro Nuevo"

    Application.Workbooks.Add 'Se omite el argumento
    MsgBox "El nuevo Libro de Trabajo se llama " & Application.ActiveWorkbook.Name

    Application.Workbooks.Add -4167
    MsgBox "El nuevo Libro de Trabajo se llama " & Application.ActiveWorkbook.Name

End Sub
```

NOTA. Si el Argumento que acompaña al **Método *Add*** es una **constante**, el nuevo Libro de Trabajo contiene **1 única Hoja del tipo especificado (*xlWBATChart, xlWBATExcel4IntlMacroSheet, xlWBATExcel4MacroSheet o xlWBATWorksheet*).** En el ejemplo el argumento es **-4167 (*xlWBATWorksheet*)** que especifica que **el nuevo Libro de Trabajo contendrá 1 única Hoja de Tipo Worksheet**. En cambio, si se **omite** el Argumento, VBA Excel creará un nuevo Libro de Trabajo con **varias Hojas en Blanco** (la cantidad de Hojas la establece la ***Propiedad SheetsInNewWorkbook***). ***https://learn.microsoft.com/es-es/office/vba/api/excel.workbooks.add***.

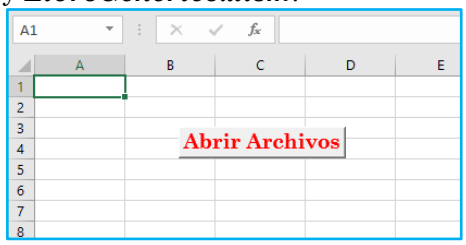
Especifica el tipo de libro que se va a crear. El nuevo libro contiene una sola hoja del tipo especificado.

Nombre	Valor	Descripción
xlWBATChart	-4109	Gráfico
xlWBATExcel4IntlMacroSheet	4	Macro de Excel versión 4
xlWBATExcel4MacroSheet	3	Macro de Excel versión 4 internacional
xlWBATWorksheet	-4167	Worksheet

El **Ejercicio Propuesto** consiste en crear los siguientes Objetos, Procedimientos y ejecutarlos.

- **Crear** (y **Ejecutar**) el **Procedimiento *GuardarTipoWorkbook*** en la ***Hoja1*** del Archivo ***EJERCICIO0305.xlsm*** que **Guarde** el **Libro Activo Hoja1** con el Nombre ***TipoWorkbook.xlsm*** en la Ruta **C:\Users\Luis\Documents** (añade la Ruta que tú prefieras). **Cerrar** el Libro de Trabajo.
- **Crear** (y **Ejecutar**) el **Procedimiento *GuardarLibroGenerico*** en la ***Hoja1*** del Archivo ***EJERCICIO0305.xlsm*** que **Guarde** el **Libro Activo Libro1** con el Nombre ***LibroGenerico.xlsm*** y que contenga **Contraseña de Apertura (tu NOMBRE)**, en la misma Ruta anterior. **Cerrar** el Libro de Trabajo

- Crear un **Botón de Formulario** en la **Hoja1** de ***EJERCICIO0305.xlsm*** que **abra** ***TipoWorkbook.xlsm*** y ***LibroGenerico.xlsm***.



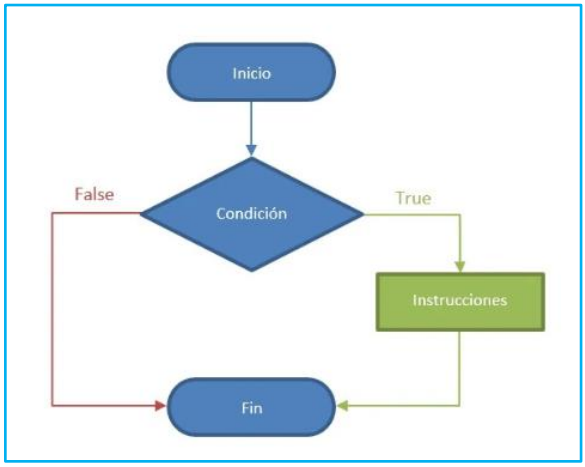
04.- TOMANDO DECISIONES

Ejercicio 04.01.-

Crearemos un Archivo vacío, llamado ***EJERCICIO0401.xlsm***. Luego creamos la siguiente Tabla. Se trata de un **Cuadrante de Notas** para Calificar la **Nota Media** de Cada Alumno.

	A	B	C	D	E	F
1		**Nota1**	**Nota2**	**Nota3**	**Nota Media**	
2	**Juan**	3,13	4,20	5,31		
3	**José**	7,48	9,13	8,68		
4	**María**	2,70	6,89	3,04		
5	**Mónica**	6,46	8,05	1,07		
6	**Paki**	6,68	5,74	6,45		
7	**Ana**	5,00	5,00	5,00		
8	**Claudia**	5,99	1,34	7,21		
9	**Alvaro**	2,20	5,51	6,67		
10						

El **Ejercicio Propuesto** consiste en ir inser un **Módulo** en el que iremos haciendo los diferentes Procedimientos, empezando por el **Procedimiento *CalcularMedia*** que se encargará de calcular la **Media Aritmética** de las 3 Notas de Cada Alumno y el Valor deberá aparecer en la **Columna Nota Media**. El Código sólo podrá usar el **Método *Offset* y la Instrucción *If...Then*** que nos permite **decidir una acción** en función de una **condición**.

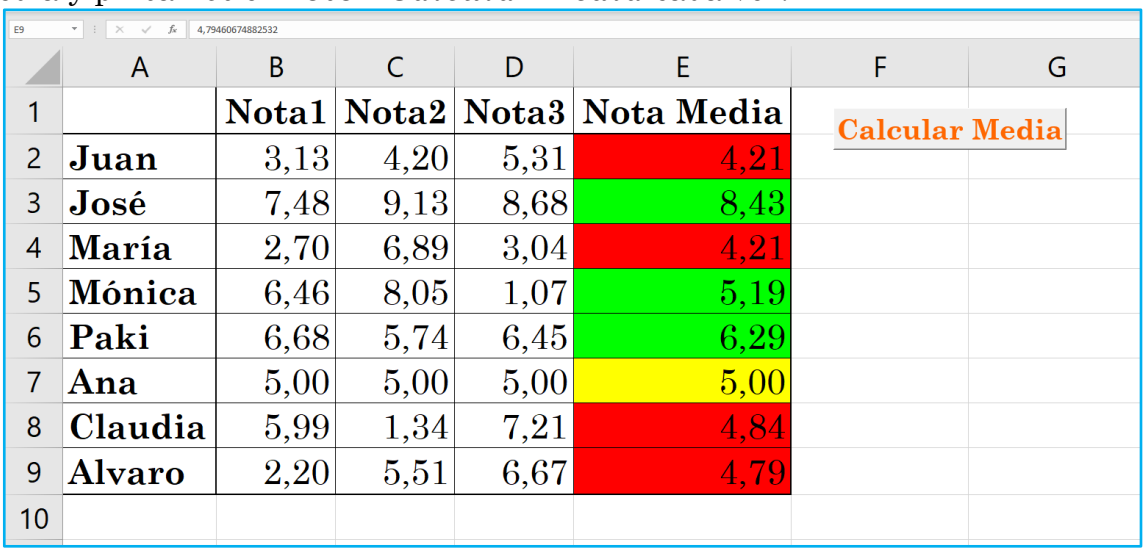


Si el resultado obtenido es **Inferior a 5**, la Celda se tornará **Roja** (***vbRed***) y hará un **Sonido (Beep)** de advertencia. La **Instrucción *Beep***, hace que el altavoz del equipo emita un sonido.

Si el resultado obtenido es **Igual a 5**, la Celda se tornará **Amarillo** (***vbYellow***). Y si el resultado obtenido fuera **Superior a 5**, la Celda se tornará **Verde** (***vbGreen***).

Añadiremos un **Botón de Formulario (*Calcular Media*)** que, al pulsarlo, ejecute el **Procedimiento *CalcularMedia***.

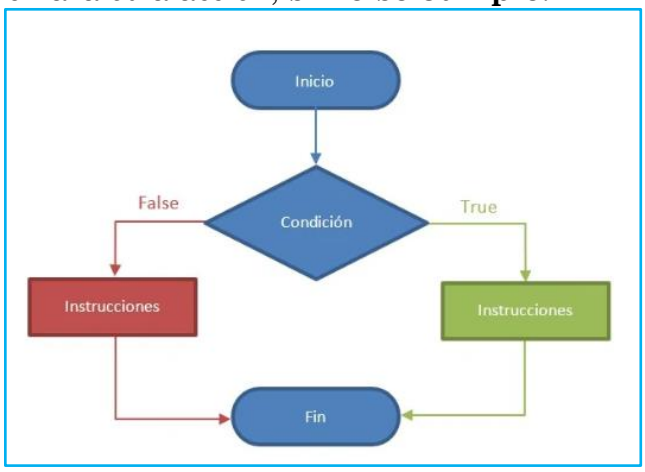
Una vez hecho el Programa, seleccionamos de uno en uno cada Rango de Nota Media y pulsamos el **Botón *Calcular Media*** cada vez.



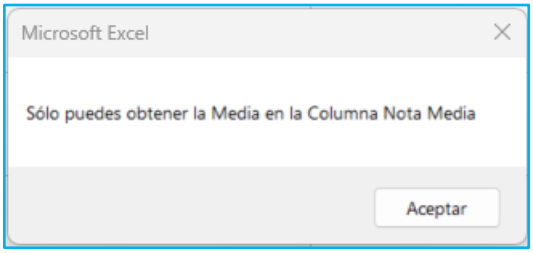
	A	B	C	D	E	F	G
1		**Nota1**	**Nota2**	**Nota3**	**Nota Media**		
2	**Juan**	3,13	4,20	5,31	4,21		
3	**José**	7,48	9,13	8,68	8,43		
4	**María**	2,70	6,89	3,04	4,21		
5	**Mónica**	6,46	8,05	1,07	5,19		
6	**Paki**	6,68	5,74	6,45	6,29		
7	**Ana**	5,00	5,00	5,00	5,00		
8	**Claudia**	5,99	1,34	7,21	4,84		
9	**Alvaro**	2,20	5,51	6,67	4,79		
10							

Ejercicio 04.02.-

Guardaremos una **Copia** de ***EJERCICIO0401.xlsm*** con el Nombre ***EJERCICIO0402.xlsm***. Borramos los resultados de la Nota Media de todos los alumnos. Ahora, el programa no sólo realizará una acción **si se cumple la condición**, sino que hará otra acción, **si no se cumple**.



El **Ejercicio Propuesto** consiste en crear el **Procedimiento *ComprobarCeldaSeleccionada***, asociado al **Botón de Formulario *Calcular Media***, que **comprobará la Fila y la Columna de la Celda seleccionada.** Si no está en el **Rango de E2 a E9**, entonces aparecerá un Mensaje que indica "**Sólo puedes obtener la Media en la Columna Nota Media**". En caso contrario, llamará al **Procedimiento *CalcularMedia***.

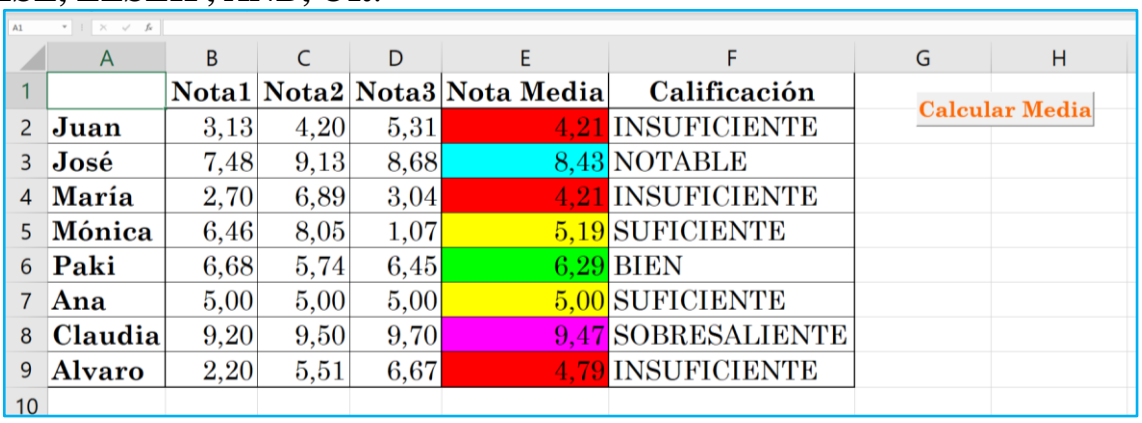


NOTA. Utiliza las **Propiedades *Selection.Row*** y ***Selection.Column*** para determinar la **Fila** y la **Columna** de la **Celda Seleccionada**, junto con la **Instrucción *If ... Then ... Else***.

Ejercicio 04.03.-

Añadimos la Columna Calificación al Archivo ***EJERCICIO0402.xlsm*** y lo guardamos con el Nombre ***EJERCICIO0403.xlsm***, **cambiando algunas Notas**.

Nuestro Programa deberá dar el siguiente resultado al aplicar el **Botón de Formulario** a cada una de los resultados de ***Nota Media*** usando **IF … THEN … ELSE, ELSEIF, AND, OR**.



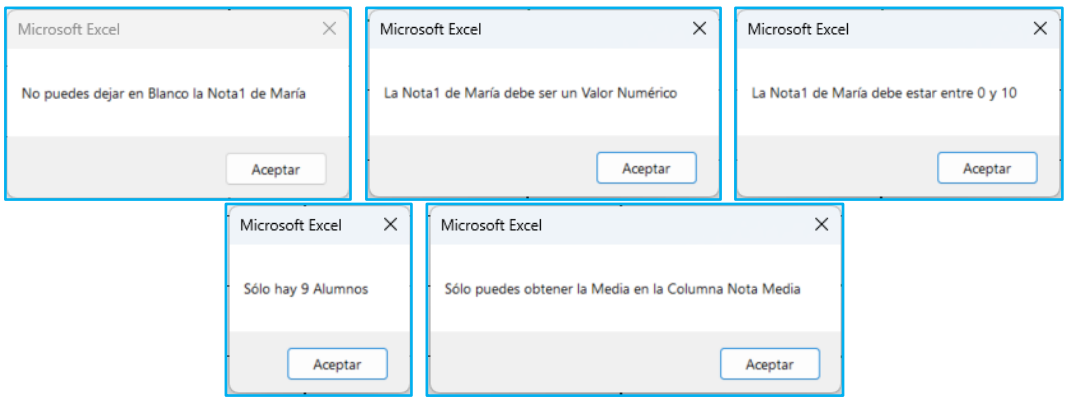
	A	B	C	D	E	F	G	H
1		**Nota1**	**Nota2**	**Nota3**	**Nota Media**	**Calificación**	Calcular Media	
2	**Juan**	3,13	4,20	5,31	4,21	INSUFICIENTE		
3	**José**	7,48	9,13	8,68	8,43	NOTABLE		
4	**María**	2,70	6,89	3,04	4,21	INSUFICIENTE		
5	**Mónica**	6,46	8,05	1,07	5,19	SUFICIENTE		
6	**Paki**	6,68	5,74	6,45	6,29	BIEN		
7	**Ana**	5,00	5,00	5,00	5,00	SUFICIENTE		
8	**Claudia**	9,20	9,50	9,70	9,47	SOBRESALIENTE		
9	**Alvaro**	2,20	5,51	6,67	4,79	INSUFICIENTE		
10								

El **Ejercicio Propuesto** consiste en hacer un Procedimiento que, al pulsar el **Botón de Formulario *Calcular Media***, calcule la **Media Aritmética de las 3 Notas**, teniendo en cuenta lo siguiente.

a) Por cada Registro se deberá tener en cuenta lo siguiente y obtendremos un Mensaje indicándolo.
   1) No puede haber ninguna Nota Vacía.
   2) No puede haber una Nota que contenga Texto.
   3) Las Notas sólo pueden valer entre 0 y 10.
   4) Sólo se puede Calcular la Nota Media en las líneas donde haya Registro.

b) Una vez obtenida la Nota Media, deberá aparecer la **Calificación** a la Derecha.
   a. 0 =< Nota Media < 5 => "**INSUFICIENTE**" (Relleno Color Rojo)
   b. 5 =< Nota Media < 6 => "**SUFICIENTE**" (Amarillo)
   c. 6 =< Nota Media < 7 => "**BIEN**" (Verde)
   d. 7 =< Nota Media < 9 => "**NOTABLE**" (Cyan)
   e. 9 =< Nota Media <=10 => "**SOBRESALIENTE**" (Magenta)

Estos son ejemplos de **mensajes de Error** que detecta el programa al pulsar el **Botón de Formulario *Calcular Media***.



NOTA. He utilizado algunas Variables para facilitar el Código.

Ejercicio 04.04.-
Tomamos el archivo ***EJERCICIO0403.xlsm*** y lo guardamos con el Nombre ***EJERCICIO0404.xlsm***, cambiando alguna Notas.

El **Ejercicio Propuesto** consiste en **sustituir** el bloque ***If … Then … ElseIf*** que pone la **Calificación** por un bloque ***Select Case*** que haga lo mismo.

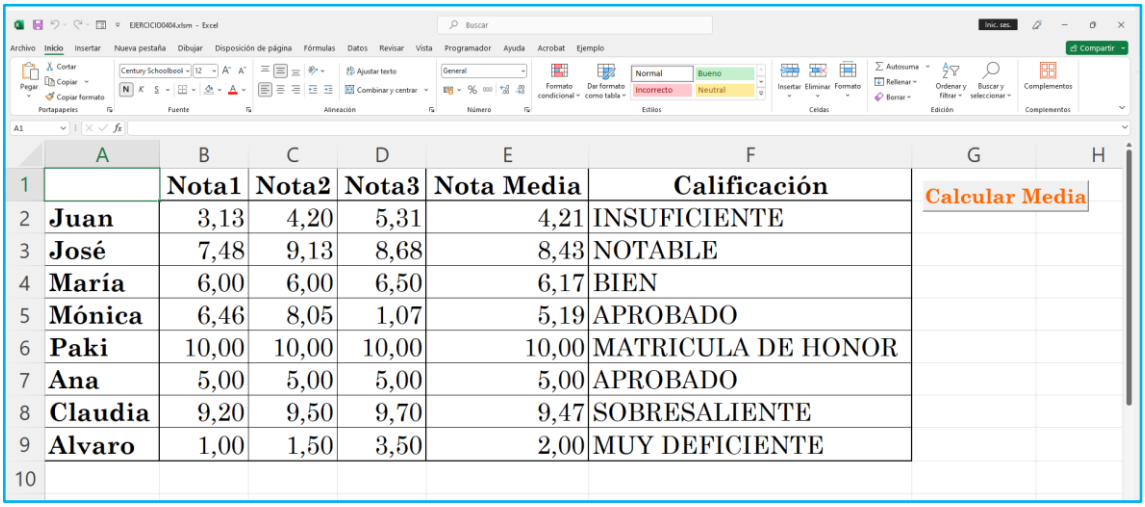
Las instrucciones ***Select Case*** en VBA Excel permiten evaluar una expresión y ejecutar una serie de instrucciones dependiendo del valor de la expresión introducida. Según las expresiones, los valores que se pueden evaluar son como las siguientes.

- Un único valor. Por ejemplo, Case 7 o Case "Resultado".
- Una Lista de Valores. Por ejemplo, Case 1,2,3 o Case "Juan", "María", "José", 7.
- Un Rango de Valores. Por ejemplo, Case 20 To 30
- Una Condición. Por ejemplo, Case Is >5, Case Is <= 25
- Cuando no se cumple ninguna Condición. Por ejemplo, Case Else: MsgBox "No se cumple"

Nuestro **Primer Programa** deberá dar el siguiente resultado al pulsar el **Botón de Formulario** a cada una de los resultados de ***Nota Media*** usando la estructura **Select Case** y cumpliendo con las **condiciones** que expongo a continuación.

a) Por cada Registro se deberá tener en cuenta lo siguiente y obtendremos un Mensaje indicándolo.
   5) No puede haber ninguna Nota Vacía.
   6) No puede haber una Nota que contenga Texto.
   7) Las Notas sólo pueden valer Números entre 0 y 10.
   8) Sólo se puede Calcular la Nota Media en las líneas donde haya Registro (Alumnos).

b) Una vez obtenida la Nota Media, deberá aparecer la **Calificación** a la Derecha (Sin Color).
   a. 0 =< Nota Media <= 3 => "**MUY DEFICIENTE**"
   b. 3 < Nota Media < 5 => "**INSUFIENTE"**
   c. 5 =< Nota Media < 6 => "**SUFICIENTE**"
   d. 6 =< Nota Media < 7 => "**BIEN**"
   e. 7 =< Nota Media < 9 => "**NOTABLE**"
   f. 9 =< Nota Media <10 => "**SOBRESALIENTE**"
   g. Nota Media = 10 => "**MATRICULA DE HONOR**"



	Nota1	Nota2	Nota3	Nota Media	Calificación
Juan	3,13	4,20	5,31	4,21	INSUFICIENTE
José	7,48	9,13	8,68	8,43	NOTABLE
María	6,00	6,00	6,50	6,17	BIEN
Mónica	6,46	8,05	1,07	5,19	APROBADO
Paki	10,00	10,00	10,00	10,00	MATRICULA DE HONOR
Ana	5,00	5,00	5,00	5,00	APROBADO
Claudia	9,20	9,50	9,70	9,47	SOBRESALIENTE
Alvaro	1,00	1,50	3,50	2,00	MUY DEFICIENTE

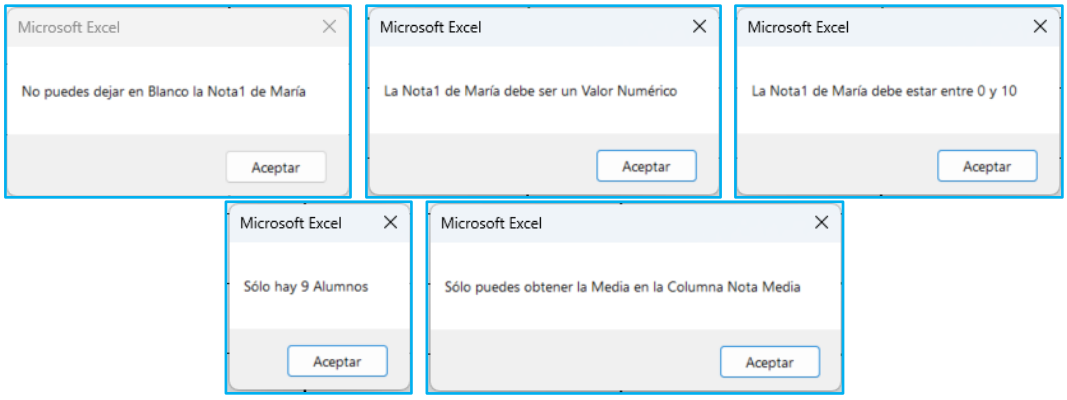
Nuestro **Segundo Programa**, también con la estructura ***Select Case***, deberá presentar la Calificación con Relleno de Color.

a) Una vez obtenida la Nota Media, deberá aparecer la ***Calificación*** a la Derecha aplicándole Colores.
   a. 0 =< Nota Media <= 3 => "**MUY DEFICIENTE**" (Relleno Color Rojo)
   b. 3 < Nota Media < 5 => "**INSUFIENTE"** (Rojo)
   c. 5 =< Nota Media < 6 => "**SUFICIENTE**" (Amarillo)
   d. 6 =< Nota Media < 7 => "**BIEN**" (Verde)
   e. 7 =< Nota Media < 9 => "**NOTABLE**" (Cyan)
   f. 9 =< Nota Media <10 => "**SOBRESALIENTE**" (Magenta)
   g. Nota Media = 10 => "**MATRICULA DE HONOR**" (Magenta)

	Nota1	Nota2	Nota3	Nota Media	Calificación
Juan	3,13	4,20	5,31	4,21	INSUFICIENTE
José	7,48	9,13	8,68	8,43	NOTABLE
María	6,00	6,00	6,50	6,17	BIEN
Mónica	6,46	8,05	1,07	5,19	APROBADO
Paki	10,00	10,00	10,00	10,00	MATRICULA DE HONOR
Ana	5,00	5,00	5,00	5,00	APROBADO
Claudia	9,20	9,50	9,70	9,47	SOBRESALIENTE
Alvaro	1,00	1,50	3,50	2,00	MUY DEFICIENTE

Calcular Media

Estos son ejemplos de mensajes de **Control de Errores** que detecta el programa al pulsar el **Botón de Formulario *Calcular Media***.



NOTA. He utilizado varias Variables para facilitar el Código.

Ejercicio 04.05.-

El **Ejercicio Propuesto** consiste en determinar el **precio final de un artículo** en función del criterio de la Cantidad Adquirida y si es Cliente Registrado. Una vez terminado guárdalo con el Nombre de ***EJERCICIO0405.xlsm***.

Las Condiciones para hacer el ejercicio son las siguientes.

1) Si adquiere más de **100 unidades** de un artículo cuyo **precio sea superior a 5€**, tendrá un **descuento del 15%**, sobre el **precio del artículo**.
2) No se puede vender más de **200 unidades** de ningún artículo, por cliente, salvo que se trate de un **Cliente Registrado (Código 01)**, frente a un **Cliente No Registrado (Código 00)**.
3) Todo **Cliente Registrado** tendrá un **descuento del 5%** sobre la **Sumatoria** de la factura (antes de impuestos).
4) El Impuesto es el **IGIC (7%)**.
5) Utiliza Condicionales ***If … Then … Else … ElseIf***, para hacer el **Procedimiento *SubTotal***.

	A	B	C	D	E
1		Precio Unitario	Cantidad	SubTotal	
2	Calcetines	4,50 €	150,00	675,00 €	
3	Camisetas	15,99 €	125,00	1.698,94 €	
4	Pantalones	25,95 €	100,00	2.595,00 €	
5	Zapatos	85,75 €	200,00	14.577,50 €	
6	Chaquetas	178,25 €	199,00	30.150,99 €	
7			Sumatoria (Sin Descuento)	49.697,43 €	
8	SubTotal	Total	Sumatoria (incluído Dto 15%)	47.212,55 €	
9	AutoAjustar	Limpiar	IGIC (7%)	3.304,88 €	
10			TOTAL	50.517,43 €	
11					
12	Tipo cliente	01			

	A	B	C	D	E
1		Precio Unitario	Cantidad	SubTotal	
2	Calcetines	4,50 €	150,00	675,00 €	
3	Camisetas	15,99 €	125,00	1.698,94 €	
4	Pantalones	25,95 €	100,00	2.595,00 €	
5	Zapatos	85,75 €	200,00	14.577,50 €	
6	Chaquetas	178,25 €	199,00	30.150,99 €	
7			Sumatoria (Sin Descuento)	49.697,43 €	
8	SubTotal	Total	Sumatoria (incluído Dto 15%)	49.697,43 €	
9	AutoAjustar	Limpiar	IGIC (7%)	3.478,82 €	
10			TOTAL	53.176,25 €	
11					
12	Tipo cliente	00			

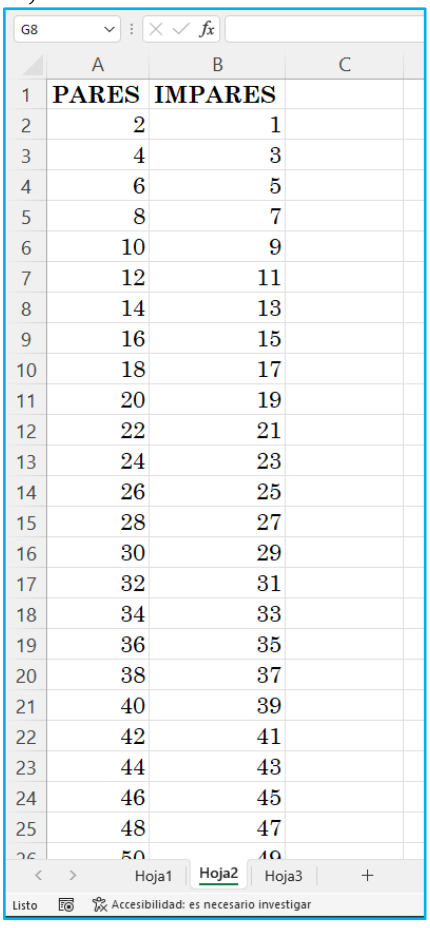
05.- BUCLES

Ejercicio 05.01.-

Crea un archivo llamado ***EJERCICIO0501.xlsm*** y guárdalo.

Dentro de este apartado haremos varios ejercicios basados en el uso del **Bucle *For … Next***. Este tipo de Estructura permite Crear bloques de Instrucciones en Bucle que se repiten un número determinado de veces. Trabajan con un contador de repeticiones y cuando este alcanza el número establecido como límite, sale del bucle (aunque se puede forzar salir antes). El contador podrá **Incrementar (+)** o **Decrementar (-)** los valores y podrá hacerse con saltos (no tiene que ser necesariamente de **1 en 1**).

El **Primer Ejercicio Propuesto** dentro de este apartado consiste en hacer un programa que recorra los números del 1 al 100 (de 1 en 1), escriba en la Columna A aquellos que sean **Pares** y, en la Columna B, aquellos que sean **Impares**. Ambas Columnas tendrán **Encabezado** (**Pares\Impares**) y debajo del último número de cada Columna, se hará la **Sumatoria** de unos y otros. El resultado deberá aparecer en ***Hoja2***. **¿Qué suma más, los Pares o los ImPares?**

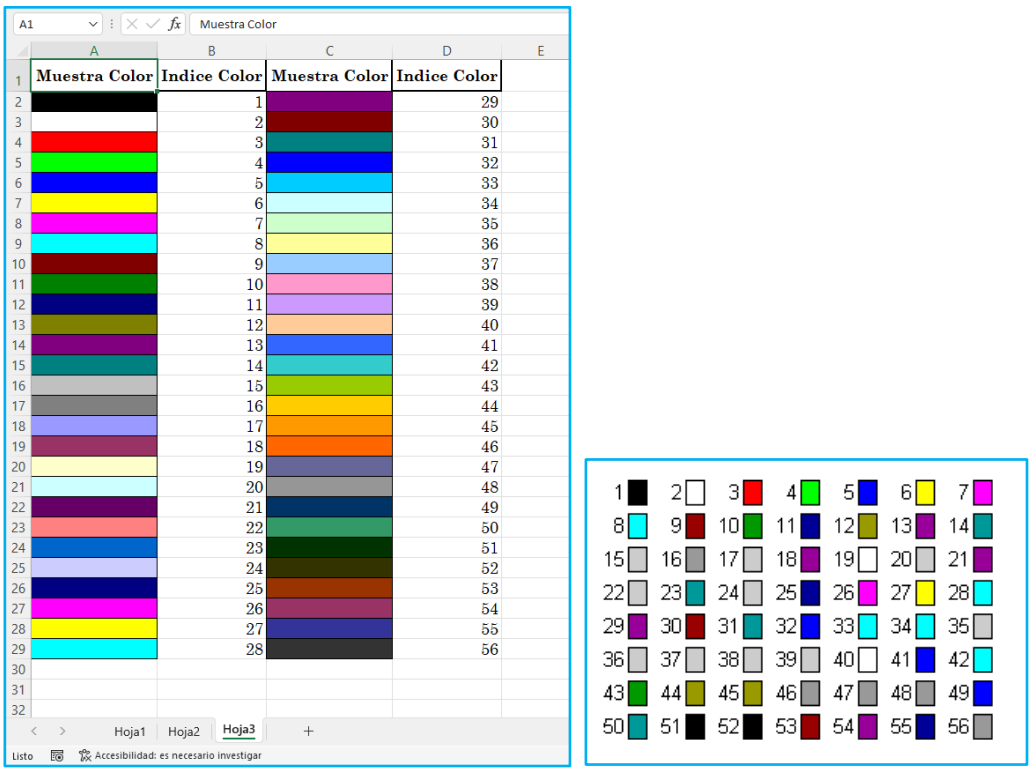


	A	B
1	PARES	IMPARES
2	2	1
3	4	3
4	6	5
5	8	7
6	10	9
7	12	11
8	14	13
9	16	15
10	18	17
11	20	19
12	22	21
13	24	23
14	26	25
15	28	27
16	30	29
17	32	31
18	34	33
19	36	35
20	38	37
21	40	39
22	42	41
23	44	43
24	46	45
25	48	47

NOTA. Recuerda que el **Operador *MOD*** divide 2 números y devuelve sólo el Resto. Todos aquellos números divididos entre 2 cuyo resto sea 0, son **Pares**.

El **Segundo Ejercicio Propuesto** lo realizaremos en la ***Hoja3*** y consiste en presentar en **2 Pares de Columnas** las Celdas rellenas con el Color que

corresponde a ***ColorIndex***, junto con el **índice del Color** al lado (a la derecha o a la izquierda del Color, como prefieras, **desde el valor 1 al 56**). Añade **Encabezados** a cada Columna.



Muestra Color	Indice Color	Muestra Color	Indice Color
	1		29
	2		30
	3		31
	4		32
	5		33
	6		34
	7		35
	8		36
	9		37
	10		38
	11		39
	12		40
	13		41
	14		42
	15		43
	16		44
	17		45
	18		46
	19		47
	20		48
	21		49
	22		50
	23		51
	24		52
	25		53
	26		54
	27		55
	28		56

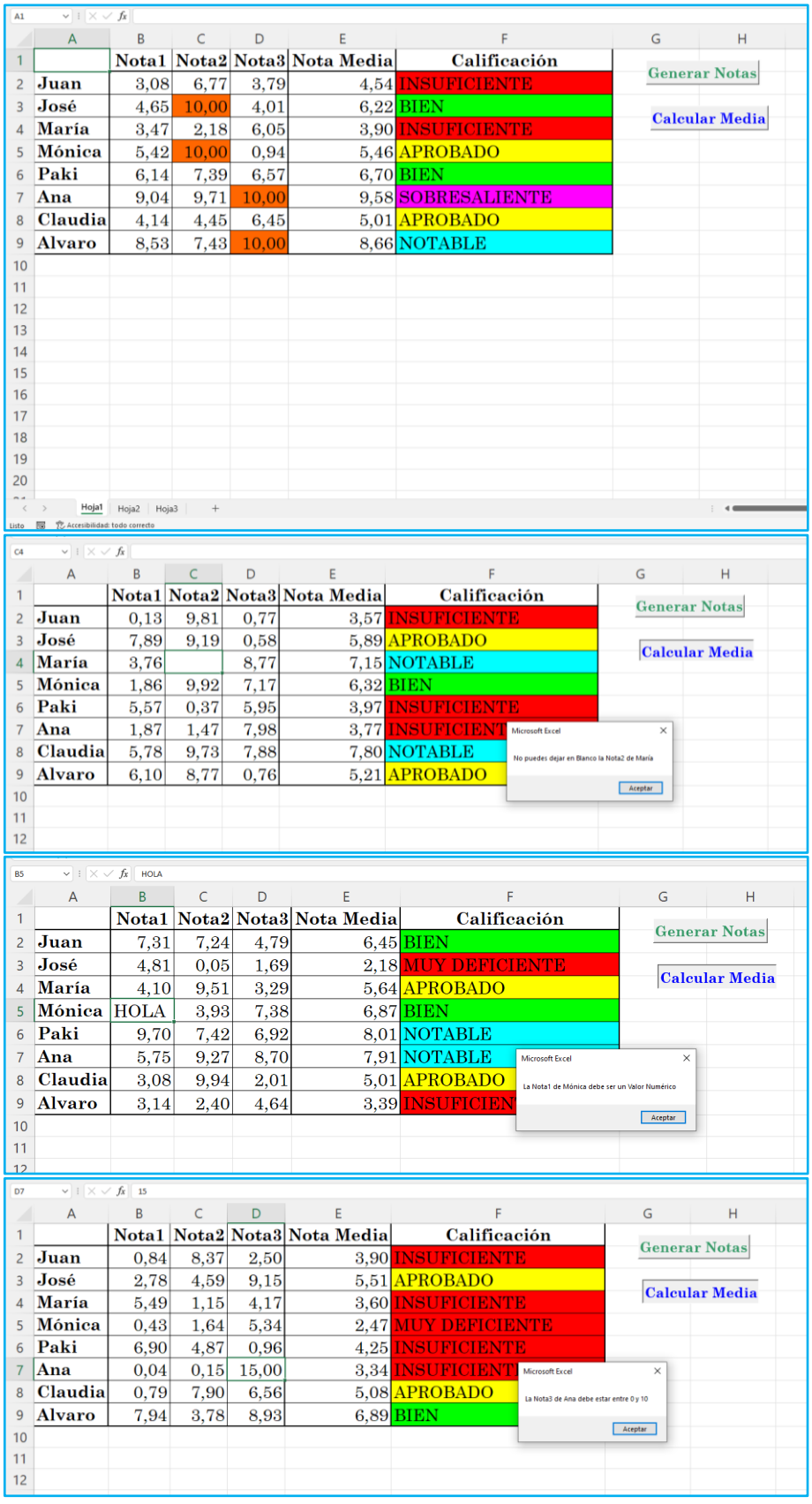
NOTA. Recuerda que la **Propiedad *Rango.Interior.ColorIndex* = *Indice*** rellena el Rango con el índice de Color referenciado.

El **Tercer Ejercicio Propuesto** lo haremos en la ***Hoja1*** y se basa en el Ejercicio ***EJERCICIO0404.xlsm***, con algunas variaciones. Nuestro programa, esta vez, rellenará de forma aleatoria, con **valores entre 0 y 10**, las Notas de cada parcial de cada alumno. Aquellas **Notas igual a 10**, deberán **rellenarse** de un Color distinto al resto.

Crearemos **2 Botones de Formulario**. Un primer Botón se encargará de **Generar las Notas Aleatorias** y, además, calcula la **Nota Media** y la **Calificación**. El segundo Botón únicamente **Calcula la Nota Media**.

Crearemos **2 Procedimientos** (***Sub GenerarNotas()*** y ***Sub CalcularMedia()***), de tal modo que ***Sub GenerarNotas()*** termina llamando (***Call CalcularMedia***) al otro Procedimiento.

Podríamos cambiar alguna nota y sustituirla por un **Texto**, por un **Número Negativo** o **Mayor que 10**, o bien, dejarla **Vacía** y el **Botón de Formulario *Calcular Media*** tendría que detectar el **Error** y anunciarlo.



	Nota1	Nota2	Nota3	Nota Media	Calificación
Juan	3,08	6,77	3,79	4,54	INSUFICIENTE
José	4,65	10,00	4,01	6,22	BIEN
María	3,47	2,18	6,05	3,90	INSUFICIENTE
Mónica	5,42	10,00	0,94	5,46	APROBADO
Paki	6,14	7,39	6,57	6,70	BIEN
Ana	9,04	9,71	10,00	9,58	SOBRESALIENTE
Claudia	4,14	4,45	6,45	5,01	APROBADO
Alvaro	8,53	7,43	10,00	8,66	NOTABLE

Generar Notas

Calcular Media

	Nota1	Nota2	Nota3	Nota Media	Calificación
Juan	0,13	9,81	0,77	3,57	INSUFICIENTE
José	7,89	9,19	0,58	5,89	APROBADO
María	3,76		8,77	7,15	NOTABLE
Mónica	1,86	9,92	7,17	6,32	BIEN
Paki	5,57	0,37	5,95	3,97	INSUFICIENTE
Ana	1,87	1,47	7,98	3,77	INSUFICIENT
Claudia	5,78	9,73	7,88	7,80	NOTABLE
Alvaro	6,10	8,77	0,76	5,21	APROBADO

Microsoft Excel

No puedes dejar en Blanco la Nota2 de María

Aceptar

Generar Notas

Calcular Media

	Nota1	Nota2	Nota3	Nota Media	Calificación
Juan	7,31	7,24	4,79	6,45	BIEN
José	4,81	0,05	1,69	2,18	MUY DEFICIENTE
María	4,10	9,51	3,29	5,64	APROBADO
Mónica	HOLA	3,93	7,38	6,87	BIEN
Paki	9,70	7,42	6,92	8,01	NOTABLE
Ana	5,75	9,27	8,70	7,91	NOTABLE
Claudia	3,08	9,94	2,01	5,01	APROBADO
Alvaro	3,14	2,40	4,64	3,39	INSUFICIEN

Microsoft Excel

La Nota1 de Mónica debe ser un Valor Numérico

Aceptar

Generar Notas

Calcular Media

	Nota1	Nota2	Nota3	Nota Media	Calificación
Juan	0,84	8,37	2,50	3,90	INSUFICIENTE
José	2,78	4,59	9,15	5,51	APROBADO
María	5,49	1,15	4,17	3,60	INSUFICIENTE
Mónica	0,43	1,64	5,34	2,47	MUY DEFICIENTE
Paki	6,90	4,87	0,96	4,25	INSUFICIENTE
Ana	0,04	0,15	15,00	3,34	INSUFICIENT
Claudia	0,79	7,90	6,56	5,08	APROBADO
Alvaro	7,94	3,78	8,93	6,89	BIEN

Microsoft Excel

La Nota3 de Ana debe estar entre 0 y 10

Aceptar

NOTA. La **Función *Rnd*** Devuelve un Valor **Tipo Single** que contiene un **número pseudoaleatorio,** menor que 1 y mayor o igual que cero (**0 <= Rnd < 1**). Antes de llamar a la **Función** usa la **Instrucción *Randomize*** sin argumento

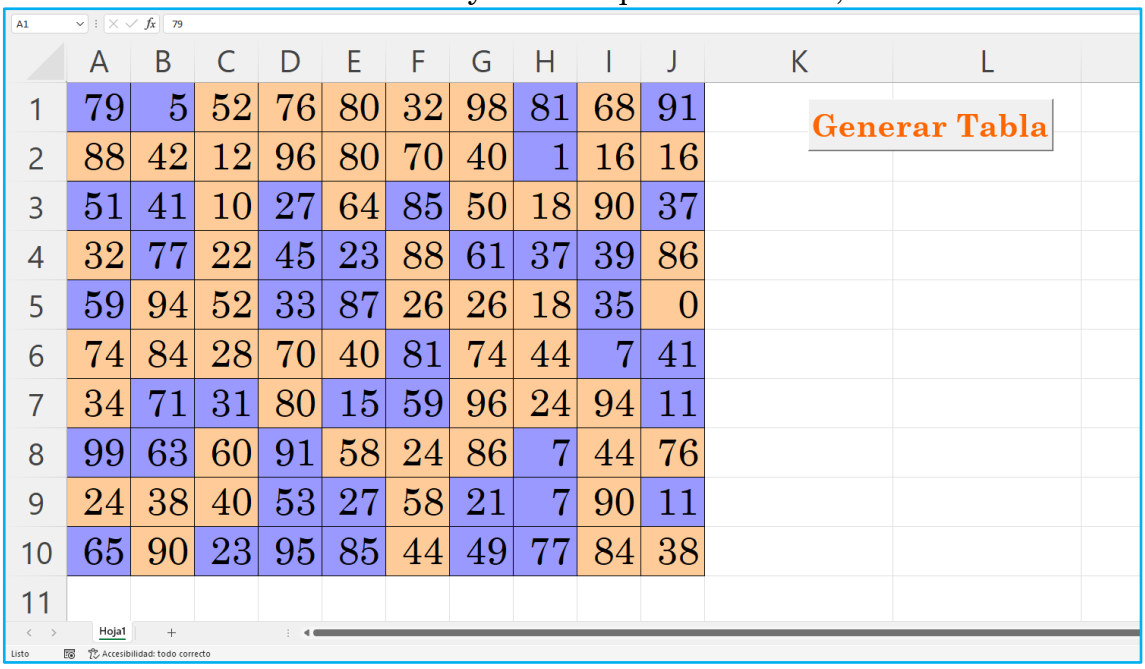
para inicializar el generador de números aleatorios con una inicialización basada en el temporizador del sistema. En el Programa he generado números aleatorios **Menores que 11**, y he añadido una **Instrucción *If … Then*** que permita que algunas Notas puedan ser igual a **10 (con el fondo relleno de un color distinto al resto)**.

Ejercicio 05.02.-
Crea un archivo llamado ***EJERCICIO0502.xlsm*** y guárdalo.

Dentro de este apartado haremos varios ejercicios basados en el uso del **Bucle *For Each … Next***. Este tipo de Estructura permite ejecutar acciones en Bucle mientras se recorre cada uno de los Elementos de la Colección. No tiene un Contador porque depende del número de Objetos que forme la Colección. En cualquier momento se puede forzar la Salida con ***Exit For***.

El **Primer Ejercicio Propuesto** lo haremos en la ***Hoja1*** y consiste en colorear aquellos **valores numéricos que sean Pares\ImPares**, dentro de un Rango dado. Los números los elegiré de forma pseudoaleatoria y luego aplico la búsqueda con ***Each***. Para ello usaré **1 Botón de Formulario.**

Crearé un **Rango de 10x10 Celdas** y a cada una le daré un valor entero entre 0 y 100 (ambos inclusive). A través de un **Bucle *For Each … Next***, voy a generar un número que será el Valor de la Celda activa. En el caso que el número será **Par**, la Celda tendrá un Color de Relleno y en caso que sea **ImPar**, otro diferente.



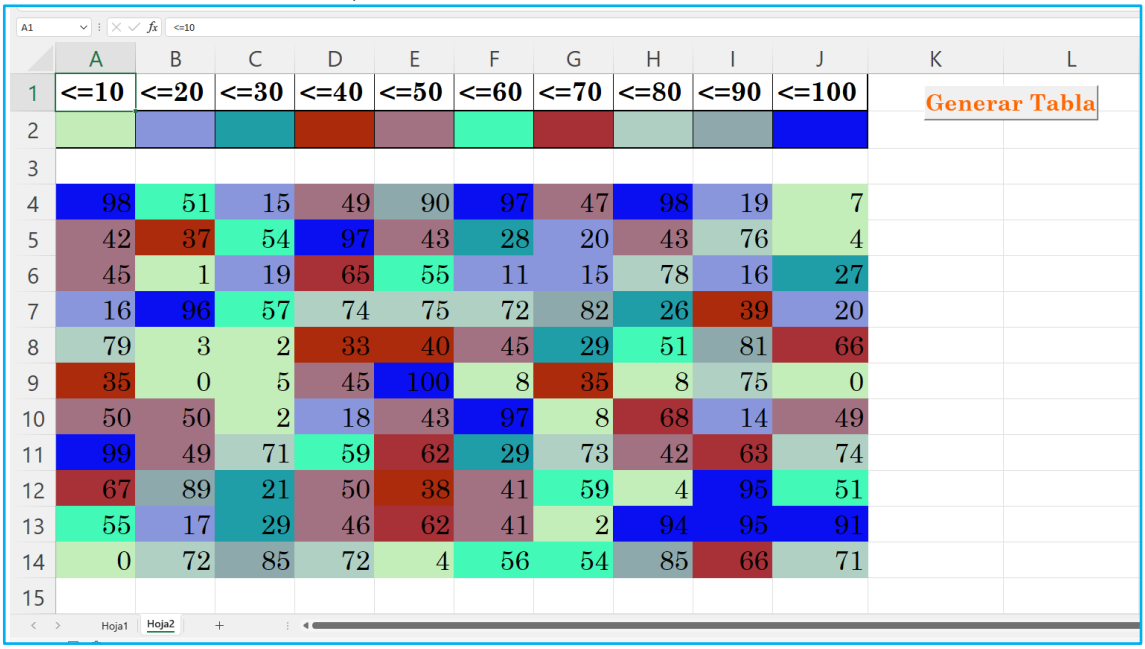
	A	B	C	D	E	F	G	H	I	J
1	79	5	52	76	80	32	98	81	68	91
2	88	42	12	96	80	70	40	1	16	16
3	51	41	10	27	64	85	50	18	90	37
4	32	77	22	45	23	88	61	37	39	86
5	59	94	52	33	87	26	26	18	35	0
6	74	84	28	70	40	81	74	44	7	41
7	34	71	31	80	15	59	96	24	94	11
8	99	63	60	91	58	24	86	7	44	76
9	24	38	40	53	27	58	21	7	90	11
10	65	90	23	95	85	44	49	77	84	38

El **Segundo Ejercicio Propuesto** lo haremos en la ***Hoja2*** y consiste en colorear el relleno de la Celda en función de si el valor de la misma es **=< 10, = 20, … , =< 100**, de un color diferente elegido **peudoaleatoriamente** eligiendo valores entre **0 y 255** por cada parámetro de la **Función *RGB***.

La sintaxis de la Función **RGB** tiene **3 Argumentos** (**Rojo, Verde, Azul**) cuyos Valores son números comprendidos entre **0 y 255**, que representan el componente de cada color básico.

En este ejercicio, asignaremos Colores pseudoaleatorios a cada tramo en el Encabezado (**=< 10, = 20, … , =< 100**) que dibujaremos debajo (en la **Fila 2**).

Luego generaremos la **Tabla** con valores pseudoaleatorios y, según el tramo que coincida en el Encabezado, la Celda deberá tener ese relleno.



<=10	<=20	<=30	<=40	<=50	<=60	<=70	<=80	<=90	<=100
98	51	15	49	90	97	47	98	19	7
42	37	54	97	43	28	20	43	76	4
45	1	19	65	55	11	15	78	16	27
16	96	57	74	75	72	82	26	39	20
79	3	2	33	40	45	29	51	81	66
35	0	5	45	100	8	35	8	75	0
50	50	2	18	43	97	8	68	14	49
99	49	71	59	62	29	73	42	63	74
67	89	21	50	38	41	59	4	95	51
55	17	29	46	62	41	2	94	95	91
0	72	85	72	4	56	54	85	66	71

Aquí el problema será "**capturar**" el Color, ya sea en **Hexadecimal** o en **RGB**. Eso se puede hacer mediante las siguientes **Funciones**.

En la **Primera Función**, el Parámetro es la Celda de la que se pretende obtener su **Indice de Color en Hexadecimal**. Usaremos la **Propiedad *Interior.Color*** que devuelve o establece el Color principal del **Objeto (Rango)** y la **Función *Hex*** que devuelve un **Valor Tipo *String*** que representa el **Valor Hexadecimal** de un Número dado.

