LENGUAJE C++

¿Qué es C? C es un lenguaje de programación de alto nivel desarrollado en el año 1972 por Dennos Ritchie en AT&T Bell Labs. La legibilidad, facilidad de mantenimiento y la portabilidad son algunas de las ventajas de este lenguaje, además que permite descender a nivel de hardware.

1. ESTRUCTURA DE UN PROGRAMA EN C

1.1 Estructura

Todo programa en **C** consta de una o más funciones, una de las cuales se llama **main**. El programa comienza en la función main, desde la cual es posible llamar a otras funciones.

Cada función estará formada por la cabecera de la función, compuesta por el nombre de la misma y la lista de argumentos (si los hubiese), la declaración de las variables a utilizar y la secuencia de sentencias a ejecutar.

Ejemplo:

```
#include <stdio.h>    libreria o biblioteca

#include <math.h>

#include <conio.h>

declaraciones globales

main( ) {
        variables locales
        bloque
}

funcion1( ) {
        variables locales
        bloque
}
```

1.2 Zona de Declaraciones y Cuerpo del Programa

Zona de declaraciones contiene las librerías y las variables que se van a utilizar en la realización del programa y la función principal además de la declaración de constantes.

Nota: Las variables pueden ir antes de iniciar las instrucciones del programa.

El cuerpo del programa se divide en inicio, instrucciones y fin.

La estructura quedaría de la siguiente manera:

****Zona de Declaraciones ****

```
#include <stdio.h>
```

main() *FUNCIÓN PRINCIPAL*

CUERPO DEL PROGRAMA

int Tipo de variables nombre; Variables

{ Inicio del programa

INSTRUCCIONES

} Fin del programa

Nota: En el lenguaje C no es lo mismo una variable en minúsculas (ejemplo nom) a una en mayúsculas (ejemplo NOM), por lo que te recomiendo que todo lo hagas en minúsculas.

1.3 Comentarios

A la hora de programar es conveniente añadir comentarios (cuantos más mejor) para poder saber que función tiene cada parte del código, en caso de que no lo utilicemos durante algún tiempo.

Además facilitaremos el trabajo a otros programadores que puedan utilizar nuestro archivo fuente.

Para poner comentarios en un programa escrito en **C** usamos los símbolos **/*** y ***/**:

```
/* Este es un ejemplo de comentario */

/* Un comentario también puede
estar escrito en varias líneas */
```

El símbolo **/*** se coloca al principio del comentario y el símbolo ***/** al final.

El comentario, contenido entre estos dos símbolos, no será tenido en cuenta por el compilador

1.4 Palabras clave

Existen una serie de indicadores reservados, con una finalidad determinada, que no podemos utilizar como identificadores.

A continuación vemos algunas de estas palabras clave:

char	**int**	**float**	**double**	**if**
else	**do**	**while**	**for**	**switch**
short	**long**	**extern**	**static**	**default**
continue	**break**	**register**	**sizeof**	**typedef**

1.5 Identificadores

Un identificador es el nombre que damos a las variables y funciones. Está formado por una secuencia de letras y dígitos, aunque también acepta el carácter de subrayado **_**. Por contra no acepta los acentos ni la **ñ**/**Ñ**.

El primer carácter de un identificador no puede ser un número, es decir que debe ser una letra o el símbolo **_**.

Se diferencian las mayúsculas de las minúsculas, así **num**, **Num** y **nuM** son distintos identificadores.

A continuación vemos algunos ejemplos de identificadores válidos y no válidos:

Válidos	**No válidos**
_num	1num
var1	número2
fecha_nac	año_nac

2. TIPOS DE DATOS

2.1 Tipos

En 'C' existen básicamente cuatro tipos de datos, aunque como se verá después, podremos definir nuestros propios tipos de datos a partir de estos cuatro. A continuación se detalla su nombre, el tamaño que ocupa en memoria y el rango de sus posibles valores.

TIPO	**Tamaño**	**Rango de valores**
char	1 byte	-128 a 127
int	2 bytes	-32768 a 32767
float	4 bytes	3'4 E-38 a 3'4 E+38
double	8 bytes	1'7 E-308 a 1'7 E+308

Tipos de Datos

TIPO		TAMAÑO EN BITS	RANGO MÍNIMO
CARACTER	char	8	-128 a 127
	unsigned char	8	0 a 255
	signed char	8	-128 a 127
ENTERO	int	16	-32768 a 32767
	unsigned int	16	0 a 65535
	signed int	16	Igual que **int**
	short int	16	Igual que **int**
	unsigned short int	16	0 a 65535
	signed short int	16	Igual que **short int**
	long int	32	-2147483648 a 2147483647
	signed long int	32	Igual a **long int**
	unsigned long int	32	0 a 4294967295
PUNTO FLOTANTE	float	32	±(1.8e-38 a 3.4e+38) 6 dígitos de precisión
	double	64	±(2.2e-308 a 1.8e+308) 15 dígitos de precisión
	long double	80	±(3.4e-4932 a 1.2e+4932) 18 dígitos de precisión
SIN TIPO	void	0	Sin valor

2.2. Calificadores de tipo

Los calificadores de tipo tienen la misión de modificar el rango de valores de un determinado tipo de variable. Estos calificadores son cuatro:

- **signed**

Le indica a la variable que va a llevar signo. Es el utilizado por defecto.

	tamaño	**rango de valores**
signed char	1 byte	-128 a 127
signed int	2 bytes	-32768 a 32767

- **unsigned**

Le indica a la variable que no va a llevar signo (valor absoluto).

	tamaño	rango de valores
unsigned char	1 byte	0 a 255
unsigned int	2 bytes	0 a 65535

- **short**

  Rango de valores en formato corto (limitado). Es el utilizado por defecto.

	tamaño	rango de valores
short char	1 byte	-128 a 127
short int	2 bytes	-32768 a 32767

- **long**

  Rango de valores en formato largo (ampliado).

	tamaño	rango de valores
long int	4 bytes	-2.147.483.648 a 2.147.483.647
long double	10 bytes	-3'36 E-4932 a 1'18 E+4932

También es posible combinar calificadores entre sí:

signed long int = long int = long

unsigned long int = unsigned long 4 bytes 0 a 4.294.967.295 (El mayor entero permitido en 'C')

2.3 LAS VARIABLES

Una variable es un tipo de dato, referenciado mediante un identificador (que es el nombre de la variable). Su contenido podrá ser modificado a lo largo del programa.

