Semana 2: C básico

Guión

- Estructura básica de un programa en C
- Sintaxis
- Tipos de datos, variables y constantes
- Operadores
- Sentencias condicionales
- Sentencias iterativas (bucles)
- Funciones

Recursos de consulta para hacer los ejercicios

- C Tutorial de Tutorialspoint
- The GNU C Programming Tutorial

Observaciones

- Editor recomendado: **Visual Studio Code** (con extensión Live Share)
- Para compilar y ejecutar desde consola:

```
$ gcc -o ejecutable fuente.c [1]
$ ./ejecutable
```

Ejercicios

Contenidos

- funciones de <stdio.h>: scanf(), printf(), getchar()
- Sentencias condicionales
- Bucles
- funciones, paso de argumentos por valor, recursividad

Enunciados y resultado de ejecución

1. Realiza un programa llamado cuadrado.c que lea un nº entero y lo eleve al cuadrado (funciones **scanf()**[2] y **printf()**)

Ejemplo de compilación y ejecución:

```
$ gcc -o cuadrado cuadrado.c
```

[1] Podríamos poner simplemente **gcc fuente.c** y entonces el ejecutable se denominaría **a.out**
[2] Si hemos declarado una variable tipo entero, ej: int numero; la función **scanf** necesitará como argumento la dirección de esa variable y eso lo expresaremos con el operador &: scanf("%d", &numero);

```
$ ./cuadrado
***Cálculo del cuadrado de un número ***
Introduce un número: 89
El cuadrado de 89 es 7921
```

2. Realiza un programa llamado area.c que lea el radio de un círculo y calcule su área (funciones **scanf()** y **printf()**). Se ha de utilizar una constante[3] PI con valor 3.141593. El resultado ha de tener 4 decimales

Ejemplo de ejecución:

```
$ ./area
***Cálculo del área de un círculo ***
Introduce el radio: 3.5
El área del círculo de radio 3.5 es: 38.4845
```

3. Realiza un programa llamado grados.c que convierta grados Fahrenheit a grados centígrados (funciones **scanf()** y **printf()**). La fórmula de conversión es C=(5/9) *(F-32)

Ejemplo de ejecución:

```
$ ./grados
***Conversión de grados Fahrenheit a grados centígrados***
Introduce grados Fahreinheit: 55
55.0 grados Fahrenheit son 12.8 grados centígrados
```

4. Realiza un programa llamado par.c que lea un número e indique si es par o impar (sentencia **if..else**)

Ejemplo de ejecución:

```
$ ./par
***Determinar si un número es par o impar***
Introduce un número: 4
```

[3] const float PI = 3.141593;
o bien
#define PI 3.141593
/* La directiva **#define** no forma parte de la sintaxis de C o C++, son ordenes para el **Preprocesador**, el cuál se ejecuta antes de que el **compilador** entre en acción. El Preprocesador busca en el código cada una de las etiquetas PI y las sustituye por el valor que se haya definido previamente; sería muy parecido a usar el buscar y reemplazar de cualquier editor de texto, carece de cualquier información de tipo. Ha de ir después de los **#include** y antes de la declaracion de funciones. */

```
El número 4 es par
$ ./par
***Determinar si un número es par o impar***
Introduce un número: 7
El número 7 es impar
```

5. Realiza un programa llamado calificacion.c que lea una calificación numérica y devuelva "suspenso" si la nota<5, "aprobado" si 5<=nota <7, "notable si 7<=nota<9 y sobresaliente si 9=<nota <=10. Indicar error si nota>10 (sentencias **if..else** **if..else**)