```
Function IndiceColorHex(Celda As Range) As String
Dim Color As String

    Color = Right("000000" & Hex(Celda.Interior.Color), 6)
    IndiceColorHex = Right(Color, 2) & Mid(Color, 3, 2) & Left(Color, 2)

End Function
```

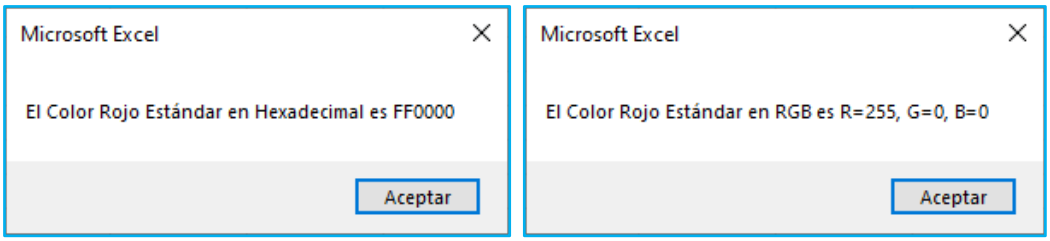
En la **Segunda Función**, el Parámetro es la Celda de la que se pretende obtener su **Indice de Color en RGB (3 Valores por separado)**. También aquí usaremos la **Propiedad *Interior.Color***.

```
Function IndiceColorRGB(Celda As Range) As String
Dim Color As Long
Dim R As Long
Dim G As Long
Dim B As Long

    Color = Celda.Interior.Color
    R = Color Mod 256
    G = Color \ 256 Mod 256
    B = Color \ 65536 Mod 256
```

```
    IndiceColorRGB = "R=" & R & ", G=" & G & ", B=" & B

End Function
```

Para comprobarlo, he dado el **Relleno** de **Color Estándar Rojo** a una Celda y he ejecutado el Siguiente Código sobre la Celda Seleccionada.

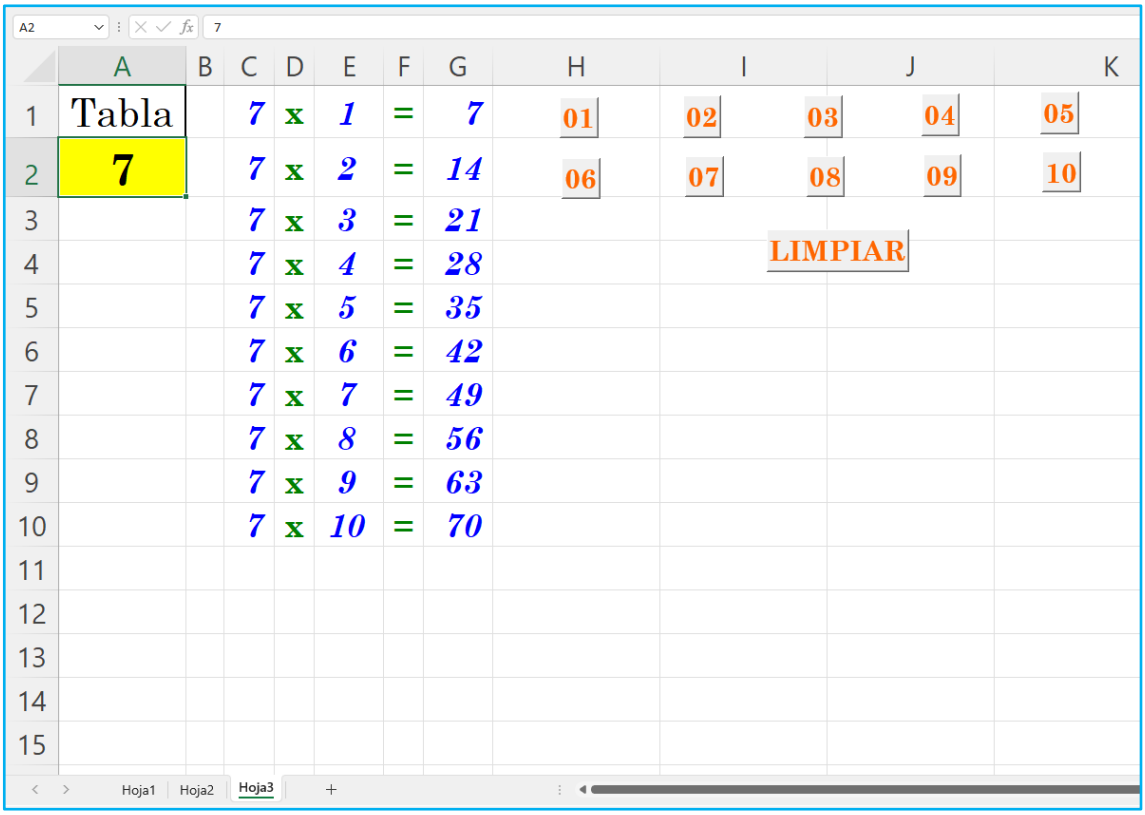
```
Sub BuscarColor()

    MsgBox "El Color Rojo Estándar en Hexadecimal es " & IndiceColorHex(Selection)
    MsgBox "El Color Rojo Estándar en RGB es " & IndiceColorRGB(Selection)

End Sub
```



Yo usé el siguiente Procedimiento en mi Código.

```
Sub ExtraerIndiceRGB()
Dim Color As Long
Dim R As Long
Dim G As Long
Dim B As Long

    Color = Rango.Interior.Color
    R = C Mod 256
    G = C \ 256 Mod 256
    B = C \ 65536 Mod 256

End Sub
```

El **Tercer Ejercicio Propuesto** se basa en realizar la **Tabla de Multiplicar del Número Pulsado**. Tendremos una serie de **Botones de Formulario** que representan los **Números del 1 al 10**, además de otro que **Limpia la Tabla**. Lo haremos en ***Hoja3*** procurando usar la **Estructura *For … Each … Next***, en la medida de lo posible.



Aprovecharemos este Ejercicio para introducir el uso de la **Instrucción *With ... End With*** que permite ejecutar una serie de instrucciones que hacen **referencia repetidamente a un único objeto o estructura**, utilizando una sintaxis simplificada y más legible al acceder, por ejemplo, a las Propiedades de un Objeto, sin más que anteceder por un **punto (.)** cada una de ellas. A menos que el bloque de ***With ... End With*** contenga un **Bucle**, las instrucciones se ejecutan una sola vez e incluso se puede anidar diferentes tipos de estructuras de control dentro.

Por ejemplo, para cambiar varias **Propiedades** de la Celda K11, se podría escribir el siguiente Código.

```
Sub UsoWith()

Application.ThisWorkbook.Sheets(Hoja3.Name).Activate

   With Cells(11, "K")
      .Value = "Curso Macros VBA de MS Excel"
      .Font.Bold = True
      .Font.Italic = True
      .Font.Name = "Tahoma"
      .Font.ColorIndex = 3
      .HorizontalAlignment = xlCenter
      .VerticalAlignment = xlCenter
   End With

End Sub
```

Se pide utilizar la **Instrucción *With ... End With*** para dar **Formato a la Tabla** aplicando **Negrita, Cursiva, Alineación Vertical\Horizontal, Color de Fuente**, etc.

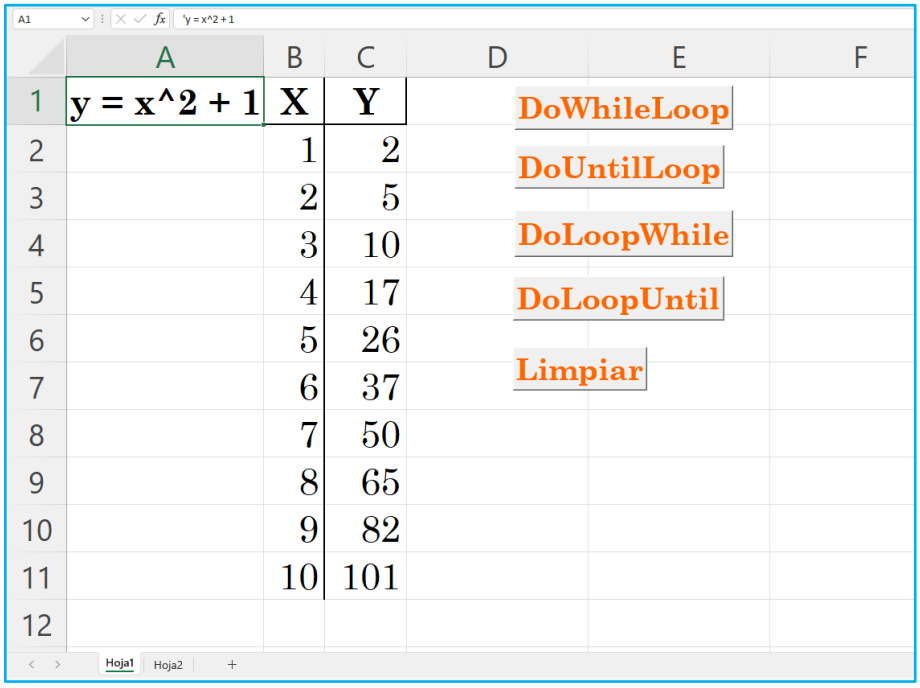
Ejercicio 05.03.-
Crea un archivo llamado ***EJERCICIO0503.xlsm*** y guárdalo.

Dentro de este apartado haremos varios ejercicios basados en el uso del **Bucle *Do ... Loop (Until, While)* y *While ... Wend***.

La **Estructura *While*** dispone de una Condición para controlar la secuencia de Repeticiones. Si antes, con el ***For ... Next*** recorríamos el Bucle un número de veces establecido (salvo que se forzase la salida), ahora, con ***Do While\Until*** el Bucle se repite **Mientras o Hasta** que se Cumpla una **Condición (*True\False*)**.

El **Primer Ejercicio Propuesto** se basa en escribir una **Tabla de Valores** que recoja el resultado de la **Función $y = x^2 + 1$** para los **Primeros 10 Números Enteros Positivos**, de forma **ascendente** al pulsar un **Botón de Formulario**. Lo haremos para los **siguientes 4 casos** y **el resultado deberá ser el mismo**. La Tabla deberá estar en la ***Hoja1*** de ***EJERCICIO0503.xlsm***. Usaremos **4 Botones de Formulario** para la ejecución.

- ***Do While Loop***
- ***Do Until Loop***
- ***Do Loop While***
- ***Do Loop Until***

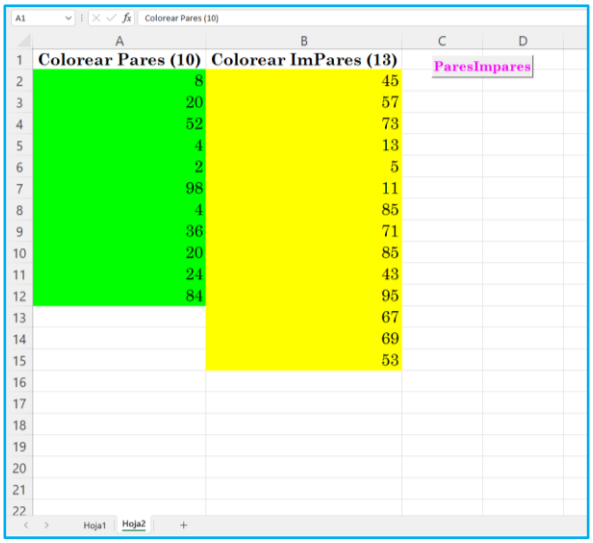


El **Segundo Ejercicio Propuesto** consiste en hacer una Tabla con 2 Columnas, una junta a la otra. En la primera, escribiremos los números **Pares** pseudoaleatorios (entre **0 y 100**) que resulten de utilizar un Contador desde **1 a 25**. En la segunda Columna irán los números que han resultado ser **ImPares**. Unos y otros tendrán un relleno de Color de Fondo de la Celda.

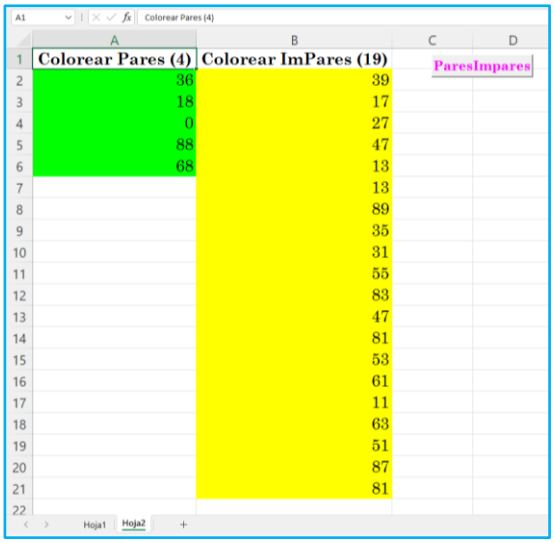
En el **Encabezado** deberá aparecer el **Número de Pares (o ImPares)** que hay en la correspondiente columna. Este **Segundo Ejercicio Propuesto** se ejecutará

en la ***Hoja2*** del archivo y usaremos la Estructura ***While … Wend*** con la Condición ***Contador < 25***.

Dispondrá de un **Botón de Formulario** que, cada vez que se pulsa, realiza lo que se pide, al tiempo que **Auto Ajusta** las 2 Columnas a su contenido y **borra** el resultado anterior.



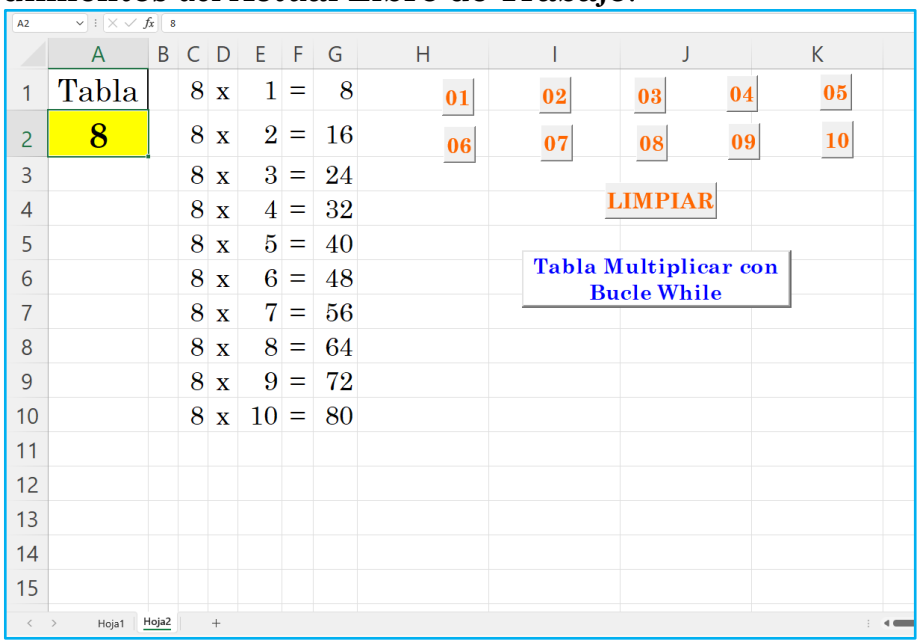
Colorear Pares (10)	Colorear ImPares (13)	ParesImpares
8	45	
20	57	
52	73	
4	13	
2	5	
98	11	
4	85	
36	71	
20	85	
24	43	
84	95	
	67	
	69	
	53	



Colorear Pares (4)	Colorear ImPares (19)	ParesImpares
36	39	
18	17	
0	27	
88	47	
68	13	
	13	
	89	
	35	
	31	
	55	
	83	
	47	
	81	
	53	
	61	
	11	
	63	
	51	
	87	
	81	

Ejercicio 05.04.-
Crea un archivo llamado ***EJERCICIO0504.xlsm*** y guárdalo.

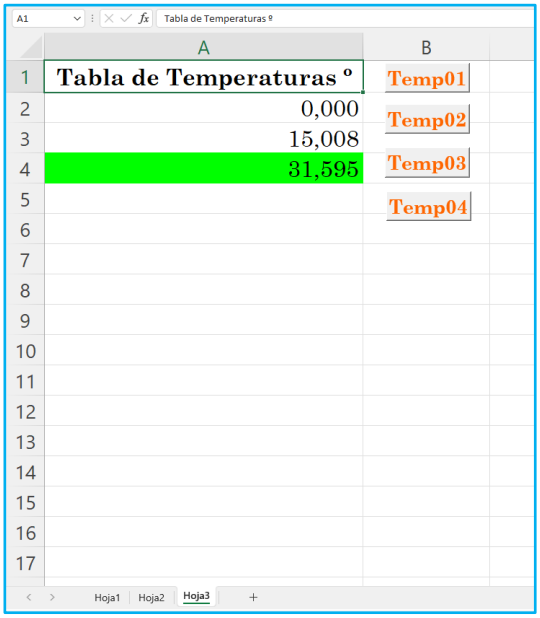
Para este **Ejercicio Propuesto** retomaremos el ejemplo de la **Tabla de Multiplicar** del ***EJERCICIO0502.xlsm*** y modificaremos el Código para que funcione con la **Estructura *While***, eliminando la **Estructura *For*** y todas aquellas **Variables** que no fueran necesarias. Se ejecutará en la ***Hoja1*** de ***EJERCICIO0504.xlsm***. Asegúrate que los **Botones de Formulario**, llaman a los **Procedimientos** del **Actual Libro de Trabajo**.

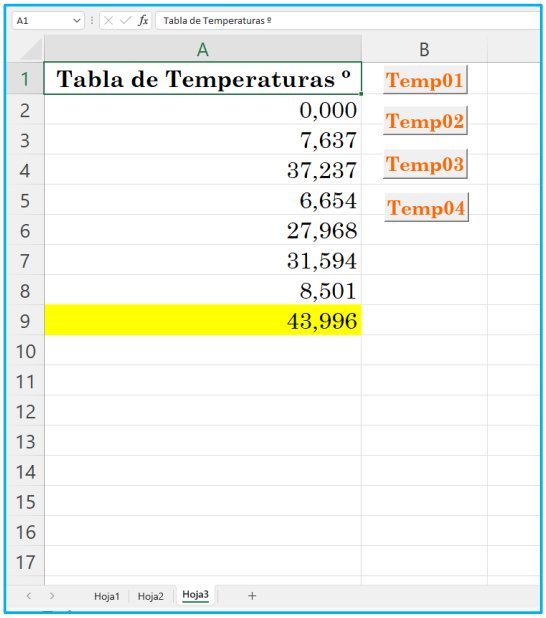


Ejercicio 05.05.-
Crea un archivo llamado ***EJERCICIO0505.xlsm*** y guárdalo.

En este **Ejercicio Propuesto** lo que buscamos es que el Programa calcule una serie de **Temperaturas** comprendidas entre **0º y 45º**, de forma **pseudoaleatoria**, y las irá escribiendo bajo el **Encabezado**, pero, en cuanto detecte un **Valor Superior a 40**, deberá escribirlo con **relleno Amarillo** y finalizar la Ejecución. Dentro del mismo Bucle, si el Valor estuviera comprendido entre **30 y 35**, deberá escribirlo con **relleno Verde** y **salir del Bucle**. El primer valor de Temperatura de la Tabla será siempre **0º**. Lo harás en la ***Hoja1***. Cada Botón corresponde a cada una de las Estructuras siguientes.

- ***Do While Loop***
- ***Do Until Loop***
- ***Do Loop While***
- ***Do Loop Until***





06.- VARIABLES, CONSTANTES Y MATRICES

Ejercicio 06.01.-

Una **Variable** o una **Constante** es un Objeto que usamos en nuestros programas para **almacenar un Valor**. Esto hace que el ordenador reserve un **espacio de memoria** para poder guardar ese contenido. Lo que guarda la Variable tiene que ser algún **Tipo de Dato** (https://learn.microsoft.com/es-es/office/vba/language/reference/user-interface-help/data-type-summary) que admita VBA Excel.

Microsoft define los **Tipos de Dato** como la característica de una Variable que determina qué Tipo de Dato puede contener, como, por ejemplo, ***Byte, Booleano, Entero, Long, Moneda, Decimal, Single, Doble, Date, String, Objeto, Variant*** (predeterminado) y otros Tipos que Defina el Usuario, así como Tipos específicos de Objetos.

Las **Constantes**, a diferencia de las **Variables**, son **Valores** que consideramos **Fijos** a lo largo de la Ejecución del Programa y las declaramos de esa manera.

Sabemos que **no es estrictamente obligatorio Declarar una Variable** antes de Usarla, salvo que hayamos activado la Opción de Requerir dicha Declaración (***Option Explicit***). Mi consejo es que las Declaremos previamente, **siempre**.

Toda variable conserva el Valor asignado hasta que **pierde el ámbito de actuación**. Al comienzo de cualquier Procedimiento, las Variables se inicializan (las **numéricas lo hacen a 0**, las cadena de longitud variable se inicializan en una **cadena de longitud cero ("")**, las cadenas de longitud fija se rellena con el carácter representado por el código de caracteres **ASCII 0** o ***Chr(0)***, las de **Tipo *Variant*** se inicializan en ***Empty***).

Cuando declaramos una **Variable de Tipo *Objeto***, el espacio se reserva en memoria, pero su valor se establece en ***Nothing*** hasta que se le asigna una Referencia de Objeto mediante la instrucción ***Set***.

Una **Variable de Nivel de Procedimiento** declarada con la instrucción ***Dim*** conserva un Valor hasta que el Procedimiento termina de ejecutarse. Si el Procedimiento llama a otros Procedimientos, la Variable conserva su valor mientras esos procedimientos se estén ejecutando.

Si declaramos una **Variable de Nivel de Procedimiento** con la **Instrucción *Static***, la Variable conserva su valor siempre y cuando el código se ejecute en cualquier Módulo del Proyecto. Cuando todo el código ha terminado de ejecutarse, la Variable pierde su ámbito y su valor. Su duración (tiempo que conserva el Valor) es la misma que la de la variable de Nivel de Módulo.

Usaremos la **Instrucción *Public*** para declarar Variables a nivel del Proyecto. Las **Variables Públicas** pueden usarse en cualquier procedimiento del proyecto. Si una variable pública se declara en un **Módulo estándar** o **Módulo de Clase**,

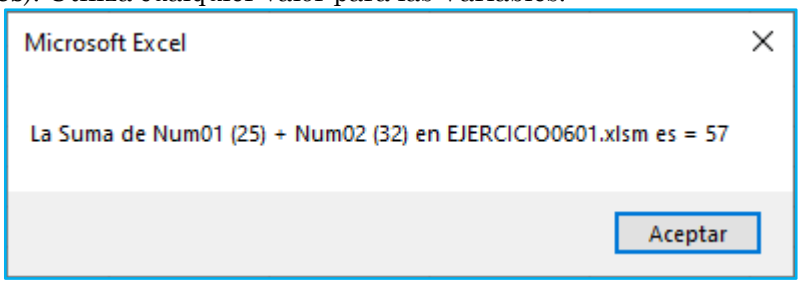
también se puede usar en cualquier proyecto que haga referencia al proyecto donde se declara la Variable Pública.

A continuación, te muestro un resumen de los Tipos de Datos más utilizados.

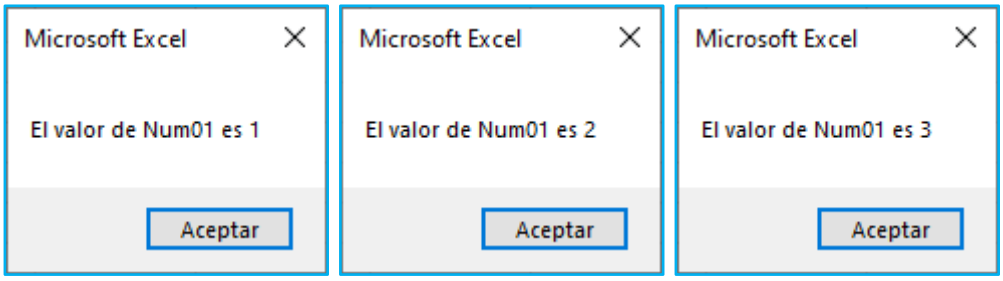
- ***Variant***: puede contener cualquier tipo de dato como, por ejemplo, valores numéricos, secuencias de caracteres y valores actuales o fechas. Se indicará automáticamente una variante si no se declara explícitamente el tipo de dato.
- ***Integer***: se utiliza con números enteros que tienen un valor de entre -32 768 y 32 767.
- ***Long***: se utiliza con números enteros que van desde -2 147 483 648 hasta 2 147 483 647.
- ***Double***: incluye todos los números con punto flotante de precisión doble desde más/menos 1,79 * 10^308.
- ***Boolean***: se trata de variables Verdaderas\Falsas. Se muestran como True\False.
- ***String***: comprende secuencias de caracteres de longitud fija o variable.
- ***Date***: se utiliza para datos de Fecha y Hora. Se puede introducir horas desde las 00:00:00 (media noche) hasta las 23:59:59.9999999. El formato de la fecha es M/D/AAA o AAAA-MM-DD, pero podemos modificarlo cambiando su formato.

Para hacer este **Ejercicio Propuesto**, comenzaremos por crear un archivo llamado ***EJERCICIO0601.xlsm*** y lo guardamos. Vamos crear **2 Módulos** y nos aseguramos que tenemos ***Hoja1*** y ***Hoja2***. Probaremos a crear diferentes Variables (Diferentes Tipos de Dato), con diferentes ámbitos.

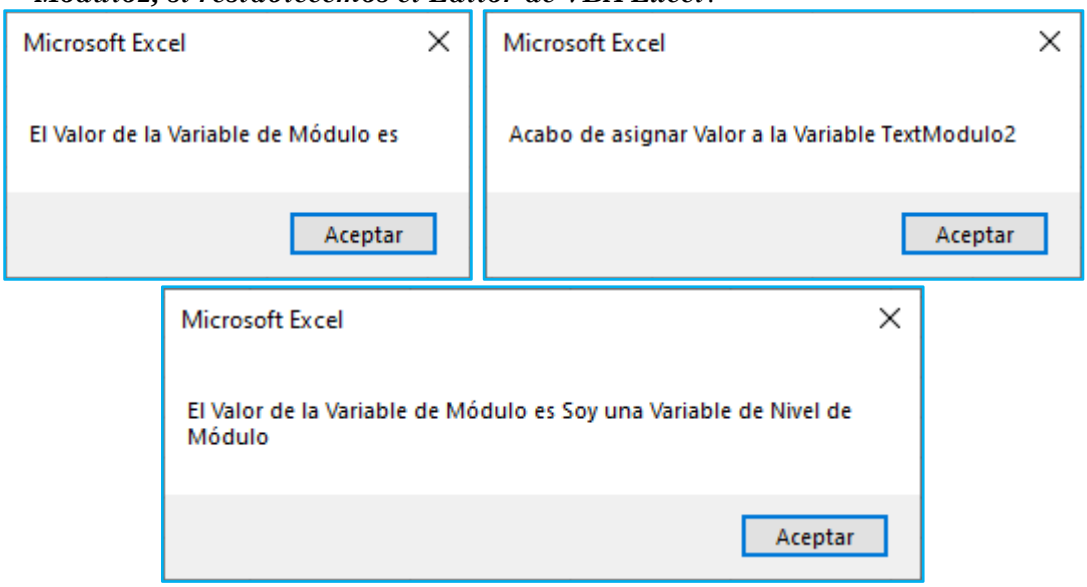
1. Crearemos un **Procedimiento** llamado ***Var01***, dentro de ***Módulo1*** que declare las Variables **Locales** Numéricas ***Tipo Double***, ***Num01*** y ***Num02***, luego le damos un Valor inicial y presentamos la Sumatoria de ambas con la frase siguiente (**usando las Variables**). Utiliza cualquier valor para las Variables.



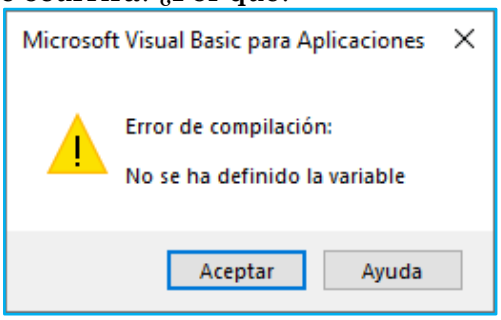
2. Ahora crearemos un **Procedimiento** llamado ***Var01*** dentro de ***Módulo2*** que declara, como **Local Estática**, la Variable Numérica ***Tipo Double***, ***Num01*** y cuyo Valor es ***Num01 = Num01 + 1***. Luego presentaremos el Valor de la Variable con la frase siguiente (**usando la Variable para cualquier valor**). ¿***Qué valor tendrá Num01 dentro de Módulo2, si restablecemos el Editor de VBA Excel***?



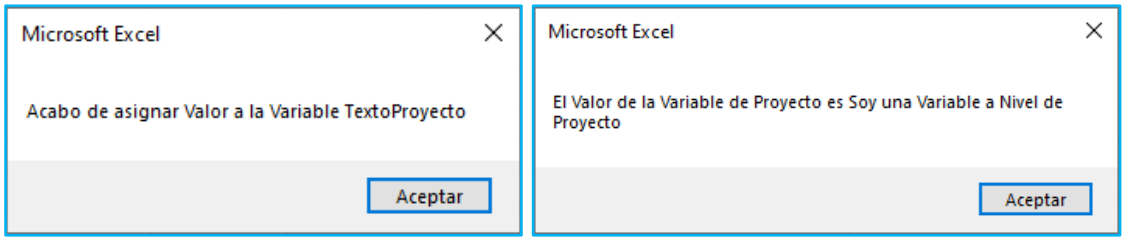
3. Ahora declararemos la Variable ***TextoModulo2*** (a nivel de ***Módulo2***). A continuación, crearemos un **Procedimiento** llamado ***Var02*** dentro de **Módulo2** que le da el valor "**Soy una Variable de Nivel de Módulo**" (ya sabemos de qué Tipo de Dato se trata) a ***TextoModulo2***. Seguimos creando otro **Procedimiento** llamado ***Var03*** que mostrará el mensaje ***"El Valor de la Variable de Módulo es " & TextoModulo2***. Llegado a este punto, ejecutamos la **Macro** ***Var03***, luego la **Macro** ***Var02*** y, de nuevo, la **Macro** ***Var03***. El resultado será lo que vemos a continuación. ¿***Qué valor tendrá TextModulo2, Módulo2, si restablecemos el Editor de VBA Excel***?



4. Ahora creamos el **Procedimiento** ***Var03*** en el ***Módulo1*** con el que intentaremos leer la Variable ***TextoModulo2*** con el mensaje ***"El Valor de la Variable de Módulo es " & TextoModulo2***. ¿**Qué ocurrirá**? ¿**Por qué**?



5. Ahora declararemos la Variable ***TextoProyecto*** (a nivel de Proyecto). A continuación, crearemos un **Procedimiento** llamado ***Var02*** dentro de ***Módulo1*** que le da el valor "***Soy una Variable a Nivel de Proyecto***" a ***TextoProyecto*** y muestra el Mensaje "***Acabo de asignar Valor a la Variable TextoProyecto***". Seguidamente, crearemos el **Procedimiento** ***Var04*** que llamará al **Procedimiento** ***Var02*** del ***Módulo1*** y mostrará el Mensaje ***"El Valor de la Variable de Proyecto es " & TextoProyecto***.


Microsoft Excel
Acabo de asignar Valor a la Variable TextoProyecto
Aceptar
Microsoft Excel
El Valor de la Variable de Proyecto es Soy una Variable a Nivel de Proyecto
Aceptar

Ejercicio 06.02.-
Cuando declaramos una **Constante** en VBA Excel, lo que estamos haciendo es asignar un **nombre significativo** a un valor, como por ejemplo **Pi = 3,1416**. Usamos la Instrucción ***Const*** para Declarar una Constante y establecer su valor. Las Constantes pueden facilitar la lectura y el mantenimiento del código VBA Excel y, una vez declaradas, no se podrán ni modificar, ni asignarles un nuevo valor.

De forma predeterminada, salvo que se diga lo contrario, al crear una **Constante**, ésta es **Privada**. En los Procedimientos, las Constantes son siempre Privadas por lo que **su visibilidad no puede cambiar**. En los **Módulos Estándar**, la **visibilidad** predeterminada se puede cambiar mediante la Instrucción ***Public***. En los **Módulos de Clase**, en cambio, las Constantes solo pueden ser **Privadas** y su visibilidad no se puede cambiar mediante la Instrucción ***Public***.

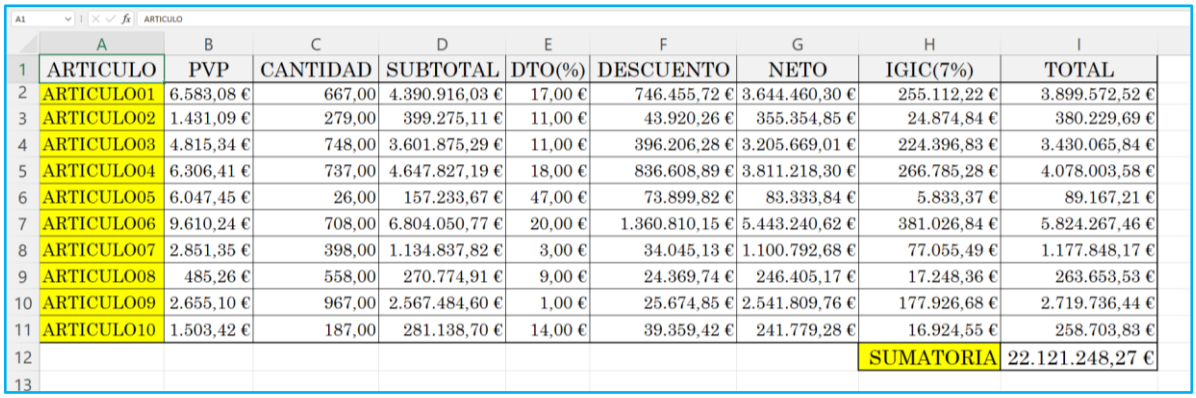
Para **combinar varias Declaraciones de Constantes** en la misma línea, separaremos cada asignación por una **Coma ","**. En este caso, si usamos la Coma "," para Declararlas, la **Instrucción *Public*** o ***Private***, se aplica a todas ellas.

No podemos usar una Variable para asignar un Valor a una Constante, ni tampoco Funciones definidas por el usuario (UDF), ni Funciones intrínsecas de VBA, como asignación (total o parcial) del valor de una Constante.

Podemos declarar una Constante dentro de un Procedimiento o en la parte superior de un Módulo, en la sección Declaraciones.

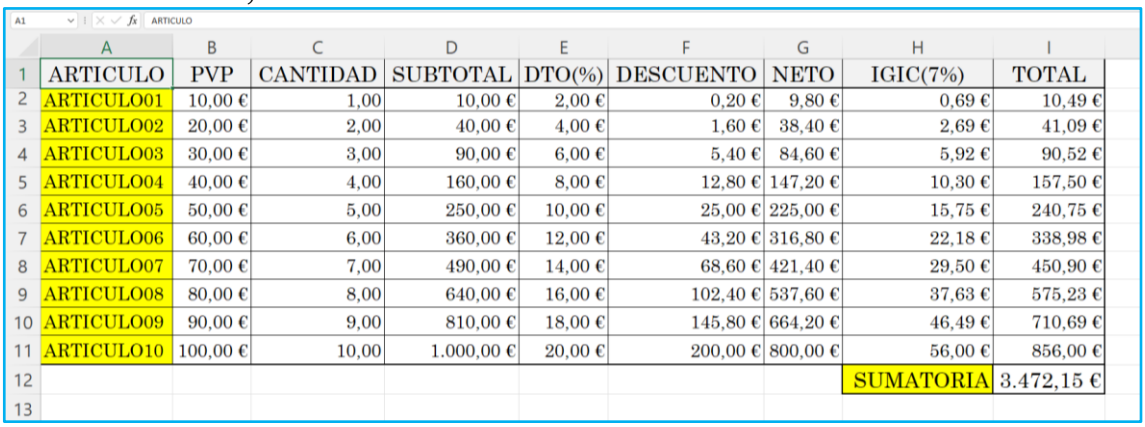
Como hacemos con las Variables, podemos Declarar las Constantes como Tipo de Dato ***Boolean, Byte, Integer, Long, Currency, Single, Double, Date, String*** *o* ***Variant***.

Crearemos un Nuevo archivo llamado ***EJERCICIO0602.xlsm*** para hacer el **siguiente Ejercicio Propuesto**. Se trata de hacer el **Procedimiento *ConstanteIGIC***, en el ***Módulo1***, que presente todo lo que se ve a continuación. El **Impuesto (IGIC = 7%)** será una **Constante** de **Tipo *Currency*** declarada a **Nivel de Proyecto en el Módulo2**. **PVP** es un valor pseudoaleatorio de **0 a 10.000 que se asigna directamente a la Celda**. **Cantidad**, en cambio, se asigna primero a una **Variable Entera** y ésta a la **Celda**, y su valor está entre **0 y 1.000**. el **Descuento** es una **Variable Entera** pseudoaleatoria con Valor entre **0 y 50**. **NETO** es el resultado de aplicar el **Descuento** al **SubTotal**. Incluiremos un **Bucle** que haga el **Cálculo por cada registro**, cambiando la Cantidad y el Descuento **en cada iteración**. Al final, obtendremos la **Sumatoria cuya Celda obtiene el Valor de una Constante Privada a Nivel de Módulo Tipo String** (***Private Const Resultado As String * 9 = "SUMATORIA"***) cuyo tamaño está limitado a **9 Caracteres**. Cada vez que ejecutemos la Macro tendremos **nuevo PVP** y **nueva Cantidad**.



	A	B	C	D	E	F	G	H	I
1	ARTICULO	PVP	CANTIDAD	SUBTOTAL	DTO(%)	DESCUENTO	NETO	IGIC(7%)	TOTAL
2	ARTICULO01	6.583,08 €	667,00	4.390.916,03 €	17,00 €	746.455,72 €	3.644.460,30 €	255.112,22 €	3.899.572,52 €
3	ARTICULO02	1.431,09 €	279,00	399.275,11 €	11,00 €	43.920,26 €	355.354,85 €	24.874,84 €	380.229,69 €
4	ARTICULO03	4.815,34 €	748,00	3.601.875,29 €	11,00 €	396.206,28 €	3.205.669,01 €	224.396,83 €	3.430.065,84 €
5	ARTICULO04	6.306,41 €	737,00	4.647.827,19 €	18,00 €	836.608,89 €	3.811.218,30 €	266.785,28 €	4.078.003,58 €
6	ARTICULO05	6.047,45 €	26,00	157.233,67 €	47,00 €	73.899,82 €	83.333,84 €	5.833,37 €	89.167,21 €
7	ARTICULO06	9.610,24 €	708,00	6.804.050,77 €	20,00 €	1.360.810,15 €	5.443.240,62 €	381.026,84 €	5.824.267,46 €
8	ARTICULO07	2.851,35 €	398,00	1.134.837,82 €	3,00 €	34.045,13 €	1.100.792,68 €	77.055,49 €	1.177.848,17 €
9	ARTICULO08	485,26 €	558,00	270.774,91 €	9,00 €	24.369,74 €	246.405,17 €	17.248,36 €	263.653,53 €
10	ARTICULO09	2.655,10 €	967,00	2.567.484,60 €	1,00 €	25.674,85 €	2.541.809,76 €	177.926,68 €	2.719.736,44 €
11	ARTICULO10	1.503,42 €	187,00	281.138,70 €	14,00 €	39.359,42 €	241.779,28 €	16.924,55 €	258.703,83 €
12								SUMATORIA	22.121.248,27 €
13									

Comprobaremos que el programa funciona, creando una variación del Procedimiento (llámalo ***ConstanteIGIC2***) que registre los siguientes **PVP**, **Cantidades** y **Descuentos** por cada Artículo. El Resultado de la **Sumatoria** deberá darte **3.472,15€.**



	A	B	C	D	E	F	G	H	I
1	ARTICULO	PVP	CANTIDAD	SUBTOTAL	DTO(%)	DESCUENTO	NETO	IGIC(7%)	TOTAL
2	ARTICULO01	10,00 €	1,00	10,00 €	2,00 €	0,20 €	9,80 €	0,69 €	10,49 €
3	ARTICULO02	20,00 €	2,00	40,00 €	4,00 €	1,60 €	38,40 €	2,69 €	41,09 €
4	ARTICULO03	30,00 €	3,00	90,00 €	6,00 €	5,40 €	84,60 €	5,92 €	90,52 €
5	ARTICULO04	40,00 €	4,00	160,00 €	8,00 €	12,80 €	147,20 €	10,30 €	157,50 €
6	ARTICULO05	50,00 €	5,00	250,00 €	10,00 €	25,00 €	225,00 €	15,75 €	240,75 €
7	ARTICULO06	60,00 €	6,00	360,00 €	12,00 €	43,20 €	316,80 €	22,18 €	338,98 €
8	ARTICULO07	70,00 €	7,00	490,00 €	14,00 €	68,60 €	421,40 €	29,50 €	450,90 €
9	ARTICULO08	80,00 €	8,00	640,00 €	16,00 €	102,40 €	537,60 €	37,63 €	575,23 €
10	ARTICULO09	90,00 €	9,00	810,00 €	18,00 €	145,80 €	664,20 €	46,49 €	710,69 €
11	ARTICULO10	100,00 €	10,00	1.000,00 €	20,00 €	200,00 €	800,00 €	56,00 €	856,00 €
12								SUMATORIA	3.472,15 €
13									

Ejercicio 06.03.-

Podemos usar las **Matrices** cuando disponemos de un conjunto de valores del mismo **Tipo de Dato** que queramos almacenar en una única Variable que tenga múltiples compartimentos donde guardar los Datos. Usaremos un mismo nombre para el conjunto de Datos y nos podremos referir a la Matriz en su conjunto o bien a cada valor individual almacenado dentro de la misma.

Por ejemplo, si quiero guardar las notas parciales de todo el alumnado que están estudiando Macros de VBA Excel, podría crear una Variable por cada alumno y por cada parcial o, sencillamente, crear una Matriz cuya dimensión me permita almacenar todos estos datos.

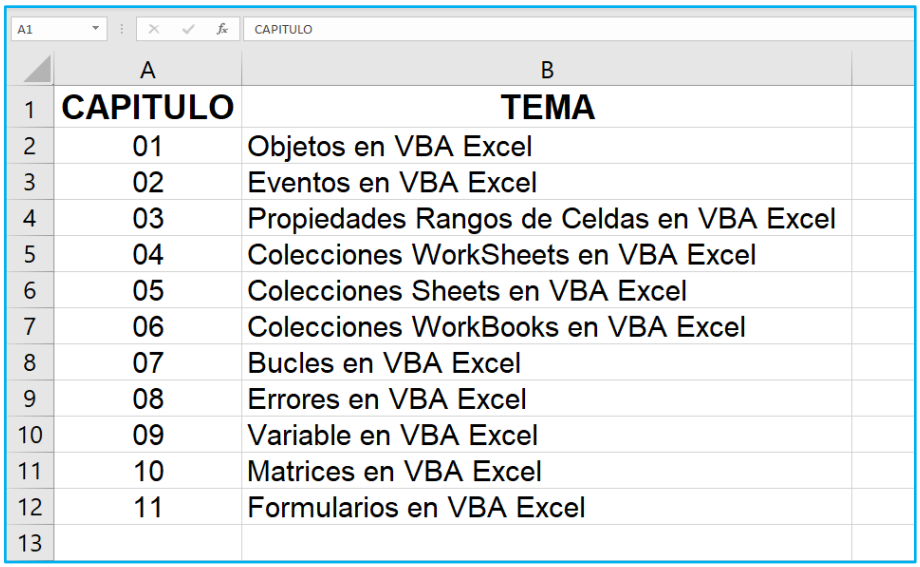
Las Matrices se declaran del mismo modo que cualquier otra Variable (usando las **Instrucciones *Dim***, ***Static***, ***Private*** o ***Public***). La diferencia entre las variables escalares (las que no son matrices) y las variables de matriz es que, normalmente, se debe especificar el tamaño de la Matriz. Las Matrices pueden tener un Tamaño especificado (**Matriz de Tamaño fijo** o **Estática**), o bien, tener un Tamaño que pueda cambiarse durante la ejecución de un programa (**Matriz Dinámica**).

Crea un archivo llamado ***EJERCICIO0603.xlsm*** y guárdalo.

Vamos a empezar con un ejemplo de **Matriz Estática** (no cambia de Tamaño) **Unidimensional (Vector)**. Imaginemos que vamos a usar Variables para guardar y enumerar varios Temas de este Curso en un Procedimiento. Lo haré dentro del ***Módulo1*** en la ***Hoja1***.