Una variable sólo puede pertenecer a un tipo de dato. Para poder utilizar una variable, primero tiene que ser declarada:

[calificador] <tipo> <nombre>

Es posible inicializar y declarar más de una variable del mismo tipo en la misma sentencia:

[calificador] <tipo> <nombre1>,<nombre2>=<valor>,<nombre3>=<valor>,<nombre4>

Ejemplo

/* Uso de las variables */

```
#include <stdio.h>

main() /* Suma dos valores */
{
        int num1=4,num2,num3=6;
        printf("El valor de num1 es
%d",num1);
        printf("\nEl valor de num3 es
%d",num3);
        num2=num1+num3;
        printf("\nnum1 + num3 =
%d",num2);
}
```

printf (“digite un numero”);

scanf(“%i”, &num1)

¿ Dónde se declaran ?

Las variables pueden ser de dos tipos según el lugar en que las declaremos: *globales* o *locales*.

La variable global se declara antes de la **main( )**. Puede ser utilizada en cualquier parte del programa y se destruye al finalizar éste.

La variable local se declara después de la **main( )**, en la función en que vaya a ser utilizada. Sólo existe dentro de la función en que se declara y se destruye al finalizar dicha función.

El identificador (nombre de la variable) no puede ser una **palabra clave** y los caracteres que podemos utilizar son las letras: **a-z** y **A-Z** (-ojo! la **ñ** o **Ñ** no está permitida), los números: **0-9** y el símbolo de subrayado _. Además hay que tener en cuenta que el primer carácter no puede ser un número.

Ejemplo

```
/* Declaración de variables */

#include <stdio.h>

int a;
main() /* Muestra dos valores */
{
        int b=4;
        printf("b es local y vale
%d",b);
        a=5;
        printf("\na es global y vale
%d",a);
}
```

2.4.- CONSTANTES

Al contrario que las **variables**, las constantes mantienen su valor a lo largo de todo el programa.

Para indicar al compilador que se trata de una constante, usaremos la directiva **#define**:

```
#define <identificador> <valor>
```

Observa que no se indica el punto y coma de final de sentencia ni tampoco el tipo de dato.

La directiva **#define** no sólo nos permite sustituir un nombre por un valor numérico, sino también por una cadena de caracteres.

El valor de una constante no puede ser modificado de ninguna manera.

Ejemplo

```
/* Uso de las constantes */

#include <stdio.h>
#define pi 3.1416
#define escribe printf
main() /* Calcula el perímetro */
{
        int radio;
        printf("Introduce el radio: ");
        scanf("%d",&radio);
        printf("El perímetro es:
%f",2*pi*radio);
}
```

2.5.- SECUENCIAS DE ESCAPE

Ciertos caracteres no representados gráficamente se pueden representar mediante lo que se conoce como secuencia de escape.

A continuación vemos una tabla de las más significativas:

\n	salto de línea \n\n\n\n
\b	retroceso
\t	tabulación horizontal
\v	tabulación vertical
****	contrabarra
\f	salto de página
\'	apóstrofe
\"	comillas dobles
\0	fin de una cadena de caracteres

Ejemplo

```
/* Uso de las secuencias de escape */

#include <stdio.h>

main() /* Escribe diversas sec. de escape */
{
        printf("Me llamo \"Nemo\" el
grande");
        printf("\n")
        printf("\nDirección: C\\ Mayor 25");
        printf("\nHa salido la letra \'L\'");
        printf("\nRetroceso\b");
        printf("\n\tEsto ha sido todo");
}
```

2.6- INCLUSIÓN DE FICHEROS

En la programación en C es posible utilizar funciones que no estén incluídas en el propio programa. Para ello utilizamos la directiva **#include**, que nos permite añadir librerías o funciones que se encuentran en otros ficheros a nuestro programa.

Para indicar al compilador que vamos a incluir ficheros externos podemos hacerlo de dos maneras (siempre antes de las declaraciones).

1. Indicando al compilador la ruta donde se encuentra el fichero.

```
#include "misfunc.h"
#include "c:\includes\misfunc.h"
```

2. Indicando que se encuentran en el directorio por defecto del compilador.

```
#include <misfunc.h>
```

3.- OPERADORES ARITMÉTICOS Y DE ASIGNACIÓN

A continuación se explican los tipos de operadores (aritméticos y de asignación) que permiten realizar operaciones matemáticas en lenguaje C.

3.1.- Operadores aritméticos

Existen dos tipos de operadores aritméticos:

Los binarios: nro=nro+1

+	Suma
-	Resta
*****	Multiplicación
/	División
%	Módulo (resto)

y los unarios:

++	Incremento (suma 1)
- -	Decremento (resta 1)
-	Cambio de signo

Su sintaxis es:

binarios:
<variable1><operador><variable2>

unarios:
<variable><operador> y al revés, **<operador><variable>.**

Ejemplo

```
/* Uso de los operadores aritméticos */

#include <stdio.h>

main() /* Realiza varias operaciones */
{
        int a=1,b=2,c=3,r;
        r=a+b;
        printf("%d + %d = %d\n",a,b,r);
        r=c-a;
        printf("%d - %d = %d\n",c,a,r);
        b++;
        printf("b + 1 = %d",b);
}
```

3.2.- Operadores de asignación

La mayoría de los operadores aritméticos binarios explicados en el capítulo anterior tienen su correspondiente operador de asignación:

=	Asignación simple suma=a+b
+=	Suma
-=	Resta
*=	Multiplicación
/=	División
%=	Módulo (resto)

Con estos operadores se pueden escribir, de forma más breve, expresiones del tipo:

n=n+3 se puede escribir **n+=3**

k=k*(x-2) lo podemos sustituir por **k*=x-2**

Ejemplo

```
/* Uso de los operadores de asignación */

#include <stdio.h>

main() /* Realiza varias operaciones */
{
        int a=1,b=2,c=3,r;
        a+=5;
        printf("a + 5 = %d\n",a);
        c-=1;
        printf("c - 1 = %d\n",c);
        b*=3;
        printf("b * 3 = %d",b);
}
```

3.3.- JERARQUÍA DE LOS OPERADORES

Será importante tener en cuenta la precedencia de los operadores a la hora de trabajar con ellos:

( )	Mayor precedencia
++, - -	
***, /, %**	
+, -	Menor precedencia

Las operaciones con mayor precedencia se realizan antes que las de menor precedencia.

Si en una operación encontramos signos del mismo nivel de precedencia, dicha operación se realiza de izquierda a derecha. A continuación se muestra un ejemplo sobre ello:

a*b+c/d-e

1. a*b resultado = x
2. c/d resultado = y
3. x+y resultado = z

4. z-e

Fijarse que la multiplicación se resuelve antes que la división ya que está situada más a la izquierda en la operación. Lo mismo ocurre con la suma y la resta.

