



Práctica 3

“Subprogramas, parámetros, cadenas de caracteres y arrays”

1. Objetivos

Los objetivos de esta práctica son los siguientes:

- Organizar el código del programa de forma eficiente mediante subprogramas.
- Comprender cómo se diseña, implementa e invoca un subprograma.
- Comprender en qué consiste y cómo se implementa el mecanismo de “paso de parámetros”
- Distinguir entre parámetros “por valor” y parámetros “por referencia”.
- Aprender a declarar y utilizar variables locales en un subprograma.

2. Proyecto a realizar: Consideraciones.

En esta tercera práctica debéis crear **un único proyecto de programación** siguiendo el mismo procedimiento que llevamos a cabo en la práctica anterior.

Nombre del proyecto:

Por ejemplo: FP_P3_GXX, donde XX representa el nº de grupo de laboratorio, por ejemplo 03 => el nombre del proyecto sería FP_P3_G03 => Proyecto 3 realizado por el grupo 03 de laboratorio

A tener en cuenta en el proyecto de programación desarrollado: En esta práctica debéis crear e invocar a todos los subprogramas que se indican a continuación en color rojo.

Cuidad la interfaz con el usuario:

- Al iniciar la ejecución del programa, mostrad un mensaje de bienvenida en pantalla (“*Bienvenido a la aplicación GestorMenu*”) (*Subprograma mensajeBienvenida()*)
- Limpiad pantalla antes de mostrar el menú y antes de ejecutar cada una de las opciones del mismo. (*system(“cls”) o system(“clear”)*)
- Añadid una **pausa** después de mostrar el mensaje de bienvenida inicial y **también antes de finalizar la ejecución de cada una de las opciones del menú** (“*Pulse <intro> para continuar... ” o system(“pause”)*) .
- Antes de finalizar la ejecución del programa, mostrad un mensaje de despedida en pantalla (“... *Gracias por utilizar nuestro programa...Pulse <Intro> para finalizar...*”) (*Subprograma mensajeDespedida()*)

3. Proyecto a realizar: Enunciado.

Escribir un programa que, de forma repetitiva, muestre al usuario el siguiente menú de opciones:

MENU PRINCIPAL

- 1.- Cálculos sobre un triángulo
- 2.- Dibujar triángulo de Pascal
- 3.- Procesar cadena de caracteres
- 4.- Procesar arrays de números enteros
- 0.- Salir

Elija opción:



Las tareas que debe realizar el programa al ejecutar cada una de las opciones del menú anterior son las siguientes:

• **Opción 1: “Cálculos sobre un triángulo”**

Al elegir esta opción, se deben ejecutar los siguientes subprogramas:

- *Subprograma leerCateto()* : Solicita al usuario que introduzca un número real “c” y devuelve al punto de llamada el valor leído.
- *Subprograma calcularHipotenusa()* : Pasándole los catetos haciendo uso de la función anterior, calcula el valor de la hipotenusa de la siguiente forma, y, devuelve al punto de llamada el resultado obtenido.

$$h = \sqrt{c1^2 + c2^2}$$

- *Subprograma calcularArea()*: Calcula el área con los valores de los catetos que recibe como parámetros y devuelve al punto de llamada el resultado obtenido.

$$A = \frac{c1 * c2}{2}$$

- *Subprograma calcularPerimetro()*: Calcula el perímetro (siendo h, c1 y c2 parámetros de este subprograma) y devuelve al punto de llamada el resultado obtenido.

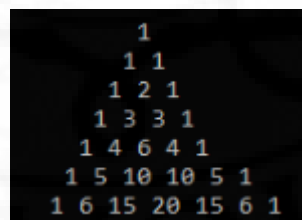
$$p = h + c1 + c2$$

- *Subprograma mostrarResultados()* : Recibe el resultado obtenido al ejecutar los subprogramas anteriores y muestra dicho resultado *del área, hipotenusa y perímetro* en pantalla, así como los catetos considerados.