Ejemplo de ejecución:

```
$ ./calificacion
***Determinar la calificación cualitativa ***
Introduce una nota: 5
Aprobado
$ ./calificacion
***Determinar la calificación cualitativa ***
Introduce una nota: 6.7
Aprobado
$ ./calificacion
***Determinar la calificación cualitativa ***
Introduce una nota: 8.9
Notable
$ ./calificacion
***Determinar la calificación cualitativa ***
Introduce una nota: 9.5
Sobresaliente
$ ./calificacion
***Determinar la calificación cualitativa ***
Introduce una nota: 3.2
Suspenso
$ ./calificacion
***Determinar la calificación cualitativa ***
Introduce una nota: 11
Introducir un valor entre 0 y 10
```

6. Realiza un programa llamado continuar.c que pregunte si se desea continuar (S) o no (N) e indique error si no se introduce ninguna de esas letras (sentencia **switch**). Si el usuario pulsa "s" o "S" le aparecerá el mensaje "Ahora seguimos", si pulsa "n" o "N" le aparecera el mensaje "Nos vemos en otra ocasión", y si no pulsa ninguna de estas letras se le indicará "Opción incorrecta" .

Ejemplo de ejecución:

```
$ ./continuar
***Determinar si continuar o no***
¿Deseas continuar (S) o no (N)? d
Opción incorrecta
$ ./continuar
***Determinar si continuar o no***
¿Deseas continuar (S) o no (N)? s
Ahora seguimos
$ ./continuar
***Determinar si continuar o no***
¿Deseas continuar (S) o no (N)? S
Ahora seguimos
$ ./continuar
***Determinar si continuar o no***
¿Deseas continuar (S) o no (N)? n
Nos vemos en otra ocasión
$ ./continuar
***Determinar si continuar o no***
¿Deseas continuar (S) o no (N)? N
Nos vemos en otra ocasión
$
```

7. Realiza un programa llamado multiplo5.c que muestre, tabulados de 5 en 5, por pantalla los números múltiplos de 5 comprendidos entre 1 y 100 (sentencia **for**, **if**)

Ejemplo de ejecución:

```
$ ./multiplos5
***Números múltiplos de 5 ***
5       10      15      20      25
30      35      40      45      50
55      60      65      70      75
80      85      90      95      100
```

8. Realiza un programa llamado sumatorio.c que lea un número N y calcule 1+2+3+...+N (sentencia **while**)

Ejemplo de ejecución:

```
$ ./sumatorio
***Cálculo de 1+2+...+N***
Introduce un número N: 5
```

```
1+2+...+5 = 15
```

9. Realiza un programa llamado parconbucle.c que lea repetidamente un nº e indique si es par o impar. El programa se repite mientras el número sea distinto de 0 (sentencia **do-while**)

Ejemplo de ejecución:

```
$ ./parconbucle
***Determinar si un número es par o impar***
Introduce un número: 3
El número 3 es impar
Introduce un número: 4
El número 4 es par
Introduce un número: 0
El número 0 es par
```

10. Realiza un programa llamado maximo.c que contenga una función para calcular el máximo de dos números enteros (**funciones**, **paso de argumentos por valor**)

Ejemplo de ejecución:

```
$ ./maximo
***Función máximo de dos números enteros***
Introduce dos números:
3
7
max(3,7)=7
$ ./maximo
***Función máximo de dos números enteros***
Introduce dos números:
8
1
max(8,1)=8
```

11. Realiza un programa llamado contarcaracteres.c que contenga una función para contar los caracteres escritos por el usuario hasta final de flujo (función **getchar()**, **funciones con paso de argumentos por valor**)

Ejemplo de ejecución:

```
$ ./contarcaracteres
***Función contar caracteres de stdin hasta EOF***
```

```
Escribo una frase y acabo con ENTER y luego CTRL+d

Has escrito 51 caracteres
```

12. Realiza un programa que contenga una **función recursiva** para calcular el factorial de un número (**funciones recursivas**)

Ejemplo de ejecución:

```
$ ./factorial
***Cálculo del factorial de un número***
Introduce un número: 10
10! = 3628800
$ ./factorial
***Cálculo del factorial de un número***
Introduce un número: 5
5! = 120
```

Más ejercicios para practicar:

- Simple programs
- Loops
- Nested loops