Ejemplo

```
/* Jerarquía de los operadores */

#include <stdio.h>

main() /* Realiza una operación */
{
        int a=6,b=5,c=4,d=2,e=1,x,y,z,r;
        x=a*b;
        printf("%d * %d = %d\n",a,b,x);
        y=c/d;
        printf("%d / %d = %d\n",c,d,y);
        z=x+y;
        printf("%d + %d = %d\n",x,y,z);
        r=z-e;
        printf("%d = %d",r,a*b+c/d-e);
}
```

4. SALIDA Y ENTRADA DE DATOS

4.1.- Sentencia printf( )

La rutina printf permite la aparición de valores numéricos, caracteres y cadenas de texto por pantalla.

El prototipo de la sentencia *printf* es el siguiente:

printf(control,arg1,arg2...);

En la cadena de control indicamos la forma en que se mostrarán los argumentos posteriores. También podemos introducir una cadena de texto (sin necesidad de argumentos), o combinar ambas posibilidades, así como **secuencias de escape**.

En el caso de que utilicemos argumentos deberemos indicar en la cadena de control tantos modificadores como argumentos vayamos a presentar.

El modificador está compuesto por el carácter **%** seguido por un carácter de conversión, que indica de qué tipo de dato se trata.

Ejemplo

```
/* Uso de la sentencia printf() 1. */

#include <stdio.h>

main() /* Saca por pantalla una suma */
{
        int a=20,b=10;
        printf("El valor de a es %d\n",a);
        printf("El valor de b es %d\n",b);
        printf("Por tanto %d+%d=%d",a,b,a+b);
}
```

Los modificadores más utilizados son:

%c	Un único carácter
%d	Un entero con signo, en base decimal
%u	Un entero sin signo, en base decimal
%o	Un entero en base octal
%x	Un entero en base hexadecimal
%e	Un número real en coma flotante, con exponente
%f	Un número real en coma flotante, sin exponente
%s	Una cadena de caracteres
%p	Un puntero o dirección de memoria

Ejemplo

```
/* Uso de la sentencia printf() 2. */

#include <stdio.h>

main() /* Modificadores 1 */
{
        char cad[]="El valor de";
        int a=-15;
        unsigned int b=3;
        float c=932.5;
        printf("%s a es %d\n",cad,a);
        printf("%s b es %u\n",cad,b);
        printf("%s c es %e o %f",cad,c,c);
}
```

El formato completo de los modificadores es el siguiente:

% [signo] [longitud] [.precisión] [l/L] conversión

Signo: indicamos si el valor se ajustará a la izquierda, en cuyo caso utilizaremos el signo menos, o a la derecha (por defecto).

Longitud: especifica la longitud máxima del valor que aparece por pantalla. Si la longitud es menor que el número de dígitos del valor, éste aparecerá ajustado a la izquierda.

***Precisión*: indicamos el número máximo de decimales que tendrá el valor.**

l/L: utilizamos l cuando se trata de una variable de tipo long y L cuando es de tipo double.

Ejemplo

```
/* Uso de la sentencia printf() 3. */

#include <stdio.h>

main() /* Modificadores 2 */
{
        char cad[ ]="El valor de";
        int a=25986;
        long int b=1976524;
        float c=9.57645;
        printf("%s a es %9d\n",cad,a);
        printf("%s b es %ld\n",cad,b);
        printf("%s c es %.3f",cad,c);
}
```

4.2.- Sentencia scanf( )

La rutina scanf permite entrar datos en la memoria del ordenador a través del teclado.

El prototipo de la sentencia *scanf* es el siguiente:

scanf(control,arg1,arg2...);

En la cadena de control indicaremos, por regla general, los modificadores que harán referencia al tipo de dato de los argumentos. Al igual que en la sentencia *printf* los **modificadores** estarán formados por el carácter **%** seguido de un carácter de conversión. Los argumentos indicados serán, nuevamente, las variables.

La principal característica de la sentencia *scanf* es que necesita saber la posición de la memoria del ordenador en que se encuentra la variable para poder almacenar la información obtenida. Para indicarle esta posición utilizaremos el símbolo *ampersand* (**&**), que colocaremos delante del nombre de cada variable. (Esto no será necesario en los arrays).

Ejemplo

```
/* Uso de la sentencia scanf(). */

#include <stdio.h>

main() /* Solicita dos datos */
{
        char nombre[10];
        int edad;
        printf("Introduce tu nombre:
");
        scanf("%s",nombre);
        printf("Introduce tu edad: ");
        scanf("%d",&edad);
}
```

5. OPERADORES RELACIONALES

Los operadores relacionales se utilizan para comparar el contenido de dos variables.

En C existen seis operadores relacionales básicos:

>	Mayor que	
<	Menor que	
>=	Mayor o igual que	
<=	Menor o igual que	
==	Igual que	nro1==nro2
!=	Distinto que	

El resultado que devuelven estos operadores es **1** para Verdadero y **0** para Falso.

Si hay más de un operador se evalúan de izquierda a derecha. Además los operadores == y **!=** están por debajo del resto en cuanto al orden de precedencia.

Ejemplo

```
/* Uso de los operadores relacionales.  */

#include <stdio.h>

main() /* Compara dos números entre ellos */
{
   int a,b;
   printf("Introduce el valor de A: ");
   scanf("%d",&a);
   printf("Introduce el valor de B: ");
   scanf("%d",&b);
   if(a>b)
      printf("A es mayor que B");
   else if(a<b)
      printf("B es mayor que A");
   else
      printf("A y B son iguales");
}
```

6. ESTRUCTURAS DE PROGRAMACIÓN EN LENGUAJE C++

6.1. ESTRUCTURAS SECUENCIALES

Programas

1. Programa que calcule la suma de dos números

```
#include <stdio.h>
  int main()
{
        int num1, num2, resultado;
        printf("Por favor, introduzca un numero:  ");
        scanf("%d",&num1);
       printf("Ahora, inserte otro:  ");
        scanf("%d",&num2);
        resultado=num1+num2;
        printf("\nEl resultado es %d\n",resultado);
}
```

2. Programa que calcula longitudes de circunferencia

```
#include <stdio.h>
  int main(){
        float radio,sol1;
        printf("Bienvenido, calcularemos la longitud de su circunferencia.\n\n");
        printf("Lo unico que debe hacer es introducir el radio: ");
        scanf("%f",&radio);
        longitud=2*3.141592*radio;
        printf("\n\nEl resultado es %f\n\n",longitud);
}
```

3. Programa que calcula la media aritmética de tres números cualesquiera.

```
#include <stdio.h>
       int main(){
```

```
        float num1,num2,num3,media;
        printf("Bienvenido, calcularemos la media aritmetica de tres numeros.\n\n");
        printf("Por favor, introduzca el primero: ");
        scanf("%f",&num1);
        printf("Ahora, inserte el segundo de ellos: ");
        scanf("%f",&num2);
        printf("Por ultimo, teclee el numero final: ");
        scanf("%f",&num3);
        media=(num1+num2+num3)/3;
        printf("\nEl resultado es %f\n\n",media);
}
```