• **Opción 2: “Dibujar triángulo de Pascal”**

Al ejecutar esta opción, se solicita un número entero mayor que 2 y que será el número de filas del triángulo de Pascal (*subprograma pedirNumero()*) y, a continuación, se invoca al *subprograma dibujarTrianguloPascal()* que dibuja en pantalla el triángulo de Pascal teniendo en cuenta el dato anterior.

Ejemplo: si el usuario introduce el número 7, el triángulo que se mostrará en pantalla será:





- **Opción 3: “Procesar cadena de caracteres”**

Al ejecutar esta opción, se muestra en pantalla el siguiente menú de opciones: (*Subprograma submenuCadena()*):

SUBMENU PROCESAR CADENA

- 1.- Anagrama
- 2.- Convertir a mayúscula las consonantes
- 3.- Invertir cadena
- 4.- Convirtiendo cadenas
- 0.- Volver

Elija opción:

- **Opción 3-1: “Anagrama”** (*Subprograma anagramaCadenas()*)

Escribe una función que reciba dos palabras (String) y retorne si son o no anagramas

- Un Anagrama consiste en formar una palabra reordenando todas las letras de otra palabra inicial.
- No hace falta comprobar que ambas palabras existan.
- Dos palabras exactamente iguales no son anagrama.

Ejemplo:

Introduzca una palabra: caso
Introduzca otra palabra: saco

Las palabras caso y saco son anagramas → este resultado se mostrará después de finalizar la ejecución del subprograma

- **Opción 3-2: “Convertir a mayúscula las consonantes”**

subprograma convertirMayuscConsonantes()

Al ejecutar esta opción, se solicita al usuario introducir una palabra y, a continuación, se crea una nueva cadena de caracteres formada por los caracteres de la palabra original pero con las consonantes en mayúsculas. Finalmente se mostrará en pantalla la nueva cadena obtenida y también un mensaje indicando cuántos caracteres han sido modificados.

Ejemplo:

Introduzca una palabra: sacapuntas

Palabra modificada con consonantes en mayúsculas: SaCaPuNTaS

Se han convertido en mayúsculas 6 consonantes → este resultado se mostrará después de que finalice la ejecución de este subprograma



- **Opción 3-3: “Invertir cadena”** (*Subprograma crearCadenaInvertida()*)

A partir de una cadena que recibe como parámetro el subprograma, crea una nueva cadena que contiene la cadena invertida y muestra la cadena resultante en pantalla.

Ejemplo: → estos mensajes se mostrarán en pantalla **antes** de que finalice la ejecución de este subprograma

Cadena original: Hola mundo
Cadena invertida: odnum aloH

- **Opción 3-4: “Convirtiendo cadenas”** (*Subprograma convertirCadenas()*)

El subprograma recibirá dos cadenas como parámetro (cad1, cad2) e imprimirá otras dos cadenas como salida (cad3, cad4).

- cad3 contendrá todos los caracteres presentes en la cad1 pero NO estén presentes en cad2.
- cad4 contendrá todos los caracteres presentes en la cad2 pero NO estén presentes en cad1

Ejemplo: → estos mensajes se mostrarán en pantalla **después** de que finalice la ejecución de este subprograma

CADENA ORIGINAL	CADENA CONVERTIDA	CARACTERES OMITIDOS
Cadena 1: El gato negro	Cadena 3: Elgtogo	6 caracteres omitidos
Cadena 2: Me parece bien	Cadena 4: Mpcbi	9 caracteres omitidos

- **Opción 3-0: “Volver”**

Al ejecutar esta opción, se mostrará de nuevo en pantalla el menú principal de la aplicación.