```
Sub TemasVBAVariables()
Dim i As Integer
Dim T1 As String, T2 As String, T3 As String, T4 As String, T5 As String
Dim T6 As String, T7 As String, T8 As String, T9 As String, T10 As String, T11 As String

Application.ThisWorkbook.Sheets(Hoja1.Name).Activate

   T1 = "Objetos en VBA Excel"
   T2 = "Eventos en VBA Excel"
   T3 = "Propiedades Rangos de Celdas en VBA Excel"
   T4 = "Colecciones WorkSheets en VBA Excel"
   T5 = "Colecciones Sheets en VBA Excel"
   T6 = "Colecciones WorkBooks en VBA Excel"
   T7 = "Bucles en VBA Excel"
   T8 = "Errores en VBA Excel"
   T9 = "Variable en VBA Excel"
   T10 = "Matrices en VBA Excel"
   T11 = "Formularios en VBA Excel"

   For i = 1 To 11
      Cells(i + 1, 1) = i
   Next i

   Range("A1").Value = "CAPITULO"
   Range("B1").Value = "TEMA"

   Cells(2, 2) = T1
```

```
    Cells(3, 2) = T2
    Cells(4, 2) = T3
    Cells(5, 2) = T4
    Cells(6, 2) = T5
    Cells(7, 2) = T6
    Cells(8, 2) = T7
    Cells(9, 2) = T8
    Cells(10, 2) = T9
    Cells(11, 2) = T10
    Cells(12, 2) = T11

End Sub
```



	A	B
1	CAPITULO	TEMA
2	01	Objetos en VBA Excel
3	02	Eventos en VBA Excel
4	03	Propiedades Rangos de Celdas en VBA Excel
5	04	Colecciones WorkSheets en VBA Excel
6	05	Colecciones Sheets en VBA Excel
7	06	Colecciones WorkBooks en VBA Excel
8	07	Bucles en VBA Excel
9	08	Errores en VBA Excel
10	09	Variable en VBA Excel
11	10	Matrices en VBA Excel
12	11	Formularios en VBA Excel
13		

Nuestra intención es hacer lo mismo, pero usando una **Matriz**, en lugar de crear las **Variables T**.

```
Sub TemasVBAMatrices()
Dim i As Integer
Dim TemasVBA(1 To 11) As String

Application.ThisWorkbook.Sheets(Hoja1.Name).Activate

    Range("A1").Value = "CAPITULO"
    Range("B1").Value = "TEMA"

    TemasVBA(1) = "Objetos en VBA Excel"
    TemasVBA(2) = "Eventos en VBA Excel"
    TemasVBA(3) = "Propiedades Rangos de Celdas en VBA Excel"
    TemasVBA(4) = "Colecciones WorkSheets en VBA Excel"
    TemasVBA(5) = "Colecciones Sheets en VBA Excel"
    TemasVBA(6) = "Colecciones WorkBooks en VBA Excel"
    TemasVBA(7) = "Bucles en VBA Excel"
    TemasVBA(8) = "Errores en VBA Excel"
    TemasVBA(9) = "Variable en VBA Excel"
    TemasVBA(10) = "Matrices en VBA Excel"
    TemasVBA(11) = "Formularios en VBA Excel"

    For i = 1 To 11
        Cells(i + 1, 1) = i
```

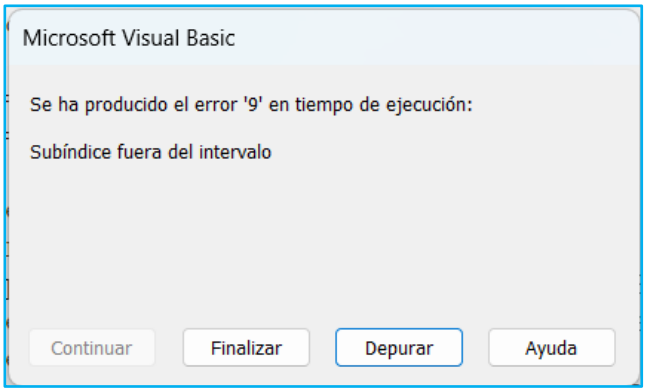
```
    Cells(i + 1, 2) = TemasVBA(i)
  Next i

End Sub
```

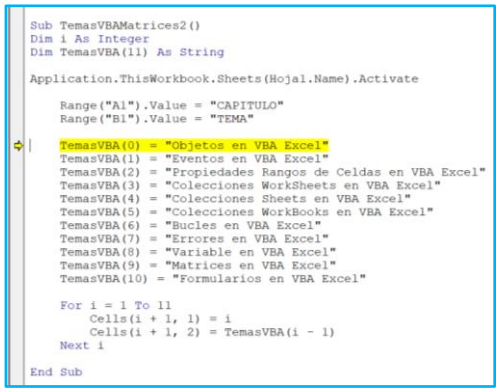
En el ejemplo anterior he creado una **Matriz de 11 Variables**, cuyos índices van de **1 a 11**. Podía haber dimensionado la Matriz declarándola con ***Dim TemasVBA(10) As String*** y seguiría teniendo **11 elementos**, pero los índices ahora van de **0 a 10**.

Los índices pueden comenzar en **0 o en 1**, según escribamos al principio del ***Módulo Option Base 0***, o bien ***Option Base 1***. Cuando no indiquemos nada, el valor por defecto es ***Option Base 0***.

```
Sub TemasVBAMatrices2()
Dim i As Integer
Dim TemasVBA(10) As String

Application.ThisWorkbook.Sheets(Hoja1.Name).Activate

  Range("A1").Value = "CAPITULO"
  Range("B1").Value = "TEMA"

  TemasVBA(0) = "Objetos en VBA Excel"
  TemasVBA(1) = "Eventos en VBA Excel"
  TemasVBA(2) = "Propiedades Rangos de Celdas en VBA Excel"
  TemasVBA(3) = "Colecciones WorkSheets en VBA Excel"
  TemasVBA(4) = "Colecciones Sheets en VBA Excel"
  TemasVBA(5) = "Colecciones WorkBooks en VBA Excel"
  TemasVBA(6) = "Bucles en VBA Excel"
  TemasVBA(7) = "Errores en VBA Excel"
  TemasVBA(8) = "Variable en VBA Excel"
  TemasVBA(9) = "Matrices en VBA Excel"
  TemasVBA(10) = "Formularios en VBA Excel"

  For i = 1 To 11
    Cells(i + 1, 1) = i
    Cells(i + 1, 2) = TemasVBA(i - 1)
  Next i

End Sub
```

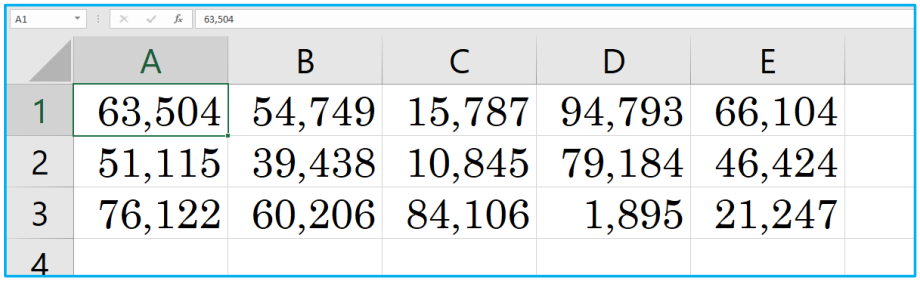
En el Código anterior, si estableciera ***Option Base 1***, nos daría **Error** porque ***TemasVBA(0)*** está fuera de índice.





Ahora haremos una **Matriz Estática de 2 Dimensiones**. Será una **Matriz de 3 Filas y 5 Columnas**, en cuyo interior, cada uno de los valores será un número **pseudoaleatorio** del **0 al 100**. Los Valores de la Matriz los asignaremos al **Rango A1:E3**, redondeado a **3 Decimales**. Lo haré dentro del ***Módulo2*** en la ***Hoja2***.

```
Sub Matriz2D3x5()
Dim i As Integer, j As Integer, Rango As Range
Dim M2D3x5(1 To 3, 1 To 5) As Double

Set Rango = Range(Cells(1, 1), Cells(3, 5))

Application.ThisWorkbook.Sheets(Hoja2.Name).Activate

   For i = 1 To 3
      For j = 1 To 5
         M2D3x5(i, j) = Rnd * (100 + 1)
      Next j
   Next i

   Rango.Value = M2D3x5

End Sub
```



	A	B	C	D	E
1	63,504	54,749	15,787	94,793	66,104
2	51,115	39,438	10,845	79,184	46,424
3	76,122	60,206	84,106	1,895	21,247
4					

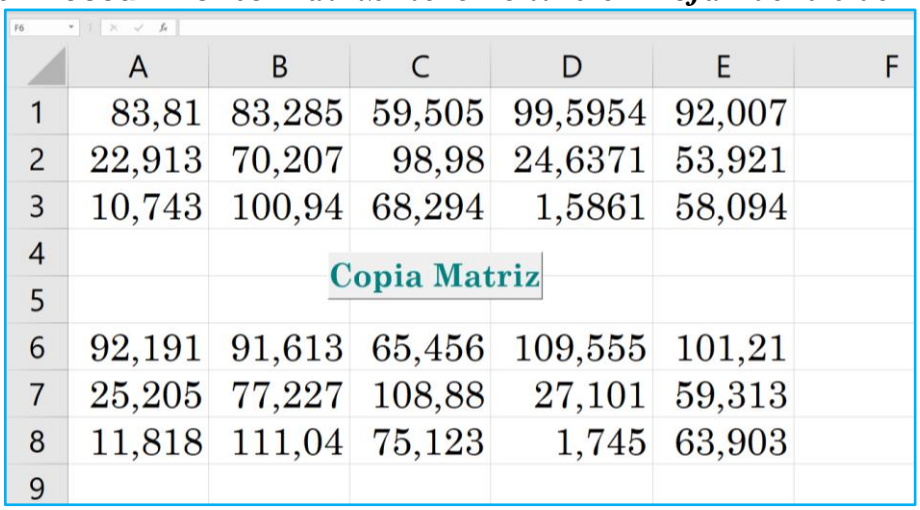
Lo que he hecho ha sido volcar la información de un **Rango a la Matriz**, pero hay que tener presente que el **Rango deberá tener el mismo tamaño (Filas, Columnas) que el Array**.

NOTA. Observa que he declarado una **Variable Tipo Objeto (Range)** que utilizo para asignarle el valor de los elementos de la Matriz a dicho rango. Esto lo podría haber hecho incluyendo la siguiente línea dentro del **Bucle j**.

```
   For j = 1 To 5
      M2D3x5(i, j) = Rnd * (100 + 1)
      Cells(i, j).Value = Round(M2D3x5(i, j), 3)
```

Next j

Como **Ejercicio Propuesto**, modifica el Código anterior para que, al pulsar un **Botón de Formulario**, haga una Copia de la **Matriz** ***Matriz2D3x5*** como ***Matriz10*** pero con todos los valores incrementados en un **10%**. Luego escribe los valores de la nueva Matriz a partir de la Celda **A6**. Guarda el nuevo Código con el nombre de **Procedimiento** ***MatrizIncrement10*** en ***Hoja2*** dentro del ***Módulo1***.



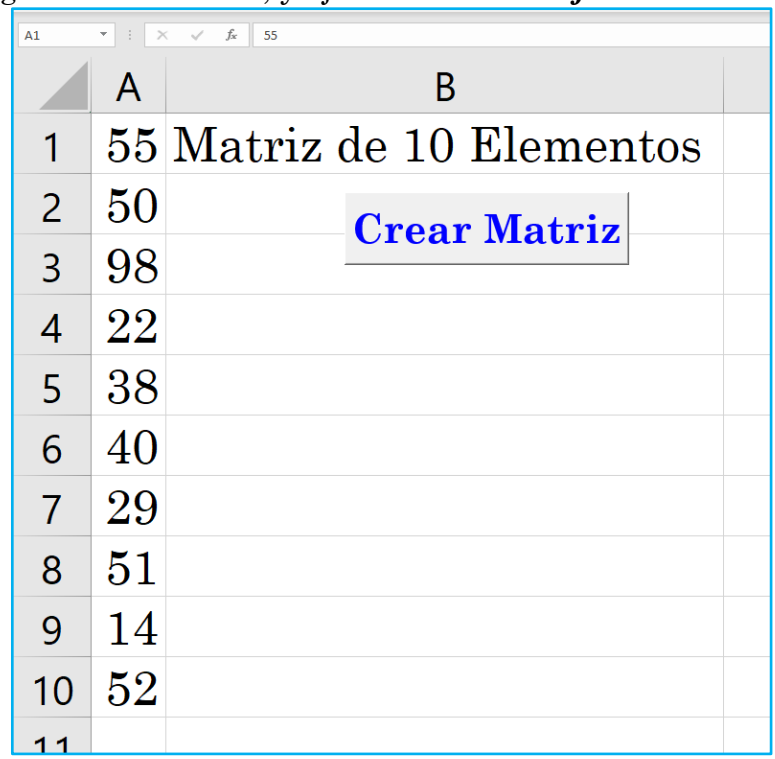
Ejercicio 06.04.-
Crea un archivo llamado ***EJERCICIO0604.xlsm*** y guárdalo.

En el ejercicio anterior, creamos **Matrices Estáticas** (su tamaño no varía). Ahora veremos las **Dinámicas**, cuyo tamaño puede variar a lo largo del Procedimiento.

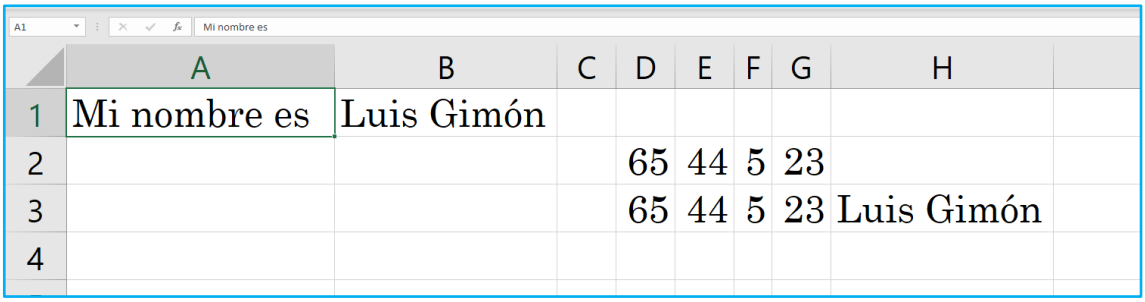
El **Primer Ejercicio Propuesto** de este apartado consiste en crear el **Procedimiento *MatrizDinamicaNoVariante* Tipo *Integer*** que genere una Matriz que cambiará de tamaño (límite de elementos) cada vez que pulsamos el Botón de Formulario asociado.

El **Límite** lo marcará un **Número pseudoaleatorio entre 1 y 10**. Una vez creada, se llenará de tantos valores **Enteros pseudoaleatorios, entre 1 y 100**, como indique el límite. A continuación, escribiremos los valores de la Matriz en la **Columna A, empezando por A1**. En **B1** aparecerá la frase que indica **cuántos valores** tiene la Matriz, cada vez.

Realiza el código en el ***Módulo1***, y ejecútalo en la ***Hoja1***.



El **Segundo Ejercicio Propuesto** en este apartado consiste en crear el **Procedimiento *MatrizDinamicaVariante*** en el que darás valor a la **Celda *B1*** con tu **nombre** y posteriormente, se genere **4 número enteros pseudoaleatorios, entre 1 y 100**, que se guardan en el **Rango (D2:G2)**. A continuación, el programa creará la **Matriz Dinámica Variante *MiMatriz***, usando la **Función *Array***, que contiene **5 valores** (los 4 número enteros pseudoaleatorios, además del valor de B1) que se escribirá en el **Rango (D3:H3)**. Al ejecutar la Macro deberá producir lo siguiente.

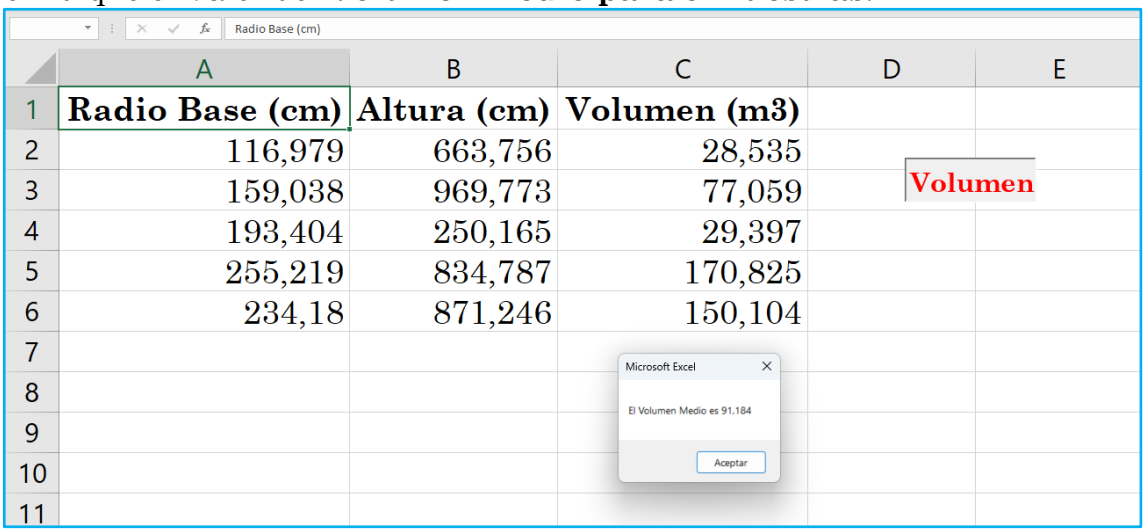


Realiza el código en el ***Módulo1***, y ejecútalo en la ***Hoja2***.

Ejercicio 06.05.-
Crea un archivo llamado ***EJERCICIO0605.xlsm*** y guárdalo.

El **Primer Ejercicio Propuesto** de este apartado consiste en calcular, pseudoaleatoriamente, las dimensiones de un **Bidón de Agua de forma cilíndrica**, para conocer su **Volumen**. En primer lugar, tenemos el **Radio** cuyos valores oscilan entre **0,5m y 2,5m**, seguido de la **Altura** que varía entre **0,5m y 10m**. Ambos aparecerán en la Tabla en **cm**, mientras que el Volumen se muestra en **m3**.

Una vez obtenidos los Valores, creamos una Tabla que recoja esa información (**Radio, Altura y Volumen**). Finalmente, deberá aparecer un Mensaje de Texto que indique el Valor del **Volumen Medio para 5 Muestras**.



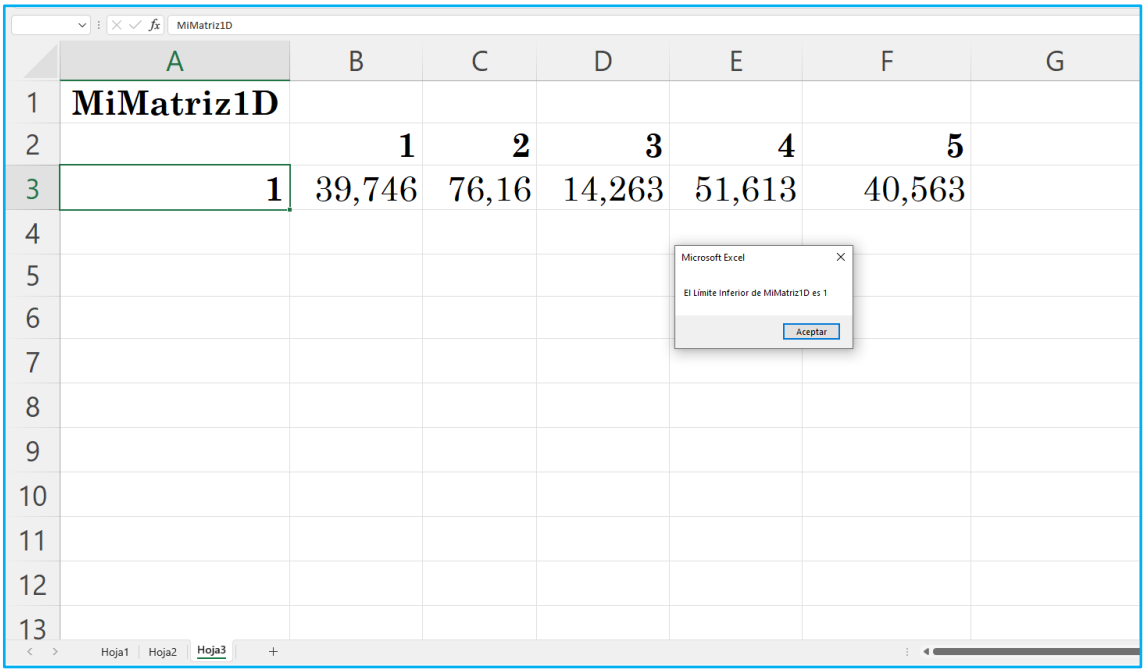
Realiza el **Ejercicio Propuesto** en el ***Módulo1*** de la ***Hoja1***.

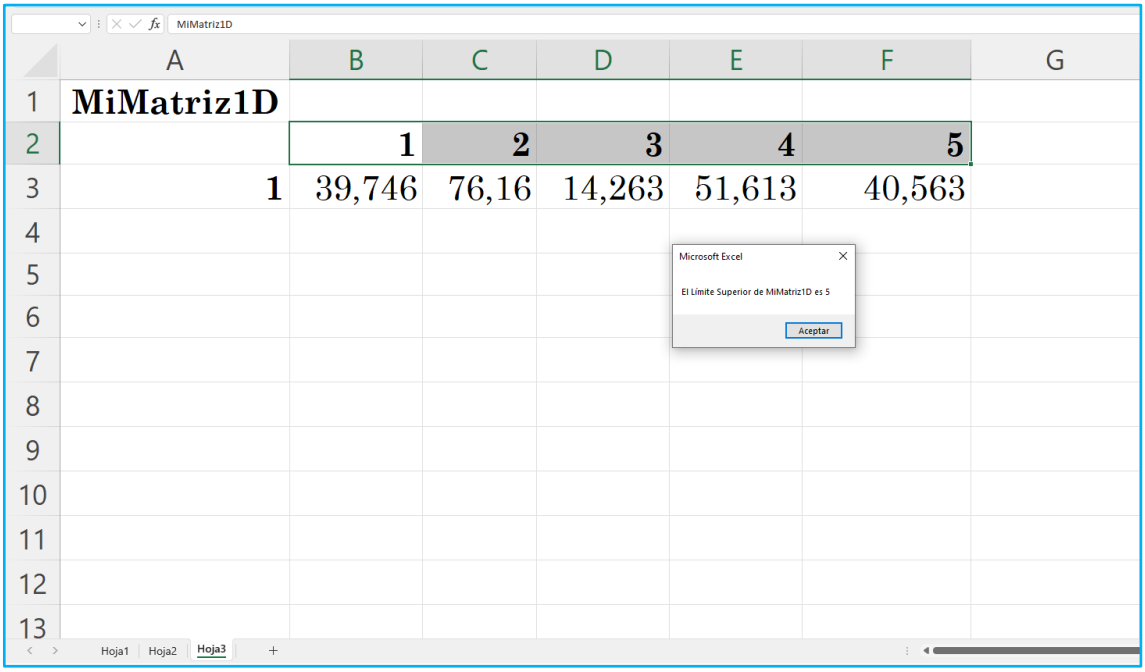
NOTA. Recuerda que puedes usar la siguiente expresión para calcular un valor pseudoaleatorio entre dos límites (Superior\Inferior).

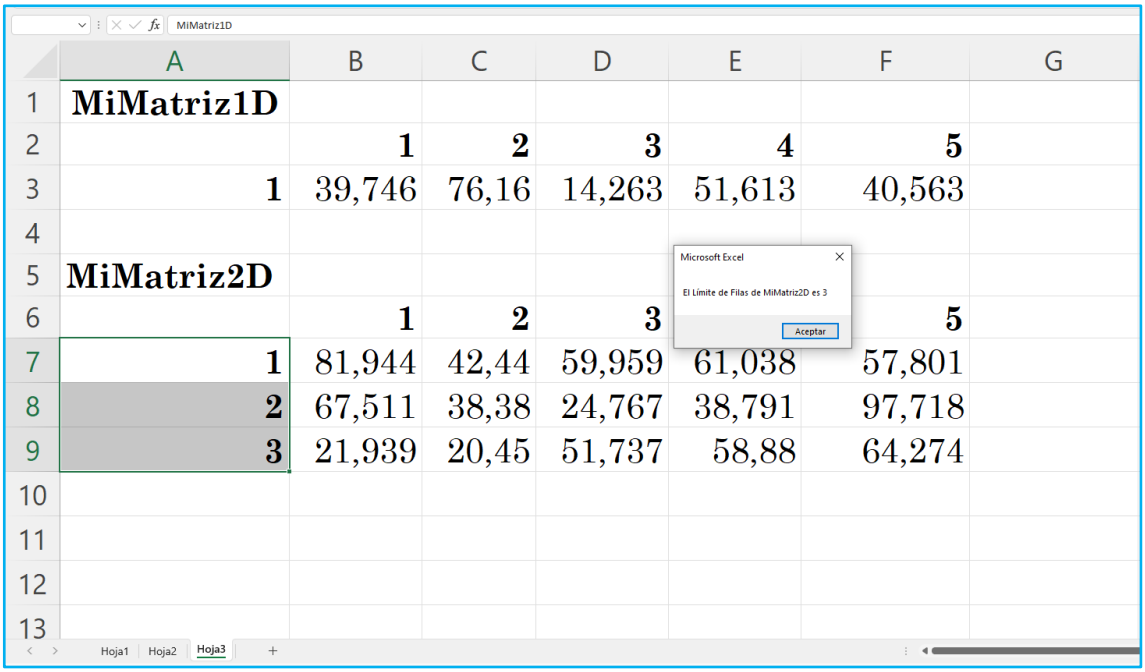
((upperbound - lowerbound + 1) * Rnd + lowerbound)

El **Segundo Ejercicio Propuesto** de este apartado lo realizarás en la ***Hoja3***, dentro del ***Módulo3***.

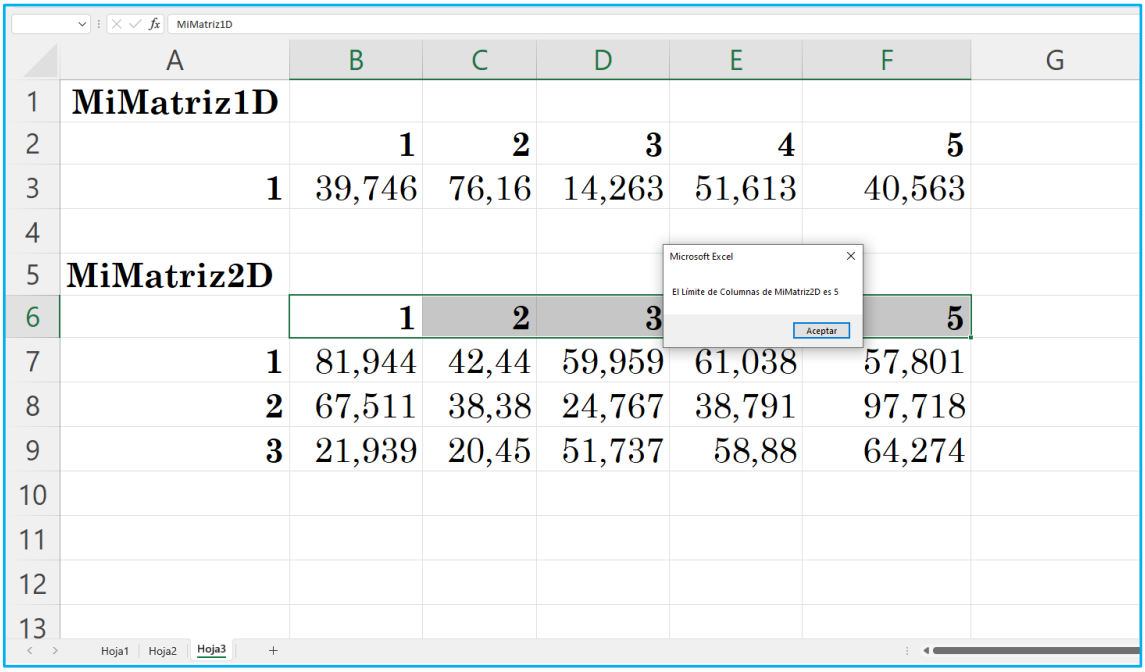
Se trata de crear **2 Matrices (1D y 2D)**, cuyos valores son números pseudoaleatorios comprendidos entre el **1 y el 100**. Para recorrer los índices deberás usar las **Funciones *LBound*** y ***UBound***. Al ejecutar la Macro ***LimiteMatrices***, la secuencia deberá ser la siguiente.


MiMatriz1D
MiMatriz1D
1 2 3 4 5
1 39,746 76,16 14,263 51,613 40,563
Microsoft Excel
El Límite Inferior de MiMatriz1D es 1
Aceptar
Hoja1 Hoja2 Hoja3

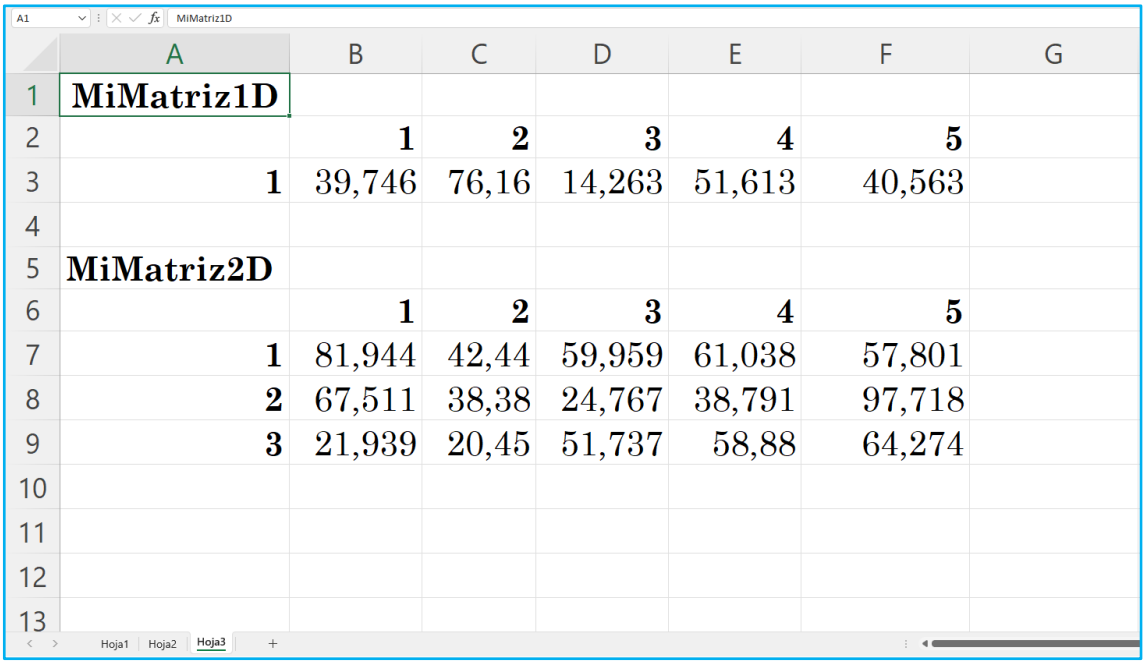

MiMatriz1D
MiMatriz1D
1 2 3 4 5
1 39,746 76,16 14,263 51,613 40,563
Microsoft Excel
El Límite Superior de MiMatriz1D es 5
Aceptar
Hoja1 Hoja2 Hoja3



	A	B	C	D	E	F	G
1	**MiMatriz1D**						
2		**1**	**2**	**3**	**4**	**5**	
3	**1**	39,746	76,16	14,263	51,613	40,563	
4							
5	**MiMatriz2D**						
6		**1**	**2**	**3**		**5**	
7	**1**	81,944	42,44	59,959	61,038	57,801	
8	**2**	67,511	38,38	24,767	38,791	97,718	
9	**3**	21,939	20,45	51,737	58,88	64,274	
10							
11							
12							
13							



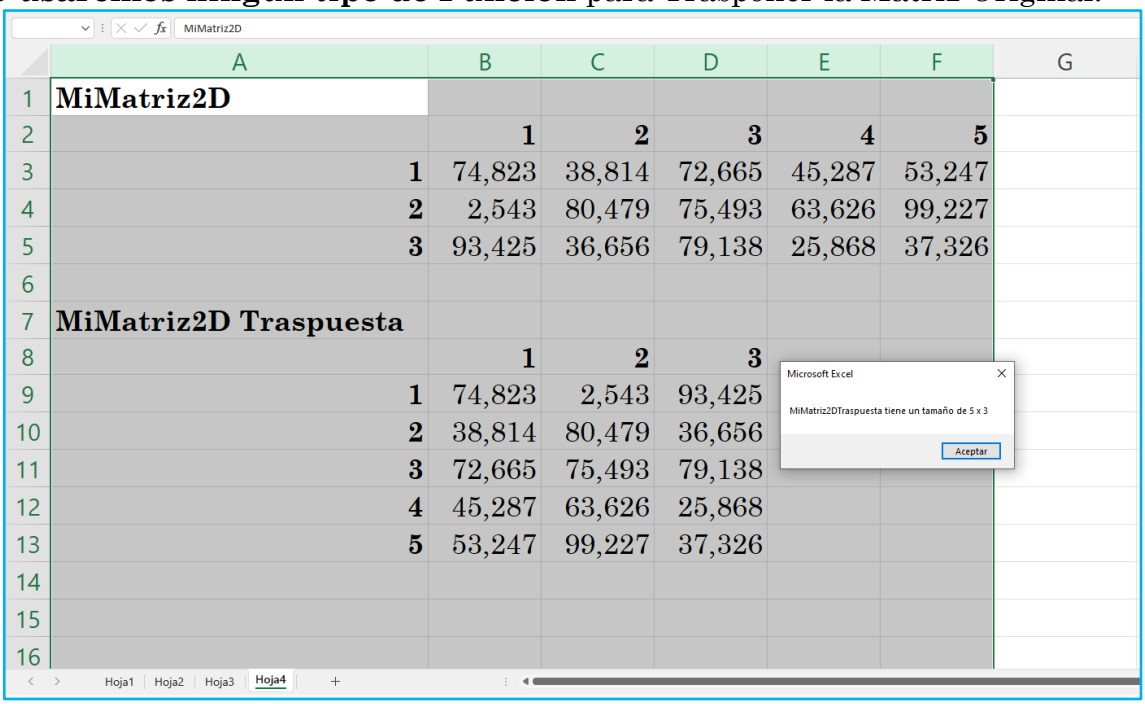
	A	B	C	D	E	F	G
1	**MiMatriz1D**						
2		**1**	**2**	**3**	**4**	**5**	
3	**1**	39,746	76,16	14,263	51,613	40,563	
4							
5	**MiMatriz2D**						
6		**1**	**2**	**3**		**5**	
7	**1**	81,944	42,44	59,959	61,038	57,801	
8	**2**	67,511	38,38	24,767	38,791	97,718	
9	**3**	21,939	20,45	51,737	58,88	64,274	
10							
11							
12							
13							



	A	B	C	D	E	F
1	**MiMatriz1D**					
2		**1**	**2**	**3**	**4**	**5**
3	**1**	39,746	76,16	14,263	51,613	40,563
4						
5	**MiMatriz2D**					
6		**1**	**2**	**3**	**4**	**5**
7	**1**	81,944	42,44	59,959	61,038	57,801
8	**2**	67,511	38,38	24,767	38,791	97,718
9	**3**	21,939	20,45	51,737	58,88	64,274
10						
11						
12						
13						

El **Tercer Ejercicio Propuesto** de este apartado lo realizarás en la ***Hoja4***, dentro del ***Módulo4***. Se trata de crear el **Procedimiento *MatrizTraspuesta*** (*Resultado de reordenar la Matriz Original mediante el cambio de Filas por Columnas y las Columnas por Filas en una nueva matriz*) que creará la **Matriz Traspuesta**.

Al ejecutar la Macro, el resultado deberá ser la siguiente secuencia. Obviamente, **no usaremos ningún tipo de Función** para Trasponer la Matriz Original.



	A	B	C	D	E	F
1	**MiMatriz2D**					
2		**1**	**2**	**3**	**4**	**5**
3	**1**	74,823	38,814	72,665	45,287	53,247
4	**2**	2,543	80,479	75,493	63,626	99,227
5	**3**	93,425	36,656	79,138	25,868	37,326
6						
7	**MiMatriz2D Traspuesta**					
8		**1**	**2**	**3**		
9	**1**	74,823	2,543	93,425		
10	**2**	38,814	80,479	36,656		
11	**3**	72,665	75,493	79,138		
12	**4**	45,287	63,626	25,868		
13	**5**	53,247	99,227	37,326		
14						
15						
16						

Microsoft Excel

MiMatriz2DTraspuesta tiene un tamaño de 5 x 3

Aceptar

Utiliza la asignación ***MiMatriz2DTraspuesta = Rango*** para crear la Matriz Traspuesta. Recuerda que esto se puede hacer sin ***Función Array*** o sin ***ReDim***.

NOTA. Al terminar, comprueba que ***MsgBox IsArray(MiMatriz2DTraspuesta)*** te devuelve ***True (Verdadero)***.

07.- MSGBOX, INPUTBOX Y CONTROL DE ERRORES

Ejercicio 07.01.-

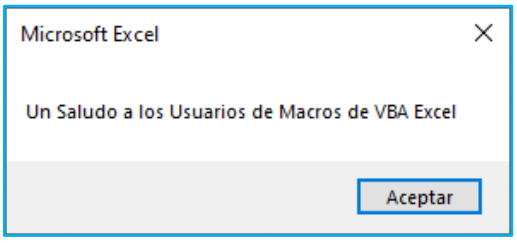
Crea un archivo llamado ***EJERCICIO0701.xlsm*** y guárdalo.

Vamos a empezar a hacer ejercicios usando la **Función *MsgBox***. Se encarga de mostrar un mensaje en un Cuadro de Diálogo, a la espera de que el usuario haga Click en un botón y devuelve un **Valor de Tipo Entero** que indica el botón que el usuario ha pulsado. Su sintaxis es la siguiente.

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

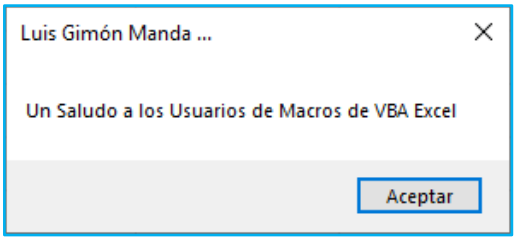
Vamos a hacer varios ejemplos en el ***Módulo1***, en la ***Hoja1***. Comenzamos con lo más básico de esta Función. Escribe lo siguiente y ejecuta la Macro.

```
Sub SaludoInicial()

   MsgBox "Un Saludo a los Usuarios de Macros de VBA Excel"

End Sub
```



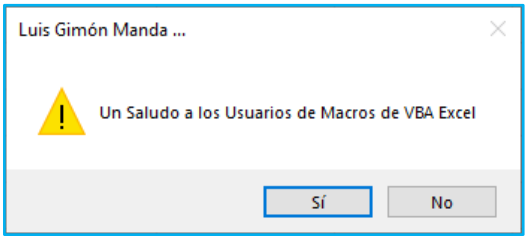
Ahora añadiremos un **Título** (***Title***) al Cuadro de Diálogo. Para eso escribiremos nuestro nombre en la Celda **A1** y redactaremos el siguiente Código.

```
Private Sub SaludoInicialPersonalizado()

   MsgBox "Un Saludo a los Usuarios de Macros de VBA Excel", , Range("A1").Value & " Manda ..."

End Sub
```



Ahora probaremos con el argumento ***[buttons]***. Además del **Botón Aceptar**, podremos mostrar otros y cambiar el comportamiento del Cuadro de Diálogo. Recuerda que los valores están divididos en **5 grupos** cuyos valores son excluyentes, dentro de cada grupo (sólo podremos elegir **1 valor de cada grupo**, aunque podemos combinarlos).

```
Sub Combinacion1()

   MsgBox "Un Saludo a los Usuarios de Macros de VBA Excel", vbYesNo + vbExclamation, Range("A1").Value & " Manda ..."

End Sub
```



El código anterior también se puede escribir como sigue.

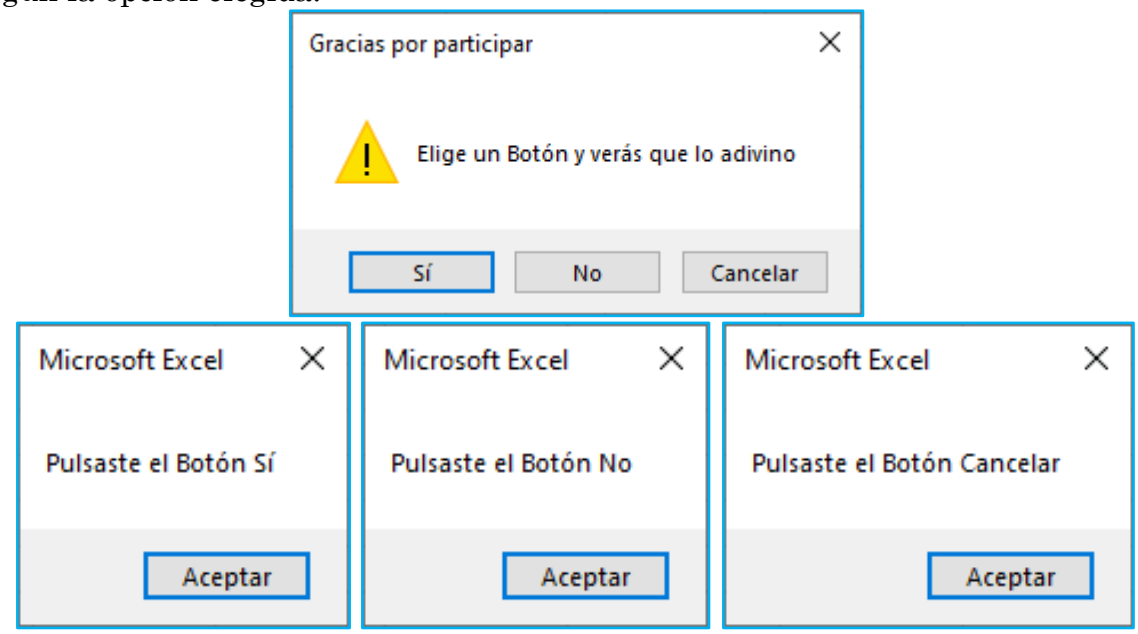
```
Sub Combinacion1()

   MsgBox "Un Saludo a los Usuarios de Macros de VBA Excel", 4 + 48, Range("A1").Value & " Manda ..."

End Sub
```

Ahora vamos a **capturar** el Botón pulsado y, en base a eso, realizaremos una acción. Recordemos que el valor que devuelve la **Función *MsgBox*** se carga sobre una Variable y los parámetros de la Función deberán ir encerrados entre **paréntesis**.