4. Programa que calcula el área de un triángulo (Fórmula de Herón).

```
 #include <stdio.h>
#include <math.h>
  int main(){
       float lado1,lado2,lado3,sp,area;
        printf("Bienvenido. Calcularemos el area del triangulo.\n\n");
        printf("Introduce el primer lado: ");
        scanf("%f",&lado1);
        printf("Ahora, inserta el segundo lado: ");
        scanf("%f",&lado2);
        printf("Por ultimo, escribe el tercer lado: ");
         scanf("%f",&lado3);
        resultado=(lado1+lado2+lado3)/2;
        area=sqrt(resultado*(resultado-lado1)*(resultado-lado2)*(resultado-lado3));
         printf("\nEl area obtenida es %f\n\n",area);
        printf("Muchas gracias por utilizar este programa.\n\n");
}
```

5. Programa que calcula el capital final de un interés simple.

```
#include <stdio.h>
 int main()
 {
       float capital,interes,tiempo,capital_final;
       printf("Bienvenido. Calcularemos el capital final de un interes simple.\n\n");
       printf("Por favor, introduce el capital inicial: ");
       scanf("%f",&capital);
       printf("Ahora, escribe el interes al que esta colocado: ");
       scanf("%f",&interes);
       printf("Por ultimo, inserta el tiempo al que se deja el capital: ");
       scanf("%f",&tiempo);
       capital_final=capital+capital*(interes/100)*tiempo;
       printf("\n\nEl capital final es de %f\n\n",capital_final);
}
```

6. Programa que calcule raíces cuadradas enteras.

```
#include <stdio.h>
#include <math.h>
 int main(){
       int num,resultado;
       printf("Por favor, inserte un numero");
       scanf("%d",&num);
      resultado=sqrt(num);
       printf("\nSu raiz cuadrada es %d\n\n",resultado);
}
```

Otro símbolo que necesitaremos a partir de ahora será el "%", que sirve para calcular el resto de una división. Podemos verlo en el siguiente ejemplo.

7. Programa que calcule el resto de cualquier división entera.

```
#include <stdio.h>
```

```
int main()
{
	int dividendo,divisor,resto;
	printf("Hola, obtendremos el resto de cualquier division entera.\n\n");
	printf("Inserte el dividendo: ");
	scanf("%d",&dividendo);
	printf("Bien, escriba el divisor: ");
	scanf("%d",&divisor);
	resto=dividendo%divisor;
	resto=dividendo mod divisor
	printf("\nEl resto de la division es %d\n\n",resto);
}
```

8) Leer un número y escribir el valor absoluto del mismo.

```
#include<stdio.h>
#include<math.h>
 int main()
{
int num,valor_absoluto;
printf("ingrese un numero para obtener su valor absoluto\n");
scanf("%d",&num);
valor_absoluto=abs(num);
printf("\nvalor dado es\n %d",valor_absoluto);
}
```

9. Programa que imprima en pantalla: HOLA COMO ESTAS.

```
  #include <stdio.h>
  #include <conio.h>
int main()
  {
	printf(“HOLA\n”);
```

```
        printf("COMO
        ESTAS\n");
        getch();
        return 0;   /*finaliza la ejecución de una función y devuelve el control a la función de
        llamada*/
  }
```

Notaras que en el programa anterior está incluida la librería <conio.h> esta librería es utilizada para getch() que se encuentra al final de las instrucciones; getch(); te obliga a presionar una tecla antes de finalizar tu programa.

10. Programa que lee 2 números, los suma, imprime el resultado de la suma y lo multiplica por 2.

```
#include <stdio.h>

#include <conio.h>

int main()

{

 int num1, num2, suma, multiplicacion;

 printf("Digite el primer numero\n");

 scanf("%i",&num1);

 printf("Digite el segundo numero\n");

 scanf("%i",&num2);

 suma=num1+num2;

 multiplicacion=res1*2;

 printf("El resultado de la suma es: %i \n",suma);

 printf("El resultado de la multiplicación es: %i\n", multplicacion);

 getch();

return 0;

}
```

7. SENTENCIAS CONDICIONALES

Este tipo de sentencias permiten variar el flujo del programa en base a unas determinadas condiciones.

Existen varias estructuras diferentes:

7.1.- Estructura IF...ELSE

Sintaxis:

```
if (condición) sentencia;
```

La sentencia solo se ejecuta si se cumple la condición. En caso contrario el programa sigue su curso sin ejecutar la sentencia.

Otro formato:

```
if (condición) sentencia1;
else sentencia2;
```

Si se cumple la condición ejecutará la sentencia1, sinó ejecutará la sentencia2. En cualquier caso, el programa continuará a partir de la sentencia2.

```
/* Uso de la sentencia condicional IF. */

#include <stdio.h>

main() /* Simula una clave de acceso */
{
  int usuario,clave=18276;
  printf("Introduce tu clave: ");
  scanf("%d",&usuario);
```

```
  if(usuario==clave)
    printf("Acceso permitido");
  else
    printf("Acceso denegado");
}
```

Otro formato:

```
if (condición) sentencia1;
else if (condición) sentencia2;
else if (condición) sentencia3;
else sentencia4;
```

Con este formato el flujo del programa únicamente entra en una de las condiciones. Si una de ellas se cumple, se ejecuta la sentencia correspondiente y salta hasta el final de la estructura para continuar con el programa.

Existe la posibilidad de utilizar llaves para ejecutar más de una sentencia dentro de la misma condición.

```
/* Uso de la sentencia condicional ELSE...IF. */

#include <stdio.h>

main() /* Escribe bebé, niño o adulto */
{
  int edad;
  printf("Introduce tu edad: ");
  scanf("%d",&edad);
  if (edad<1)
    printf("Lo siento, te has equivocado.");
  else if (edad<3) printf("Eres un bebé");
  else if (edad<13) printf("Eres un niño");
  else printf("Eres adulto");
}
```

OPERADORES LÓGICOS

Los operadores lógicos básicos son tres:

&&	AND
||	OR
!	NOT (El valor contrario)

Estos operadores actúan sobre expresiones lógicas. Permiten unir expresiones lógicas simples formando otras más complejas.

