



Objetivos:

- ❖ **Diseño** de diagramas de flujo con **funciones**
- ❖ **Codificación** de **funciones** en Visual Basic
- ❖ Llamada a funciones propias y del sistema

Programa de demostración del uso de funciones

Interfaz

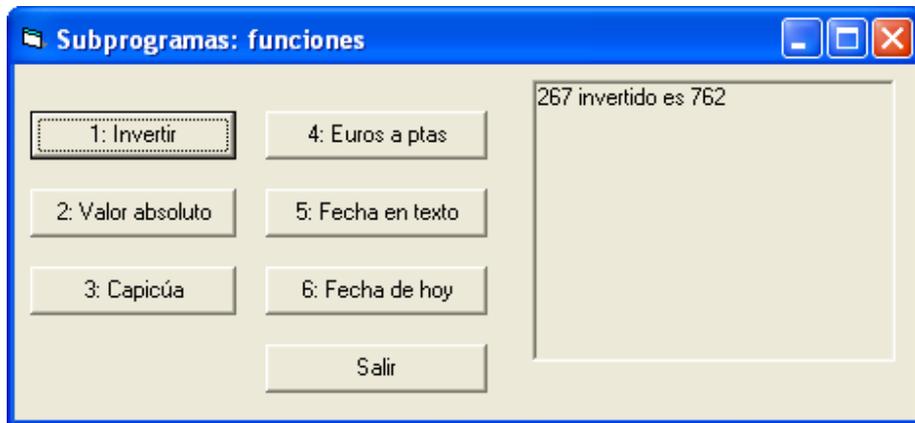


Figura 8.1. Objetos presentes en la interfaz: botones y caja de dibujo

Funcionamiento general

1. Cada **ejercicio** tendrá un botón de ejecución propio (`cmdEj1`, `cmdEj2`, ..., `cmdEj6`).
2. Al hacer **click** sobre cada botón, borraremos inicialmente el contenido de la **caja de dibujo** (PictureBox) del resultado, `pctRes`. Utilizaremos para ello el método `cls` (`pctRes.cls`).
3. Al hacer **click** en el botón **Salir**, el **programa finalizará**.
4. Se proporciona un modelo de programa **ejecutable** para clarificar los enunciados.

Ejercicio 8.1: Invertir número (resolución)

Funcionamiento

Llamaremos `cmdEj1` al botón asociado al ejercicio 1. Cuando el usuario pulse el botón etiquetado “1: Invertir”, el programa pedirá un número positivo que no termine en 0 mediante una instrucción `InputBox` y mostrará en la caja de dibujo (PictureBox) mediante una sentencia `Print` (`pctRes.Print`) este mismo número invertido, tal y como se ejemplifica en la Figura 8.1.