```
Sub RellenoCeldaA1()
Dim Respuesta As Integer

   Respuesta = MsgBox("¿Quieres rellenar de Color Amarillo la celda A1?", 4, "Rellenar de Color la Celda A1")

   MsgBox "Valor de Respuesta = " & Respuesta

   If Respuesta = 6 Then
      With Range("A1")
         .Interior.Color = vbYellow
         .Value = "Relleno Amarillo"
      End With
   Else
      With Range("A1")
         .Interior.Color = xlNone
         .Value = "Sin Color de Relleno"
      End With
   End If

   Range("A1").Select

End Sub
```

El **Ejercicio Propuesto** consiste en lanzar un MsgBox diferente por cada Botón Pulsado. Lanzaremos un ***MsgBox*** para que el usuario pulse entre diferentes

opciones. Hazlo en el ***Módulo1*** y en la ***Hoja1***. El resultado deberá ser como sigue, según la opción elegida.



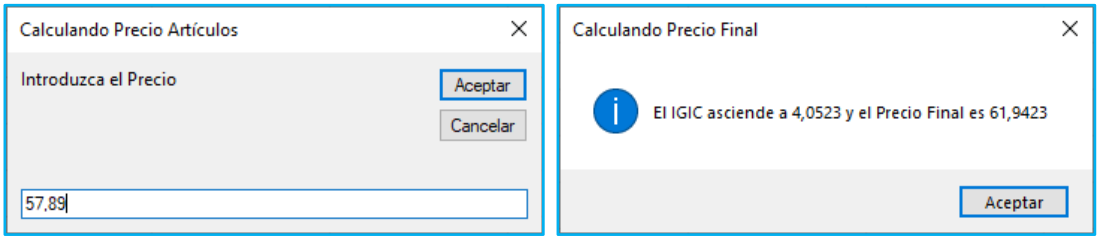
Ejercicio 07.02.-
Crea un archivo llamado ***EJERCICIO0702.xlsm*** y guárdalo.

Ahora haremos ejercicios con ***InputBox*** **(Función y Método).** La **Función** ***InputBox*** muestra una pregunta en un cuadro de diálogo, espera a que el usuario escriba el texto o haga clic en un botón y devuelve una ***String*** que contiene el contenido del cuadro de texto. Su sintaxis es la siguiente, en la que tan sólo el parámetro ***prompt*** (expresión de cadena que aparece como mensaje en el cuadro de diálogo) es obligatorio rellenar.

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

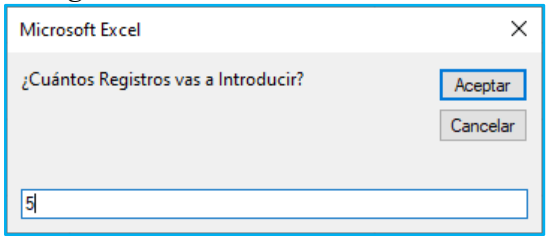
Con el siguiente ejemplo, el usuario ingresa el **precio de un artículo** y el programa calcula el **impuesto (7%)** y el **precio final**.

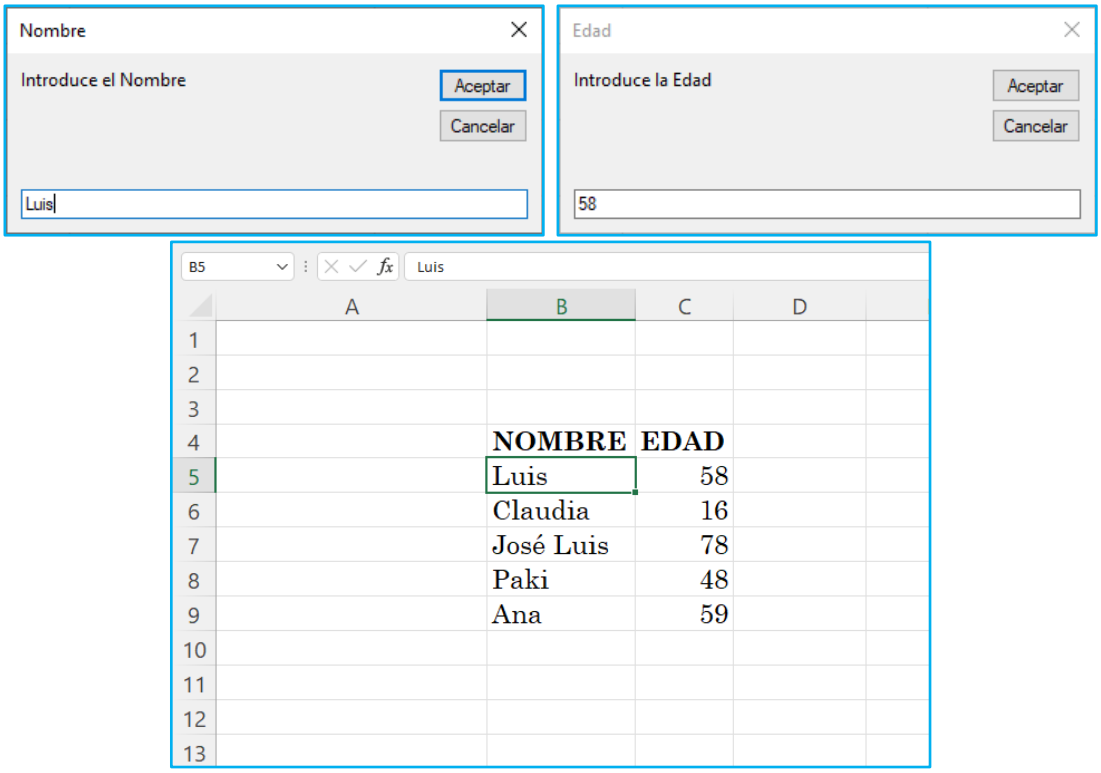
```
Sub PrecioArticulo()
Dim IGIC As Double, Cantidad As Double

IGIC = 7

Application.ThisWorkbook.Sheets(Hoja2.Name).Activate

   Cantidad = InputBox("Introduzca el Precio", "Calculando Precio Artículos")

   MsgBox "El IGIC asciende a " & Cantidad * IGIC / 100 & " y el Precio Final es " &
Cantidad * 1.07, vbInformation, "Calculando Precio Final"

End Sub
```



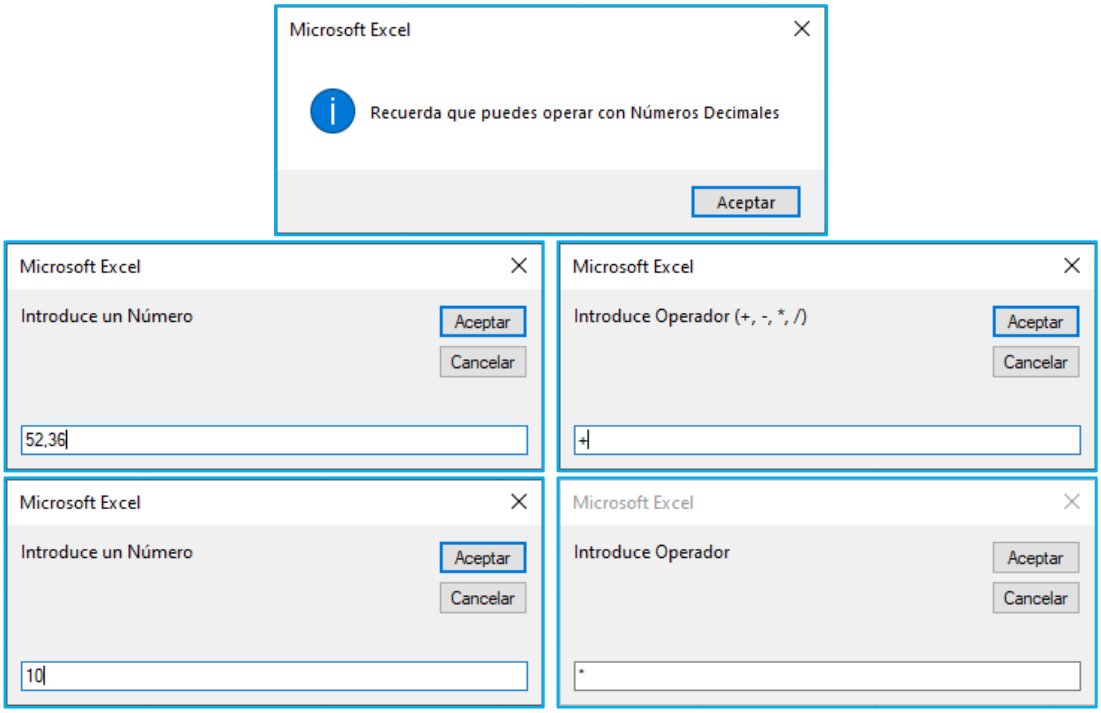
El **Primer Ejercicio Propuesto** consiste en introducir una serie de **Nombres** y la **Edad (Entero)**, guardarlos en una **Variable de Matriz (*Datos*)** y volcarla en la ***Hoja1 (B4)***. Haz el ejercicio en el ***Módulo1***.

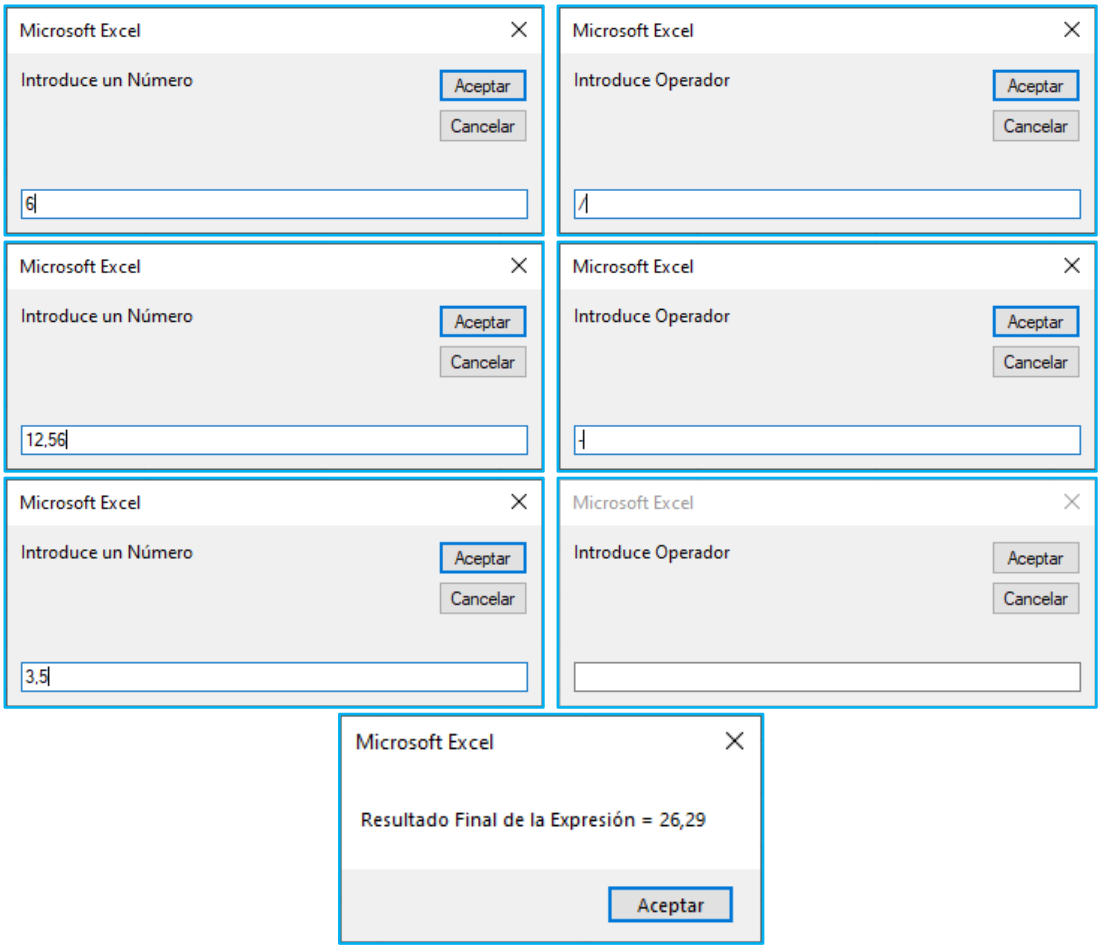
El programa, deberá preguntar cuántos registros se van a introducir, entonces se podrá dimensionar la **Matriz *Datos***. Para un total de **5 Registros**, procura que el resultado se parezca a lo siguiente.



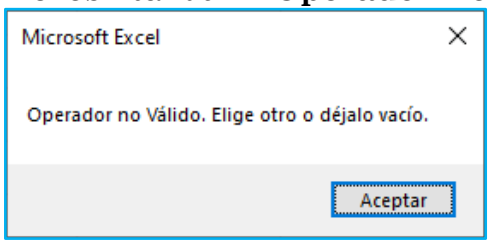


El **Segundo Ejercicio Propuesto** consiste en hacer una **Calculadora** muy sencilla que permite operar con **+ - * /**, únicamente. Crea el **Procedimiento *Calculadora*** en el ***Módulo1***. Ten en cuenta que no tenemos **Control de Errores**. Prueba a hacer esta secuencia.





Intenta operar con **2 Números** usando un **Operador No Válido**.



Ejercicio 07.03.-
Crea un archivo llamado ***EJERCICIO0703.xlsm*** y guárdalo.

Ahora vamos a trabajar con el **Método *InputBox*** (***Application.InputBox***). a diferencia de la **Función**, el **Método** devuelve un **Tipo de Dato *Variant***.

El **Método *InputBox*** difiere de la **Función *InputBox*** en que permite la validación selectiva de la entrada de datos del usuario, y se puede usar con objetos, valores de error y fórmulas de Excel.

Application.InputBox invoca al **Método *InputBox***. En cambio, ***InputBox*** sin calificador de objeto, invoca la **Función *InputBox***. Su sintaxis es la siguiente.

Application.InputBox (Prompt, [Title], [Default], [Left], [Top[, [HelpFile, HelpContextID], [Type])

Lo más destacado de este **Método**, será el uso de ***Type***. En la tabla siguiente se enumeran los valores que se pueden pasar en el argumento ***Type***. Puede ser alguno de ellos o bien la suma de ellos. Por ejemplo, para que un cuadro de entrada acepte Textos y Números, ***Type = 1 + 2***.

VALOR	DESCRIPCION
0	Una fórmula
1	Un número
2	Texto (una cadena)
4	Un valor lógico (*True* o *False*)
8	Una referencia a una celda, como un objeto *Range*
16	Un valor de error, como por ejemplo #N/A
64	Una matriz de valores

En el ejemplo de ***Type:=64*** (**Matrices**) usaré la **Función *VarType*** que devuelve un **entero** que indica el subtipo de una variable o el tipo de la propiedad predeterminada de un objeto.

Constante	Valor	Descripción
vbEmpty	0	Empty (no inicializado)
vbNull	1	Null (datos no válidos)
vbInteger	2	Integer
vbLong	3	Entero largo
vbSingle	4	Número de punto flotante de precisión sencilla
vbDouble	5	Número de punto flotante de doble precisión
vbCurrency	6	Valor de divisa
vbDate	7	Valor de fecha
vbString	8	Cadena
vbObject	9	Objeto
vbError	10	Valor de error
vbBoolean	11	Valor booleano
vbVariant	12	Variante (usado solo con matrices de variantes)
vbDataObject	13	Un objeto de acceso a datos

vbDecimal	14	Valor decimal
vbByte	17	Valor de bytes
vbLongLong	20	Entero longlong (válido solo en plataformas de 64 bits)
vbUserDefinedType	36	Variantes que contienen tipos definidos por el usuario
vbArray	+8192	Matriz (siempre agregada a otra constante cuando la devuelve esta función)

NOTA. En base a la Tabla anterior, si una **Variable** ***Application.InputBox Type:=64*** nos devuelve un ***VarType = 8204***, se debe la suma de **vbVariant = 12 + 8192**.

Voy a escribir debajo un **Ejemplo** con casi todos los ***Type*** en **1 único Módulo**. Recuerda que no tenemos **Control de Errores** en los códigos siguientes. El **Ejercicio Propuesto** consiste en **probarlos todos para que conozca su funcionamiento**.

1. Al principio declaro una serie de **Constantes** que se refieren a cada tipo. Obviamente se puede introducir directamente el valor entero que corresponde a cada ***Type*** (por ejemplo ***Type:=0***.
2. El Procedimiento ***PulsadoCancelar*** lo usaré cada vez que el Usuario pulse la **Tecla Cancelar.**
3. ***ActiveWindow.Zoom = True*** permite ajustar la Vista de la pantalla a la selección.
4. ***Type:=0*** corresponde a introducir una **Fórmula**. Podrás introducir tanto **=A1+B2** (se convierte a notación ***R1C1***), como =**25+75**, por ejemplo.
5. ***Type:=1*** corresponde a introducir un valor **Numérico**.
6. ***Type:=2*** corresponde a introducir un **Texto**. En este ejemplo hay que introducir un Rango de Celda, por ejemplo, ***A1:E3*** y el programa toma el texto correspondiente y lo usa para seleccionar el Rango introducido.
7. ***Type:=4*** corresponde a introducir una expresión **Booleana**. Por ejemplo, introduce la expresión ***+10<2*** y el resultado será que es ***Falsa***.
8. ***Type:=8*** corresponde a introducir un **Rango de Celdas**. Por ejemplo, introduce la expresión ***A1:F15***. He usado el comando ***On Error Resume Next*** para evitar que el error detenga el programa.
9. ***Type:=16*** corresponde a introducir un **Error**.
10. ***Type:=64*** corresponde a introducir una Matriz. En este ejemplo, el programa permite seleccionar una tabla con datos, lo convierte en una Matriz y la traspone.

```
Option Explicit

Const TypeFormula = 0
Const TypeNumber = 1
Const TypeString = 2
Const TypeBoolean = 4
Const TypeRange = 8
Const TypeError = 16
Const TypeArray = 64
Const Titulo = "Curso Macros VBA Excel por Luis Gimón"

Sub PulsadoCancelar()
   MsgBox "Has pulsado Cancelar"
End Sub

Sub Type0_formula()
Dim VarType0 As Variant
```

```
Application.ThisWorkbook.Sheets(Hoja1.Name).Activate

    Cells.Select
    Selection.ClearContents

    Range("A1").CurrentRegion.Select
    Selection.ClearContents

    Range("A1").Value = 10
    Range("A2").Value = 20
    Range("B1").Value = 30
    Range("B2").Value = 40
    Range("C1").Value = "Luis"
    Range("C2").Value = "Gimón"
    Range("D1").Value = "Resultado"

    VarType0 = Application.InputBox(prompt:="Introduce una Fórmula: ", Title:=Titulo,
Type:=TypeFormula)

        If VarType0 = False Then 'Cuando se pulsa Cancelar
            PulsadoCancelar
        Else:
            MsgBox "La Suma " & VarType0 & " la introduzco en D1"
            Range("D2").Value = VarType0
        End If

    Range("A1:D2").Select
    ActiveWindow.Zoom = True 'Amplía la vista en pantalla

    Range("A1").Select

End Sub

Sub Type1_number()
Dim VarType1 As Double

Application.ThisWorkbook.Sheets(Hoja1.Name).Activate

    Cells.Select
    Selection.ClearContents

    Range("A1").Value = 10
    Range("A2").Value = 20
    Range("B1").Value = 30
    Range("B2").Value = 40
    Range("C1").Value = "Luis"
    Range("C2").Value = "Gimón"
    Range("D1").Value = "Resultado"

    VarType1 = Application.InputBox(prompt:="Introduce un Número: ", Title:=Titulo,
Type:=TypeNumber)

        If VarType1 = 0 Then
            PulsadoCancelar
        Else:
            MsgBox "El Número introducido es " & VarType1
            Range("D2").Value = VarType1
        End If
```

```
    Range("A1:D2").Select
    ActiveWindow.Zoom = True 'Amplía la vista en pantalla

    Range("A1").Select

End Sub

Sub Type2_String()
Dim VarType2 As String

Application.ThisWorkbook.Sheets(Hoja1.Name).Activate

    Cells.Select
    Selection.ClearContents

    Range("A1").Value = 10
    Range("A2").Value = 20
    Range("B1").Value = 30
    Range("B2").Value = 40
    Range("C1").Value = "Luis"
    Range("C2").Value = "Gimón"
    Range("D1").Value = "Resultado"

    Range("A1").CurrentRegion.Select

    VarType2 = Application.InputBox(prompt:="Introduce un Rango (p.e. A1:D2) para
Activarlo: ", Title:=Titulo, Default:=Selection.Address, Type:=TypeString)

        If VarType2 = "Falso" Then
            PulsadoCancelar
        ElseIf VarType2 = "" Then
            MsgBox "Has introducido el Valor Vacío"
        Else
            Range("D2").Value = VarType2
            Application.ThisWorkbook.Sheets(Hoja1.Name).Range(VarType2).Select
            ActiveWindow.Zoom = True 'Amplía la vista en pantalla
            MsgBox "Este es el Rango que elegiste", vbInformation
        End If

    Range("A1").Select

End Sub

Sub Type4_Boolean()
Dim VarType4 As Boolean, Expresion As String

Application.ThisWorkbook.Sheets(Hoja1.Name).Activate

    Cells.Select
    Selection.ClearContents

    Range("A1").Value = 10
    Range("A2").Value = 20
    Range("B1").Value = 30
    Range("B2").Value = 40
    Range("C1").Value = "Luis"
    Range("C2").Value = "Gimón"
```

```
    Range("D1").Value = "Expresión"
    Range("E1").Value = "Resultado"

    Expresion = InputBox("Introduce una Expresión Booleana", "Expresión Tipo
Booleana")

    Range("D2").Value = Expresion

    VarType4 = Application.InputBox(prompt:="Introduce una Expresión Booleana: ",
Title:=Titulo, Default:=Expresion, Type:=TypeBoolean)

    Range("E2").Value = VarType4

    Application.ThisWorkbook.Sheets(Hoja1.Name).Range("A1").CurrentRegion.Select
    Selection.Columns.AutoFit
    ActiveWindow.Zoom = True 'Amplía la vista en pantalla

    MsgBox "La Expresión " & Expresion & " es " & VarType4

    Range("A1").Select

End Sub

Sub Type8_Range()
Dim VarType8 As Range

Application.ThisWorkbook.Sheets(Hoja2.Name).Activate

    Cells.Select
    Selection.ClearContents

    Range("A1").Value = "Primer Valor"
    Range("B1").Value = "Segundo Valor"
    Range("C1").Value = "Tercer Valor"
    Range("D1").Value = "Sumatoria"

    Randomize

    Range("A2").Value = Int(Rnd * 100 + 1)
    Range("B2").Value = Int(Rnd * 100 + 1)
    Range("C2").Value = Int(Rnd * 100 + 1)

    Range("D2").Value = Range("A2").Value + Range("B2").Value + Range("C2").Value

    Range("A3").Value = Int(Rnd * 100 + 1)
    Range("B3").Value = Int(Rnd * 100 + 1)
    Range("C3").Value = Int(Rnd * 100 + 1)

    Range("D3").Value = Range("A3").Value + Range("B3").Value + Range("C3").Value

    Range("A1").CurrentRegion.Select 'Rango que se usará por Defecto en el InputBox

    On Error Resume Next

    Set VarType8 = Application.InputBox(prompt:="Seleeciona un Rango o Introduce un
Rango: ", Default:=Selection.Address, Type:=TypeRange)

    VarType8.Select
```

```
    If VarType8 Is Nothing Then
       PulsadoCancelar
    Else:
       Selection.Columns.AutoFit
       ActiveWindow.Zoom = True 'Amplía la vista en pantalla
       MsgBox "El Rango Seleccionado es " & VarType8.Address
    End If

End Sub

Sub Type64_Array()
Dim VarType64 As Variant, i As Integer, j As Integer

Application.ThisWorkbook.Sheets(Hoja3.Name).Activate

   Cells.Select
   Selection.ClearContents

   Range("A1").Value = "Primer Valor"
   Range("B1").Value = "Segundo Valor"
   Range("C1").Value = "Tercer Valor"
   Range("D1").Value = "Sumatoria"

   Range("A2").Value = Int(Rnd * 100 + 1)
   Range("B2").Value = Int(Rnd * 100 + 1)
   Range("C2").Value = Int(Rnd * 100 + 1)

   Range("D2").Value = Range("A2").Value + Range("B2").Value + Range("C2").Value

   Range("A3").Value = Int(Rnd * 100 + 1)
   Range("B3").Value = Int(Rnd * 100 + 1)
   Range("C3").Value = Int(Rnd * 100 + 1)

   Range("D3").Value = Range("A3").Value + Range("B3").Value + Range("C3").Value

   Range("A1").CurrentRegion.Select 'Rango que se usará por Defecto en el InputBox

   VarType64 = Application.InputBox(prompt:="Introduce\Selecciona un Rango: ",
Title:=Titulo, Default:=Selection.Address, Type:=TypeArray)

    If VarType(VarType64) = 11 Then 'Indica que la Variable contiene un Valor
Booleano
       PulsadoCancelar
       Exit Sub
    End If

   'Bucle para trasponer la Matriz
   For i = LBound(VarType64, 2) To UBound(VarType64, 2)
      For j = LBound(VarType64, 1) To UBound(VarType64, 1)
         Cells(UBound(VarType64, 1) + 1 + i, j).Value = VarType64(j, i)
      Next j
   Next i

   i = UBound(VarType64, 1) + UBound(VarType64, 2) + 1 'Filas
   j = UBound(VarType64, 2) 'Columnas

   Range(("A1"), Cells(i, j)).Select 'Seleccionar ambos rangos
```

```
    Selection.Columns.AutoFit
    ActiveWindow.Zoom = True 'Amplía la vista en pantalla

End Sub
```

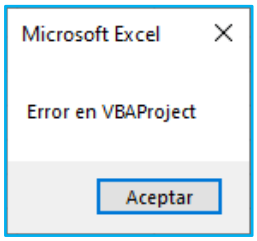
Ejercicio 07.04.-
Crea un archivo llamado ***EJERCICIO0704.xlsm*** y guárdalo.

la **Instrucción *On Error***, habilita una rutina de control de errores, al tiempo que indica lo que debe hacer nuestro programa en caso de que se produzca un error controlable. Sirve, también, para deshabilitar una rutina de control de errores.

Una rutina de control de errores no es ni un procedimiento ***Sub*** ni una ***Function***. Es una parte del código de nuestro programa que viene precedido de una Etiqueta de Línea, en el que le decimos qué hay que hacer en caso de error durante la ejecución.

Las rutinas de control de errores se basan en el valor en la propiedad ***Number*** del **Objeto *Err*** para determinar la causa del error. Efectivamente, cada error tiene su propio ***Err.Number***, así como una cadena que contiene la descripción (***Err.Description***) y siempre se refleja el **último Error** producido. Debemos controlarlo para gestionar el error ya que comprobar el **Err** elimina cualquier duda sobre **qué objeto** lo ha provocado.

La **Propiedad *Err.Source*** contiene el nombre del Objeto o de la Aplicación que generó el error. Por ejemplo, trabajando con MS Excel, ***Err.Number*** recoge el código de error producido y establece ***Err.Source*** al nombre del Proyecto en el que se generó el Error.



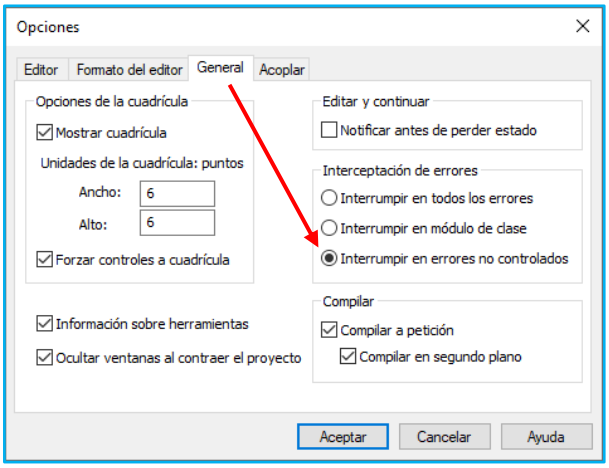
NOTA. Ten en cuenta que la **Instrucción *Err.Clear*** se usa para **borrar las Propiedades** del Objeto ***Err*** una vez que el error está controlado.

On Error Resume Next hace que la ejecución continúe con la instrucción inmediatamente posterior a la instrucción en la que se produjo el error. Esto significa que el programa continúa a pesar del error. ***On Error Resume Next*** no actúa si cambiamos de procedimiento (***Call***), así que tendríamos que colocar una nueva rutina de este tipo en cada procedimiento de nuestra aplicación.

On Error GoTo 0 deshabilita el control de errores en el procedimiento actual (desactiva la captura de errores). Al salir de cualquier procedimiento se desactiva automática la captura de errores.

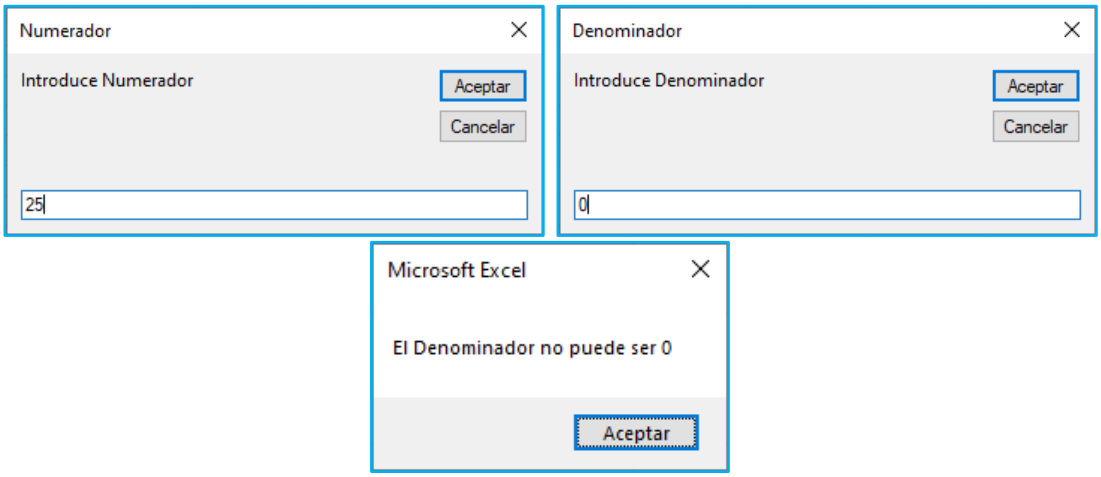
NOTA. Ten presente que, para evitar que se ejecute el código de control de errores cuando no se haya producido ningún error, habrá que colocarla la **Instrucción *Exit Sub*** o ***Exit Function*** justo antes de la rutina de control de errores.

NOTA. Antes de escribir el Código, nos aseguraremos de activar la **Opción *Interrumpir en errores no controlados***, dentro del **Menú Herramientas\Opciones\Interceptación de errores**. Recuerda que, si no lo hacemos, el manejo de errores no funcionará correctamente.



Vamos a empezar por controlar un **Error al Dividir por 0**. Como verás, en cuanto se produce un **Error**, activa la **Etiqueta *ControlErrores***, lanza el ***MsgBox*** y termina la aplicación. En el caso que no haya Error, no pasará por el Control de Errores porque tenemos la Instrucción ***ExitSub***.

```
Sub Dividirx0()
Dim Valor1 As Double, Valor2 As Double

On Error GoTo ControlErrores

   Valor1 = InputBox(prompt:="Introduce Numerador", Title:="Numerador")
   Valor2 = InputBox(prompt:="Introduce Denominador", Title:="Denominador")

   MsgBox "La División entre " & Valor1 & "/" & Valor2 & " = " & Valor1 / Valor2

Exit Sub

ControlErrores:

   MsgBox "El Denominador no puede ser 0"

End Sub
```

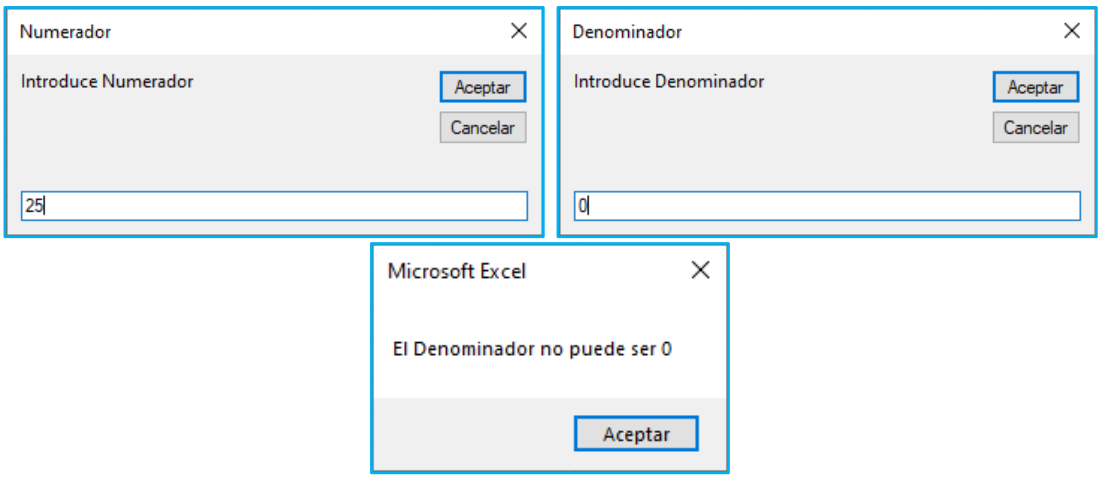


A continuación, cambiamos el Código y usamos ***On Error Resume Next***. Veremos que, si hay un Error, no aparecerá el mensaje y el programa terminará sin que pase nada.

```
Sub Dividirx0ResumeNext()
Dim Valor1 As Double, Valor2 As Double

On Error Resume Next

   Valor1 = InputBox(prompt:="Introduce Numerador", Title:="Numerador")
   Valor2 = InputBox(prompt:="Introduce Denominador", Title:="Denominador")

   MsgBox "La División entre " & Valor1 & "/" & Valor2 & " = " & Valor1 / Valor2

End Sub
```

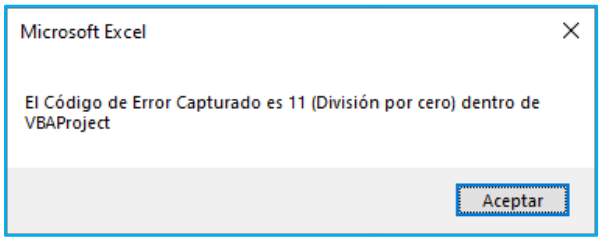
Vamos a hacer un ejemplo que nos permita **Dividir 2 números** y controle el **Error** que se produce al **Dividir por 0**, usando ***Resume Número de Línea***.

```
Sub Dividir2Numeros()
Dim Valor1 As Double, Valor2 As Double

On Error GoTo ControlErrores

   Valor1 = InputBox(prompt:="Introduce Numerador", Title:="Numerador")
1  Valor2 = InputBox(prompt:="Introduce Denominador", Title:="Denominador")

   Range("A1").Value = Valor1
   Range("A2").Value = Valor2
   Range("A3").Value = Valor1 / Valor2

Exit Sub

ControlErrores:

   MsgBox "El Denominador no puede ser 0"
   Resume 1

End Sub
```

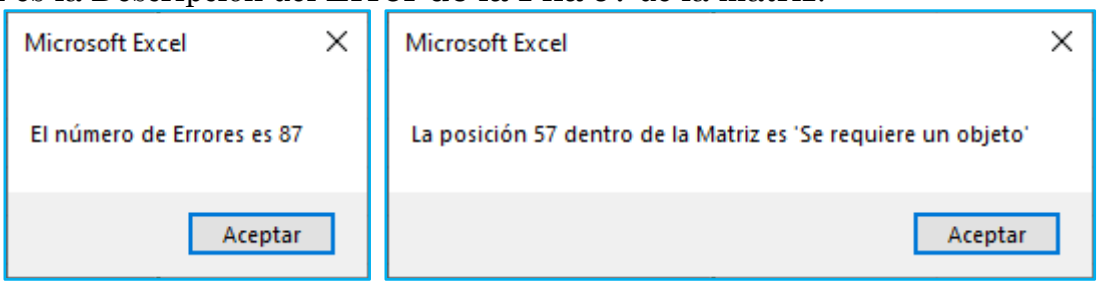


El siguiente ejemplo captura el Error y nos da la Descripción del mismo.

```
Sub Dividirx0Err()
Dim Valor1 As Double, Valor2 As Double

On Error GoTo ControlErrores

   Valor1 = InputBox(prompt:="Introduce Numerador", Title:="Numerador")
   Valor2 = InputBox(prompt:="Introduce Denominador", Title:="Denominador")

   MsgBox "La División entre " & Valor1 & "/" & Valor2 & " = " & Valor1 / Valor2

Exit Sub

ControlErrores:

   MsgBox "El Código de Error Capturado es " & Err.Number & " (" & Err.Description &
") dentro de " & Err.Source

End Sub
```



El **Ejercicio Propuesto** consiste en crear un **Procedimiento** llamado ***ListadoErrores*** que genere una **Matriz** llamada ***MatrizErrores*** que contenga el **Listado de Errores**, cuyos **Códigos** estén comprendido entre el **1 y el 750** y cuya **Descripción** no coincida con "***Error definido por la aplicación o el objeto***". Haz que salga un mensaje que indique el **número de Errores** que te aparecen y cuál es la Descripción del **Error de la Fila 57** de la matriz.

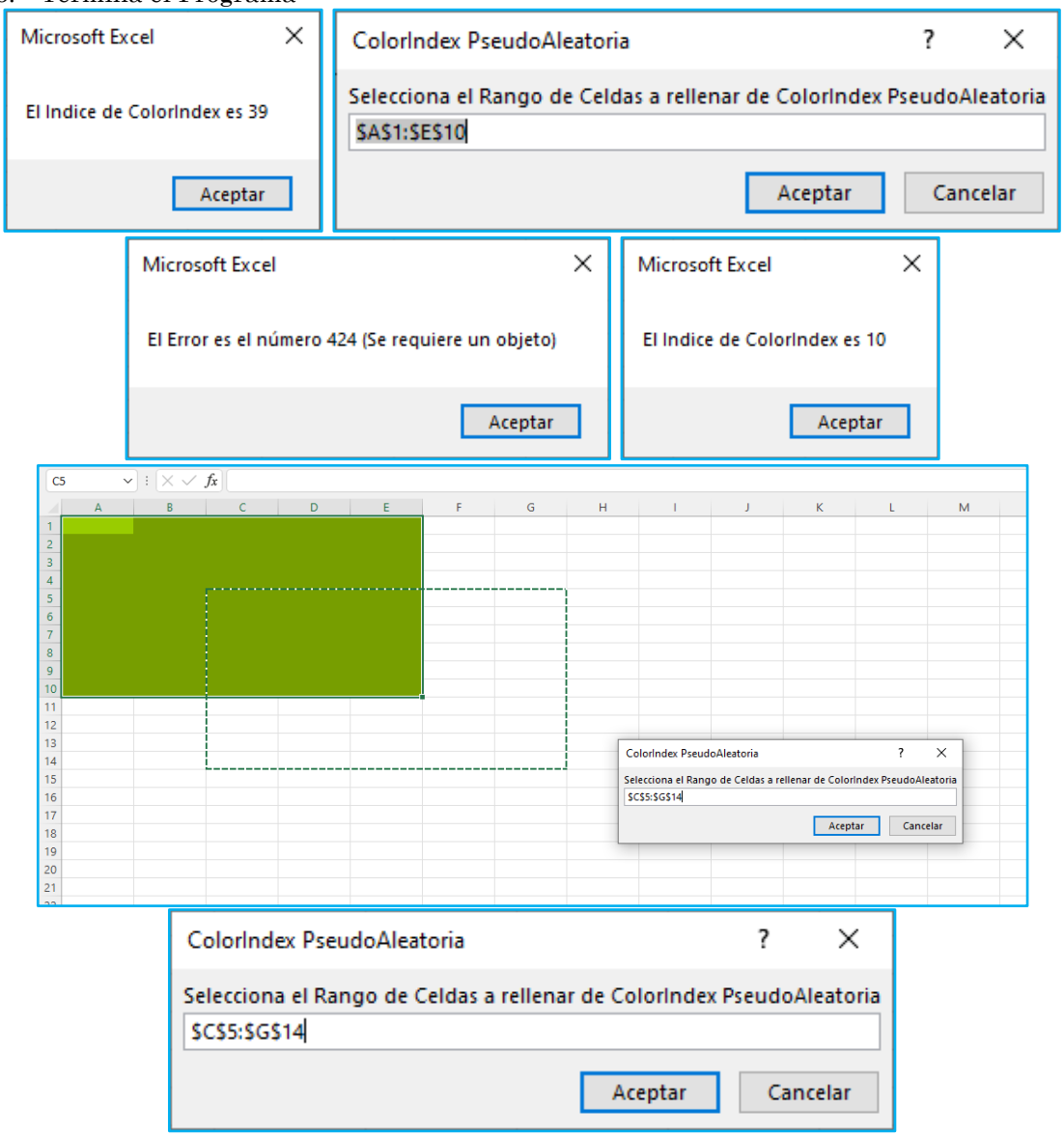


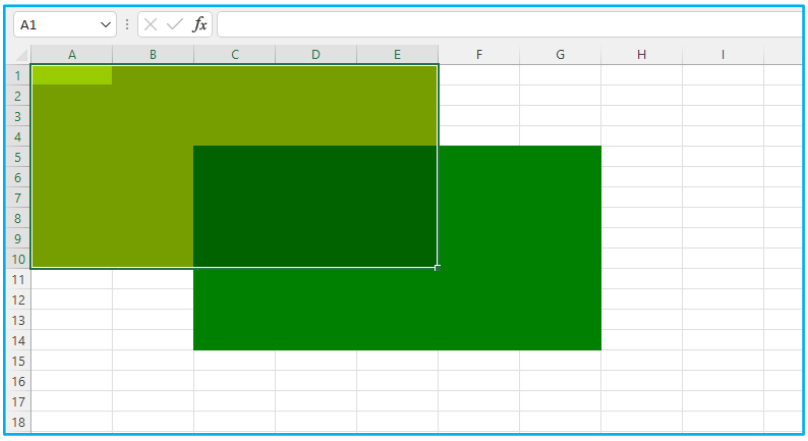
Ejercicio 07.05.-
Crea un archivo llamado ***EJERCICIO0703.xlsm*** y guárdalo.

El **Ejercicio Propuesto** consiste en hacer un programa (**Procedimiento *CeldasRellenar***) que utilice el **Método *InputBox*** para solicitar seleccionar un Rango para rellenar de ***ColorIndex* PseudoAleatorio**. El parámetro ***Default*** del **Método** vale el **Rango A1:E10**.

Te muestro la secuencia de lo que debe ocurrir al ejecutar el Programa.

1. Se **elimina** los Colores de todas las Celdas de la ***Hoja1*** y aparece un Cuadro de Diálogo con el **ColorIndex PseudoAleatorio**. Pulso **Aceptar**.
2. Aparece el Cuadro de Diálogo con el **Rango por defecto**. Pulso **Cancelar**.
3. Aparece el Cuadro de Diálogo con el **Código de Error** y su **Descripción**. Pulso **Aceptar.**
4. Se rellena el **Rango por Defecto** con el **ColorIndexPseudoAleatorio**, al tiempo que aparece el Cuadro de Diálogo con un ***nuevo*** **ColorIndex PseudoAleatorio** y Pulso **Aceptar**.
5. Aparece el Cuadro de Diálogo con el **Rango por defecto** y selecciono el **Rango C5:G14**. Pulso **Aceptar**.
6. Termina el Programa




A1
A
B
C
D
E
F
G
H
I
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18

08.- FUNCIONES DE EXCEL, INSERTAR FORMULAS Y UDF

Ejercicio 08.01.-

Crea un archivo llamado ***EJERCICIO0801.xlsm*** y guárdalo.

Vamos a utilizar ciertas Funciones Preestablecidas de MS Excel en VBA Excel a partir de ahora y para ello usaremos el **Objeto *Application.WorksheetFunction***.

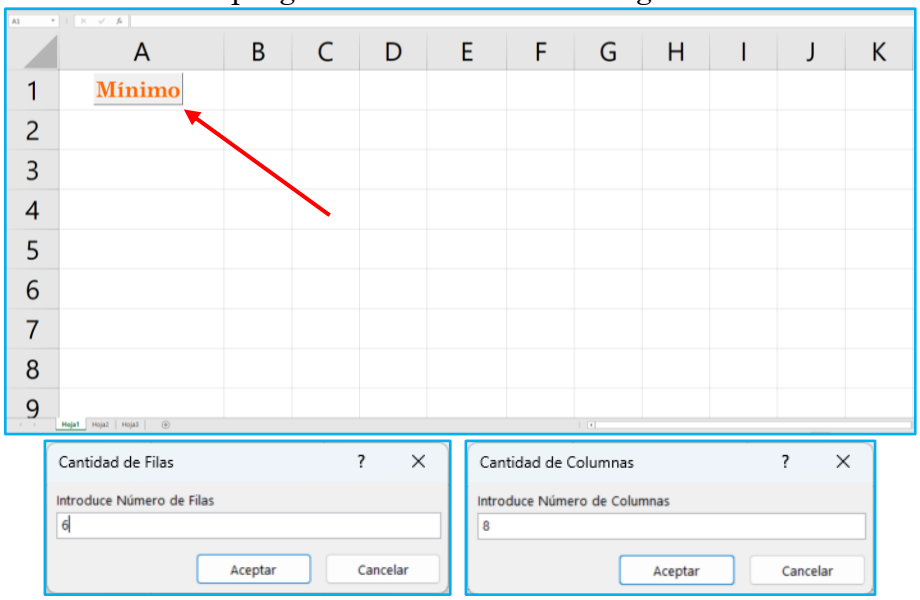
Application.WorksheetFunction.Sum(Rango)

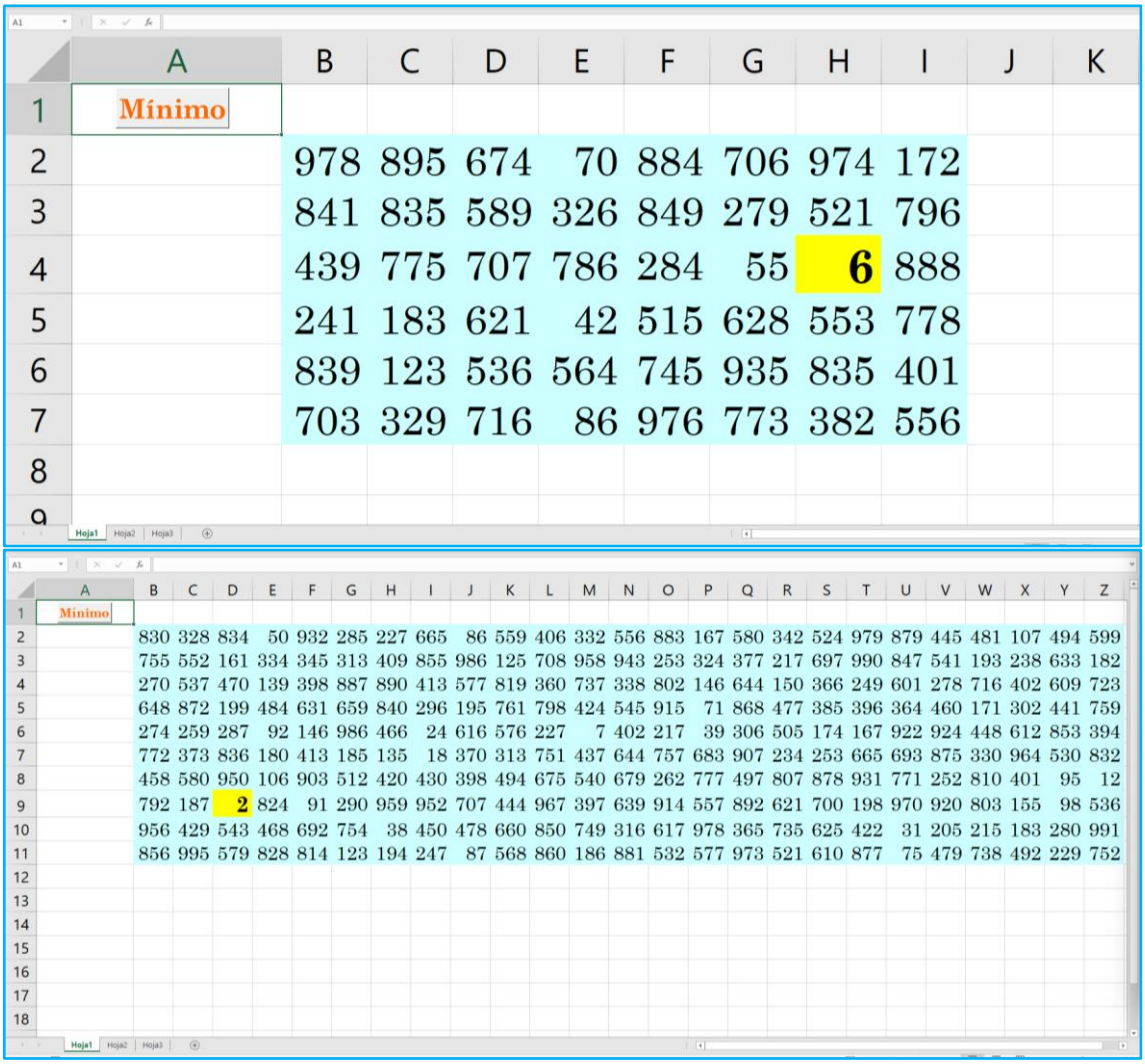
El **Primer Ejercicio Propuesto** consiste en hacer un programa (**Procedimiento *FuncMin***) que utilice el **Método *Application.InputBox*** para solicitar las **Filas** y **Columnas** del Rango de Celdas que contendrá, cada una, un **Número Entero entre 1 y 1000**, **pseudoaleatorio**. Escribe el Código en el **Módulo1** y haz que se ejecute en la ***Hoja1***.

Al pulsar el **Botón de Formulario**, el Código detectará **el número más bajo (puede estar repetido)** y lo resaltará. El resultado siempre se adapta a la pantalla (sea cual sea su matriz).

En caso de pulsar ***Cancelar*** (o ***Esc***) al introducir las Filas o las Columnas, el programa determinará pseudoaleatoriamente, ambos valores (**entre 1 y 15**).

Las pantallas de nuestro programa serán como las siguientes.

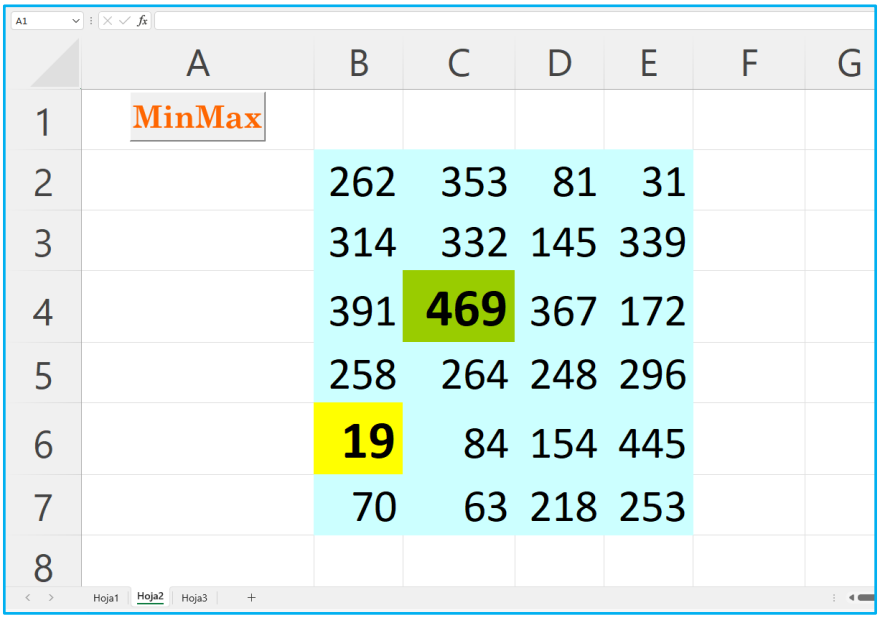




	A	B	C	D	E	F	G	H	I	J	K
1	Mínimo										
2		978	895	674	70	884	706	974	172		
3		841	835	589	326	849	279	521	796		
4		439	775	707	786	284	55	6	888		
5		241	183	621	42	515	628	553	778		
6		839	123	536	564	745	935	835	401		
7		703	329	716	86	976	773	382	556		
8											
9											

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	Mínimo																									
2		830	328	834	50	932	285	227	665	86	559	406	332	556	883	167	580	342	524	979	879	445	481	107	494	599
3		755	552	161	334	345	313	409	855	986	125	708	958	943	253	324	377	217	697	990	847	541	193	238	633	182
4		270	537	470	139	398	887	890	413	577	819	360	737	338	802	146	644	150	366	249	601	278	716	402	609	723
5		648	872	199	484	631	659	840	296	195	761	798	424	545	915	71	868	477	385	396	364	460	171	302	441	759
6		274	259	287	92	146	986	466	24	616	576	227	7	402	217	39	306	505	174	167	922	924	448	612	853	394
7		772	373	836	180	413	185	135	18	370	313	751	437	644	757	683	907	234	253	665	693	875	330	964	530	832
8		458	580	950	106	903	512	420	430	398	494	675	540	679	262	777	497	807	878	931	771	252	810	401	95	12
9		792	187	2	824	91	290	959	952	707	444	967	397	639	914	557	892	621	700	198	970	920	803	155	98	536
10		956	429	543	468	692	754	38	450	478	660	850	749	316	617	978	365	735	625	422	31	205	215	183	280	991
11		856	995	579	828	814	123	194	247	87	568	860	186	881	532	577	973	521	610	877	75	479	738	492	229	752
12																										
13																										
14																										
15																										
16																										
17																										
18																										

El **Segundo Ejercicio Propuesto** consiste en modificar el Ejercicio anterior para que, al pulsar el Botón de Formulario llame al **Procedimiento *FuncMinMax*** que, además de encontrar los **valores más pequeños**, encuentre los **valores más grandes**, entre el **1 y el 500**. Recuerda que ambos valores pueden aparecer repetidos. Escribe el Código en el **Módulo1** y haz que se ejecute en la ***Hoja2***.


A1
A B C D E F G
1 MinMax
2 262 353 81 31
3 314 332 145 339
4 391 469 367 172
5 258 264 248 296
6 19 84 154 445
7 70 63 218 253
8
Hoja1 Hoja2 Hoja3

Ejercicio 08.02.-
Crea un archivo llamado ***EJERCICIO0802.xlsm*** y guárdalo.

El **Primer Ejercicio Propuesto** consiste en modificar el **Procedimiento *FuncMinMax***, del Ejercicio anterior, para que, además de lo que hace, calcule la **Sumatoria de todos los Valores de cada Columna** y los coloque debajo.

Este **Procedimiento** se llamará ***SumarColumnas***. Su Código está dentro de ***Módulo1*** y se ejecuta en ***Hoja1***. El resultado final al ejecutarlo será como sigue.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	SumarColumnas															
2		400	430	73	279	293	384	479	428	212	362	43	236	421	366	265
3		69	156	161	145	193	49	482	51	343	468	461	115	151	177	97
4		338	484	412	260	85	108	306	80	124	258	357	59	188	65	434
5		196	410	398	169	491	158	23	62	311	64	172	113	290	261	480
6		190	426	350	313	425	89	231	437	318	409	137	87	370	388	201
7		394	447	25	423	228	314	320	304	352	75	140	210	1	126	119
8		274	309	468	388	129	275	203	51	376	399	97	371	415	49	15
9		350	479	224	225	486	69	189	460	149	449	181	353	102	355	463
10		404	81	419	138	481	85	141	104	216	380	22	95	109	200	428
11		378	161	179	492	53	238	316	214	19	472	478	462	10	366	431
12	Sumatoria	2.993	3.383	2.709	2.832	2.864	1.769	2.690	2.191	2.420	3.336	2.088	2.101	2.057	2.353	2.933

El **Segundo Ejercicio Propuesto**, modifica el anterior y añade, además de la **Sumatoria**, el **Mínimo**, el **Máximo** y el **Promedio (con 2 decimales)** de cada Columna. El Procedimiento se ejecuta en la ***Hoja2*** y se llamará ***FuncVarias***. El resultado será como sigue.

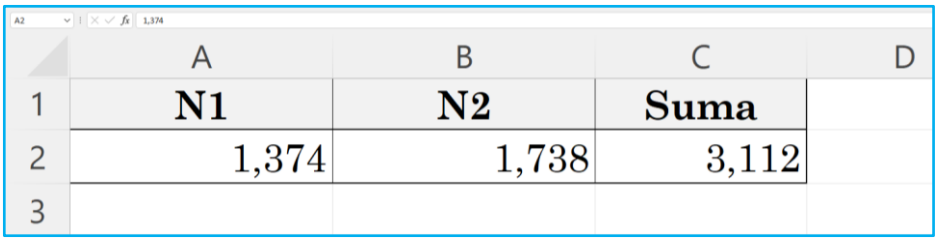
	A	B	C	D	E	F
1	Funciones Varias					
2		102	372	61	361	93
3		265	14	267	375	316
4		493	331	50	393	497
5		53	16	128	292	434
6		7	21	112	340	290
7		73	91	404	465	150
8	*Sumatoria*	*993*	*845*	*1.022*	*2.226*	*1.780*
9	*Mínimo*	*7*	*14*	*50*	*292*	*93*
10	*Máximo*	*493*	*372*	*404*	*465*	*497*
11	*Promedio*	*165,50*	*140,83*	*170,33*	*371,00*	*296,67*

Ejercicio 08.03.-
Crea un archivo llamado ***EJERCICIO0803.xlsm*** y guárdalo.

Toda Celda en VBA Excel tiene la **Propiedad *Formula* y *FormulaLocal*** que permite **almacenar Fórmulas** asociadas. En el caso del **Objeto *Application.WorksheetFunction*** lo que hacíamos era almacenar el resultado d la operación dentro de una Celda, pero si los valores de los que dependía la Celda variaban, **la Celda no actualizaba su valor**. Ahora, al insertar una Fórmula, **la Celda sí actualiza su contenido**.

En el siguiente ejemplo, el valor de la Celda ***C2*** es ***=A2+B2***, por lo que cada vez que cambie el valor de ***A2*** o de ***B2***, se actualizará el de ***C2***.