OPERANDOS		AND	OR
V	V	V	V
V	F	F	V
F	V	F	V
F	F	F	F

V = Verdadero F = Falso

Ejemplo

```
/* Uso de los op. lógicos AND,OR,NOT. */

#include <stdio.h>

main() /* Compara un número introducido */
{
  int numero;
  printf("Introduce un número: ");
  scanf("%d",&numero);
  if(!(numero>=0))
    printf("El número es negativo");
  else if((numero<=100)&&(numero>=25))
    printf("El número está entre 25 y 100");
  else if((numero<25)||(numero>100))
    printf("El número no está entre 25 y 100");
}
```

Problemas Condicionales

Ejemplo:

1. Programa para determinar si un número es par o impar.

```
#include <stdio.h>

int main() {

  int num; int res;

 printf("Programa para determinar naturaleza par o impar de un numero\n\n");

  printf ("Introduzca un numero entero: ");
```

```
  scanf ("%d", &num);
        res = num%2;
        if (res==0) {
      printf ("El numero es par\n");
        } else {
      printf ("El numero es impar\n");
        }
}
```

2. Programa que pida un número del 1 al 5 y diga si es primo o no.

```
 #include <stdio.h>
#include <stdlib.h>  // librería para utilizar system("PAUSE)
int main(){
        int num;
        printf("Introduzca número del 1 al 5:\n");
   scanf("%d",&num);
        if (num!=4) {
     printf("Es primo\n\n");
        }
        else
        {
     printf("No es primo\n\n");
        }
   system("PAUSE");
        return 0;
}
```

3. Programa que lee dos números y dice cual es el mayor y cual es el menor.

```
#include <stdio.h>
#include <conio.h>
int main()
{
int num1, num2;
```

```
printf("Escriba un numero\n");
scanf("%i",&num1);
printf("Escriba otro numero\n");
scanf("%i",&num2);
if (num1>num2)
        printf("El numero %i es mayor que %i\n",num1, num2);
else
        printf("El numero %i es menor que %i\n",num1, num2);
getch();
return 0;
}
```

4. Programa que lee un número e imprime si es positivo o negativo.

```
#include <stdio.h>
#include <conio.h>
int main()
{
int num;
printf("Escribe un numero\n");
scanf("%i",&num);
if (num>=0)
        printf("EL NUMERO %i ES POSITIVO",num);
else
        printf("EL NUMERO %i ES NEGATIVO",num);
getch();
return 0;
}
```

5. Programa que lee dos números e imprime si es divisible.

```
#include<stdio.h>

int main(){

        int num1, num2;

        printf("Por favor digite el primer numero\n");

        scanf("%i",&num1);

        printf("Por favor digite segundo numero\n");

        scanf("%i",&num2);

        if (num1%num2==0){

                printf("el numero %i es divisible entre %i\n",num1,num2);

        }

return 0;

}
```

ELSE-IF

Básicamente el else-if se utiliza para escribir una decisión múltiple y su forma general es de la siguiente manera:

```
if(condición)
        {
        Instrucciones;
        }
 else if (condición)
        }
        Instrucciones;
 else
        {
        Instrucciones;
        }
```

Es importante remarcar que no importa la cantidad de else-if que pongas no existe un límite. El último else se maneja en caso de que ninguno de los anteriores cumpla con la condición.

Programas Decisiones Múltiples

1. Programa que muestra en pantalla una serie de opciones a elegir e imprime el costo del producto elegido.

```
#include <stdio.h>
#include <conio.h>
int main()
{
int p;
printf("SELECCIONE UN PRODUCTO\n\n");
printf("1. REFRESCO\n");
printf("2. PAPAS FRITAS\n");
printf("3. HAMBURGUESA\n");
printf("4. JUGO\n");
scanf("%i",&p);
if (p == 1)
printf("EL COSTO ES: $5.00");
else if(p == 2)
printf("EL COSTO ES: $10.00");
else if(p == 3)
printf("EL COSTO ES: $20.00");
else
printf("EL COSTO ES: $8.00");
getch();
return 0;
}
```

2. Programa que escriba en pantalla un comentario con respecto a la temperatura del día.

```
#include <stdio.h>
int main(){
	int temperatura,Y;
	printf("Bienvenido. Introduzca la temperatura");
	scanf("%d",&temperatura);
	if(temperatura<15){
		printf("\nBrrr... Que frio!\n");
	}else if(temperatura<25){
		printf("\nClima templado\n");
	}else{
		printf("\nBuf!..Que calor!\n");
	}
}
```

3. Programa que resuelve ecuaciones de segundo grado.

```
#include <stdio.h>
#include <math.h>
int main(){
	float A,B,C,D,resultado,sol1,sol2;
	printf("Bienvenido, Resolveremos su ecuacion de segundo grado.\n\n");
	printf("\nPor favor, introduzca el coeficiente A: ");
	scanf("%f",&A);
	printf("\nAhora, escriba el coeficiente B: ");
	scanf("%f",&B);
	printf("\nPor ultimo, inserte el coeficiente C: ");
	scanf("%f",&C);
```

```
        D=B*B-4*A*C;
        if(D<0){
                printf("\n\nDisculpe, no tiene solucion real\n\n");
        }else if(D==0){
                resultado=-B/2*A;
                printf("\n\nEl resultado de la ecuacion es %f\n\n",resultado);
        }else{
                sol1=(-B+sqrt(D))/(2*A);
                sol2=(-B-sqrt(D))/(2*A);
                printf("\n\nLos resultados de la ecuacion son %f y %f\n\n",sol1,sol2);
        }
        printf("\nGracias por utilizar este programa\n\n");
}
```

4. Programa que resuelva la ecuación cuadrática tipo $ax^2 + bx + c$

```
 #include <stdio.h>
#include <math.h>
int main() {
  //Declaración de variables
          double a, b, c, d, e;
   printf("Programa para calcular ecuacion cuadratica tipo a*x^2 + b*x + c = 0\n\n");
   //Obtención de datos
          printf ("Introduzca valor parametro a: ");          scanf ("%lf", &a);
          printf ("Introduzca valor parametro b: ");          scanf ("%lf", &b);
          printf ("Introduzca valor parametro c: ");          scanf ("%lf", &c);
//Cálculo y muestra de resultados
          d = pow (b,2) - 4 * a * c;          e = 2 * a;
          if (d == 0) { printf ("El resultado es x1 = x2 =%lf", - b / e);
          } else {
```

```
if (d > 0) {
   printf ("El resultado es:\n");
   printf ("x1 = %lf \n", (-b + sqrt(d)) / e);
   printf ("x2 = %lf \n", (-b - sqrt(d)) / e);
    } else {
   printf ("El resultado es: \n");
   printf ("x1 = %lf + %lf * i \n", -b / e, sqrt(-d)/e);
   printf ("x2 = %lf - %lf * i \n", -b / e, sqrt(-d)/e);
    }
    }
return 0;
    }
```