Diagrama de flujo

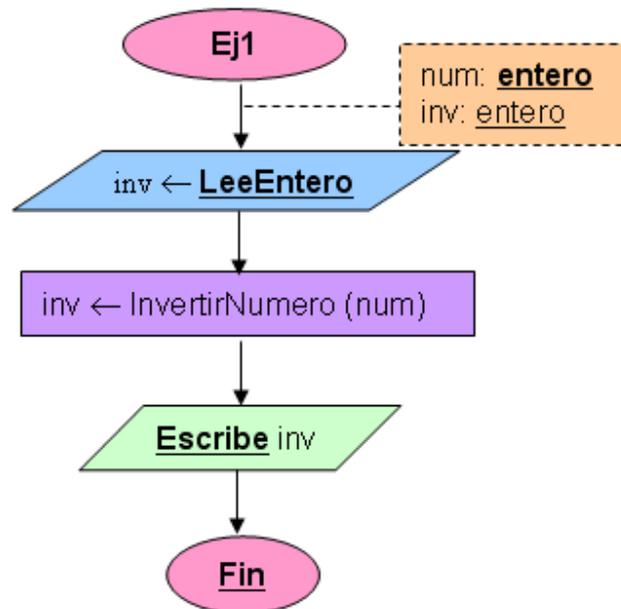


Figura 8.2. Diagrama de flujo del ejercicio 1

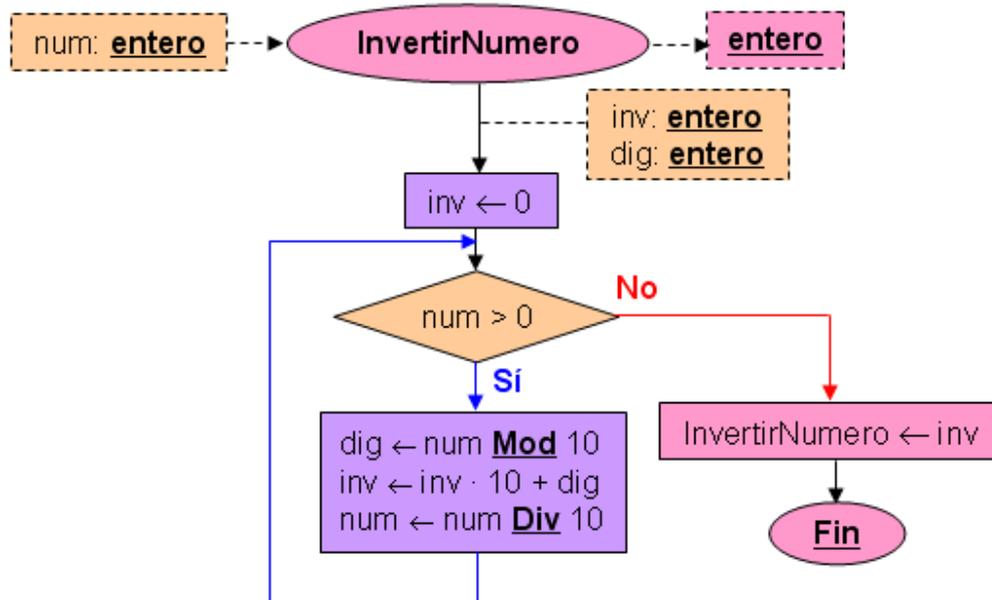


Figura 8.3 Diagrama de flujo de la función `InvertirNumero` del ejercicio 1

Pasos a seguir

1. Crearemos los objetos del tipo y forma mostrados en la Figura 8.1. Guardaremos todo.
2. Añadir el código a los eventos, es decir, el clic sobre los botones:

- **Código del botón “1: Invertir”**: es el procedimiento o subprograma asociado al evento clic, como hemos venido haciendo. Nótese que llama a la función InvertirNumero.

```

Sub cmdEj1_Click()
    Dim s As String
    Dim num As Integer
    Dim inv As Integer

    pctRes.Cls
    s = InputBox("Introduce un número positivo que no termine en 0")
    num = CInt(s)
    inv = InvertirNumero(num)
    pctRes.Print num & " invertido es " & CStr(inv)
End Sub

```

Figura 8.4 Código del programa principal con la llamada a la función.

Habrà que definir igualmente (normalmente a continuación) la función nueva:

```

Function InvertirNumero(ByVal num As Integer) As Integer
    Dim dig As Integer
    Dim inv As Integer

    inv = 0
    While num > 0
        dig = num Mod 10
        inv = inv * 10 + dig
        num = num \ 10
    Wend
    InvertirNumero = inv
End Function

```

Figura 8.5 Código de la función InvertirNumero.

2. **Diseña** los diagramas de flujo y **codifica** un programa VB que lea un número y calcule su **valor absoluto**, mostrando el resultado en el cuadro de dibujo. **Diseña** y utiliza para ello la **función** ValAbs que calcule el valor absoluto de un número cuya cabecera se muestra en la Figura 8.6.



Figura 8.6 Cabecera de la función que calcula el valor absoluto

3. **Diseña** los diagramas de flujo y **codifica** un programa VB que lea un número y muestra en el cuadro de dibujo **si es capicúa** (se lee igual del derecho y del revés). **Diseña** y utiliza para ello la **función** EsCapicua cuya cabecera se muestra en la Figura 8.7 que llame a la función InvertirNumero vista en el ejercicio 8.1.



Figura 8.7 Cabecera de la función que dice si un número es capicúa

4. **Diseña** los diagramas de flujo y **codifica** un procedimiento que pida una cantidad en euros y nos muestre su valor en pesetas, sabiendo que 1 € son 166,386 pesetas. **Diseña** y utiliza para ello la función `EurosPtas` cuya cabecera se muestra en la Figura 8.8.



Figura 8.8. Cabecera de la función que convierte euros a pesetas

5. **Diseña** los diagramas de flujo y **codifica** el programa que pida un día, mes y año (mediante llamadas a la función `InputBox`) y muestre la cadena de la fecha con el siguiente formato: día de mes de año, por ejemplo “31 de marzo de 2009”. Obtendrá la cadena mediante una función `CadenaFecha` a diseñar cuya cabecera se muestra en la Figura 8.9. Para obtener la cadena del mes diseñará y utilizará una función específica `CadenaMes` cuya cabecera se proporciona en la Figura 8.10. Esta función no verifica si el día, mes y año se corresponden con una fecha correcta, por ejemplo: “0 de no-mes de -123”.



Figura 8.9 Cabecera de la función que obtiene la **cadena** de una **fecha**



Figura 8.10 Cabecera de la función que obtiene la **cadena** de un **mes**

6. **Diseña** los diagramas de flujo y **codifica** el programa que muestre la fecha del sistema. Para ello se diseñará y utilizará la función `CadHoy` cuya cabecera se proporciona en la Figura 8.11 hará uso de las funciones del sistema (de Visual Basic) que se estimen necesarias de la Tabla 8.1.

Más concretamente, habrá que declarar una variable de tipo `Date` llamada `hoy` que se inicializará con el valor de la fecha del sistema, mediante una llamada a la función `Date` (nótese que la función y el tipo se llaman de la misma manera). La variable `hoy` será utilizada como parámetro de entrada a las funciones VB `Day`, `Month` y `Year` para obtener el día, mes y año respectivamente en formato numérico. Con estos datos podemos llamar directamente a la función de `CadenaFecha` del ejercicio anterior.



Figura 8.11 Cabecera de la función que obtiene la **cadena** de la fecha del sistema

Tablas de referencia rápida

Función	Descripción
Date As Date	Fecha actual del sistema
Day (ByVal dat As Date) As Integer	Día de una fecha
Month (ByVal dat As Date) As Integer	Mes de una fecha
Year (ByVal dat As Date) As Integer	Año de una fecha

Tabla 8.1. Lista de funciones con fechas de Visual Basic