```
Sub SumarCeldas()

Application.ThisWorkbook.Sheets(Hoja1.Name).Activate

Cells(1, 1).Value = "N1"
Cells(1, 2).Value = "N2"
Cells(1, 3).Value = "Suma"

   Randomize
   Cells(2, 1).Value = Application.Round((Rnd * 10 + 1), 3)
   Cells(2, 2).Value = Application.Round((Rnd * 10 + 1), 3)

   Cells(2, 3).Formula = "=A2+B2"

End Sub
```

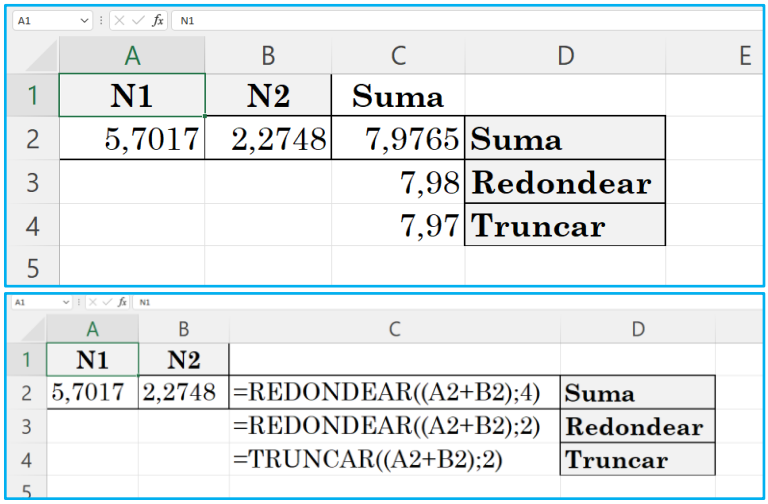


	A	B	C	D
1	N1	N2	Suma	
2	1,374	1,738	3,112	
3				

Ahora toca usar otras funciones para que aparezcan dentro de las Fórmulas.

```
Sub TruncarCeldas()

Application.ThisWorkbook.Sheets(Hoja2.Name).Activate

Cells(1, 1).Value = "N1"
Cells(1, 2).Value = "N2"
Cells(2, 4).Value = "Suma"
Cells(3, 4).Value = "Redondear"
Cells(4, 4).Value = "Truncar”

Range(Cells(2, 1), Cells(2, 3)).Select

   With Selection
      .NumberFormat = "#,##0.0000"
   End With
```

```
Range(Cells(3, 3), Cells(4, 3)).Select

    With Selection
       .NumberFormat = "#,##0.00"
    End With

    Randomize
    Cells(2, 1).Value = Application.Round((Rnd * 10 + 1), 4)
    Cells(2, 2).Value = Application.Round((Rnd * 10 + 1), 4)

    Cells(2, 3).Formula = "=Round((A2+B2),4)"
    Cells(3, 3).Formula = "=Round((A2+B2),2)"
    Cells(4, 3).Formula = "=Trunc((A2+B2),2)"

    Range("A:D").Select
    Selection.Columns.AutoFit
    Selection.Rows.AutoFit

    Range("A1").Select

End Sub
```



	A	B	C	D
1	N1	N2	Suma	
2	5,7017	2,2748	7,9765	Suma
3			7,98	Redondear
4			7,97	Truncar
5				

	A	B	C	D
1	N1	N2		
2	5,7017	2,2748	=REDONDEAR((A2+B2);4)	Suma
3			=REDONDEAR((A2+B2);2)	Redondear
4			=TRUNCAR((A2+B2);2)	Truncar
5				

Otra forma de conseguir el mismo resultado es usar la **Propiedad *FormulaLocal***.

```
Cells(2, 3).FormulaLocal = "=Redondear((A2+B2);4)"
Cells(3, 3).FormulaLocal = "=Redondear((A2+B2);2)"
Cells(4, 3).FormulaLocal = "=Truncar((A2+B2);2)"
```

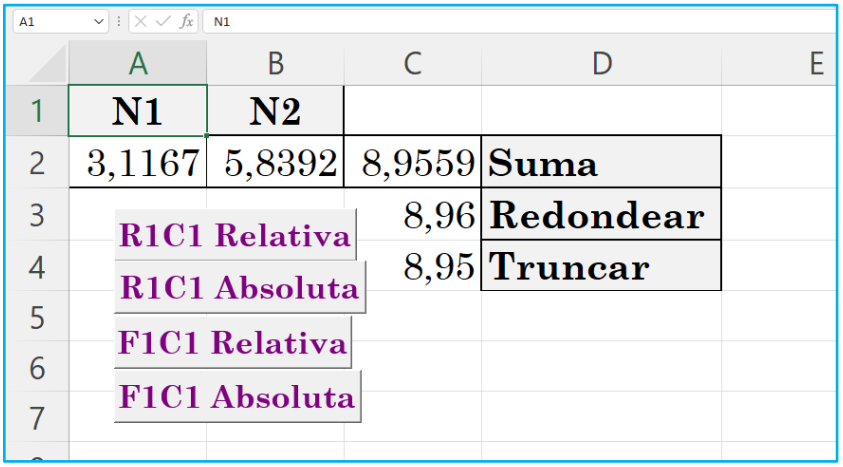
Al usar la notación ***R1C1*** para insertar Fórmulas, tenemos que tener presente la tabla a continuación (teniendo en cuenta **R para Row en Formula** y **F para Fila en FormulaLocal**).

Relativa	Absoluta
R\F[+- Número de Fila] C[+- Número de Columna]	R\F Número de Fila C Número de Columna

El **Ejercicio Propuesto** consiste en adaptar el código anterior a ***FormulaR1C1\FormulaF1C1*** y ***FormulaLocalR1C1\ FormulaLocalF1C1***.

Cada **Procedimiento** se llamará como aparece debajo, se guardarán en **Módulo2** y se ejecutarán en ***Hoja3*** con un Botón de Formulario para cada uno.

TruncarCeldasR1C1Relativa
TruncarCeldasR1C1Absoluta
TruncarCeldasF1C1Relativa
TruncarCeldasF1C1Absoluta



Ejercicio 08.04.-
Crea una copia del archivo ***EJERCICIO0803.xlsm*** con el nombre ***EJERCICIO0804.xlsm*** y guárdalo. Prepara el archivo para que tenga sólo la ***Hoja1***, ***Módulo1*** y ***Módulo2***.

El **Ejercicio Propuesto** consiste en crear varios Procedimientos que guardaremos en el Módulo correspondiente, a los que llamaremos como indico debajo. La ***Hoja1*** tendrá **4 Botones** que llamarán a cada uno de los **Procedimientos** del ***Módulo1***. Cada uno de estos **Procedimientos** calcula el valor de **N1** y **N2** y lo pasa como Argumento de la llamada a los **4 Procedimientos** que están en el ***Módulo2***.

Procedimientos en:	
Módulo1	***Módulo2***
RCRelativa	***TruncarCeldasR1C1Relativa***
RCAbsoluta	***TruncarCeldasR1C1Absoluta***
FCRelativa	***TruncarCeldasF1C1Relativa***
FCAbsoluta	***TruncarCeldasF1C1Absoluta***

Por ejemplo, al pulsar el **Botón de Formulario *R1C1 Absoluta***, se ejecuta el **Procedimiento *RCAbsoluta***, que calcular ***N1*** y ***N2*** y lo pasa como **Argumentos** de la llamada al Procedimiento ***TruncarCeldassR1C1Absoluta***.

Ejercicio 08.05.-
Crea un archivo llamado ***EJERCICIO0805.xlsm*** y guárdalo.

Los Procedimientos que hemos hecho hasta ahora **no devuelven valor alguno**. Ahora trabajaremos con **Procedimientos que sí devuelven valores**. Hablamos de crear **Funciones Personalizadas por el Usuario (UDF)**.

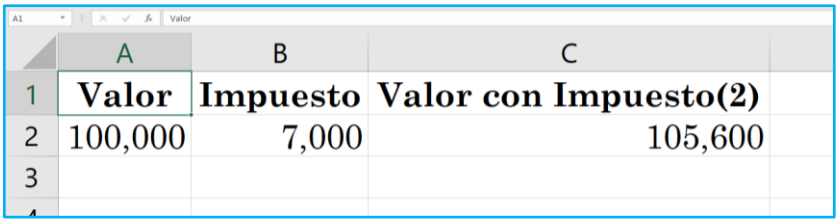
Crearemos UDF cuando sabemos positivamente que MS Excel no dispone de la Función que hace, exactamente, lo que yo necesito.

Trabajaremos en diferentes ***Módulos*** y crearemos uno llamado ***Funciones***, dónde meteremos el Código de todas las **Función** que hagamos.

Vamos a hacer un ejemplo en el que el Programa nos pregunte el valor de un **Producto**, el valor del **Impuesto** y si queremos aplicar dicho Impuesto o uno **Reducido** (un **20% menos**). A la **Función** la llamaremos ***CalcularImpuesto***.

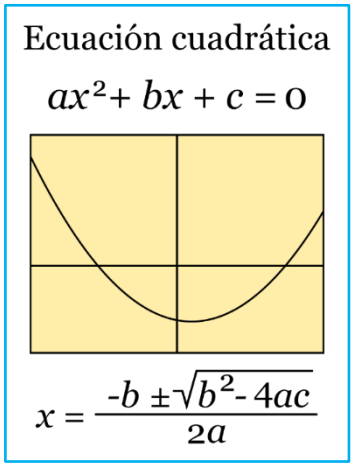
```
Sub Impuesto()
Dim Valor As Double, Impuesto As Double, i As Integer

Application.ThisWorkbook.Sheets(Hoja1.Name).Activate

Cells.ClearContents

   For i = 1 To 3
      Cells(2, i).NumberFormat = "#,##0.000"
   Next i

   Valor = Application.InputBox(prompt:="Introduce el Precio", Title:="Calcular
Impuesto", Default:=0, Type:=1)
   Impuesto = Application.InputBox(prompt:="Introduce el Impuesto", Title:="Impuesto",
Default:=7, Type:=1)

   Cells(1, 1).Value = "Valor"
   Cells(1, 2).Value = "Impuesto"
   Cells(2, 1).Value = Valor
   Cells(2, 2).Value = Impuesto

   Cells(2, 3).Value = CalcularImpuesto(Valor, Impuesto)

   Range("A:D").Select
   Selection.Rows.AutoFit
   Selection.Columns.AutoFit

   Range("A1").Select

End Sub

Function CalcularImpuesto(Valor As Double, Impuesto As Double) As Double
Dim Eleccion As String

   Eleccion = Application.InputBox(prompt:="Elige 1(Impuesto Sin Reducción) o
2(Impuesto Reducido)", Title:="Elegir Tipo Impositivo", Default:="1", Type:=2)
```

```
    Do While Eleccion <> "1" And Eleccion <> "2"
        Eleccion = Application.InputBox(prompt:="Elige 1(Impuesto Sin Reducción) o
2(Impuesto Reducido)", Title:="Elegir Tipo Impositivo", Default:="1", Type:=2)
    Loop

    Cells(1, 3).Value = "Valor con Impuesto" & "(" & Eleccion & ")"

    If Eleccion = 1 Then
        CalcularImpuesto = (Valor * Impuesto / 100) + Valor
    Else: CalcularImpuesto = (Valor * Impuesto * 0.8 / 100) + Valor
    End If

End Function
```



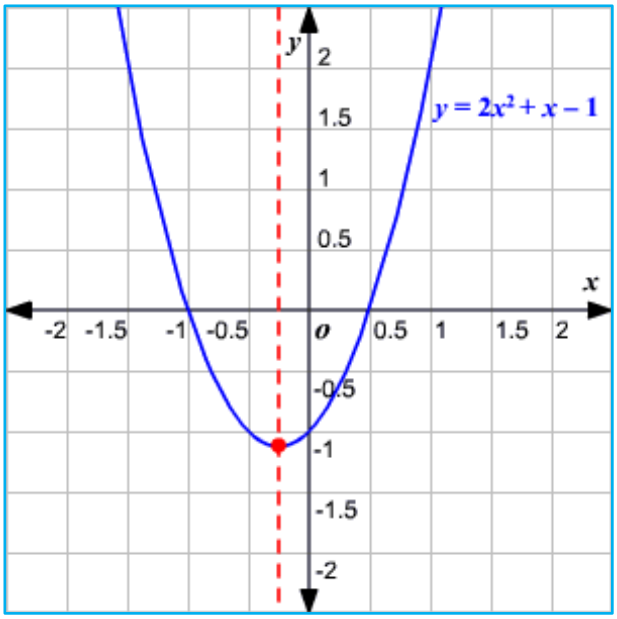
	A	B	C
1	Valor	Impuesto	Valor con Impuesto(2)
2	100,000	7,000	105,600
3			

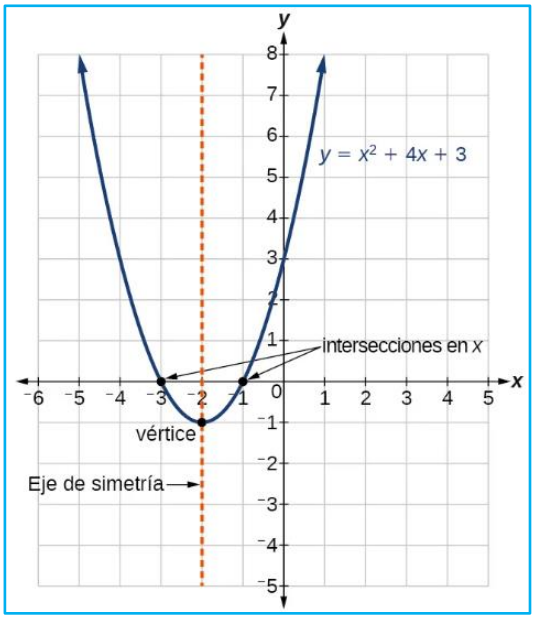
El **Ejercicio Propuesto** consiste en crear una **Función** que calcule la ***archifamosa*** **Fórmula Cuadrática que resuelve las Ecuaciones Cuadráticas.**



La forma general de una **Función Cuadrática** es $F(x) = ax^2 + bx + c$. La gráfica de una Función Cuadrática es una **parábola**, un tipo de **Curva 2D**. Por ejemplo, $10x^2 + 6x - 23 = 0$ es una Función Cuadrática.

Los Parámetros ***a, b*** y ***c*** son valores conocidos. Además, ***a*** no puede ser ***0***. ***x*** es la incógnita que tenemos que resolver. Para eso, tenemos que encontrar aquellos valores que hacen que la ecuación sea **= *0***. Normalmente hay **2 soluciones** (los cortes con el Eje X).





Pero hay veces que la curva **sólo** toca al **Eje X** o incluso que la curva **no corte** al **Eje X** (***Números Complejos***). En estos casos hay que fijarse en la parte de la fórmula que dice ***(b^2 – 4ac)*** a la que llamaremos ***Discriminante*** y es lo que determina varias posibles respuestas.

- Cuando el Discriminante es > 0, tendremos 2 Soluciones de Números Reales.
- Cuando el Discriminante es = 0, sólo hay 1 única Solución (Número Real).
- Cuando el Discriminante es < 0, obtendremos 2 Soluciones de Números Complejos.

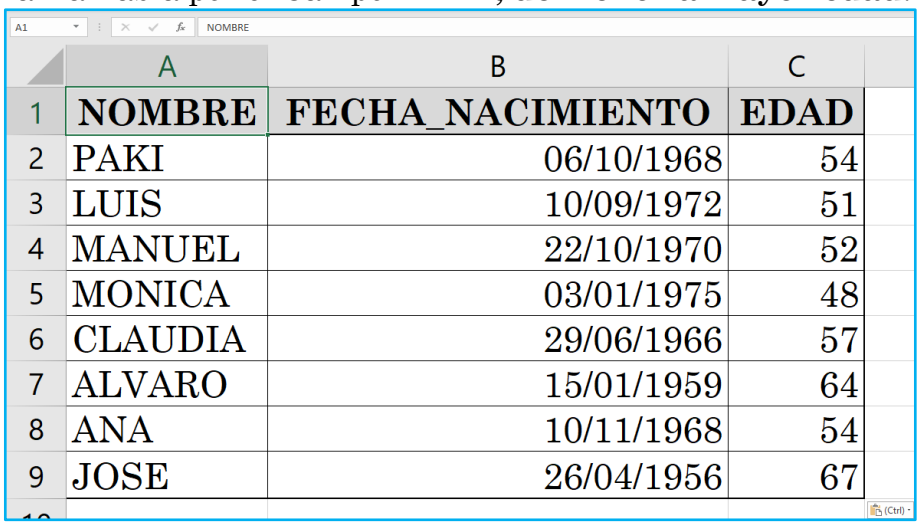
Así pues, nuestro objetivo es crear una **Función Personalizada UDF** que calcule el valor de x de una Función Cuadrática. A la Función VBA Excel la llamaremos ***EcuacionCuadratica*** y al **Procedimiento** que llama a la Función lo llamaremos ***FuncionCuadratica*** y lo guardamos en ***Módulo2*** y se ejecuta en la ***Hoja2***.

09.- ORDENAR, FILTRAR, IMPRIMIR, PDF RANGOS

Ejercicio 09.01.-

Crea un archivo llamado ***EJERCICIO0901.xlsm*** y guárdalo.

En esta ocasión, vamos a Ordenar una serie de valores contenidos en un Rango. Partimos de una tabla de **3 Columnas** (con **Encabezado**) y **8 Registros** (ya veremos que el número de filas puede ser cualquiera). Vamos a desarrolla el Código para ordenar la Tabla por el Campo ***EDAD*, de menor a mayor edad**.

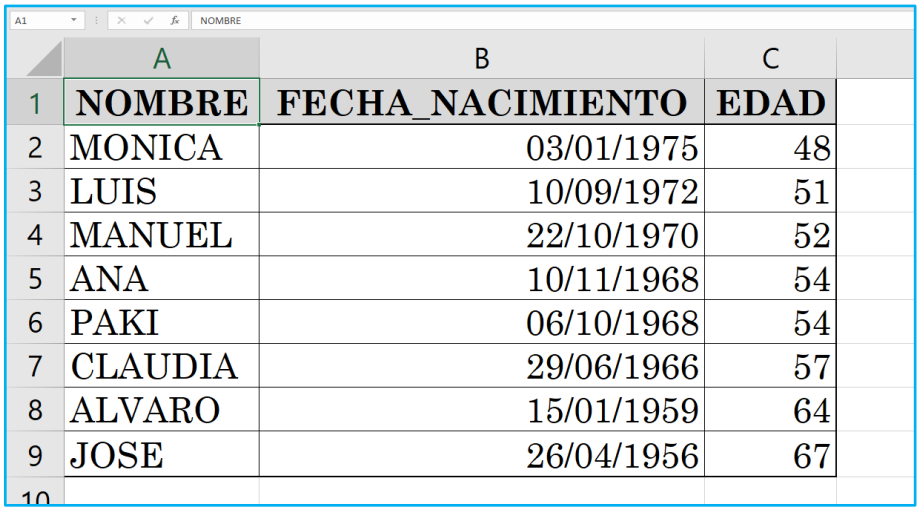


	A	B	C
1	NOMBRE	FECHA_NACIMIENTO	EDAD
2	PAKI	06/10/1968	54
3	LUIS	10/09/1972	51
4	MANUEL	22/10/1970	52
5	MONICA	03/01/1975	48
6	CLAUDIA	29/06/1966	57
7	ALVARO	15/01/1959	64
8	ANA	10/11/1968	54
9	JOSE	26/04/1956	67

Sub OrdenarTabla()

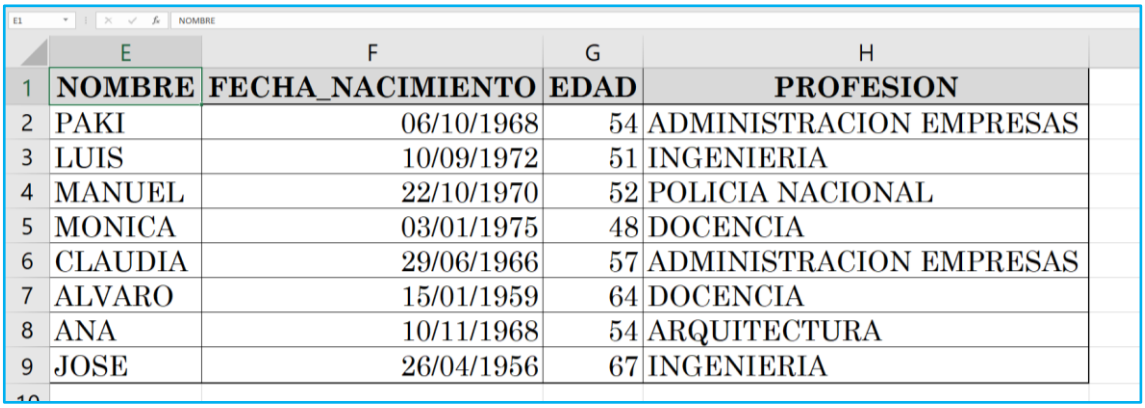
Application.Sheets(Hoja1.Name).Range("A1").Sort key1:=Application.Worksheets("Hoja1").Range("A1"), order1:=xlAscending, Header:=xlYes

End Sub



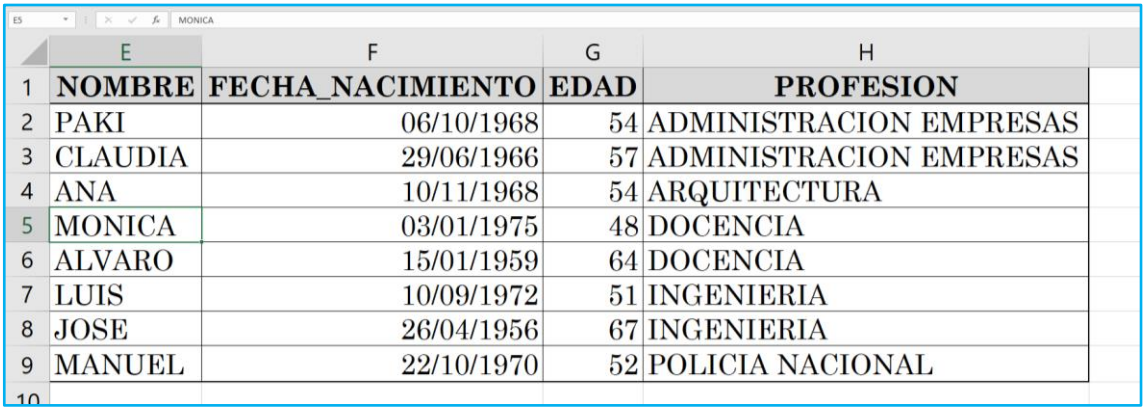
	A	B	C
1	NOMBRE	FECHA_NACIMIENTO	EDAD
2	MONICA	03/01/1975	48
3	LUIS	10/09/1972	51
4	MANUEL	22/10/1970	52
5	ANA	10/11/1968	54
6	PAKI	06/10/1968	54
7	CLAUDIA	29/06/1966	57
8	ALVARO	15/01/1959	64
9	JOSE	26/04/1956	67

Vamos a añadir el Campo ***PROFESION*** y ahora ordenaremos primero por ese campo y segundo por la ***EDAD***.



	E	F	G	H
1	NOMBRE	FECHA_NACIMIENTO	EDAD	PROFESION
2	PAKI	06/10/1968	54	ADMINISTRACION EMPRESAS
3	LUIS	10/09/1972	51	INGENIERIA
4	MANUEL	22/10/1970	52	POLICIA NACIONAL
5	MONICA	03/01/1975	48	DOCENCIA
6	CLAUDIA	29/06/1966	57	ADMINISTRACION EMPRESAS
7	ALVARO	15/01/1959	64	DOCENCIA
8	ANA	10/11/1968	54	ARQUITECTURA
9	JOSE	26/04/1956	67	INGENIERIA

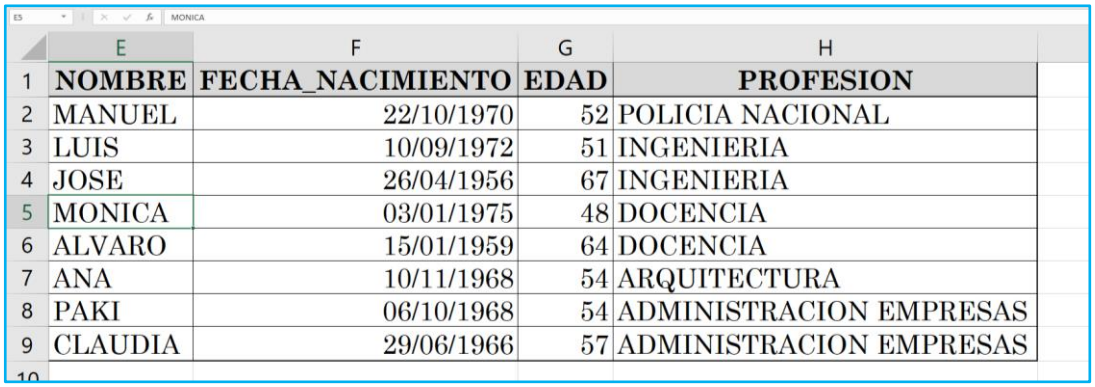
```
Sub OrdenarTabla()

    Application.Sheets(Hoja1.Name).Range("E1").Sort
key1:=Application.Worksheets("Hoja1").Range("H1"), order1:=xlAscending, _
    key2:=Application.Worksheets("Hoja1").Range("G1"), order1:=xlAscending,
Header:=xlYes

End Sub
```



	E	F	G	H
1	NOMBRE	FECHA_NACIMIENTO	EDAD	PROFESION
2	PAKI	06/10/1968	54	ADMINISTRACION EMPRESAS
3	CLAUDIA	29/06/1966	57	ADMINISTRACION EMPRESAS
4	ANA	10/11/1968	54	ARQUITECTURA
5	MONICA	03/01/1975	48	DOCENCIA
6	ALVARO	15/01/1959	64	DOCENCIA
7	LUIS	10/09/1972	51	INGENIERIA
8	JOSE	26/04/1956	67	INGENIERIA
9	MANUEL	22/10/1970	52	POLICIA NACIONAL

Ahora cambio el Código para que ordene de Mayor a Menor por el Primer Campo (***PROFESION***) y de Menor a Mayor por el segundo (***EDAD***).

```
Sub OrdenarTabla()

    Application.Sheets(Hoja1.Name).Range("E1").Sort
key1:=Application.Worksheets("Hoja1").Range("H1"), order1:=xlDescending, _
    key2:=Application.Worksheets("Hoja1").Range("G1"), order1:=xlAscending,
Header:=xlYes

End Sub
```

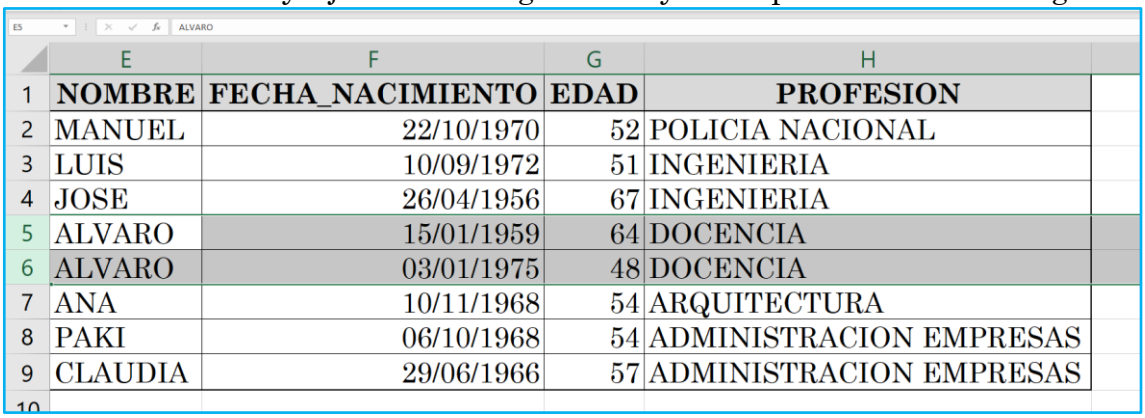


	E	F	G	H
1	NOMBRE	FECHA_NACIMIENTO	EDAD	PROFESION
2	MANUEL	22/10/1970	52	POLICIA NACIONAL
3	LUIS	10/09/1972	51	INGENIERIA
4	JOSE	26/04/1956	67	INGENIERIA
5	MONICA	03/01/1975	48	DOCENCIA
6	ALVARO	15/01/1959	64	DOCENCIA
7	ANA	10/11/1968	54	ARQUITECTURA
8	PAKI	06/10/1968	54	ADMINISTRACION EMPRESAS
9	CLAUDIA	29/06/1966	57	ADMINISTRACION EMPRESAS

El **Ejercicio Propuesto** consiste en Ordenar la Tabla por la ***PROFESION*** (**Descendente**), luego por el ***NOMBRE*** (**Descendente**) y, finalmente, por la ***FECHA_NACIMIENTO*** (**Ascendente**). Hazlo en el ***Módulo1*** de la ***Hoja1*** y llama al **Procedimiento *Orden3Campos***. La Tabla de partida es la siguiente.

NOMBRE	FECHA_NACIMIENTO	EDAD	PROFESION
PAKI	06/10/1968	54	ADMINISTRACION EMPRESAS
LUIS	10/09/1972	51	INGENIERIA
MANUEL	22/10/1970	52	POLICIA NACIONAL
ALVARO	03/01/1975	48	DOCENCIA
CLAUDIA	29/06/1966	57	ADMINISTRACION EMPRESAS
ALVARO	15/01/1959	64	DOCENCIA
ANA	10/11/1968	54	ARQUITECTURA
JOSE	26/04/1956	67	INGENIERIA

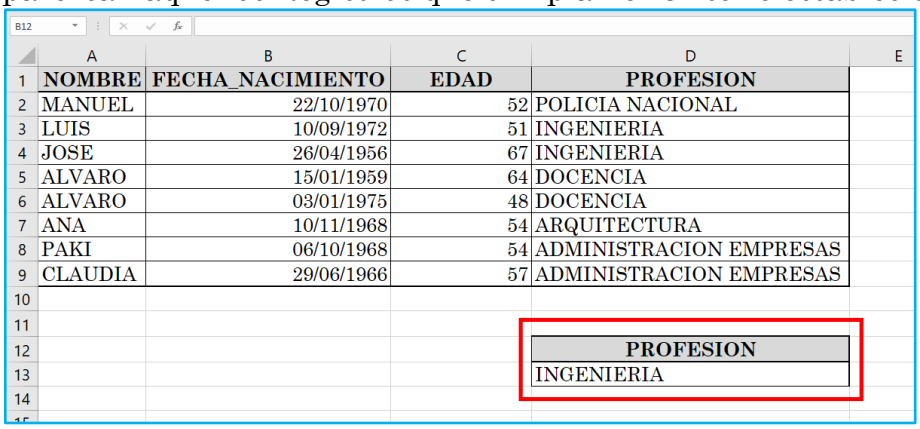
Observa el resultado y fíjate en los registros cuyo Campo ***NOMBRE*** son iguales.



	E	F	G	H
1	NOMBRE	FECHA_NACIMIENTO	EDAD	PROFESION
2	MANUEL	22/10/1970	52	POLICIA NACIONAL
3	LUIS	10/09/1972	51	INGENIERIA
4	JOSE	26/04/1956	67	INGENIERIA
5	ALVARO	15/01/1959	64	DOCENCIA
6	ALVARO	03/01/1975	48	DOCENCIA
7	ANA	10/11/1968	54	ARQUITECTURA
8	PAKI	06/10/1968	54	ADMINISTRACION EMPRESAS
9	CLAUDIA	29/06/1966	57	ADMINISTRACION EMPRESAS

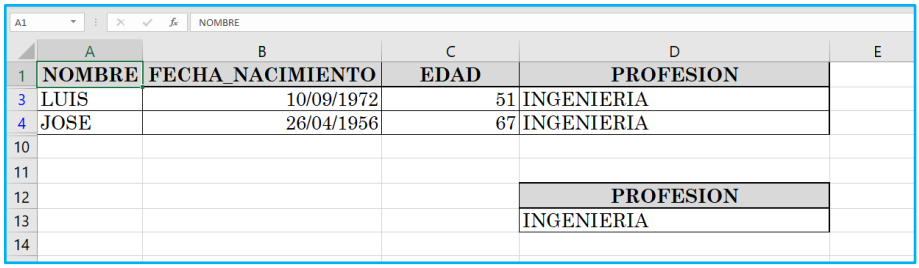
Ejercicio 09.02.-
Haz una copia del archivo ***EJERCICIO0901.xlsm*** con el nombre ***EJERCICIO0902.xlsm*** y guárdalo.

Vamos a tomar el resultado de la Tabla anterior y vamos a aplicarle un Filtro para que sólo aparezcan aquellos Registros que cumplan el Criterio establecido.



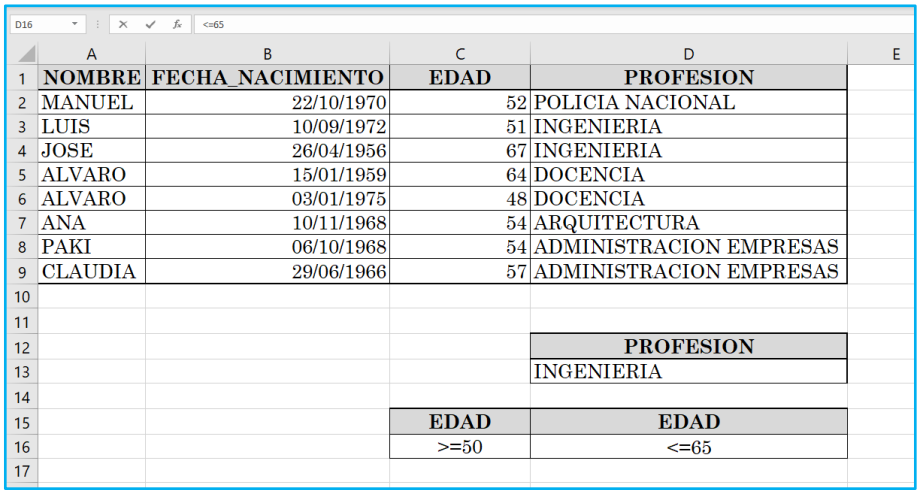
	A	B	C	D
1	NOMBRE	FECHA_NACIMIENTO	EDAD	PROFESION
2	MANUEL	22/10/1970	52	POLICIA NACIONAL
3	LUIS	10/09/1972	51	INGENIERIA
4	JOSE	26/04/1956	67	INGENIERIA
5	ALVARO	15/01/1959	64	DOCENCIA
6	ALVARO	03/01/1975	48	DOCENCIA
7	ANA	10/11/1968	54	ARQUITECTURA
8	PAKI	06/10/1968	54	ADMINISTRACION EMPRESAS
9	CLAUDIA	29/06/1966	57	ADMINISTRACION EMPRESAS
10				
11				
12				PROFESION
13				INGENIERIA
14				

```
Sub FiltrarEnOrigen () 'Filtra en la misma Hoja dónde están los Datos

    Application.ThisWorkbook.Sheets(Hoja2.Name).Activate

    Selection.AutoFilter

    Range("A1").CurrentRegion.Select

    Selection.AdvancedFilter Action:=xlFilterInPlace, _
    CriteriaRange:=Range("D12:D13"), Unique:=False

    Range("A1").Select

End Sub
```



	A	B	C	D
1	NOMBRE	FECHA_NACIMIENTO	EDAD	PROFESION
3	LUIS	10/09/1972	51	INGENIERIA
4	JOSE	26/04/1956	67	INGENIERIA
10				
11				
12				PROFESION
13				INGENIERIA
14				

Ahora voy a filtrar aquellos Registros cuya ***EDAD*** sea ***>=50 y <=65*** y quiero que la Tabla se muestre Ordenada (**Ascendente**) por el Campo ***EDAD***.

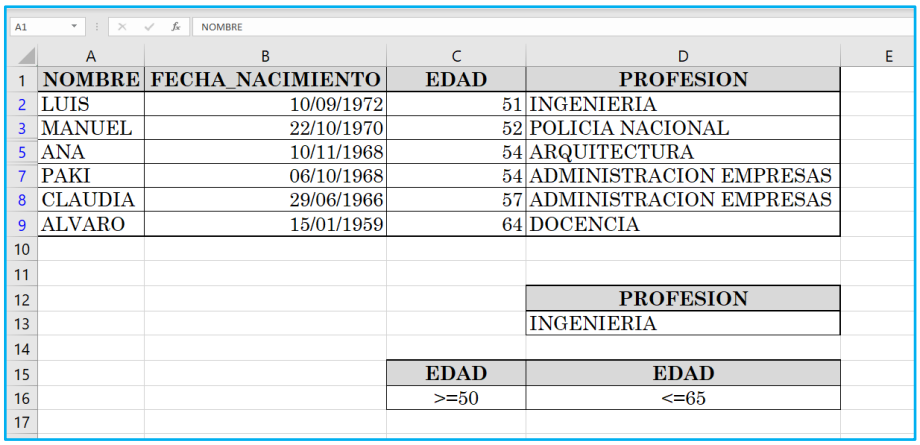


NOMBRE	FECHA_NACIMIENTO	EDAD	PROFESION
MANUEL	22/10/1970	52	POLICIA NACIONAL
LUIS	10/09/1972	51	INGENIERIA
JOSE	26/04/1956	67	INGENIERIA
ALVARO	15/01/1959	64	DOCENCIA
ALVARO	03/01/1975	48	DOCENCIA
ANA	10/11/1968	54	ARQUITECTURA
PAKI	06/10/1968	54	ADMINISTRACION EMPRESAS
CLAUDIA	29/06/1966	57	ADMINISTRACION EMPRESAS

PROFESION
INGENIERIA

EDAD	EDAD
>=50	<=65

```
Sub FiltrarEnOrigen() 'Filtra en la misma Hoja dónde están los Datos

    Application.ThisWorkbook.Sheets(Hoja2.Name).Activate

    Selection.AutoFilter

    Range("A1").CurrentRegion.Select

    Selection.AdvancedFilter Action:=xlFilterInPlace, _
    CriteriaRange:=Range("C15:D16"), Unique:=False

    Application.Sheets(Hoja2.Name).Range("A1").Sort
key1:=Application.Worksheets("Hoja2").Range("C1"), order1:=xlAscending,
Header:=xlYes

    Range("A1").Select

End Sub
```

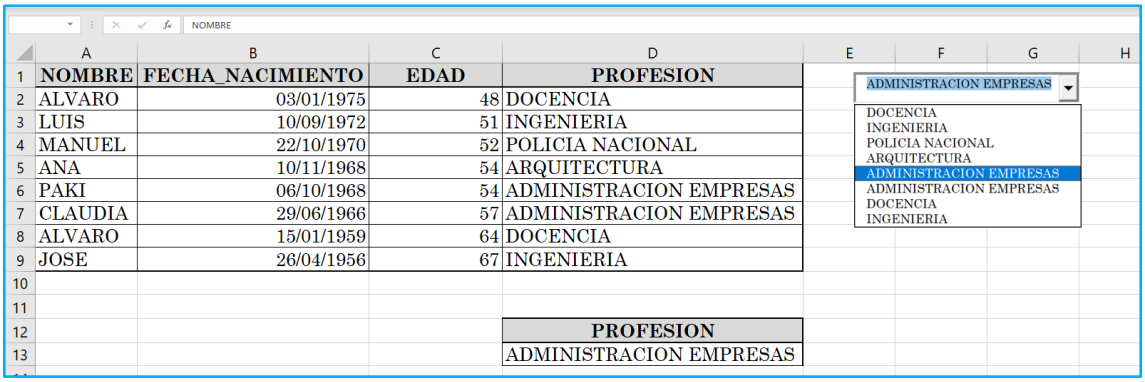


NOMBRE	FECHA_NACIMIENTO	EDAD	PROFESION
LUIS	10/09/1972	51	INGENIERIA
MANUEL	22/10/1970	52	POLICIA NACIONAL
ANA	10/11/1968	54	ARQUITECTURA
PAKI	06/10/1968	54	ADMINISTRACION EMPRESAS
CLAUDIA	29/06/1966	57	ADMINISTRACION EMPRESAS
ALVARO	15/01/1959	64	DOCENCIA

PROFESION
INGENIERIA

EDAD	EDAD
>=50	<=65

Ahora insertaré un **ComboBox (ActiveX)** llamado ***cboProfesion*** que lee el Campo **PROFESION**. Modificaré la ***Propiedad ListFillRange = Hoja2!$D$2:$D$9*** y la **Propiedad *LinkedCell = Hoja2D12$D$13***, para que aparezca el valor que seleccione manualmente.

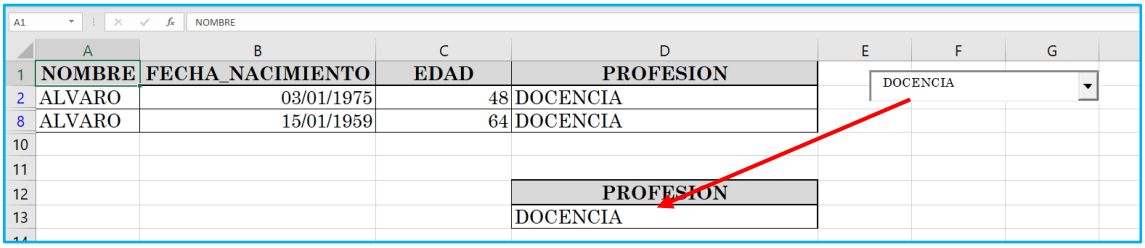


NOMBRE	FECHA_NACIMIENTO	EDAD	PROFESION
ALVARO	03/01/1975	48	DOCENCIA
LUIS	10/09/1972	51	INGENIERIA
MANUEL	22/10/1970	52	POLICIA NACIONAL
ANA	10/11/1968	54	ARQUITECTURA
PAKI	06/10/1968	54	ADMINISTRACION EMPRESAS
CLAUDIA	29/06/1966	57	ADMINISTRACION EMPRESAS
ALVARO	15/01/1959	64	DOCENCIA
JOSE	26/04/1956	67	INGENIERIA

PROFESION
ADMINISTRACION EMPRESAS

Ahora cada vez que elija un valor del **ComboBox**, ejecutaré el Filtro, de **forma manual**.

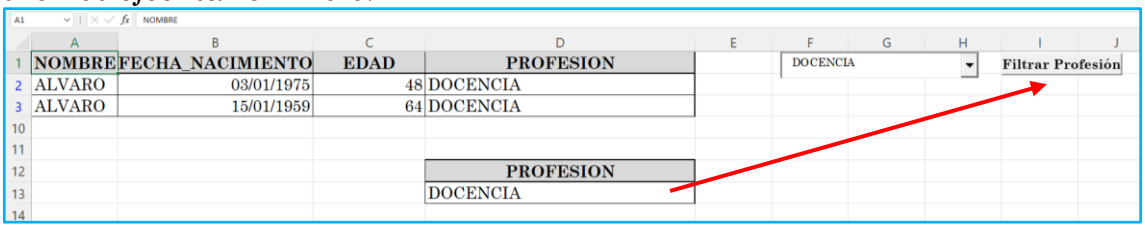
```
Sub FiltrarEnOrigen() 'Filtra en la misma Hoja dónde están los Datos