5. visualizar la tarifa de la luz según el gasto de corriente eléctrica. Para un gasto menor de 1000 kwxh la tarifa es de 1.2, entre 1000 y 1850 kwxh es de 1.0 y mayor de 1850 kwxh la tarifa es de 0.9.

```
#include<stdio.h>
#define tarifa1 1.2
#define tarifa2 1
#define tarifa3 0.9
int main(){
    float gasto, tasa;
    printf("digite el gasto de energia ");
    scanf("%f",&gasto);
    if (gasto<1000){
        tasa = tarifa1;
    }
    if(gasto>1000 && gasto<1850){
        tasa = tarifa2;
    }
        if(gasto>1850){
```

```
            tasa = tarifa3;

        }

        printf("la tarifa a pagar es :%.3f",tasa);

}
```

6. *Un proveedor de estéreos ofrece un descuento del 10% sobre el precio sin IVA, de algún aparato si este cuesta $200.000 o más. Además, independientemente de esto, ofrece un 5% de descuento si la marca es "NOSY". Determinar cuanto pagara, con IVA incluido, un cliente cualquiera por la compra de su aparato.*

```
#include <stdio.h>
#include <math.h>
#include <string.h>
int main()
{
float producto, valor_sin_iva, descuento, total, valor_apagar1, valor_apagar, descuento_marca, iva,
descuento_marc;
char marca[]="nosy";

printf("introduzca el valor del producto:\n");
scanf("%f", &producto);

printf("digite la marca del producto:\n");
scanf("%s",marca);

        iva=(producto*19)/100 ;
printf("valor del iva es :%.2f\n",iva);
        valor_sin_iva=(producto-iva) ;
printf("valor del producto sin iva:%.2f\n",valor_sin_iva);

if (producto>=200000)  {

descuento=(valor_sin_iva*10/100);
printf("el descuento del 10 por ciento del estereo es %.2f\n",descuento);
total=producto-descuento;
printf("el valor a pagar es de: %.2f\n",total);

} else{

printf("el producto no tiene descuento del 10 por ciento %.2f\n",producto);
}if (strcmp (marca,"nosy")== 0 ){

descuento_marca=(valor_sin_iva*5)/100 ;

printf("El descuento por la marca nosy es:%.2f\n",descuento_marca);
valor_apagar=(producto-descuento-descuento_marca);

printf("el valor a pagar con el descuento por ser marca nosy es %.2f\n", valor_apagar);
}
}
```

7.2. LA ESTRUCTURA CONDICIONAL SWITCH … CASE

Esta estructura se suele utilizar en los menús, de manera que según la opción seleccionada se ejecuten una serie de sentencias.

Su sintaxis es:

```
switch (variable){
        case contenido_variable1:
                sentencias;
                break;
        case contenido_variable2:
                sentencias;
                break;
        default:
                sentencias;
}
```

Cada case puede incluir una o más sentencias sin necesidad de ir entre llaves, ya que se ejecutan todas hasta que se encuentra la sentencia **BREAK**. La variable evaluada sólo puede ser de tipo **entero** o **carácter**. **default** ejecutará las sentencias que incluya, en caso de que la opción escogida no exista.

```
/* Uso de la sentencia condicional SWITCH. */
#include <stdio.h>

main() /* Escribe el día de la semana */
{
   int dia;
   printf("Introduce el día: ");
   scanf("%d",&dia);
   switch(dia){
      case 1: printf("Lunes"); break;
      case 2: printf("Martes"); break;
      case 3: printf("Miércoles"); break;
      case 4: printf("Jueves"); break;
      case 5: printf("Viernes"); break;
      case 6: printf("Sábado"); break;
```

```
        case 7: printf("Domingo"); break;
    }
}
```

El condicional switch permite elegir entre varias opciones posibles. En realidad, este condicional puede ser expresado también usando condicionales if, por lo que podríamos decir que no es estrictamente necesario. Sin embargo, en algunos casos resulta más claro usar el condicional switch y resulta útil.

La sintaxis a emplear con C es la siguiente:

```
switch (selector) {

case [valor selector 1]:
Instrucción 1;
Instrucción 2;

case [valor selector 2]:
Instrucción 3;
Instrucción 4;
.
.
.
case [valor selector n]:
Instrucción j;
Instrucción k;

default:
Instrucción x;
Instrucción y;
}
```

La variable a evaluar o selector ha de ser un valor con equivalencia ordinal (equivalencia numérica entera). Por ejemplo puede ser una variable tipo int ó char (ya que cada char lleva un número entero asociado), pero no puede ser un double ni un float ni una cadena de caracteres (string). Sólo se puede evaluar una expresión y no múltiples expresiones.

Las sentencias break; hacen que una vez verificado que se cumple una opción dentro del switch, se salga del mismo y se continúe la ejecución en la siguiente instrucción después del switch. El uso de

break; es opcional, pero es habitual incluir un break; después de cada evaluación. En caso de no hacerlo, se ejecutarán todas las instrucciones dentro de cualquier case (incluso aunque no se verifique) después del que ha verificado la condición hasta encontrar una sentencia break; o terminar la ejecución del switch. Esto puede generar resultados contradictorios o no esperados, de ahí que en general siempre se incluyan sentencias break;

La cláusula default hace que se ejecuten las instrucciones a continuación de ella en caso de que la ejecución del programa llegue a alcanzarla. Si se usan las sentencias break; en cada evaluación, sólo se ejecutará lo indicado en default si no se ha encontrado una coincidencia previa.

La evaluación de expresiones es de coincidencia entre la variable evaluada y el valor indicado en case. Por ejemplo, case 12 indicaría que si la expresión evaluada vale 12 se ejecutarán las instrucciones anexas. También se puede usar case empleando constantes simbólicas (pero no con constantes definidas con const).

Veamos un ejemplo: ***programa que permita calcular el salario de un trabajador, dependiendo del nivel de antigüedad,***

1. *//ejemplo con la condición switch*

```
#include <stdio.h>

#include <stdlib.h>

#include <math.h>

int main()

{

        int nivel;

        float salario, salario_nuevo;

        printf("Introduce el nivel de antiguedad del trabajador:\n");

        scanf("%d",&nivel);

        printf("Introduce tu salario:\n");

        scanf("%f",&salario);

        switch (nivel) {

        case 5:

          salario_nuevo=salario+(salario*.035);

        printf("\nTu salario es:%.2f\n",salario_nuevo);

        break;

        case 6:

        salario_nuevo=salario+(salario*.041);
```

```
        printf("\nTu salario es: %.2f\n",salario_nuevo);
        break;
        case 7:
          salario_nuevo=salario+(salario*.048);
        printf("\nTu salario es: %.2f\n",salario_nuevo);
            break;
        case 8:
          salario_nuevo=salario+(salario*.053);
        printf("\nTu salario es: %.2f\n",salario_nuevo);
        break;
        default:
        printf("\nTu salario es:%.2f\n",salario);
}
system("PAUSE");
return 0;
}
```