- **Opción 4: “Procesar arrays de números enteros”**

Al ejecutar esta opción, se solicita la introducción de 10 números enteros y se almacenan los 5 primeros números en el array tabla1, y los otros 5 números en el array tabla2. A continuación, se presenta en pantalla el siguiente menú de opciones: (*Subprograma submenuArraysEnteros()*):

SUBMENU PROCESAR ARRAYS DE ENTEROS

- 1.- Suma
- 2.- Multiplicación
- 3.- Intersección
- 0.- Volver

Elija opción:

- **Opción 4-1: “Suma”** (*Subprograma sumaArrays()*)

Se reciben como parámetros ambos arrays (tabla1, tabla2) y se suman. El array resultante (tablaSuma) se mostrará **después** de finalizar la ejecución de este subprograma.

- **Opción 4-2: “Multiplicación”** (*Subprograma multiplicacionArrays()*)

Se reciben como parámetros ambos arrays (tabla1, tabla2) y se multiplican. El array resultante (tablaMultiplicación) se muestra **antes** de finalizar la ejecución de este subprograma.

- **Opción 4-3: “Intersección”** (*Subprograma interseccionArrays()*)

Se reciben como parámetros ambos arrays (tabla1, tabla2) y se obtiene el array resultante de la intersección de los elementos de los dos arrays anteriores. El array resultante (tablaIntersección) se muestra **antes** de finalizar la ejecución de este subprograma.



- **Opción 4-0: “Volver”**

Al ejecutar esta opción, se mostrará de nuevo en pantalla el menú principal de la aplicación.

- **Opción 0 (“Salir”):**

Al ejecutar esta opción se muestra el mensaje de despedida en pantalla y finaliza la ejecución del programa. (*Subprograma mensajeDespedida()*)

4. Normas de realización y entrega de la práctica.

- Se debe realizar una **entrega en el Campus virtual** (*un único fichero comprimido .zip o .rar*) que incluya el Proyecto de programación desarrollado y la Memoria completa de la práctica (.pdf).
- **El CÓDIGO FUENTE debe estar correctamente comentado.**

Al principio del programa debe añadir un bloque de comentarios similares a los siguientes:

```


/*****
Programa: GestionDeMenu
Tarea que realiza: Presenta en pantalla un menú de opciones y ejecuta cada una de ellas con ayuda de
subprogramas.
Autor : Carlos Sanz Porta
Fecha: 10 noviembre de 2024
*****/
  
```

Además, antes de cada subprograma debe incluir también un bloque con comentarios que definan el subprograma, tal y como se muestra en el siguiente ejemplo:

```

/*****
Subprograma: SumarNumeros
Tarea que realiza: Calcula y devuelve la suma de dos números enteros pasados por parámetros
Parámetros de E/: num1(entero) y num2 (entero)
Parámetros de S/: suma (entero)
*****/
  
```

- **La MEMORIA DE LA PRÁCTICA debe constar de :**

- **Portada:** donde se especifique el título de la práctica, la fecha de realización, el/los autor/es de la misma, número de grupo de laboratorio, turno (Lunes/Martes/Miércoles/Viernes) y profesor de laboratorio (Pilar, Berta o Francisco). (*¡Imprescindible!*)
- **Índice** (*apartados de la memoria y números de página*)
- **Diagrama de descomposición modular** que incluya el paso de parámetros
- **Ficha de descripción completa de cada subprograma** (similar a las realizadas en clase de teoría): 

<ul style="list-style-type: none"> • Nombre del subprograma • Parámetros de entrada • Parámetros de Salida • Proceso
--
- **Pruebas de ejecución del programa** (*“Capturas de pantallas de ejecución de cada una de las opciones del menú”*).

Fecha límite de entrega (en el campus virtual): **Domingo, 1 de diciembre de 2024, 22 h.**

- **Pasada esta fecha de entrega, no se recogerán más prácticas.** No se admiten entregas por correo electrónico ni por cualquier otro medio que no sea la tarea de entrega de la práctica que tenéis en el campus virtual.
- La entrega en el campus virtual **se cierra automáticamente a las 22 h.** del día indicado.

¡¡ No esperes al último momento para entregar tu trabajo!!