    Application.ThisWorkbook.Sheets(Hoja2.Name).Activate

    Range("A1").CurrentRegion.Select

    Selection.AdvancedFilter Action:=xlFilterInPlace,
CriteriaRange:=Sheets(Hoja2.Name).Range("D12:D13"), Unique:=False

    Application.Sheets(Hoja2.Name).Range("A1").Sort
key1:=Application.Worksheets("Hoja2").Range("A1"), order1:=xlAscending,
Header:=xlYes

    Range("A1").Select

End Sub
```



NOMBRE	FECHA_NACIMIENTO	EDAD	PROFESION
ALVARO	03/01/1975	48	DOCENCIA
ALVARO	15/01/1959	64	DOCENCIA

PROFESION
DOCENCIA

Si ahora asociamos la **Macro *FiltrarEnOrigen*** a un **Botón de Formulario**, podremos ejecutar el Filtro.



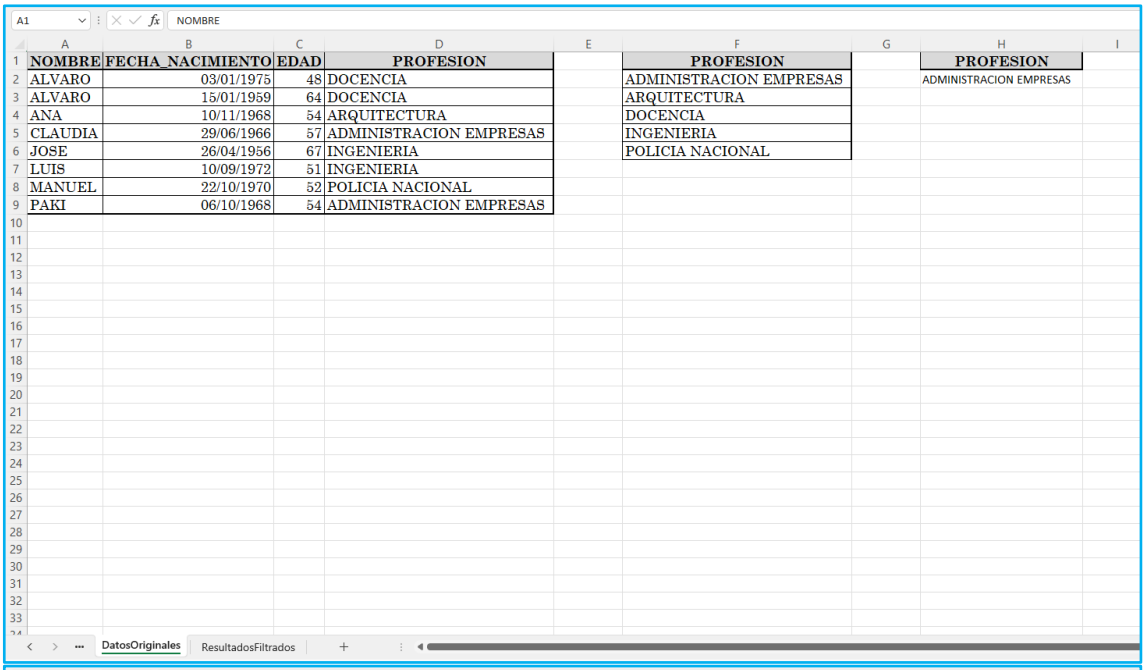
NOMBRE	FECHA_NACIMIENTO	EDAD	PROFESION
ALVARO	03/01/1975	48	DOCENCIA
ALVARO	15/01/1959	64	DOCENCIA

PROFESION
DOCENCIA

El **Ejercicio Propuesto** consiste en generar un Código en el ***Módulo2*** que ejecute el **Filtro automáticamente** cada vez que se selecciona un valor del ***ComboBox***. Para ello cumplirás lo siguiente.

- Dentro del archivo ***EJERCICIO0902.xlsm***, crearás las **Hojas *Hoja3(DatosOriginales)*** y ***Hoja4(DatosFiltrados)***.

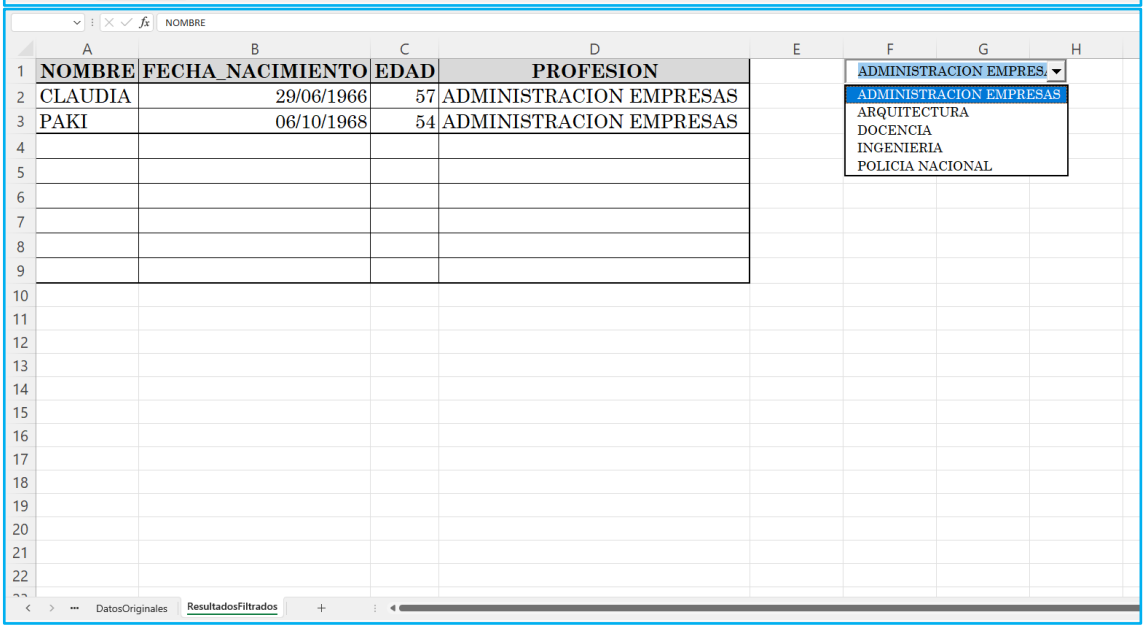
- El Filtro lo harás por el **Campo *PROFESION***.
- En la **Hoja *DatosOriginales*** tendrás, además de la Tabla Original, el **Campo *PROFESION* sin Duplicados (ordenada Ascendente)** y el Campo ***PROFESION*** con el valor del ***ComboBox.Value*** que usarás como Filtro.
- En la **Hoja *DatosFiltrados*** tendrás la Tabla Filtrada, junto con el ***ComboBox*** que usarás para elegir el valor por el que quieres Filtrar.
- Cuando borres el valor del **ComboBox**, te aparecerá la **Tabla Completa** (sin Filtro).
- Cada vez que ejecutes el Filtro, se deberá **Ordenar** la Tabla Filtrada por el **Campo *NOMBRE***.
- El **Evento *Workbook_Open*** del archivo, será el que actualice el Campo ***PROFESION* sin Duplicados**. De modo que, si añades nuevas Profesiones, deberás **Cerrar\Abrir** el archivo.



NOMBRE	FECHA_NACIMIENTO	EDAD	PROFESION
ALVARO	03/01/1975	48	DOCENCIA
ALVARO	15/01/1959	64	DOCENCIA
ANA	10/11/1968	54	ARQUITECTURA
CLAUDIA	29/06/1966	57	ADMINISTRACION EMPRESAS
JOSE	26/04/1956	67	INGENIERIA
LUIS	10/09/1972	51	INGENIERIA
MANUEL	22/10/1970	52	POLICIA NACIONAL
PAKI	06/10/1968	54	ADMINISTRACION EMPRESAS

PROFESION
ADMINISTRACION EMPRESAS
ARQUITECTURA
DOCENCIA
INGENIERIA
POLICIA NACIONAL

PROFESION
ADMINISTRACION EMPRESAS



NOMBRE	FECHA_NACIMIENTO	EDAD	PROFESION
CLAUDIA	29/06/1966	57	ADMINISTRACION EMPRESAS
PAKI	06/10/1968	54	ADMINISTRACION EMPRESAS

ADMINISTRACION EMPRES.

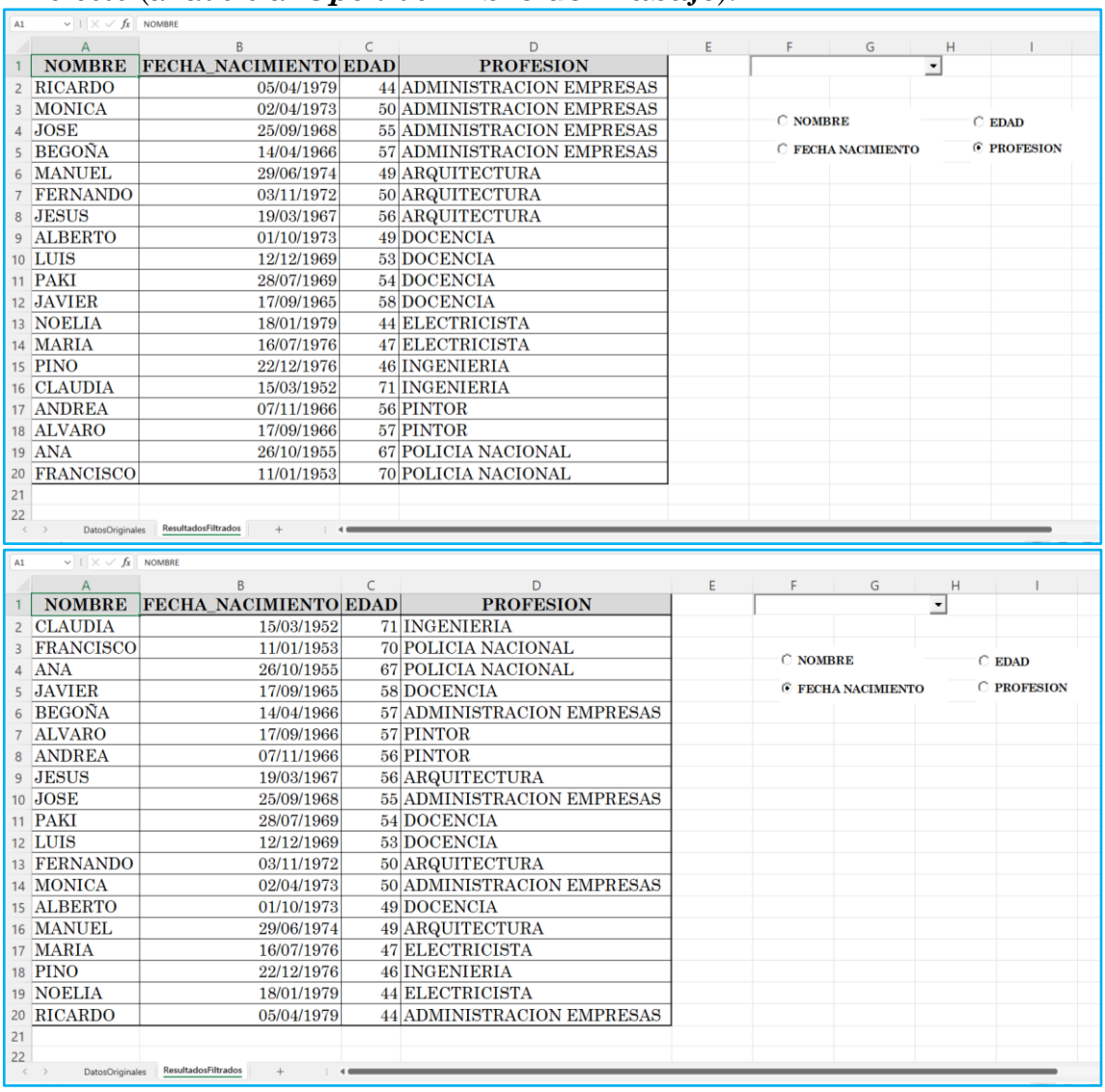
ADMINISTRACION EMPRESAS
ARQUITECTURA
DOCENCIA
INGENIERIA
POLICIA NACIONAL

NOTA. Ten presente **Borrar** el Valor del **ComboBox** (***ComboBox.Value =""*)**, nada más abrir el archivo.

Ejercicio 09.03.-

Haz una copia del archivo ***EJERCICIO0902.xlsm*** con el nombre ***EJERCICIO0903.xlsm***. Guárdalo y quédate únicamente con las **Hojas *DatosOriginales*** y ***ResultadosFiltrados***.

El **Ejercicio Propuesto** consiste en añadir un grupo de **Botones de Opción** que permite **Ordenar por cualquiera de los 4 Campos**. El **Campo *NOMBRE*** será por Defecto (añádelo al ***Open*** del **Libro de Trabajo**).



NOMBRE	FECHA_NACIMIENTO	EDAD	PROFESION
RICARDO	05/04/1979	44	ADMINISTRACION EMPRESAS
MONICA	02/04/1973	50	ADMINISTRACION EMPRESAS
JOSE	25/09/1968	55	ADMINISTRACION EMPRESAS
BEGOÑA	14/04/1966	57	ADMINISTRACION EMPRESAS
MANUEL	29/06/1974	49	ARQUITECTURA
FERNANDO	03/11/1972	50	ARQUITECTURA
JESUS	19/03/1967	56	ARQUITECTURA
ALBERTO	01/10/1973	49	DOCENCIA
LUIS	12/12/1969	53	DOCENCIA
PAKI	28/07/1969	54	DOCENCIA
JAVIER	17/09/1965	58	DOCENCIA
NOELIA	18/01/1979	44	ELECTRICISTA
MARIA	16/07/1976	47	ELECTRICISTA
PINO	22/12/1976	46	INGENIERIA
CLAUDIA	15/03/1952	71	INGENIERIA
ANDREA	07/11/1966	56	PINTOR
ALVARO	17/09/1966	57	PINTOR
ANA	26/10/1955	67	POLICIA NACIONAL
FRANCISCO	11/01/1953	70	POLICIA NACIONAL

NOMBRE EDAD FECHA NACIMIENTO PROFESION

NOMBRE	FECHA_NACIMIENTO	EDAD	PROFESION
CLAUDIA	15/03/1952	71	INGENIERIA
FRANCISCO	11/01/1953	70	POLICIA NACIONAL
ANA	26/10/1955	67	POLICIA NACIONAL
JAVIER	17/09/1965	58	DOCENCIA
BEGOÑA	14/04/1966	57	ADMINISTRACION EMPRESAS
ALVARO	17/09/1966	57	PINTOR
ANDREA	07/11/1966	56	PINTOR
JESUS	19/03/1967	56	ARQUITECTURA
JOSE	25/09/1968	55	ADMINISTRACION EMPRESAS
PAKI	28/07/1969	54	DOCENCIA
LUIS	12/12/1969	53	DOCENCIA
FERNANDO	03/11/1972	50	ARQUITECTURA
MONICA	02/04/1973	50	ADMINISTRACION EMPRESAS
ALBERTO	01/10/1973	49	DOCENCIA
MANUEL	29/06/1974	49	ARQUITECTURA
MARIA	16/07/1976	47	ELECTRICISTA
PINO	22/12/1976	46	INGENIERIA
NOELIA	18/01/1979	44	ELECTRICISTA
RICARDO	05/04/1979	44	ADMINISTRACION EMPRESAS

NOMBRE EDAD FECHA NACIMIENTO PROFESION

Ejercicio 09.04.-
Crea el archivo ***EJERCICIO0904.xlsm*** y guárdalo.

El **Ejercicio Propuesto** consiste en hacer un programa que, inicialmente, genere una Tabla con los Nombres que hay debajo, cuyas **Fechas de Nacimiento** será un **valor aleatorio** generado por VBA Excel **entre 2 fechas** establecidas.
ValorInferior = "01/01/1950" y ValorSuperior = "01/01/1980"

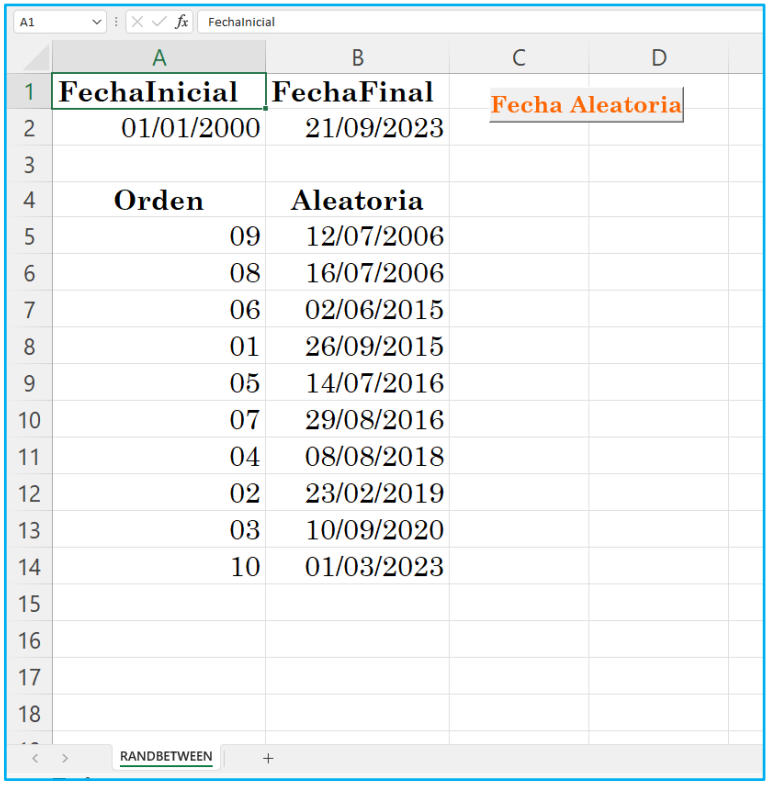
Hay varias maneras de generar fechas aleatorias. Yo he empleado el **Método *WorksheetFunction.RandBetween*** que **devuelve un número entero aleatorio** entre los números que se especifique. Cada vez que se recalcula la Hoja de Cálculo, se genera un nuevo valor. Entre los Parámetros que permite la función ***ALEATORIO.ENTRE (RANDBETWEEN)*** tenemos los siguientes.

- Valor Inferior (obligatorio): Límite Inferior de los números aleatorios.
- Valor Superior (obligatorio): Límite Superior de los números aleatorios.

El **Método *RandBetween*** genera un número aleatorio, pero se puede usar para elegir entre números o entre fechas, por ejemplo. Debajo añado un ejemplo declarando un Objeto Hoja de Cálculo e introduciendo 2 fechas en A2 y B2. Luego le indico al programa que busque 10 fechas aleatorias dentro de este Siglo, hasta la fecha de hoy. Finalmente, se ordenan por antigüedad.

```
Sub MetodoRANDBETWEEN()
Dim Hoja As Worksheet, FechaInicial As Date, FechaFinal As Date, i As Integer

Set Hoja = Worksheets("RANDBETWEEN")

Application.ThisWorkbook.Sheets(Hoja1.Name).Activate

Cells(1, 1).Value = "FechaInicial"
Cells(1, 2).Value = "FechaFinal"
Range(Cells(2, 1), Cells(2, 2)).NumberFormat = "dd/mm/yyyy"
Cells(2, 1).Value = "01/01/2000"
Cells(2, 2).Value = Date

FechaInicial = Hoja.Range("A2")
FechaFinal = Hoja.Range("B2")

Cells(4, 1).Value = "Orden"
Cells(4, 2).Value = "Aleatoria"

   For i = 5 To 14
      Cells(i, 1).NumberFormat = "00"
      Cells(i, 1).Value = i - 4
      Cells(i, 2).NumberFormat = "dd/mm/yyyy"
      Cells(i, 2).Value = Application.WorksheetFunction.RandBetween(FechaInicial,
FechaFinal)
   Next

   Application.Sheets(Hoja1.Name).Range("A5").Sort key1:=Range("B5"),
order1:=xlAscending, Header:=xlYes