2. Programa que pida por teclado un número de un día de la semana y muestre por pantalla el nombre correspondiente a dicho día.

```
#include <stdio.h>
int main()
{
  int dia;
  printf( "\n   Introduzca día de la semana en número: " );
  scanf( "%d", &dia );
  switch ( dia )
  {
        case 1 : printf( "\n   Lunes" );
        break;
```

```
        case 2 : printf( "\n   Martes" );
        break;
        case 3 : printf( "\n   Miercoles" );
        break;
        case 4 : printf( "\n   Jueves" );
        break;
        case 5 : printf( "\n   Viernes" );
        break;
        case 6 : printf( "\n   Sabado" );
        break;
        case 7 : printf( "\n   Domingo" );
        break;
  default : printf( "\n   ERROR: Dia incorrecto." );
  }
  return 0;
}
```

3. Programa que realice el cálculo del día del año.

```
#include <stdio.h>
#include <stdlib.h>
 #define MAXDIA 31
#define MAXMES 12
int main() {
        int nDia = 0; int nMes = 0; int diaDelAnyo = 0;
        printf("Calculo del dia del a%co (no bisiesto)\n", 164);
        print ("");
        printf ("Introduzca el numero del dia: ");
        scanf("%d", &nDia);
```

```
printf ("Introduzca el numero del mes: ");
scanf("%d", &nMes);
if ( nDia >= 1 && nDia <= MAXDIA && nMes >= 1 && nMes <= MAXMES) {
switch (nMes) {
case 1: diaDelAnyo = nDia; break;
case 2: diaDelAnyo = nDia + 31; break;
case 3: diaDelAnyo = nDia + 59; break;
case 4: diaDelAnyo = nDia + 90; break;
case 5: diaDelAnyo = nDia + 120; break;
case 6: diaDelAnyo = nDia + 151; break;
case 7: diaDelAnyo = nDia + 181; break;
case 8: diaDelAnyo = nDia + 212; break;
case 9: diaDelAnyo = nDia + 243; break;
case 10: diaDelAnyo = nDia + 273; break;
case 11: diaDelAnyo = nDia + 304; break;
case 12: diaDelAnyo = nDia + 334; break;
default: puts ("Datos no validos");
}
}
if (diaDelAnyo !=0) {
printf ("El %d del %d es el dia %d del a%co\n", nDia, nMes, diaDelAnyo, 164);
} else {
puts ("Los datos no son validos");
}
return 0;
}
```

4. Se quiere escribir un programa que:

1º) Muestre el listado de los signos del zodíaco, con sus números asociados.

2º) Pida por teclado un número (dato entero) asociado a un signo del zodíaco.

3º) Muestre la categoría a la que pertenece el signo del zodíaco seleccionado.

Nota: Si el número introducido por el usuario, no está asociado a ningún signo del zodíaco, se mostrará el mensaje: "ERROR: <número> no está asociado a ningún signo.".

```
#include <stdio.h>

int main()

{

  int numero;

  printf( "\n   Listado de signos del zodiaco:" );

  printf( "\n\n   1. Aries" );

  printf( "\n   2. Tauro" );

  printf( "\n   3. Geminis" );

  printf( "\n   4. Cancer" );

  printf( "\n   5. Leo" );

  printf( "\n   6. Virgo" );

  printf( "\n   7. Libra" );

  printf( "\n   8. Escorpion" );

  printf( "\n   9. Sagitario" );

  printf( "\n   10. Capricornio" );

  printf( "\n   11. Acuario" );

  printf( "\n   12. Piscis" );

  printf( "\n\n   Introduzca numero de signo: " );

  scanf( "%d", &numero );

  switch ( numero )

  {

        case  1 :

        case  5 :

        case  9 : printf( "\n   Es un signo de Fuego." );
```

```
        break;
        case  2 :
        case  6 :
        case 10 : printf( "\n   Es un signo de Tierra." );
        break;
        case  3 :
        case  7 :
        case 11 : printf( "\n   Es un signo de Aire." );
        break;
        case  4 :
        case  8 :
        case 12 : printf( "\n   Es un signo de Agua." );
        break;
        default : printf( "\n   ERROR: %d no está asociado a ningún signo.", numero );
  }
  return 0;
}
```

5. *Realizar un programa con un menú de 4 opciones:*

- ***1. Calcular el doble de un número entero.***
- ***2. Calcular la mitad de un número entero.***
- ***3. Calcular el cuadrado de un número entero.***
- ***4. Salir.***

2º) Pida por teclado la opción deseada (dato entero).

3º) Ejecute la opción del menú seleccionada.

4º) Repita los pasos 1º, 2º y 3º, mientras que, el usuario no seleccione la opción 4 (Salir) del menú.

```
#include <math.h>
```

```
#include <stdio.h>
int main()
{
        int n, opcion;
        do
        {
        printf( "\n   1. Calcular el doble de un n%cmero entero.", 163 );
        printf( "\n   2. Calcular la mitad de un n%cmero entero.", 163 );
        printf( "\n   3. Calcular el cuadrado de un n%cmero entero.", 163 );
        printf( "\n   4. Salir." );
        printf( "\n\n   Introduzca opci%cn (1-4): ", 162 );
        scanf( "%d", &opcion );
        /* Inicio del anidamiento */
        switch ( opcion )
        {
        case 1: printf( "\n   Introduzca un n%cmero entero: ", 163 );
                scanf( "%d", &n );
                printf( "\n   El doble de %d es %d\n\n", n, n * 2 );
                break;
        case 2: printf( "\n   Introduzca un n%cmero entero: ", 163 );
                scanf( "%d", &n );
                printf( "\n   La mitad de %d es %f\n\n", n, ( float ) n / 2 );
                break;
        case 3: printf( "\n   Introduzca un n%cmero entero: ", 163 );
                scanf( "%d", &n );
                printf( "\n   El cuadrado de %d es %d\n\n", n, ( int ) pow( n, 2 ) );
        }
        /* Fin del anidamiento */
```

```
    } while ( opcion != 4 );

    return 0;

}
```

- *Véase que, se han realizado dos castings, (float) y (int), para cambiar, respectivamente, los tipos de datos de los valores resultantes de las expresiones n / 2 y pow(n, 2).*
- *·Por otra parte, fíjese que el bucle do...while iterará, mientras que, opcion sea distinto del valor 4. Normalmente, en un menú, la opción de salir (opción 4 en este caso) no se debe contemplar en la alternativa múltiple, es decir, si el usuario introduce un 4, no se debe hacer nada. Pero, ¿qué ocurre si el usuario teclea un número mayor que 4 ó menor que 1?.*
- *Al introducir un número menor que 1 ó mayor que 4, se muestra de nuevo el menú. Para evitar que ocurra esto, es conveniente utilizar un filtro al leer la opción que introduce el usuario.*