   Cells(1, 1).Select
```

End Sub



El **Procedimiento** principal se llamará ***GenerarDatos*** y estará dentro de ***Módulo1***. El Libro tendrá dos **Hojas** (***Datos*** y ***Filtrado***). Este **Procedimiento** llamará a la **Función *GenerarDatosAleatorios*** que, a su vez, devolverá un **Objeto** tipo **Matriz** llamado ***MatrizEdad(1 To Recuento, 1 To 4)***, donde ***Recuento*** es el número de Registros que tiene la Tabla. La **Función** se guardará en el **Módulo** llamado ***Funciones***.

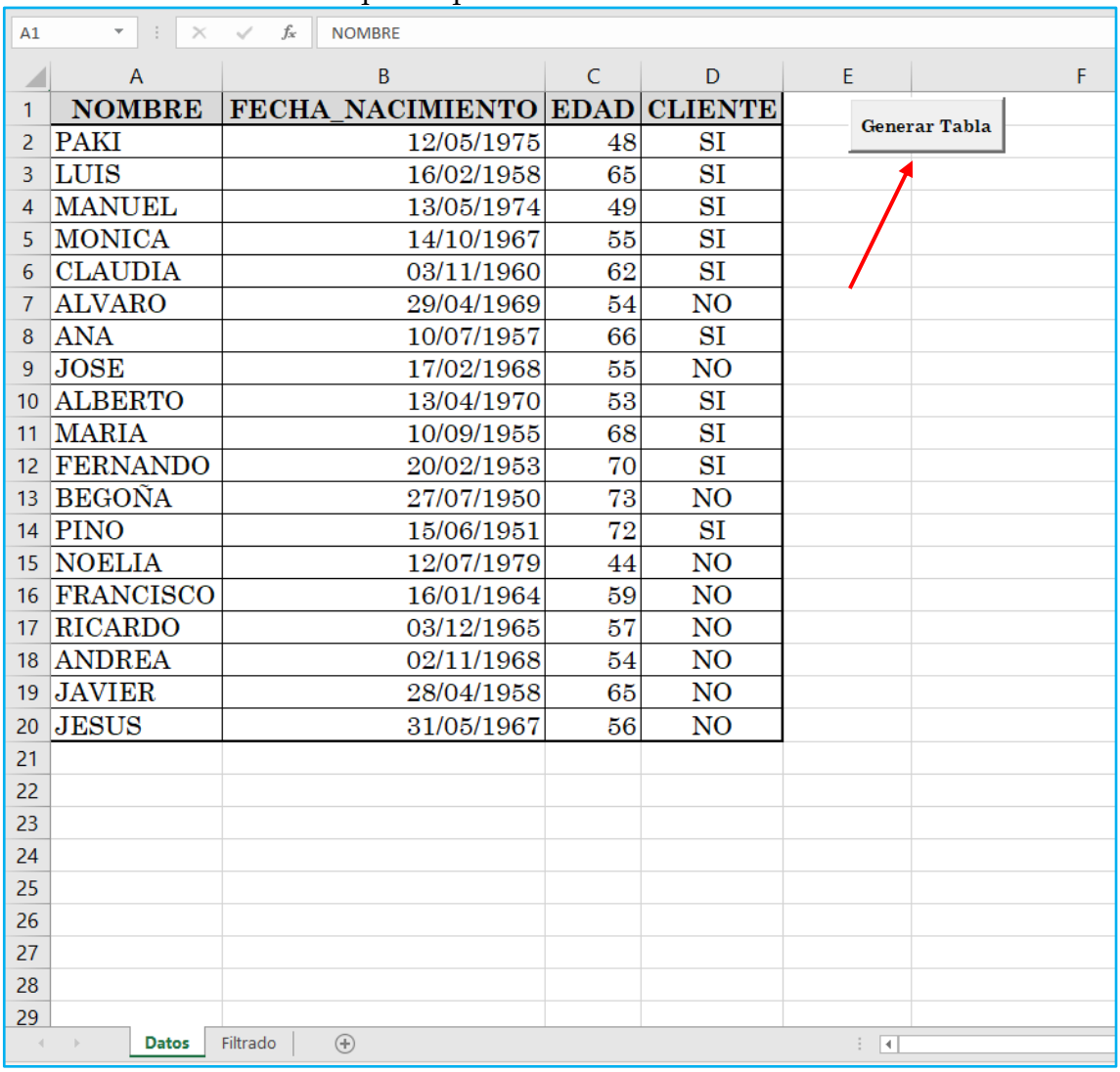
El valor del **Campo *CLIENTE*** es **SI\NO** obtenido, también **aleatoriamente**, como producto de elegir **un número entre el 1-100**, el cual, si es **PAR**, el valor será **SI** y, en caso contrario, **NO**.

NOMBRE	FECHA_NACIMIENTO	EDAD	CLIENTE
PAKI	30/06/1952	71	NO
LUIS	25/05/1962	61	SI
MANUEL	05/08/1961	62	SI
MONICA	13/12/1968	54	NO
CLAUDIA	11/07/1967	56	NO
ALVARO	20/04/1957	66	SI
ANA	03/09/1950	73	SI
JOSE	26/07/1963	60	NO
ALBERTO	18/07/1954	69	SI
MARIA	02/02/1957	66	NO
FERNANDO	11/09/1977	46	NO

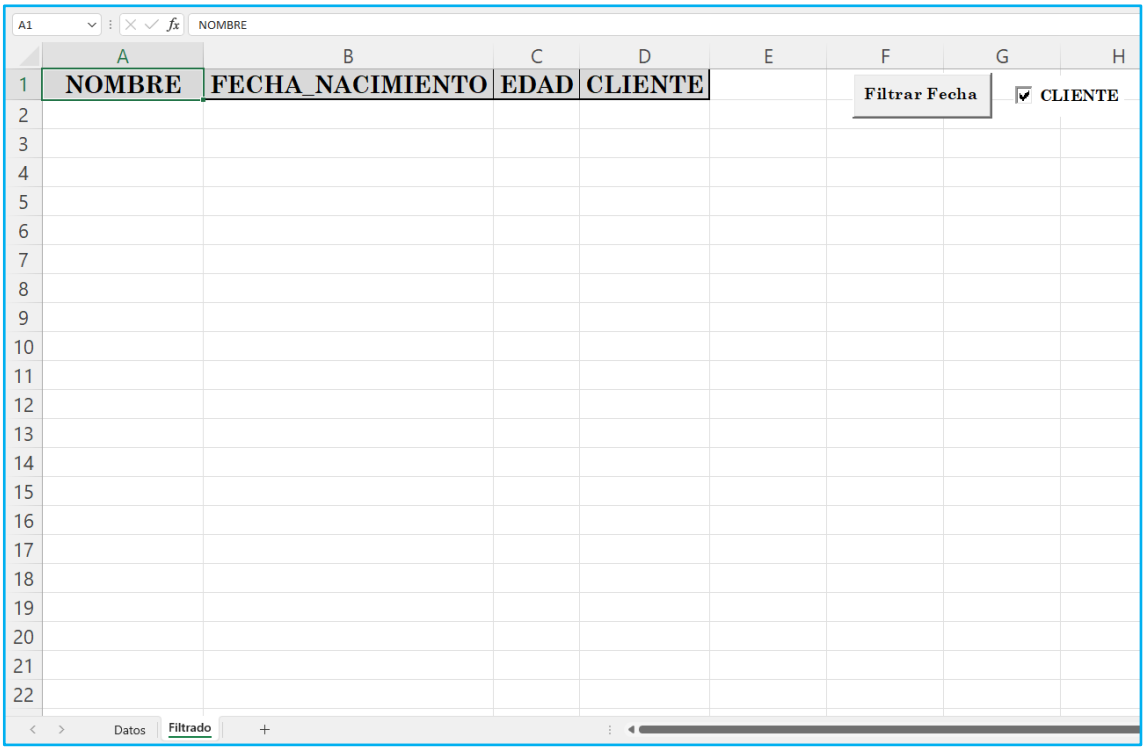
BEGOÑA	18/09/1953	70	SI
PINO	27/06/1978	45	NO
NOELIA	19/05/1973	50	SI
FRANCISCO	03/08/1976	47	SI
RICARDO	24/04/1951	72	NO
ANDREA	02/03/1956	67	SI
JAVIER	18/03/1977	46	SI
JESUS	18/08/1953	70	NO

Llegado a este punto, el programa nos pedirá **2 fechas** y activar (o no) un ***CheckBox*** que servirá para Filtrar aquellos Registros cuya **Fecha de Nacimiento** esté comprendida entre esas dos fechas dadas y el ***CheckBox*** nos dirá si el **Campo *CLIENTE*** es **SI\NO**. Se deberá cumplir todas las condiciones para filtrar.

El Programa dispondrá de un **Botón de Comando ActiveX** llamado ***cmbGenerarTabla*** que llamará al **Procedimiento *GenerarDatos*** y generará una nueva Tabla cada vez que se pulse.

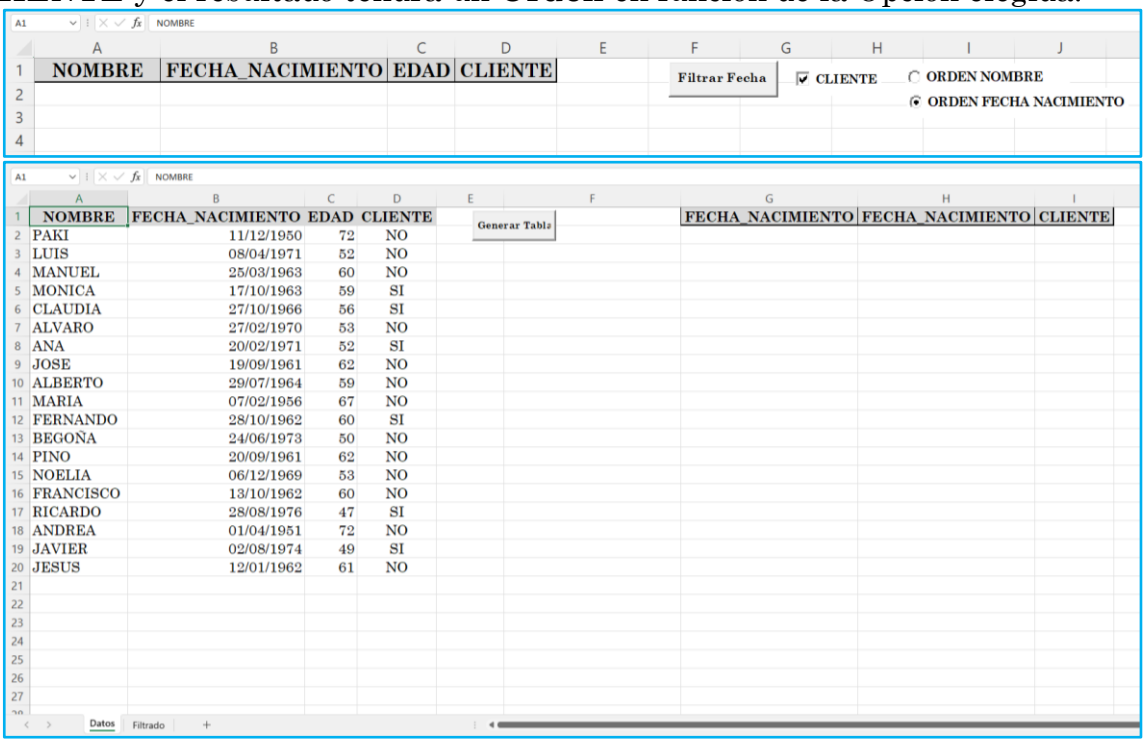


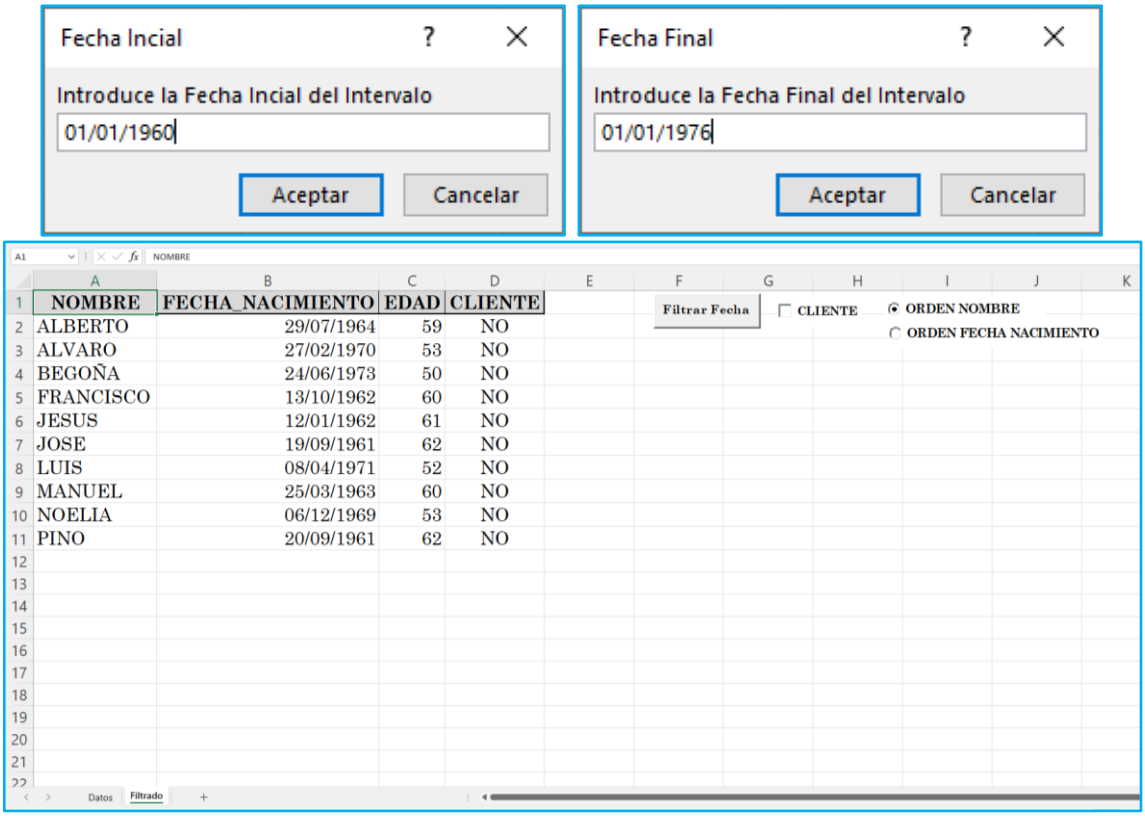
	NOMBRE	FECHA_NACIMIENTO	EDAD	CLIENTE
2	PAKI	12/05/1975	48	SI
3	LUIS	16/02/1958	65	SI
4	MANUEL	13/05/1974	49	SI
5	MONICA	14/10/1967	55	SI
6	CLAUDIA	03/11/1960	62	SI
7	ALVARO	29/04/1969	54	NO
8	ANA	10/07/1957	66	SI
9	JOSE	17/02/1968	55	NO
10	ALBERTO	13/04/1970	53	SI
11	MARIA	10/09/1955	68	SI
12	FERNANDO	20/02/1953	70	SI
13	BEGOÑA	27/07/1950	73	NO
14	PINO	15/06/1951	72	SI
15	NOELIA	12/07/1979	44	NO
16	FRANCISCO	16/01/1964	59	NO
17	RICARDO	03/12/1965	57	NO
18	ANDREA	02/11/1968	54	NO
19	JAVIER	28/04/1958	65	NO
20	JESUS	31/05/1967	56	NO



El **Procedimiento** que ejecutaremos se llama ***FiltrarFechas***, asociado al **Botón de Comando ActiveX *cmdFiltrarFechas***.

Además de **Filtrar por las Fechas Introducidas**, lo hará por el **Campo *CLIENTE*** y el resultado tendrá un **Orden** en función de la Opción elegida.





NOMBRE	FECHA_NACIMIENTO	EDAD	CLIENTE
ALBERTO	29/07/1964	59	NO
ALVARO	27/02/1970	53	NO
BEGOÑA	24/06/1973	50	NO
FRANCISCO	13/10/1962	60	NO
JESUS	12/01/1962	61	NO
JOSE	19/09/1961	62	NO
LUIS	08/04/1971	52	NO
MANUEL	25/03/1963	60	NO
NOELIA	06/12/1969	53	NO
PINO	20/09/1961	62	NO

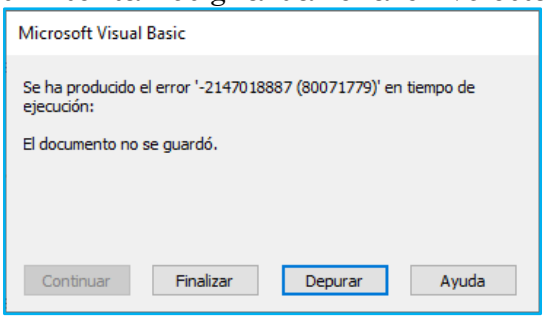
Observa cómo el Filtro se ha escrito al pulsar el **Botón *Filtrar Fechas***.

FECHA_NACIMIENTO	FECHA_NACIMIENTO	CLIENTE
>=01/01/1960	<=01/01/1976	NO

Ejercicio 09.05.-
Haz una copia del archivo ***EJERCICIO0904.xlsm*** con el nombre ***EJERCICIO0905.xlsm*** y guárdalo.

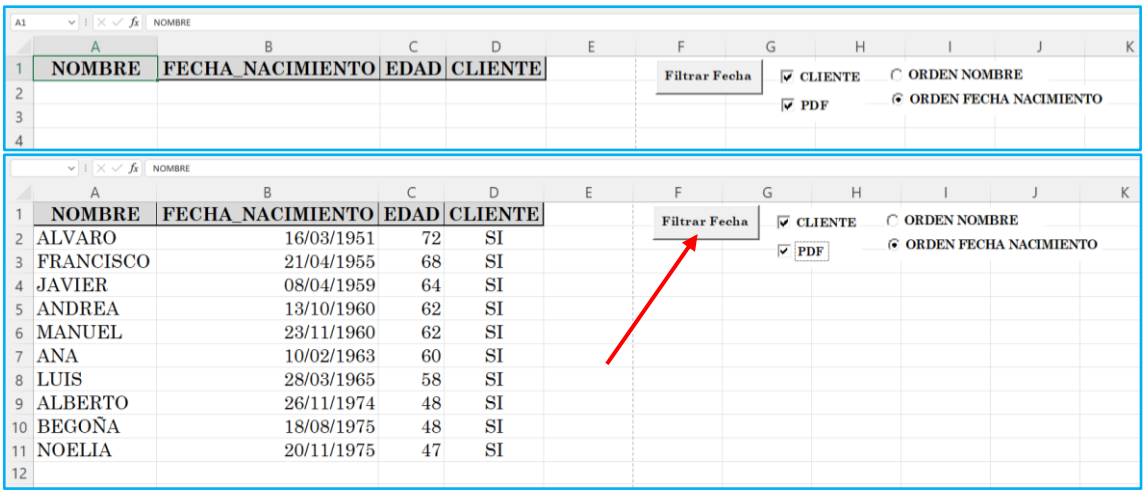
El **Ejercicio Propuesto** consiste en generar un archivo **PDF** con el listado obtenido. La forma de funcionamiento consiste en pulsar el **Botón *Filtrar Fecha*** y, previamente, seleccionar la **Casilla de Verificación *chkPDF*** y haber elegido un Orden concreto.

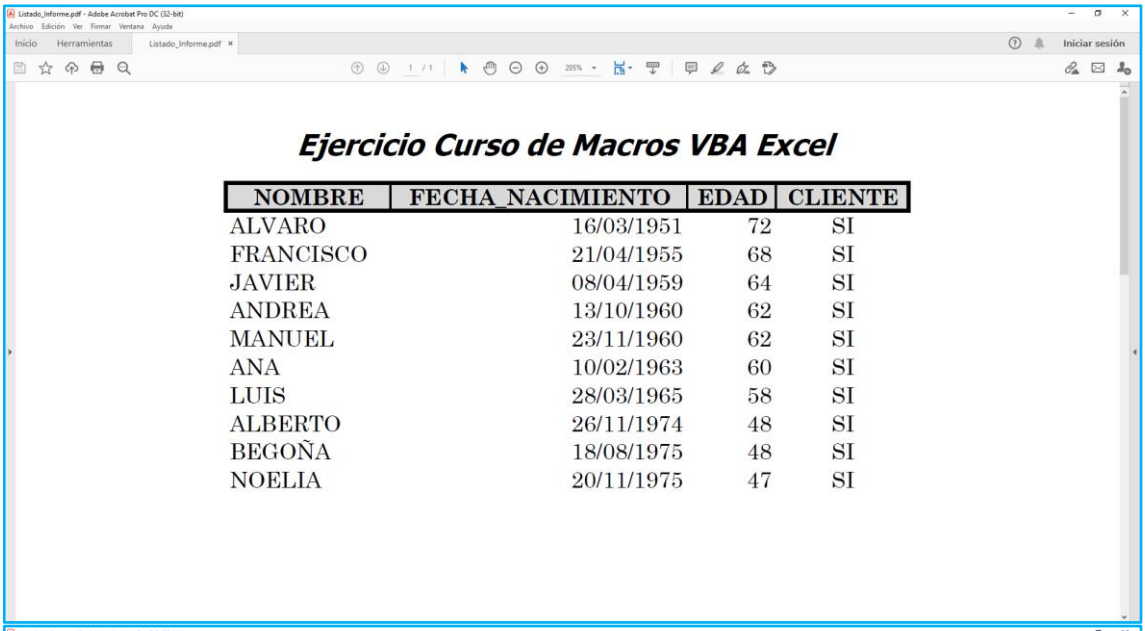
Podemos usar el **Método *ExportAsFixedFormat*** o bien el **Método *PrintOut***. Además, incluiremos una rutina de Control de Error para evitar el siguiente Error que se puede producir si intentamos guardar el archivo estando ya abierto.



Los **Parámetros** que usaremos para configurar el **Objeto *PageSetup*** que representa la descripción de la **configuración de página**, serán los siguientes.

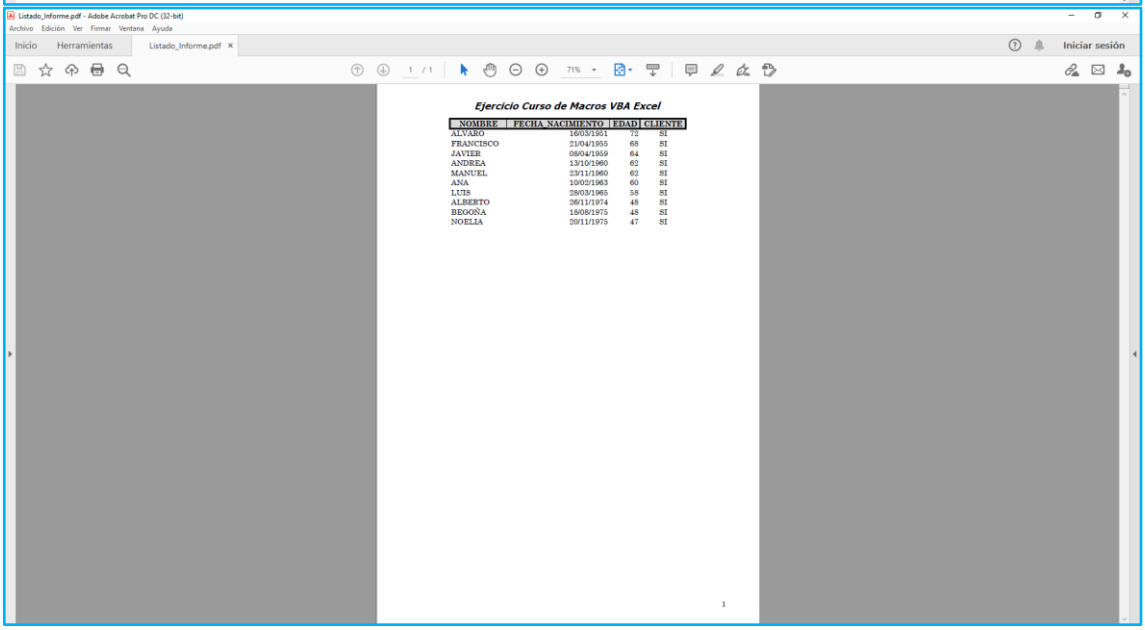
```
With Hoja2.PageSetup
   .PrintQuality = 600
   .CenterHorizontally = True
   .CenterVertically = False
   .Orientation = xlPortrait
   .PaperSize = xlPaperA4
   .CenterHeader = "&""Tahoma,Negrita Cursiva""&16Ejercicio Curso de Macros VBA Excel"
   .RightFooter = "&P"
   .FirstPageNumber = xlAutomatic
   .Zoom = 100
End With
```





Ejercicio Curso de Macros VBA Excel

NOMBRE	FECHA_NACIMIENTO	EDAD	CLIENTE
ALVARO	16/03/1951	72	SI
FRANCISCO	21/04/1955	68	SI
JAVIER	08/04/1959	64	SI
ANDREA	13/10/1960	62	SI
MANUEL	23/11/1960	62	SI
ANA	10/02/1963	60	SI
LUIS	28/03/1965	58	SI
ALBERTO	26/11/1974	48	SI
BEGOÑA	18/08/1975	48	SI
NOELIA	20/11/1975	47	SI



Ejercicio Curso de Macros VBA Excel

NOMBRE	FECHA_NACIMIENTO	EDAD	CLIENTE
ALVARO	16/03/1951	72	SI
FRANCISCO	21/04/1955	68	SI
JAVIER	08/04/1959	64	SI
ANDREA	13/10/1960	62	SI
MANUEL	23/11/1960	62	SI
ANA	10/02/1963	60	SI
LUIS	28/03/1965	58	SI
ALBERTO	26/11/1974	48	SI
BEGOÑA	18/08/1975	48	SI
NOELIA	20/11/1975	47	SI

10.- FORMULARIOS DE USUARIOS (USERFORMS)

Ejercicio 10.01.-

Los Formularios son Cuadros de Diálogo que contienen una serie de **Controles ActiveX (Cuadros de Texto, Listas Desplegables, ComboBox, Etiquetas, etc.)**, a los que asociaremos código VBA Excel para crear nuestros programas con un entorno gráfico amigable que permita entrada\salida de información.

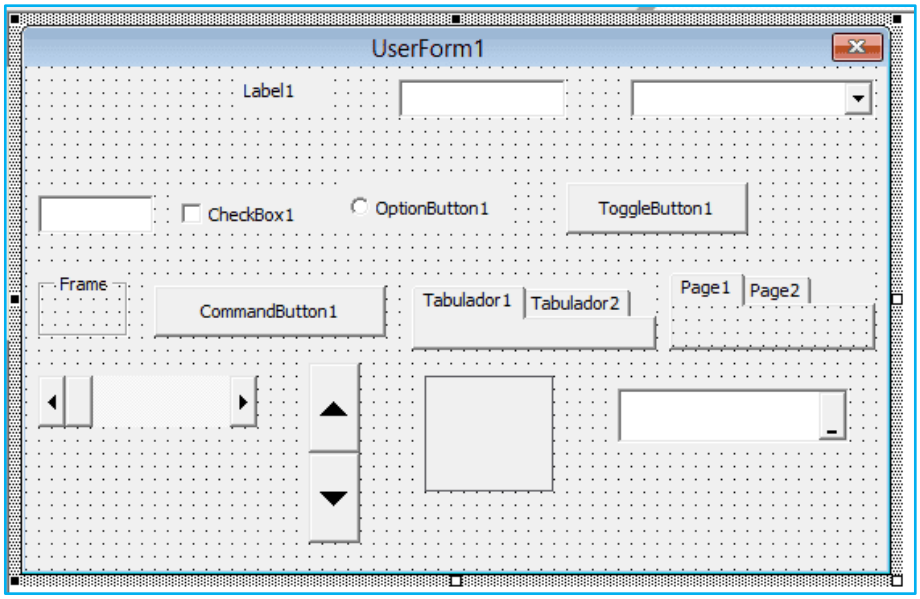
Se trata de Objetos independientes de los Módulos por lo que se almacenan en carpetas distintas dentro de Editor de VBA Excel.

Usaremos los Formulario, principalmente, para lo siguiente.

- Espacio de intercambio para introducir Datos y obtener el Resultado de operaciones de Cálculo (o cualquier otra), realizado con dichos Datos.
- Introducir o Leer Datos de\desde Libros de Trabajo (Hojas de Cálculo).

Existen diferentes tipos de Controles de Formulario que ofrecen diversos tipos de funcionalidad e interacción con el usuario. Disponemos desde una sencilla Etiqueta hasta Controles que permiten Selecciones Múltiple. A continuación, te muestro una breve descripción de algunos de ellos.

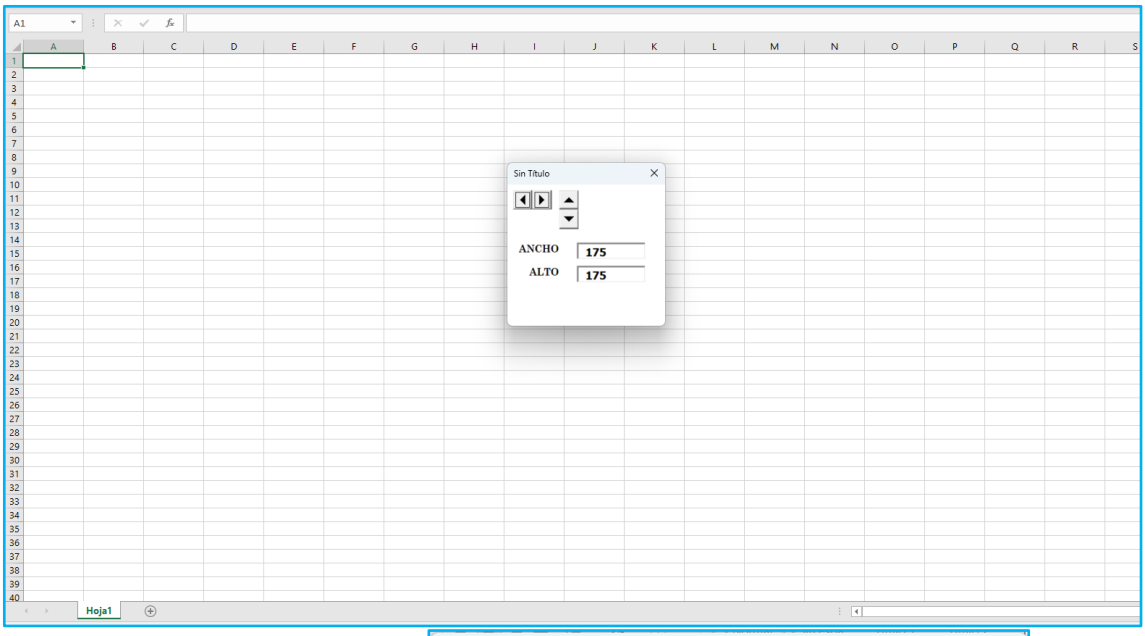
- **Barra de Desplazamiento**. Al hacer click en las flechas se va desplazando la barra dentro de un intervalo predefinido.
- **Botón de Comando**. Permite ejecutar un Procedimiento al pulsarlo.
- **Botón de Opción**. Permite una única selección dentro de un conjunto de opciones.
- **Casilla de Verificación**. Permite la Seleccionar\No Seleccionar una opción.
- **Control de Número**. Nos permite controlar el Aumento\Disminución de un valor numérico.
- **Cuadro Combinado**. Es una combinación de un Cuadro de Texto con un Cuadro de Lista.
- **Cuadro de Lista**. Muestra una Lista de valores entre los que podemos elegir una o múltiples opciones a la vez (según la configuración de las Propiedades).
- **Marco**. Agrupa varios controles dentro de un rectángulo.
- **Etiqueta**. Permite especificar un texto o breves instrucciones en el formulario.

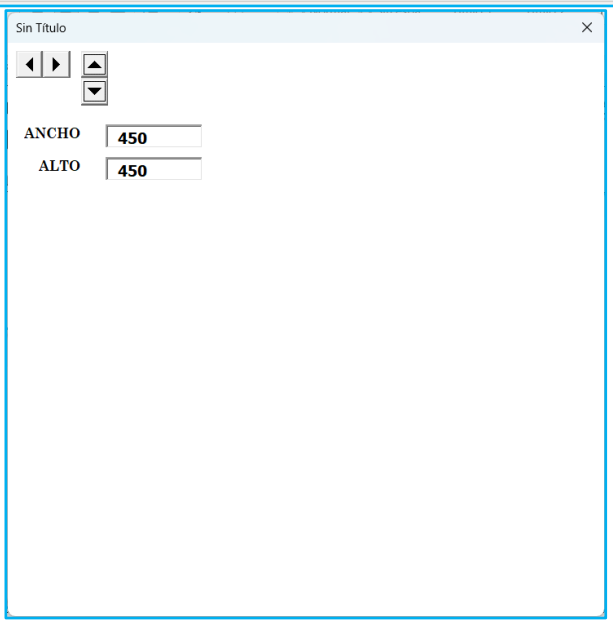


Crea el archivo ***EJERCICIO1001.xlsm*** y guárdalo.

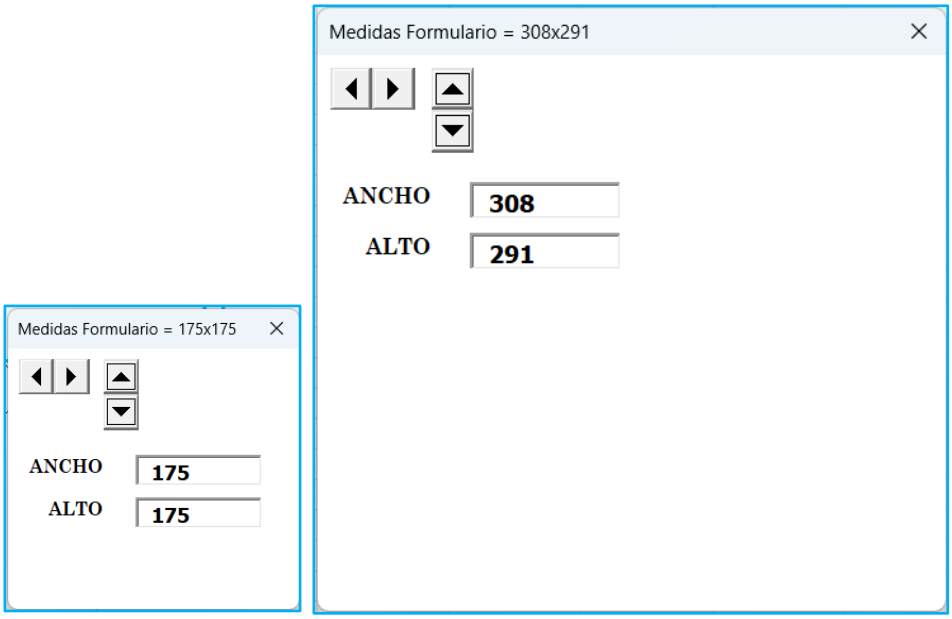
El **Primer Ejercicio Propuesto** es hacer un **Formulario** llamado ***frmFormulario*** que tenga **2 Controles *SpinButton*** (Controles de Número) que nos va a permitir modificar el **Ancho** y el **Alto** del Formulario.

El **Formulario**, inicialmente medirá **175x175 (AnchoxAlto)** puntos y podrá llegar hasta **450x450 puntos**. Dispondrá de **2 Etiquetas** y **2 Cuadros de Texto** que indicarán las medidas del formulario en cada momento.



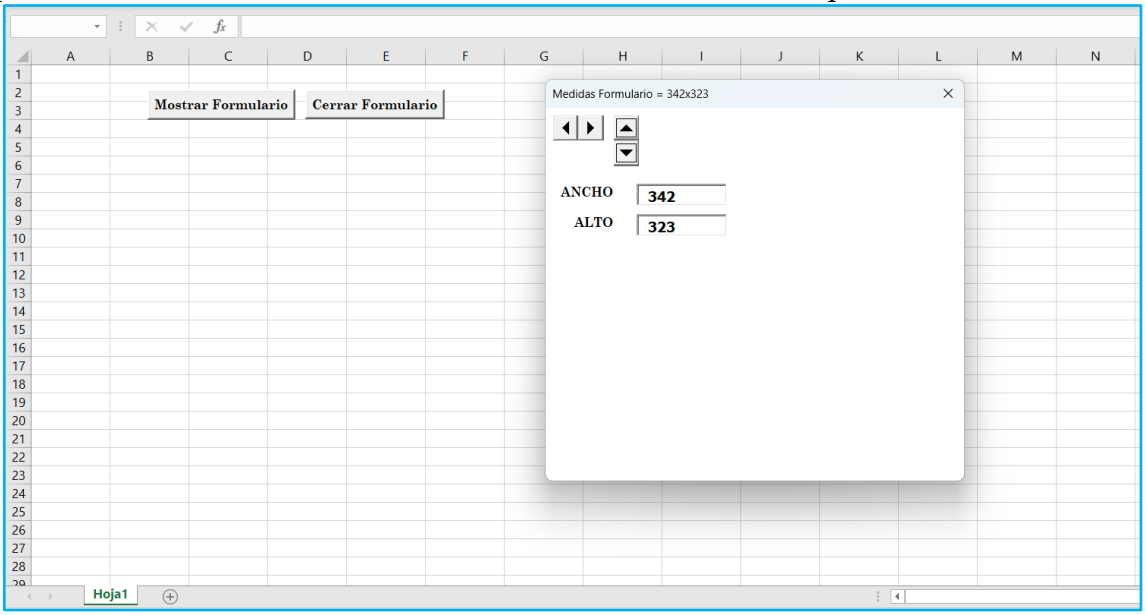


El **Segundo Ejercicio Propuesto** consiste en que, por cada cambio de tamaño del Formulario, su **Propiedad *Caption*** indique sus dimensiones.

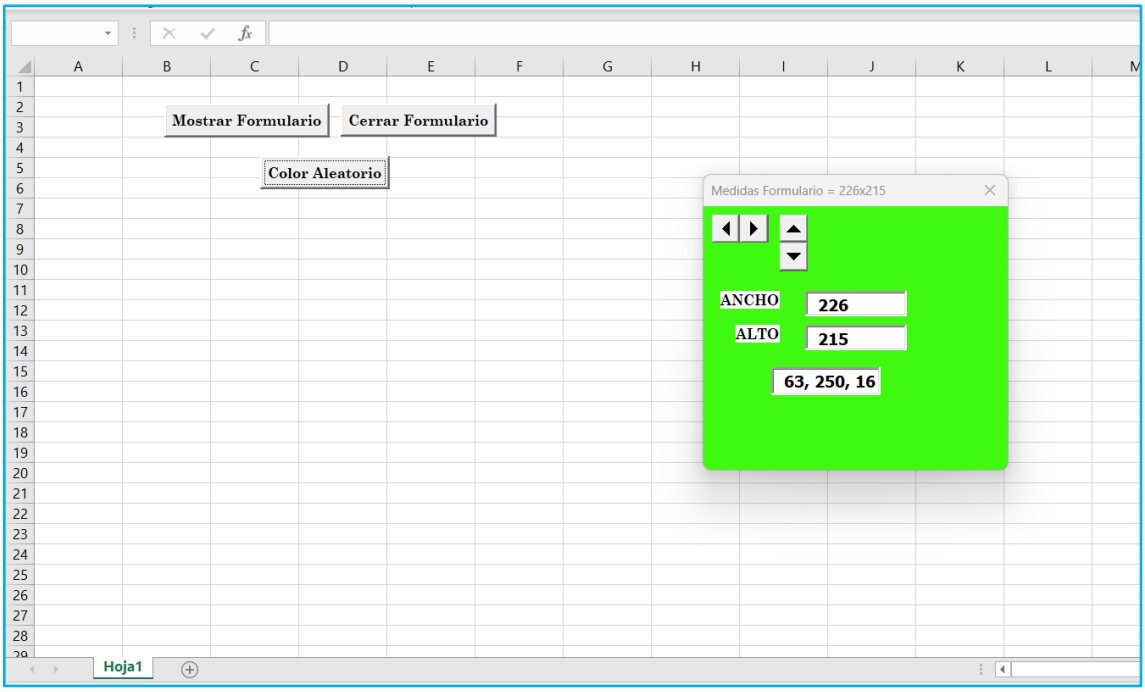


El **Tercer Ejercicio Propuesto** consiste en añadir **2 Botones de Comando ActiveX** en la **Hoja**. Uno de los **Botones Muestra el Formulario** y el otro lo **Cierra** (pero que lo mantendrá en memoria).

NOTA. Ten presente que para que puedas acceder a controles fuera del formulario, **estando activo**, tendrás que modificar la **Propiedad *ShowModal = True\False***. Si es ***Modal (True)***, el usuario deberá proporcionar información o cerrar el formulario para poder usar cualquier otra parte de la aplicación. Puedes cambiar la **Propiedad** a ***No Modal (False)*** ejecutando ***frmFormulario.show 0*** y podrás ver otros formularios u otras ventanas sin tener que cerrar el formulario

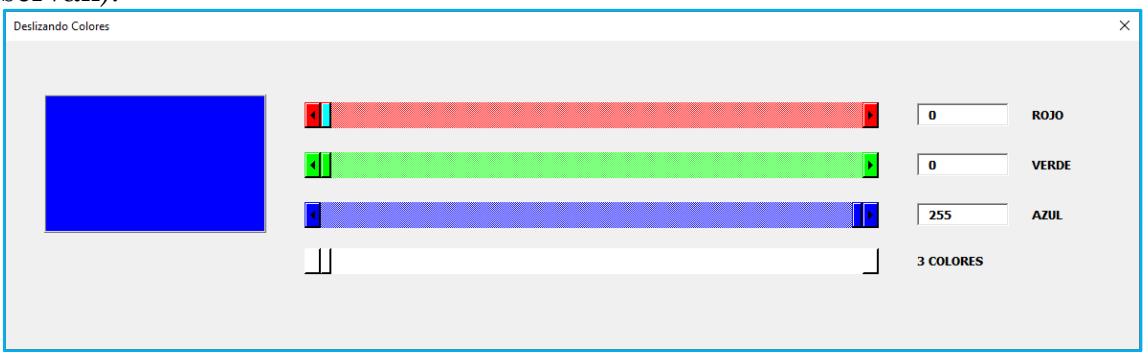


El **Cuarto Ejercicio Propuesto** consiste en crear un Botón que, a cada pulsación, cambie el Color de Fondo del Formulario con **Colores RGB Aleatorios**. Deberá aparecer un **Cuadro de Texto** (***txtColor***) con el formato (***255, 255 ,255***).


Mostrar Formulario
Cerrar Formulario
Color Aleatorio
Medidas Formulario = 226x215
ANCHO
226
ALTO
215
63, 250, 16
Hoja1

Ejercicio 10.02.-
Crea el archivo ***EJERCICIO1002.xlsm*** y guárdalo.

El **Primer Ejercicio Propuesto** tendrá un **Formulario** llamado ***frmColoresRGB*** y dispondrá de los elementos que se observan en la imagen a continuación. La imagen inicial debe ser la siguiente (con los colores que se observan).

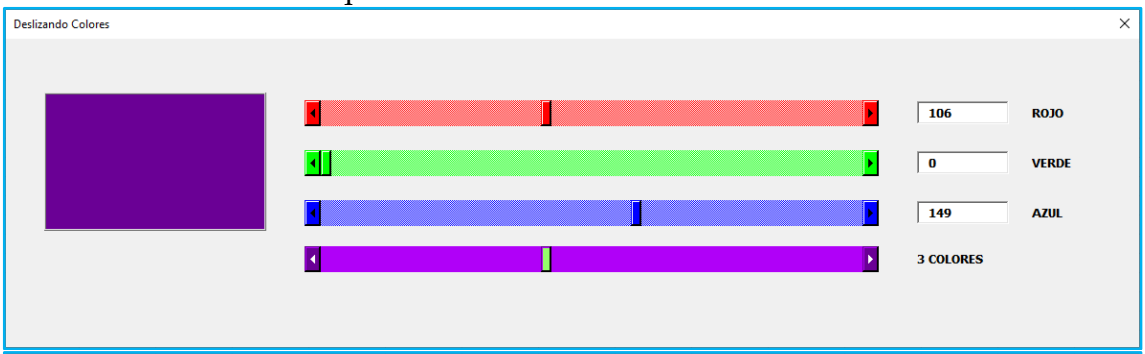


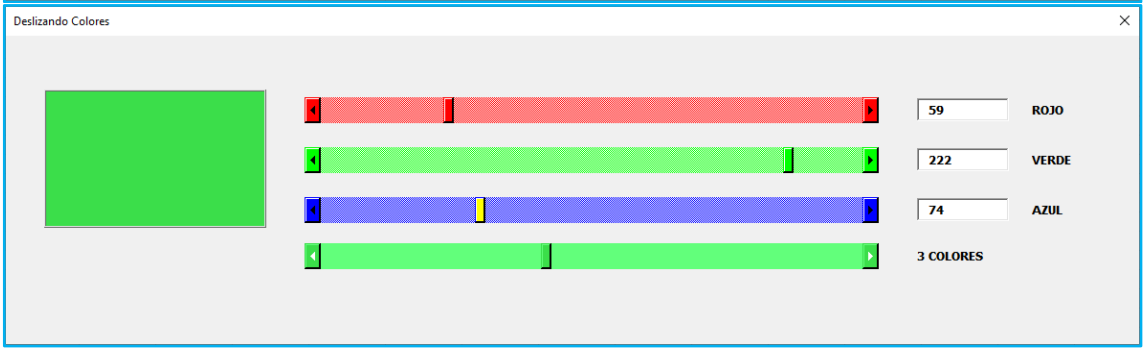
El objetivo del programa es conseguir mostrar la mezcla de colores que va desde el **Azul(0,0,255)** al **Rojo(255,0,0),** partiendo de **RGB(0,0,255)** y para la tinta ***Verde=0*,** en todo momento.

Al arrastrar el deslizador ***3 COLORES*** hacia la derecha, irá aumentando el deslizador del color Rojo e irá disminuyendo el de Azul, hasta llegar al Rojo.

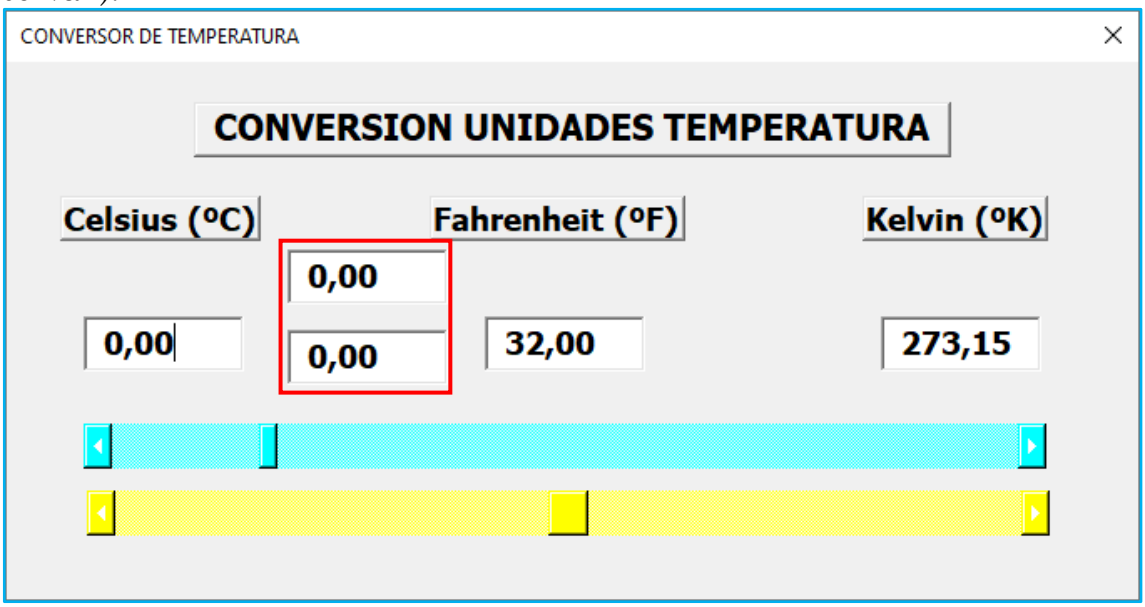
En cualquier momento podremos desplazar los deslizadores de los colores a mano y se reflejará el Color en la muestra, pero en cuanto toquemos el deslizador ***3 COLORES***, el **Verde volverá a valor 0**.

Cada deslizador tendrá el Color que le corresponde por su nombre, salvo el deslizador ***3 COLORES*** que tendrá el Color de la muestra en ese momento.





El **Segundo Ejercicio Propuesto** tendrá un **Formulario** llamado ***frmTemperatura*** y dispondrá de los elementos que se observan en la imagen a continuación. La imagen inicial debe ser la siguiente (con los colores que se observan).



NOTA. Los **2 Cuadros de Textos** marcados son **auxiliares** y no deberán verse durante la ejecución.

El planteamiento es que disponemos de **2 Barras de Deslizamiento** para ir **aumentando\disminuyendo** la **Temperatura en ºC**. La **Barra Superior** varía en valores **Enteros** y la **Inferior** en valores **Decimales**.

El ***SpinButton*** es un Control que, por medio de **2 flechas**, permite **aumentar\disminuir números enteros** al presionarlas. Pero su valor no se puede **incrementar\decrementar** en valores **Decimales**. Es por ello que hay que buscar fórmulas alternativas, como por ejemplo incluir 2 Controles y usar uno con los valores enteros y el otro, dividiendo por 100. Así, si **sumas\restas** ambos valores, obtienes un **número decimal**. Es por eso que usé 2 Cuadros de Texto Auxiliares para guardar ambos valores.

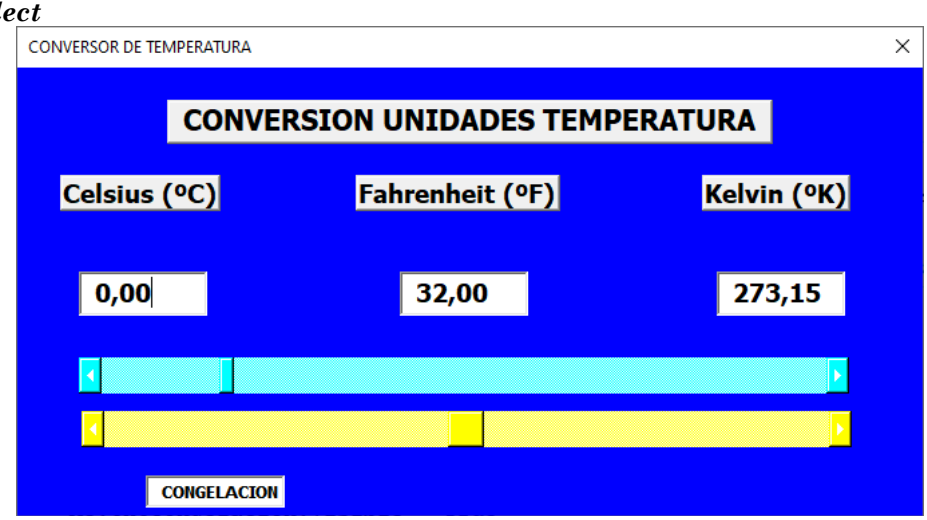
- ***La Fórmula para pasar de ºC a ºF es*** $ºF = \frac{9}{5} * ºC + 32$
- ***La Fórmula para pasar de ºC a ºK es*** $ºK = ºC + 273,15$

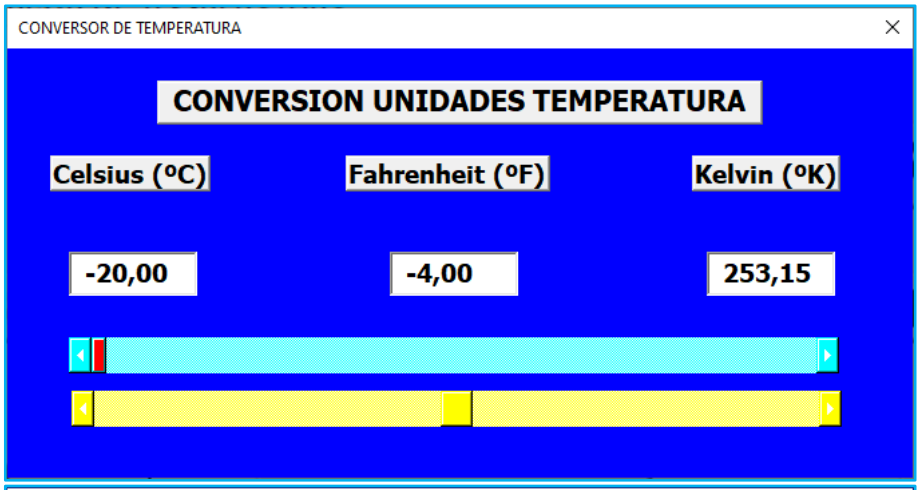
En el momento de **Inicializarse** el **Formulario**, el Programa escribirá en la **Hoja de Cálculo**, la Tabla de Conversión para **valores de Temperatura de -20 a 100 ºC**.

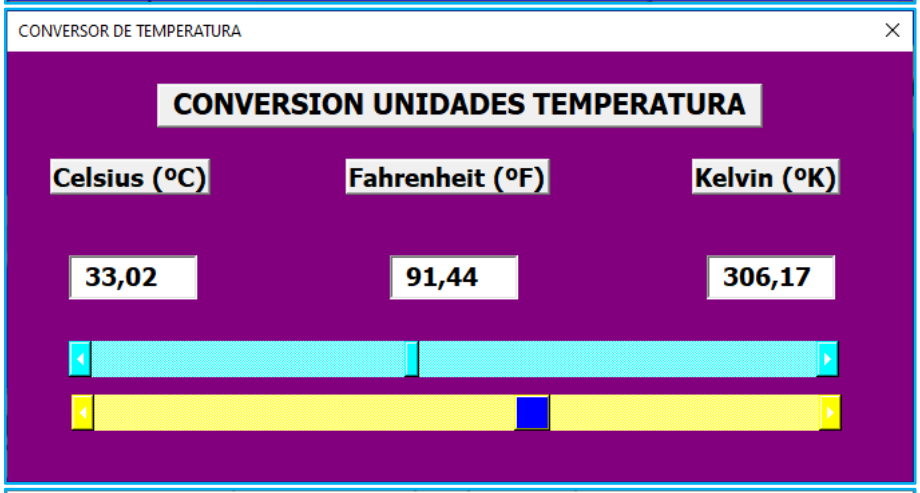
	A	B	C	D
1	CELSIUS	FARENHEIT	KELVIN	
2	-20	-4,00	253,15	
3	-19	-2,20	254,15	
4	-18	-0,40	255,15	
5	-17	1,40	256,15	
6	-16	3,20	257,15	
7	-15	5,00	258,15	
8	-14	6,80	259,15	
9	-13	8,60	260,15	
10	-12	10,40	261,15	
11	-11	12,20	262,15	
12	-10	14,00	263,15	
13	-9	15,80	264,15	
14	-8	17,60	265,15	

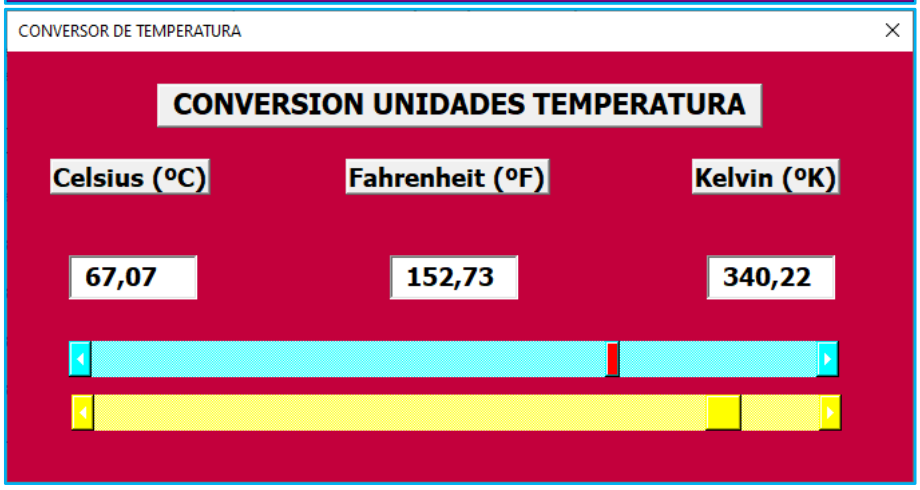
En este ejemplo, a medida que **aumenta\disminuye la Temperatura**, la **Propiedad *BackColor*** del **Formulario** la he ido cambiando. He pasado de Azul a Rojo en **4 Colores**.

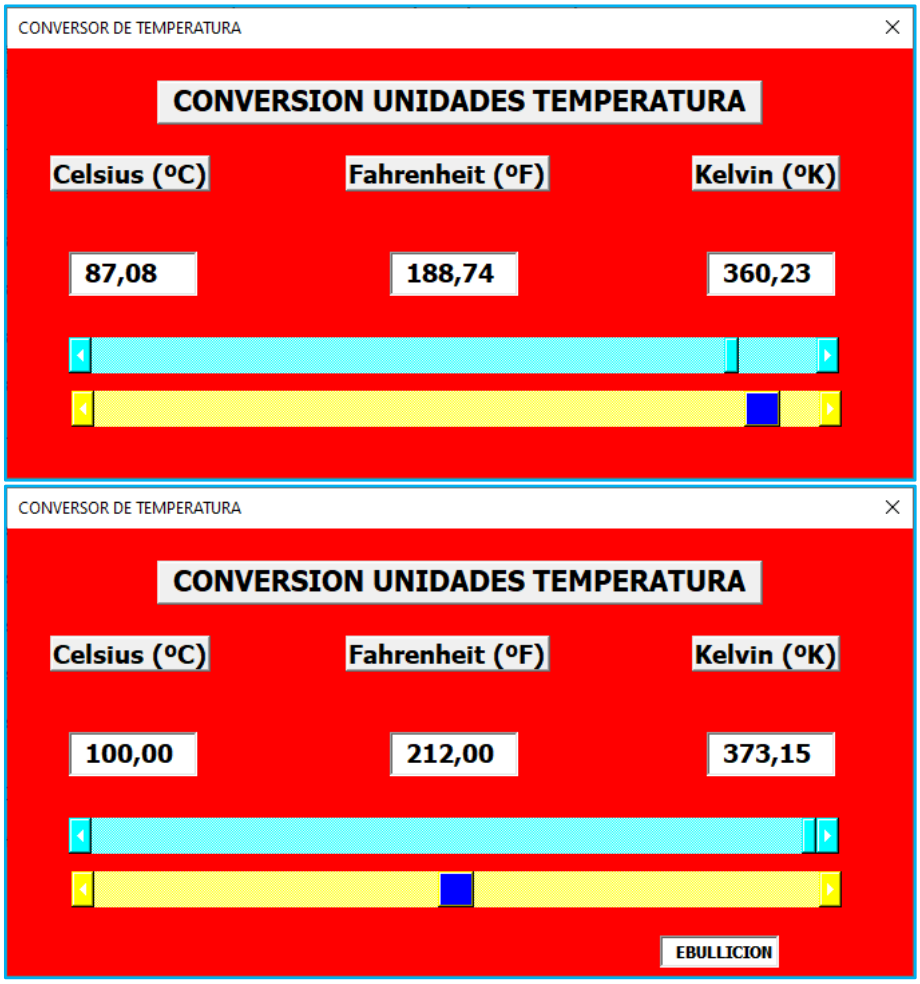
```
Select Case Me.scbTemperatura.Value

   Case Is < 15
      Me.BackColor = RGB(0, 0, 255)
   Case Is < 30
      Me.BackColor = RGB(65, 0, 190)
   Case Is < 60
      Me.BackColor = RGB(130, 0, 125)
   Case Is < 80
      Me.BackColor = RGB(195, 0, 60)
   Case Else
      Me.BackColor = RGB(255, 0, 0)

End Select
```




CONVERSOR DE TEMPERATURA
CONVERSION UNIDADES TEMPERATURA
Celsius (ºC)
Fahrenheit (ºF)
Kelvin (ºK)
-20,00
-4,00
253,15


CONVERSOR DE TEMPERATURA
CONVERSION UNIDADES TEMPERATURA
Celsius (ºC)
Fahrenheit (ºF)
Kelvin (ºK)
33,02
91,44
306,17


CONVERSOR DE TEMPERATURA
CONVERSION UNIDADES TEMPERATURA
Celsius (ºC)
Fahrenheit (ºF)
Kelvin (ºK)
67,07
152,73
340,22

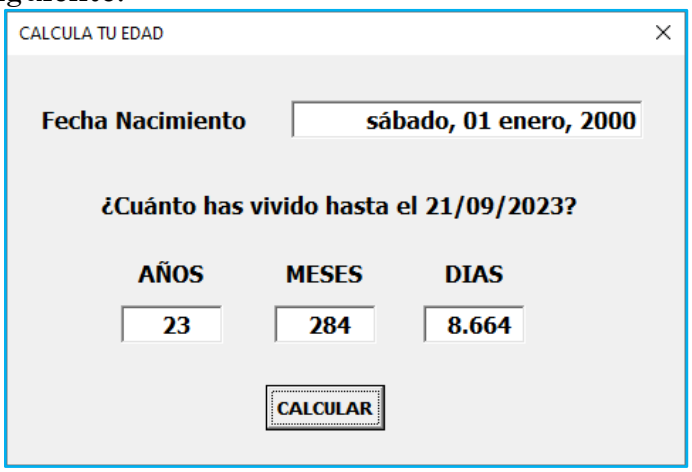


El Programa deberá calcular la **Conversión** para valores de ***-20 a 100 ºC*** y marcar con un mensaje cuando se posicione a ***0ºC*** y a ***100ºC*** como ***CONGELACION*** y ***EBULLICION***, respectivamente.

Finalmente, al hacer ***Doble Click*** sobre el Título "***CONVERSION UNIDADES TEMPERATURA***", se **cerrará** el Formulario.

Ejercicio 10.03.-
Crea el archivo ***EJERCICIO1003.xlsm*** y guárdalo.

El **Ejercicio Propuesto** consiste en hacer un Programa que nos pregunta la **Fecha de Nacimiento** y nos devuelve el tiempo que hemos vivido (hasta la **fecha actual**) en **Años**, **Meses** y **Días**. Por ejemplo, una persona nacida el ***01/01/2000*** habría vivido lo siguiente.



Para este ejercicio puedes utilizar la **Función *VBA.DATEDIFF*** que devuelve un valor ***Variant*** (***Long***) que especifica el número de intervalos de tiempo entre **2 fechas** específicas (https://learn.microsoft.com/es-es/office/vba/language/reference/user-interface-help/datediff-function), como por ejemplo:

Variable = VBA.DateDiff(Intervalo, FechaInicial, FechaFinal)
Variable = VBA.DateDiff("yyyy", "01/01/2000", Now)

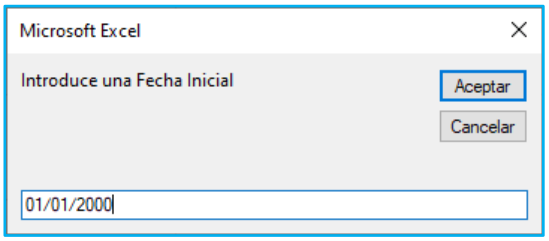
El valor del **Parámetro *Intervalo*** tiene la siguiente configuración.

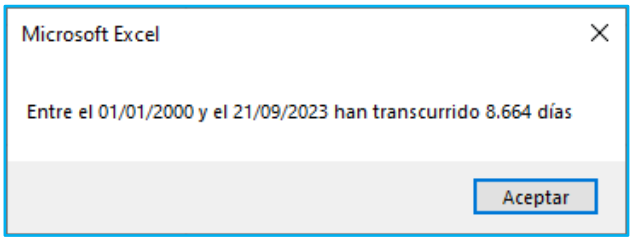
Configuración	Descripción
yyyy	Año
q	Trimestre
m	Mes
y	Día del año
d	Día
w	Día de la semana
ww	Semana
h	Hora
n	Minuto
s	Segundo

```
Sub Entre2Fechas()
Dim Variable As Variant, FechaInicial As Date

    FechaInicial = InputBox("Introduce una Fecha Inicial")
    Variable = DateDiff("d", FechaInicial, Now)
    Variable = Format(Variable, "#,##0")
```

MsgBox "Entre el " & FechaInicial & " y el " & Date & " han transcurrido " & Variable & " días"

End Sub





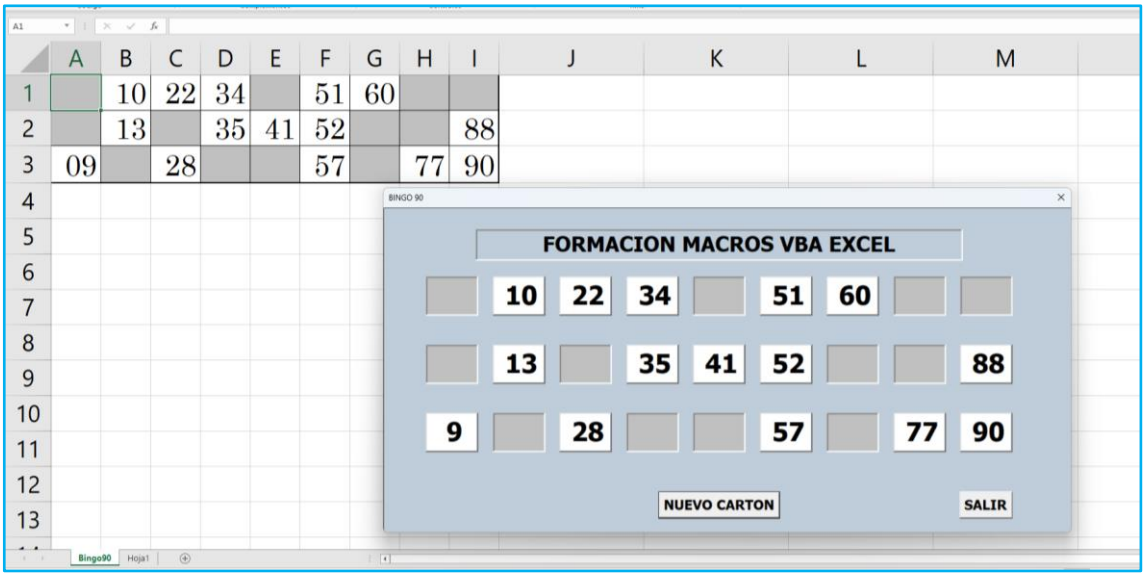
Ejercicio 10.04.-
Crea el archivo ***EJERCICIO1004.xlsm*** y guárdalo.

El **Ejercicio Propuesto** consiste en hacer un programa que genere los cartones que se utilizan para jugar al conocido **Bingo de 90 Bolas**.

Se trata de un juego que utiliza **cartones de 3 filas con 9 casillas cada una**, es decir, una **matriz de 3 x 9**. Dentro de cada fila solo **5 casillas contienen números** y el resto permanecen vacías. En consecuencia, cada cartón tiene **15 números del bingo**. Los números con los que se juega can del **1 al 90**, tal y como su propio nombre indica. En total, hay **9 Columnas**, en la que los números deberán ir **ordenados de menor a mayor tamaño**. La primera Columna sólo admite números del **1 al 9**, la segunda, del **10 al 19**, la tercera, del **20 al 29**, la cuarta del **30 al 39**, la quinta del **40 al 49**, la sexta del **50 al 59**, la séptima del **60 al 69**, la octava del **70 al 79** y, finalmente, en la novena solo encontraremos número del **80 al 90**. No podrá haber una Columna entera sin número alguno y, evidentemente, no se puede repetir ningún número.

Este será el aspecto que tendrá nuestro cartón. Por cada pulsación del **Botón *NUEVO CARTON*** se generará una nueva sucesión aleatoria de números que cumplen los requisitos ya mencionados.

Observa que los números se van a generar en la **Hoja de Cálculo** "***Bingo90***" y de ahí, pasarán al **Formulario *frmBingo90***.



Aplicaremos el **Método de Ordenar** llamado **Burbuja (Bubble Sort)**, basado en comparar elementos adyacentes de la lista o la matriz, al tiempo que se intercambian sus valores si están desordenados. Se dice que los valores más pequeños burbujean hacia el primer elemento de la lista, mientras que los valores más grandes se hunden hacia el final de la lista.

Se necesita 4 pasadas para ordenar un grupo de números de 5 elementos (n-1 pasadas para n elementos).

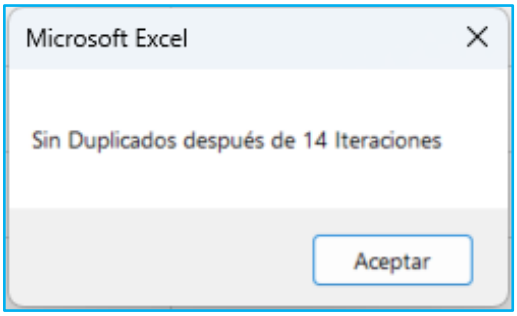
El algoritmo Burbuja se puede mejorar añadiendo un indicador que registre si se hubiera producido algún intercambio en la pasada. De no haber cambios, la lista estaría ordenada y no sería necesario seguir comparando.

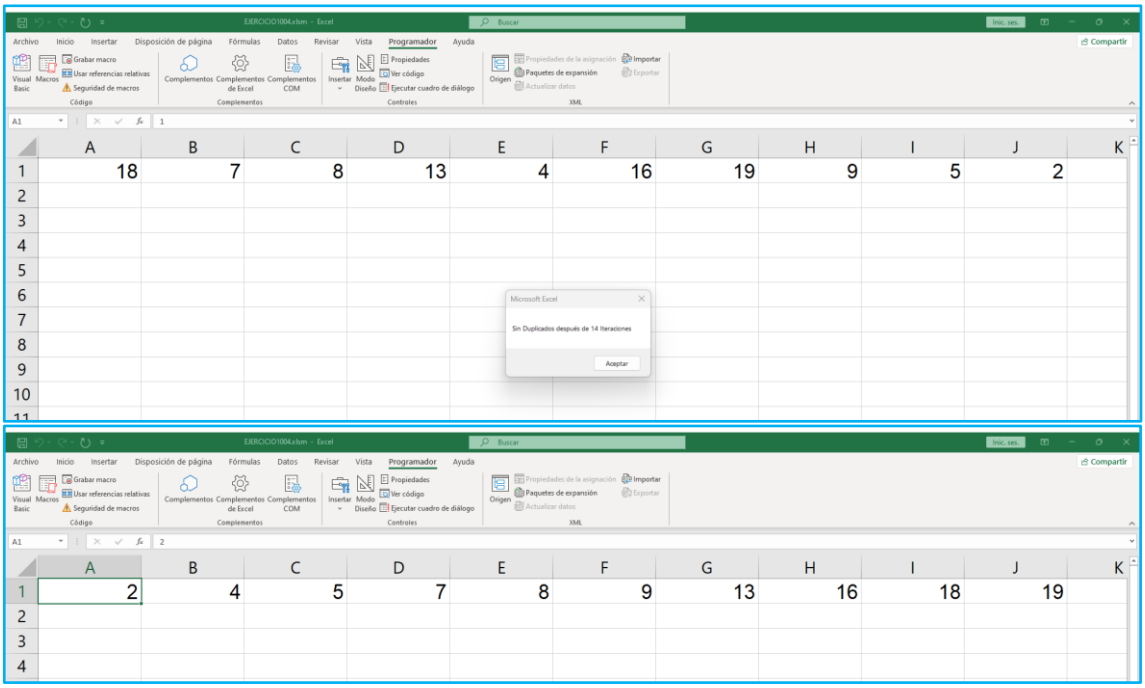
Existe otros muchos métodos de ordenación, pero emplearemos este (también llamado ***Método de Intercambio Directo***) porque eses muy sencillo y el número de elementos a ordenar es muy pequeño.

A continuación, te escribo el Código que hice basado en este Método de Ordenación. El ejemplo siguiente, ordena **10 números aleatorios** elegidos **entre el 1 y el 20**. Para ello, cada vez que elige los 10 números, comprueba si alguno se repite. En caso afirmativo, vuelve a generar los números.

```
Sub OrdenBurbuja()
Dim i As Integer, j As Integer, Auxiliar As Integer, MiMatriz As Variant
Dim Recuento As Boolean, Iteraciones As Long, Limite As Integer

    Hoja2.Activate

    Iteraciones = 1

    Limite = 10

    ReDim MiMatriz(1 To Limite)
```

```
    Do
        Recuento = False

        For i = 1 To UBound(MiMatriz)
            MiMatriz(i) = Application.WorksheetFunction.RandBetween(1, 20)
        Next i

        Hoja2.Range(Cells(1, 1), Cells(1, UBound(MiMatriz))).Value = MiMatriz

        'Localiza si hay números duplicados en la Matriz
        For i = 1 To UBound(MiMatriz) - 1
            For j = i + 1 To UBound(MiMatriz)
                If MiMatriz(i) = MiMatriz(j) Then
                    Recuento = True
                    Iteraciones = Iteraciones + 1
                End If
            Next
        Next

    Loop Until Recuento = False

    MsgBox "Sin Duplicados después de " & Iteraciones & " Iteraciones"

    'Ordena los números de la Matriz por el Método de la burbuja
    For i = 1 To UBound(MiMatriz) - 1
        For j = i + 1 To UBound(MiMatriz)
            If MiMatriz(i) > MiMatriz(j) Then
                Auxiliar = MiMatriz(i)
                MiMatriz(i) = MiMatriz(j)
                MiMatriz(j) = Auxiliar
            End If
        Next
    Next

    Hoja2.Range(Cells(1, 1), Cells(1, UBound(MiMatriz))).Value = MiMatriz

    Hoja2.Range("A1").Select

End Sub
```



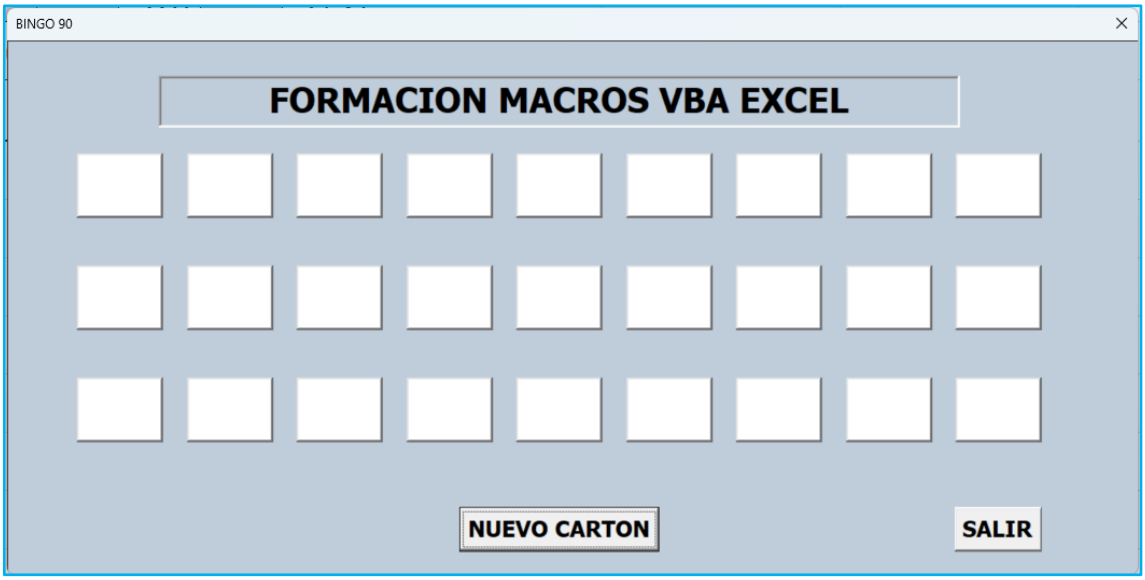


Para el Ejercicio hice un **Módulo** llamado ***Funciones*** y dentro incluí una **Función** llamada ***MatrizColumna*** que genera los números de cada columna del cartón (recuerda que son 9). Los números los devuelve ordenados de menor a mayor, aunque puede devolverlos con números repetidos.

En otro Módulo, tengo un Procedimiento llamado ***CrearCarton*** que se encarga de lo siguiente.

1. Llama a la Función (lo hace 9 veces) con los Parámetros ***Columna***, ***Número Inferior*** y ***Número Superior*** (en cada Columna).
2. Crea una Matriz Inicial que rellenar el cartón completo (**27 Números**). No deja ningún hueco vacío.
3. **Elimina 4 Números** de cada Fila (los elige al azar).
4. Genera el Primer Cartón que puede ser válido. Pero si tiene, al menos, **1 Columna vacía**, repite de nuevo proceso desde el principio.

La Matriz Final es el Cartón que buscamos, cumpliendo los requisitos prefijados.



El cartón es un **Formulario** (***frmBingo90***) con una serie de controles que toman los datos de la **Hoja *Bingo90*** que contiene los números del cartón.

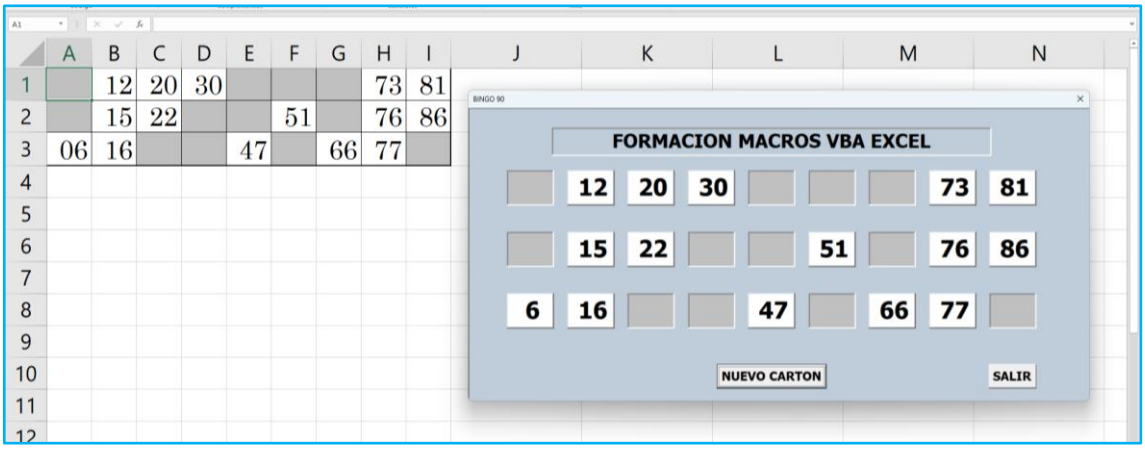
El Formulario contiene una **Etiqueta** como título, un **Botón** que genera el Cartón y otro que cierra y sale del **Formulario**. Disponemos, además, de los **27 Cuadros de Texto**, donde podrán ir los números.

En este Formulario, he usado una serie de Controles (***Cuadros de Texto***), que están incluidos dentro de la **Colección *Controls* del Formulario**, lo cual me permite acceder a ellos y establecer sus Propiedades de una manera muy cómoda.

Por ejemplo, con el siguiente código podría acceder a una serie de Cuadros de Texto (***CuadroTexto1, CuadroTexto2, CuadroTexto3, … , CuadroTexto10***) (Controles dentro de la Colección del Formulario) y modificar su **Propiedad *Value***, usando un patrón muy práctico. Obviamente, tendré que tener presente que debo usar nombres que se ajusten al patrón que voy a usar.

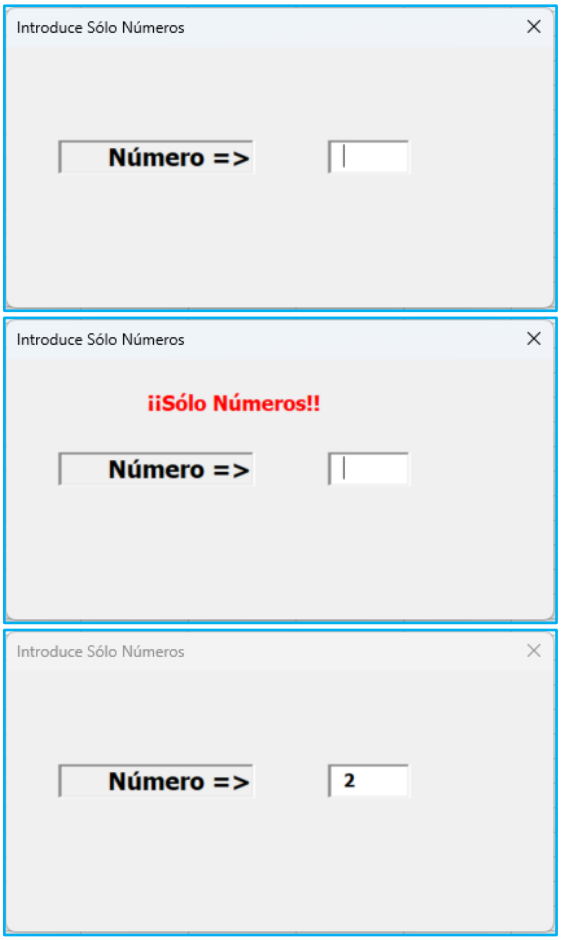
```
For i = 1 To 10
   Controls("CuadroTexto" & i).Value = Int(Rnd * 100 + 1)
Next i
```

Nuestro objetivo es conseguir que, al pulsar el **Botón *NUEVO CARTON***, se genere un Cartón de ***Bingo90***, válido.


FORMACION MACROS VBA EXCEL
12 20 30 73 81
15 22 51 76 86
6 16 47 66 77
NUEVO CARTON
SALIR

Ejercicio 10.05.-
Crea el archivo ***EJERCICIO1005.xlsm*** y guárdalo.

Vamos a empezar haciendo que un **Cuadro de Texto** tan sólo admita números. Para ello usaremos un **Evento** del **Cuadro de Texto** que responde cuando se pulse una Tecla.

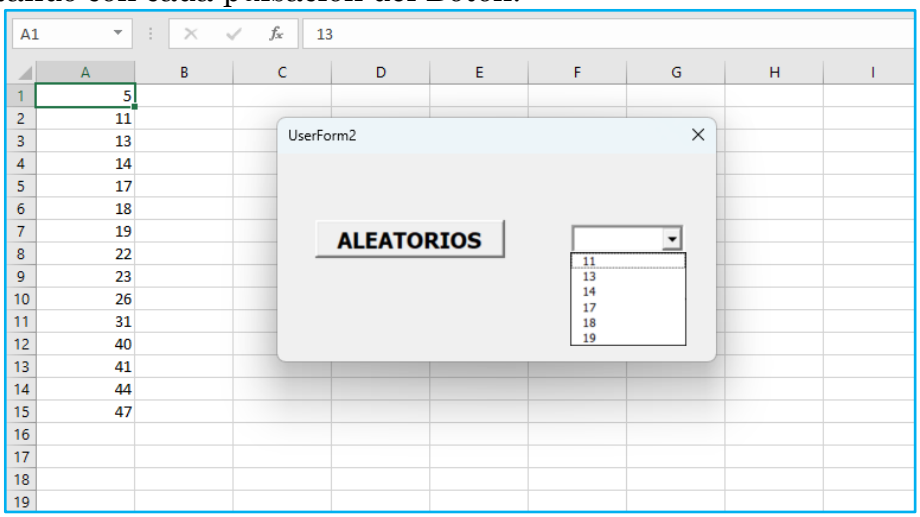



```
Private Sub UserForm_Initialize()
   Me.Height = 140
   Me.Width = 280
End Sub

Private Sub TextBox1_KeyPress(ByVal KeyAscii As MSForms.ReturnInteger)

   Select Case KeyAscii
      Case Is < vbKey0, Is > vbKey9
         Me.lblAdvertencia.Caption = "¡¡Sólo Números!!"
         KeyAscii = 0
         Beep
      Case Else
         Me.lblAdvertencia.Caption = ""
   End Select

End Sub
```

El siguiente ejemplo consiste en rellenar un ***ComboBox*** con los valores contenidos en un Rango, pero sólo con aquellos que **comiencen por el número 1** (sin contar

el propio 1). Tendremos un **Botón de Comando** que, al pulsarlo, genera **15 números enteros aleatorios entre el 1 y el 50** y si aparecen **duplicados**, vuelve a generar la lista. Finalmente **ordena** la lista y rellenar de datos el ***ComboBox***.

NOTA. Si no usamos el **Método *ComboBox.Clear***, los valores se irán incrementando con cada pulsación del Botón.



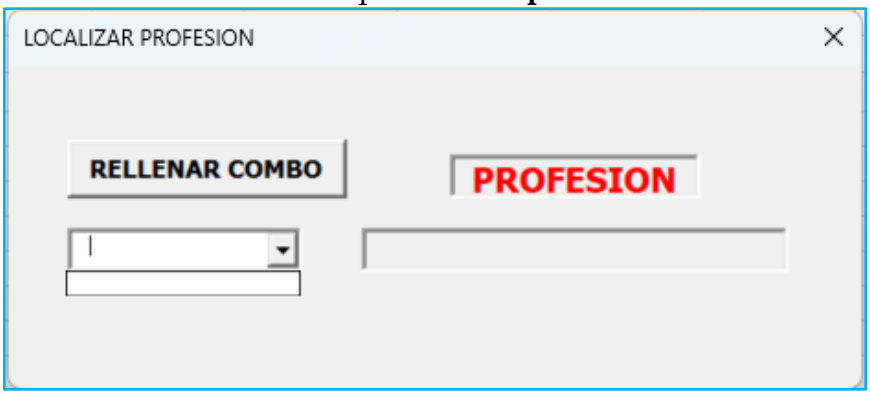

```
Private Sub UserForm_Initialize()
   Me.Height = 160
   Me.Width = 290
End Sub

Private Sub cmdAleatorios_Click()
Dim i, j As Integer, Duplicado As Boolean

Hoja1.Activate

   For i = 1 To 15
      Cells(i, 1).Value = Application.WorksheetFunction.RandBetween(1, 50)
   Next i

   Do
      Duplicado = False

      'Localiza si hay números duplicados en la Matriz
      For i = 1 To 15 - 1
         For j = i + 1 To 15
            If Cells(i, 1).Value = Cells(j, 1).Value Then
               Cells(j, 1).Value = Application.WorksheetFunction.RandBetween(1, 50)
               Duplicado = True
            End If
         Next
      Next

   Loop Until Duplicado = False

   Application.Range("A1").Sort key1:=Range("A1"), order1:=xlAscending, Header:=xlNo

   Me.cmbAleatorio.Clear
```

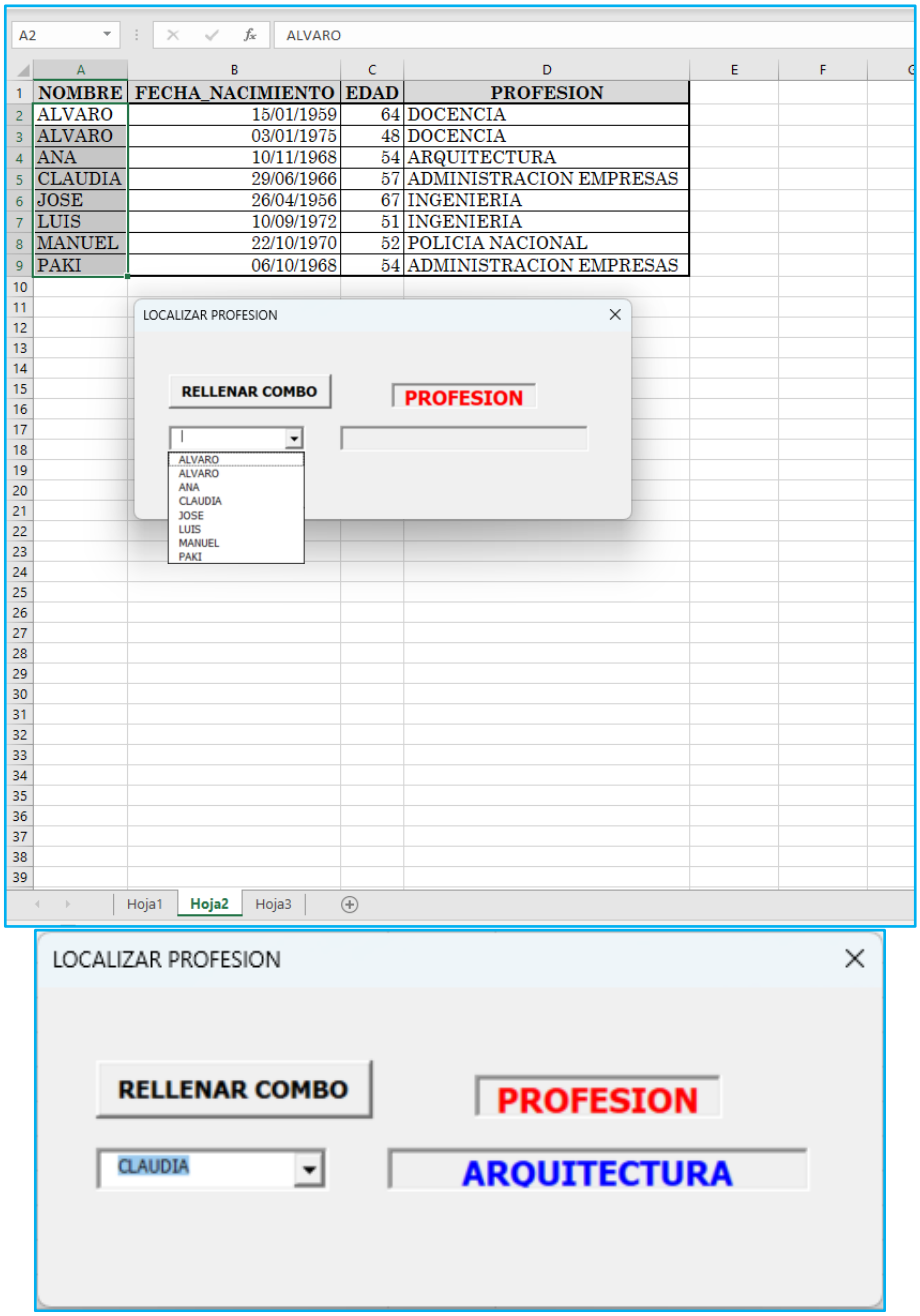
```
    For i = 1 To 15
       If Left(Cells(i, 1).Value, 1) = 1 And Cells(i, 1).Value <> 1 Then
          Me.cmbAleatorio.AddItem Cells(i, 1).Value
       End If
    Next

End Sub
```

El **Primer Ejercicio Propuesto** consiste en utilizar un ***ComboBox*** para que, al seleccionar un **Nombre** de la Lista, te muestre la **Profesión**. Partiremos de un listado con los Datos contenidos en la ***Hoja2***.

NOMBRE	FECHA_NACIMIENTO	EDAD	PROFESION
CLAUDIA	29/06/1966	57	ADMINISTRACION EMPRESAS
PAKI	06/10/1968	54	ADMINISTRACION EMPRESAS
ANA	10/11/1968	54	ARQUITECTURA
ALVARO	15/01/1959	64	DOCENCIA
ALVARO	03/01/1975	48	DOCENCIA
JOSE	26/04/1956	67	INGENIERIA
LUIS	10/09/1972	51	INGENIERIA
MANUEL	22/10/1970	52	POLICIA NACIONAL

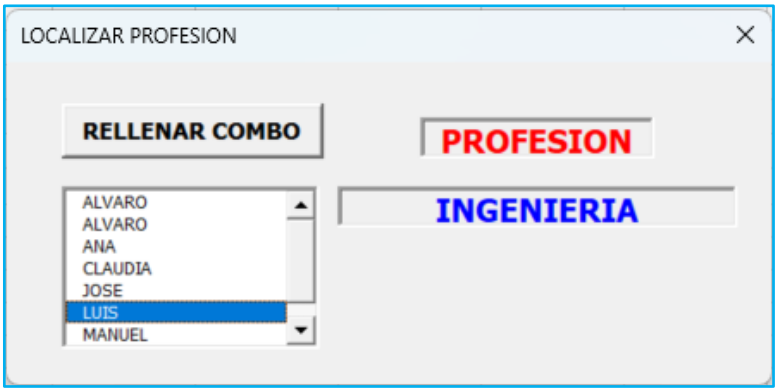
Al iniciarse el programa, lo primero que haremos es Rellenar el ***ComboBox***. Par ello, primero ordenaremos la Tabla por el **Campo *NOMBRE***.





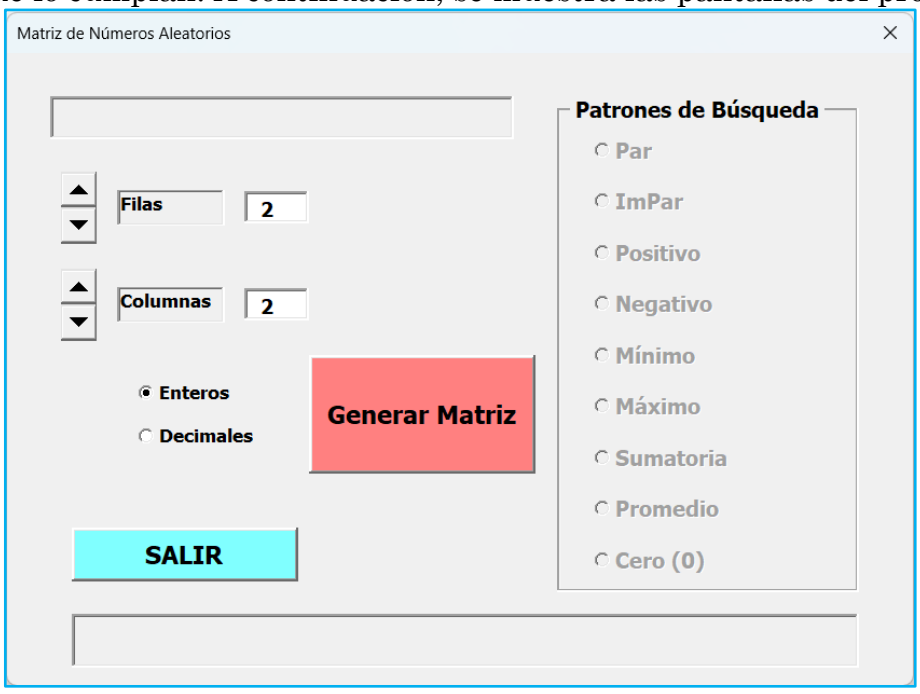
NOTA. Puede usar la **Propiedad *ComboBox.ListIndex*** para saber qué línea del Rango tienes que devolver, teniendo en cuenta que ***ListIndex*** empieza en **0**.

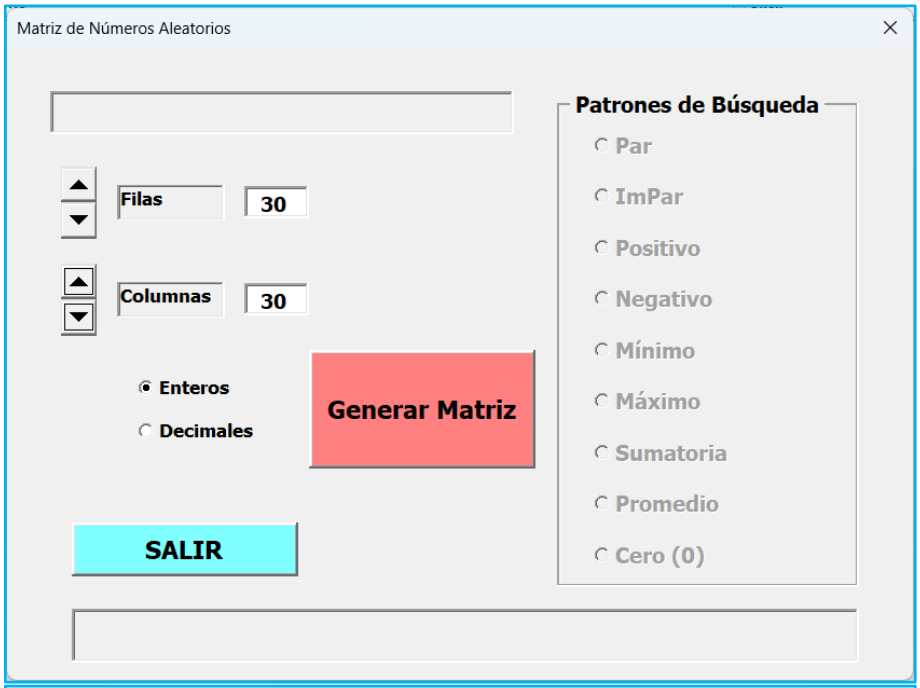
El **Segundo Ejercicio Propuesto** consiste en hacer exactamente lo mismo que el anterior, sólo que usarás un ***ListBox*** para esta ocasión. Partiremos de un listado con los Datos contenidos en la ***Hoja2***.

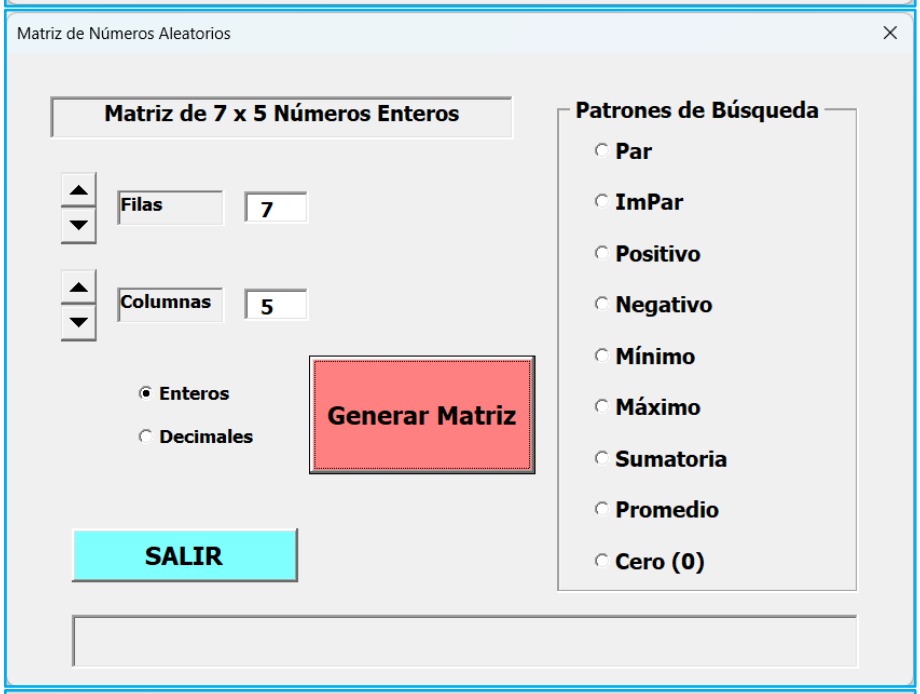


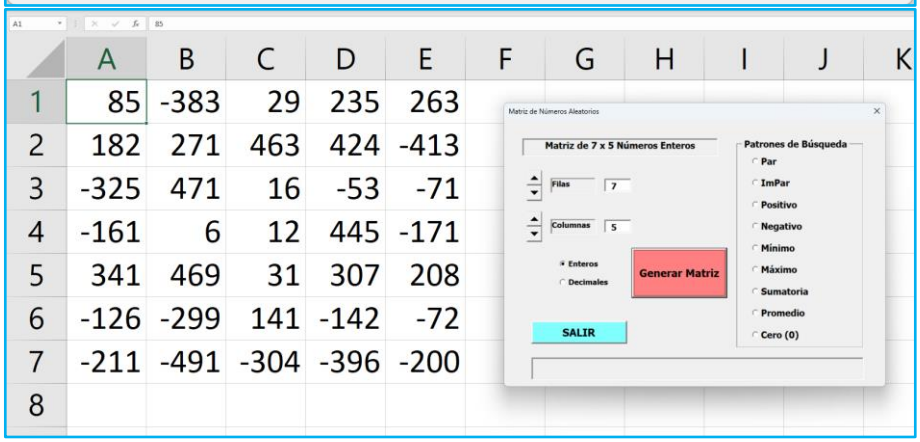
El **Tercer Ejercicio Propuesto** consiste en hacer una aplicación que genere **Matrices Cuadradas o Rectangulares** cuya dimensión va desde **2x2 a 30x30**. Los valores son números desordenados comprendidos entre el **-500 y 500** que pueden ser **Enteros** o **Decimales** y admite **repeticiones**.

A la derecha tendremos **Patrones de Búsqueda** para que se resalte aquellos valores que lo cumplan. A continuación, se muestra las pantallas del programa.

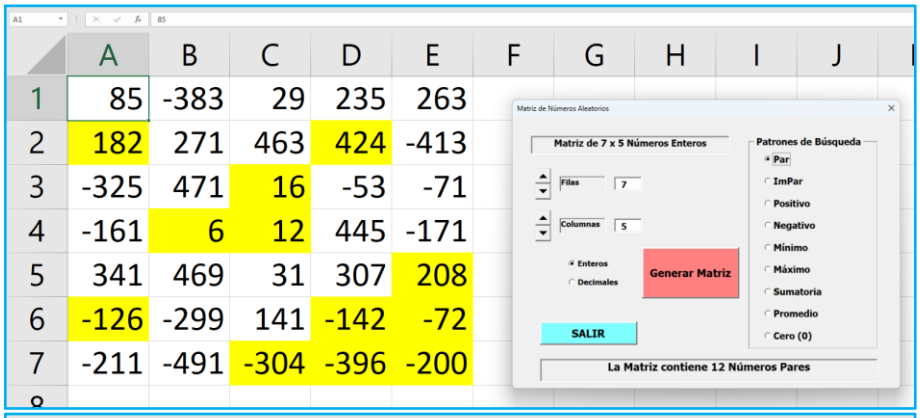


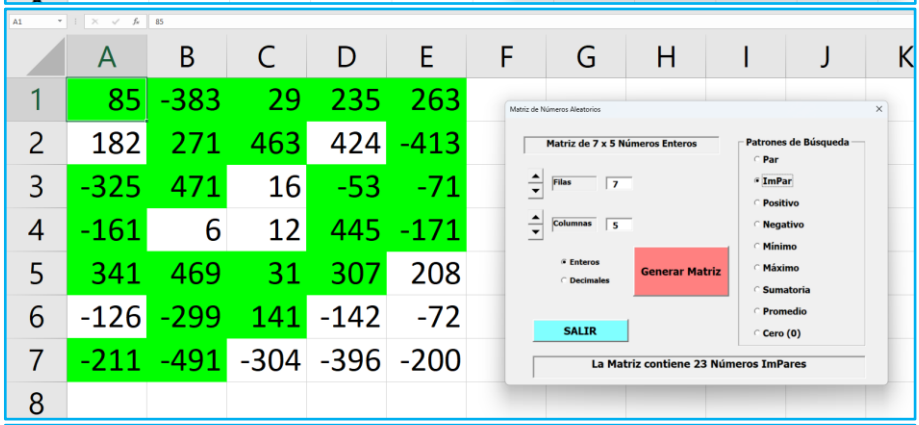


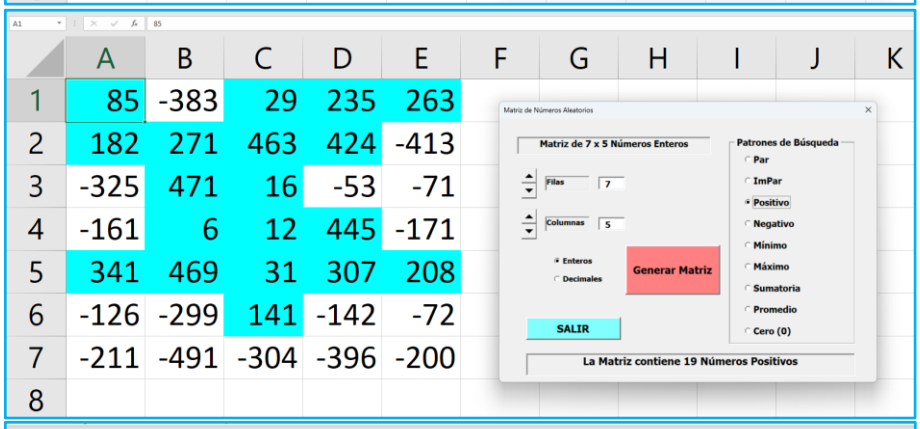


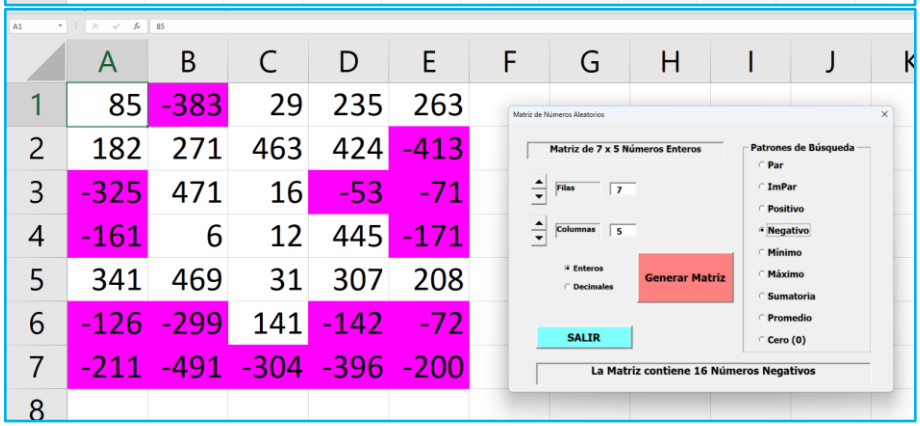


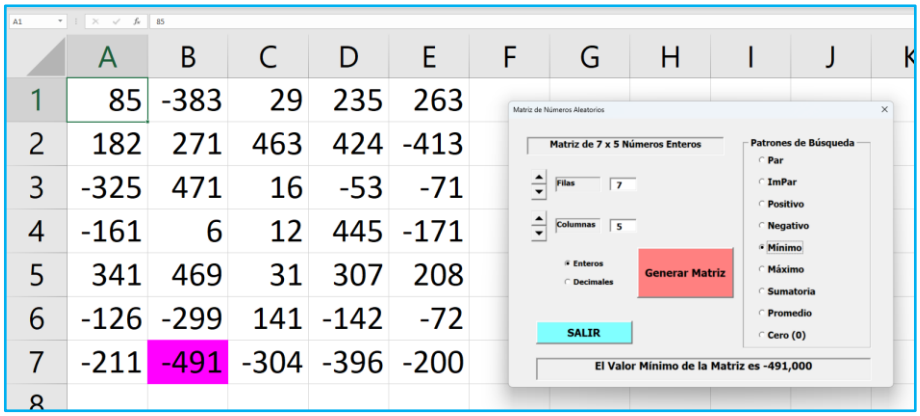
	A	B	C	D	E
1	85	-383	29	235	263
2	182	271	463	424	-413
3	-325	471	16	-53	-71
4	-161	6	12	445	-171
5	341	469	31	307	208
6	-126	-299	141	-142	-72
7	-211	-491	-304	-396	-200
8					

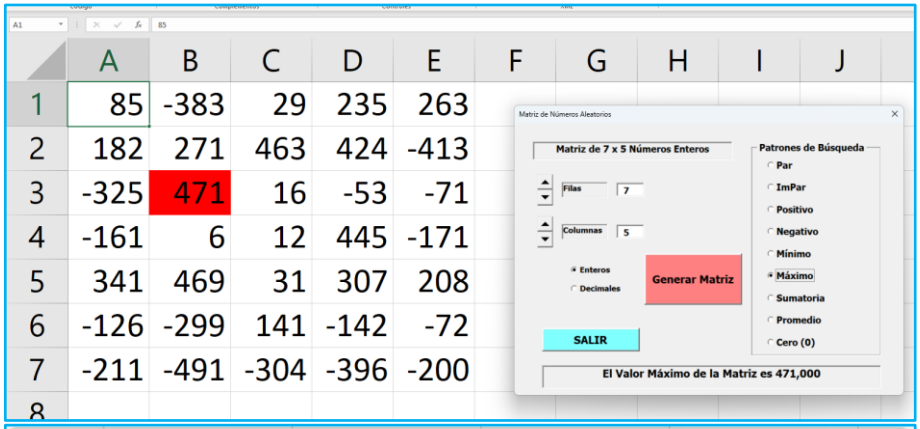

Matriz de Números Aleatorios
Matriz de 7 x 5 Números Enteros
Filas 7
Columnas 5
Enteros
Decimales
Generar Matriz
SALIR
Patrones de Búsqueda
Par
ImPar
Positivo
Negativo
Mínimo
Máximo
Sumatoria
Promedio
Cero (0)
La Matriz contiene 12 Números Pares

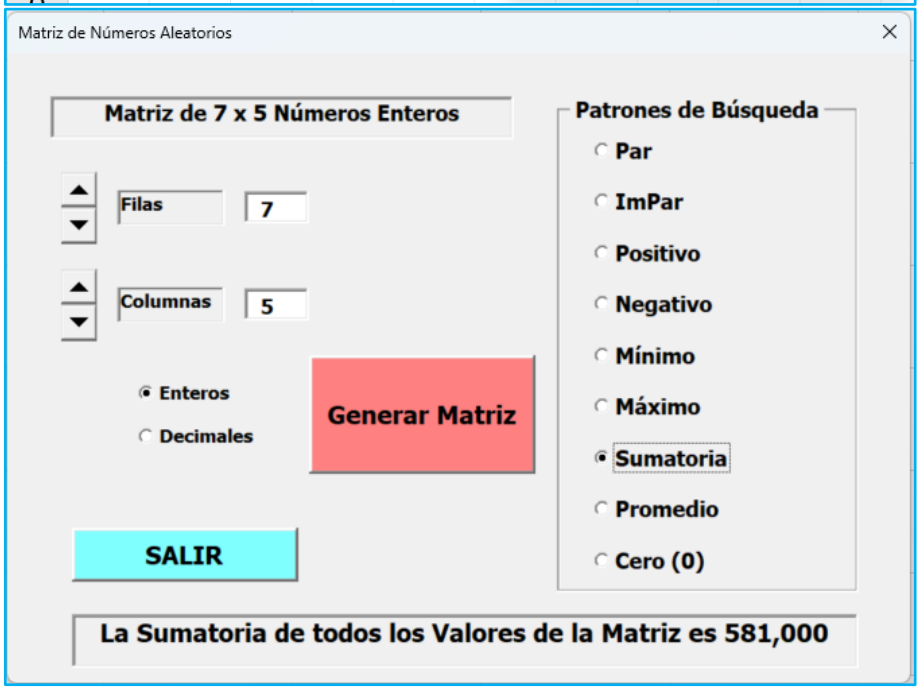

Matriz de Números Aleatorios
Matriz de 7 x 5 Números Enteros
Filas 7
Columnas 5
Enteros
Decimales
Generar Matriz
SALIR
Patrones de Búsqueda
Par
ImPar
Positivo
Negativo
Mínimo
Máximo
Sumatoria
Promedio
Cero (0)
La Matriz contiene 23 Números ImPares

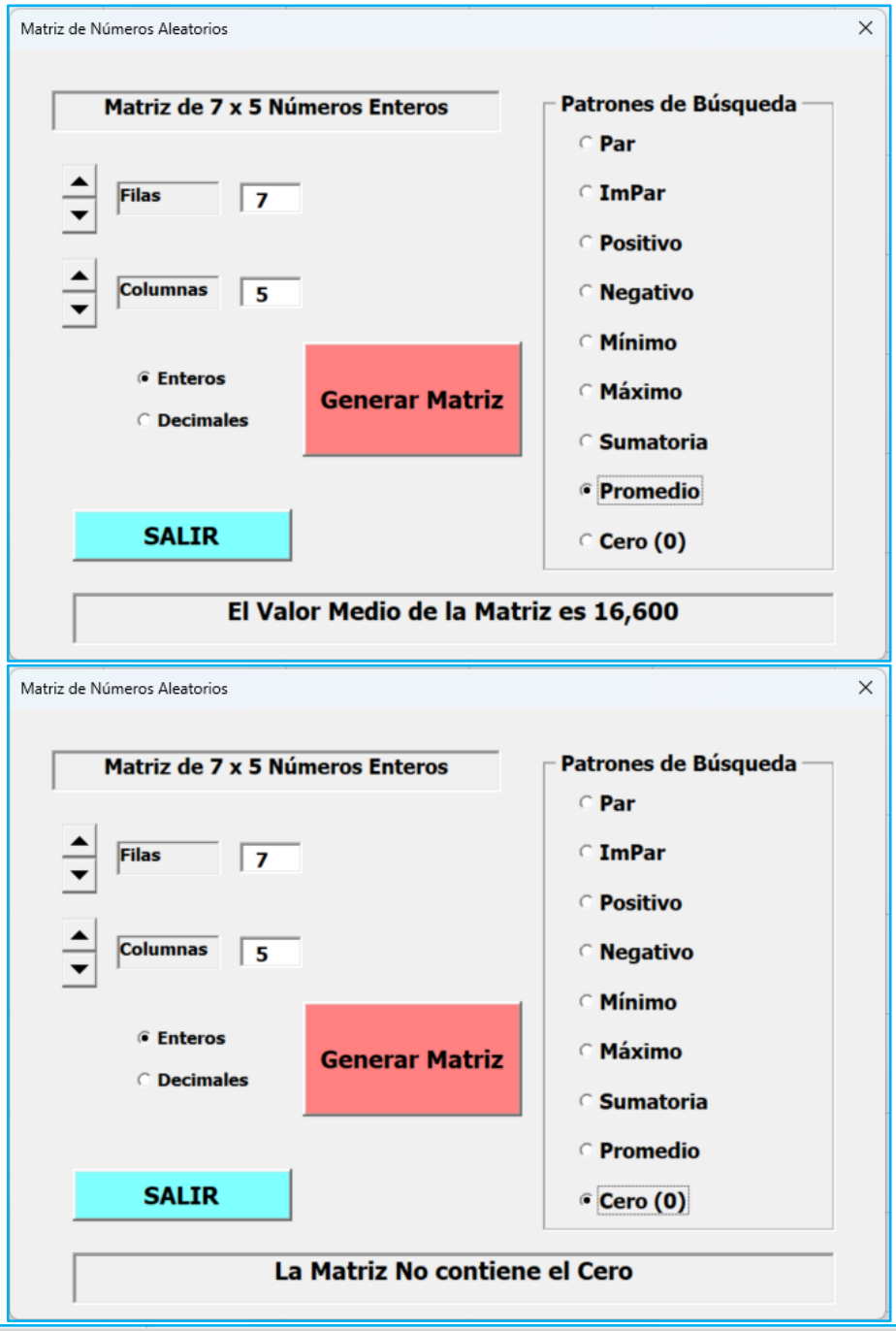

Matriz de Números Aleatorios
Matriz de 7 x 5 Números Enteros
Filas 7
Columnas 5
Enteros
Decimales
Generar Matriz
SALIR
Patrones de Búsqueda
Par
ImPar
Positivo
Negativo
Mínimo
Máximo
Sumatoria
Promedio
Cero (0)
La Matriz contiene 19 Números Positivos

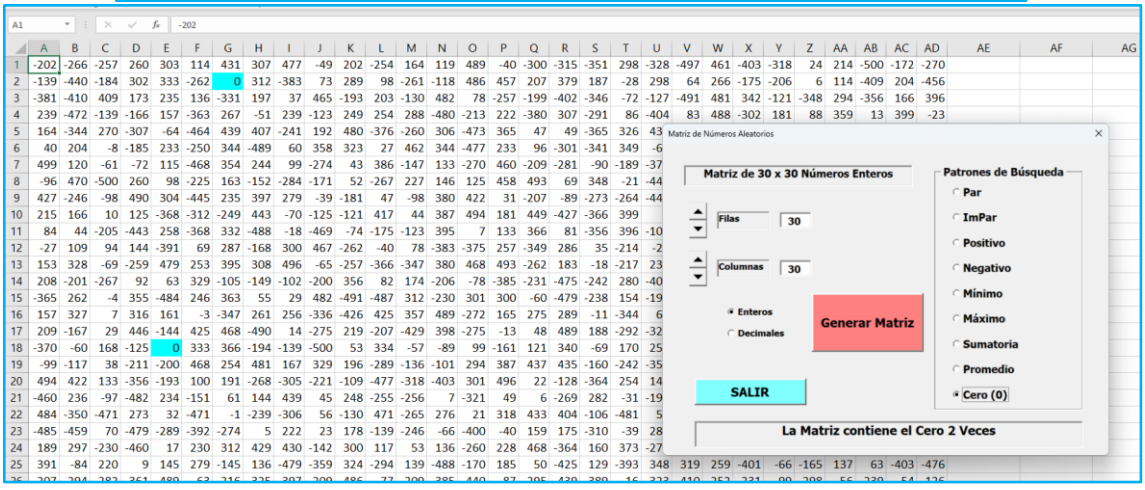

Matriz de Números Aleatorios
Matriz de 7 x 5 Números Enteros
Filas 7
Columnas 5
Enteros
Decimales
Generar Matriz
SALIR
Patrones de Búsqueda
Par
ImPar
Positivo
Negativo
Mínimo
Máximo
Sumatoria
Promedio
Cero (0)
La Matriz contiene 16 Números Negativos

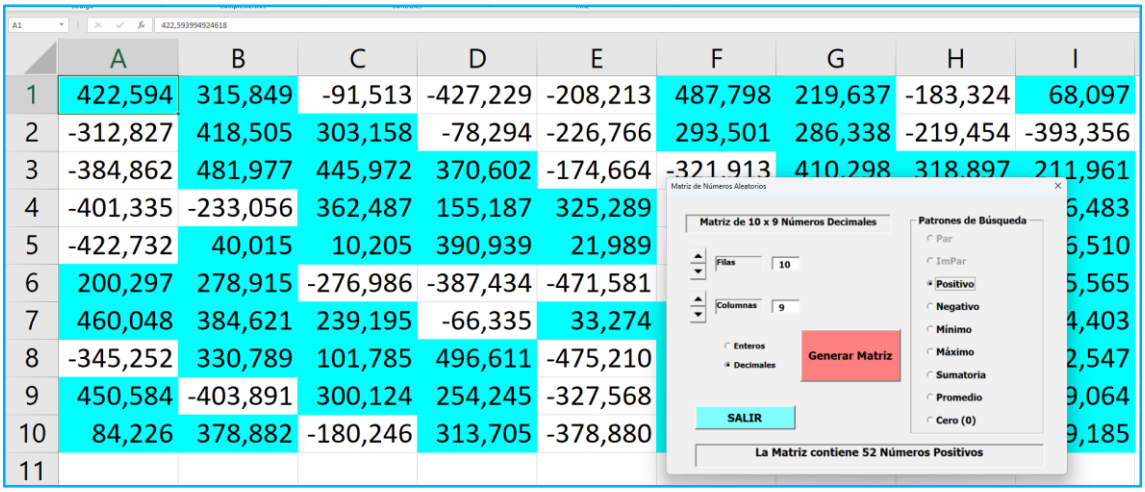

Matriz de Números Aleatorios
Matriz de 7 x 5 Números Enteros
Filas 7
Columnas 5
Enteros
Decimales
Generar Matriz
SALIR
Patrones de Búsqueda
Par
ImPar
Positivo
Negativo
Mínimo
Máximo
Sumatoria
Promedio
Cero (0)
El Valor Mínimo de la Matriz es -491,000


Matriz de Números Aleatorios
Matriz de 7 x 5 Números Enteros
Filas 7
Columnas 5
Enteros
Decimales
Generar Matriz
SALIR
Patrones de Búsqueda
Par
ImPar
Positivo
Negativo
Mínimo
Máximo
Sumatoria
Promedio
Cero (0)
El Valor Máximo de la Matriz es 471,000


Matriz de Números Aleatorios
Matriz de 7 x 5 Números Enteros
Filas 7
Columnas 5
Enteros
Decimales
Generar Matriz
SALIR
Patrones de Búsqueda
Par
ImPar
Positivo
Negativo
Mínimo
Máximo
Sumatoria
Promedio
Cero (0)
La Sumatoria de todos los Valores de la Matriz es 581,000


Matriz de Números Aleatorios
Matriz de 7 x 5 Números Enteros
Filas
7
Columnas
5
Enteros
Decimales
Generar Matriz
SALIR
Patrones de Búsqueda
Par
ImPar
Positivo
Negativo
Mínimo
Máximo
Sumatoria
Promedio
Cero (0)
El Valor Medio de la Matriz es 16,600
Matriz de Números Aleatorios
Matriz de 7 x 5 Números Enteros
Filas
7
Columnas
5
Enteros
Decimales
Generar Matriz
SALIR
Patrones de Búsqueda
Par
ImPar
Positivo
Negativo
Mínimo
Máximo
Sumatoria
Promedio
Cero (0)
La Matriz No contiene el Cero


Matriz de Números Aleatorios
Matriz de 30 x 30 Números Enteros
Filas
30
Columnas
30
Enteros
Decimales
Generar Matriz
SALIR
Patrones de Búsqueda
Par
ImPar
Positivo
Negativo
Mínimo
Máximo
Sumatoria
Promedio
Cero (0)
La Matriz contiene el Cero 2 Veces


A1
422,593994924618
A B C D E F G H I
1 422,594 315,849 -91,513 -427,229 -208,213 487,798 219,637 -183,324 68,097
2 -312,827 418,505 303,158 -78,294 -226,766 293,501 286,338 -219,454 -393,356
3 -384,862 481,977 445,972 370,602 -174,664 -321,913 410,298 318,897 211,961
4 -401,335 -233,056 362,487 155,187 325,289 6,483
5 -422,732 40,015 10,205 390,939 21,989 6,510
6 200,297 278,915 -276,986 -387,434 -471,581 5,565
7 460,048 384,621 239,195 -66,335 33,274 4,403
8 -345,252 330,789 101,785 496,611 -475,210 2,547
9 450,584 -403,891 300,124 254,245 -327,568 9,064
10 84,226 378,882 -180,246 313,705 -378,880 9,185
11
Matriz de Números Aleatorios
Matriz de 10 x 9 Números Decimales
Filas 10
Columnas 9
Enteros
Decimales
Generar Matriz
SALIR
Patrones de Búsqueda
Par
ImPar
Positivo
Negativo
Mínimo
Máximo
Sumatoria
Promedio
Cero (0)
La Matriz contiene 52 Números Positivos