Solución 2: filtrando la opción introducida por el usuario

```
#include <math.h>
#include <stdio.h>
int main()
{
   int n, opcion;
   do
   {
        printf( "\n   1. Calcular el doble de un n%cmero entero.", 163 );
        printf( "\n   2. Calcular la mitad de un n%cmero entero.", 163 );
        printf( "\n   3. Calcular el cuadrado de un n%cmero entero.", 163 );
        printf( "\n   4. Salir." );

        /* Filtramos la opción elegida por el usuario */
        do
        {
        printf( "\n   Introduzca opci%cn (1-4): ", 162 );
        scanf( "%d", &opcion );

        } while ( opcion < 1 || opcion > 4 );
```

```
        /* La opción sólo puede ser 1, 2, 3 ó 4 */

        switch ( opcion )
        {
        case 1: printf( "\n   Introduzca un n%cmero entero: ", 163 );
                scanf( "%d", &n );
                printf( "\n   El doble de %d es %d\n\n", n, n * 2 );
                break;

        case 2: printf( "\n   Introduzca un n%cmero entero: ", 163 );
                scanf( "%d", &n );
                printf( "\n   La mitad de %d es %f\n\n", n, ( float ) n / 2 );
                break;

        case 3: printf( "\n   Introduzca un n%cmero entero: ", 163 );
              scanf( "%d", &n );
                printf( "\n   El cuadrado de %d es %d\n\n", n, ( int ) pow( n, 2 ) );
        }

    } while ( opcion != 4 );
    return 0;
}
```

- La variable opción, también puede ser un dato de tipo carácter, en vez de tipo entero.

Solución 3: declarando la variable opcion de tipo carácter

```
/* Programa: Menú de opciones (Solución 3) */
#include <stdio.h>
#include <math.h>
int main()
{
        char opcion;
        int n;
        do
```

```
    {
    printf( "\n   1. Calcular el doble de un n%cmero entero.", 163 );
    printf( "\n   2. Calcular la mitad de un n%cmero entero.", 163 );
    printf( "\n   3. Calcular el cuadrado de un n%cmero entero.", 163 );
     printf( "\n   4. Salir." );
    do
    {
    printf( "\n   Introduzca opci%cn (1-4): ", 162 );
    fflush( stdin );   // limpieza del buffer
    scanf( "%c", &opcion );
    } while ( opcion < '1' || opcion > '4' );
    switch ( opcion )
    {
    case '1': printf( "\n   Introduzca un n%cmero entero: ", 163 );
              scanf( "%d", &n );
              printf( "\n   El doble de %d es %d\n\n", n, n * 2 );
              break;

    case '2': printf( "\n   Introduzca un n%cmero entero: ", 163 );
              scanf( "%d", &n );
              printf( "\n   La mitad de %d es %f\n\n", n, ( float ) n / 2 );
              break;
    case '3': printf( "\n   Introduzca un n%cmero entero: ", 163 );
              scanf( "%d", &n );
              printf( "\n   El cuadrado de %d es %d\n\n", n, ( int ) pow( n, 2 ) );
    }
    } while ( opcion != '4' );
return 0;
```

```
}
```

6. Programa que pida un número del 1 al 7 y diga el día de la semana correspondiente.

```
 #include <stdio.h>
#include <stdlib.h>
int main()
{
        int i;
        printf("Introduzca numero del 1 al 7:\t"); // \t tabulador – espacio
   scanf("%d",&i);
    switch(i){
          case 1:
              printf ("Lunes\n\n");
              break;
          case 2:
              printf ("Martes\n\n");
              break;
          case 3:
              printf ("Miercoles\n\n");
              break;
          case 4:
              printf ("Jueves\n\n");
              break;
          case 5:
              printf ("Viernes\n\n");
              break;
          case 6:
              printf ("Sabado\n\n");
              break;
```

```
        case 7:
            printf ("Domingo\n\n");
                break;
        default:
            printf ("Opción no valida\n\n");
            break;
        }
   system("PAUSE");
        return 0;
}
```

7. Programa que pida una letra y detecte si es una vocal

```
 #include <stdio.h>
#include <stdlib.h>
 int main()
{
        char c;
        printf("Introduzca un carácter:\t");
          scanf("%c",&c);
        switch (c)
        {
       case 'a':
           printf ("Es vocal\n\n");
           break;
       case 'e':
             printf ("Es vocal\n\n");
           break;
       case 'i':
```

```
            printf ("Es vocal\n\n");
            break;
        case 'o':
            printf ("Es vocal\n\n");
            break;
        case 'u':
              printf ("Es vocal\n\n");
            break;
        default:
            printf ("No es vocal\n\n");
            break;
          }
   system("PAUSE");
          return 0;
}
```

8. Programa que muestre un menú donde las opciones sean "Equilátero", "Isósceles" y "Escaleno", pida una opción y calcule el perímetro del triángulo seleccionado.

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
          int lado, base, opcion;
          printf("Introduzca lado del triangulo:\t");
          scanf("%d",&lado);
           printf("Introduzca base del triangulo:\t");
           scanf("%d",&base);
           printf ("\n") ;
           printf("Seleccione opcion:\n\n");
           printf("1 - Equilatero\n");
```

```
        printf("2 - Isosceles\n");
        printf("3 - Escaleno\n");
        scanf("%d",&opcion);
switch (opcion)
{
    case 1:
        printf("El perímetro es:%d\n",3*lado);
        break;
    case 2:
        printf("El perímetro es:%d\n",(2*lado)+base);
        break;
    case 3:
        printf("El perímetro es:%d\n",lado + lado + lado);
        break;
    default:
    printf("Opcion no valida.");
    break;
}
system("PAUSE");
return 0;
}
```

8. BUCLES

Los bucles son estructuras que permiten ejecutar partes del código de forma repetida mientras se cumpla una condición.

Esta condición puede ser simple o compuesta de otras condiciones unidas por operadores lógicos.

8.1 Sentencia WHILE

Su sintaxis es:

while (condición) sentencia;

Con esta sentencia se controla la condición antes de entrar en el bucle. Si ésta no se cumple, el programa no entrará en el bucle.

Naturalmente, si en el interior del bucle hay más de una sentencia, éstas deberán ir entre llaves para que se ejecuten como un bloque.

```
/* Uso de la sentencia WHILE. */
#include <stdio.h>
main() /* Escribe los números del 1 al 10 */
{
  int numero=1;
  while(numero<=10)
  {
        printf("%d\n",numero);
        numero++;
  }
}
```

La diferencia básica entre el bucle while y el bucle do…while, es que en el segundo se ejecutan como mínimo una vez las acciones contenidas dentro del bucle.

1. *Programa que visualice los números del 1 al 10.*

```
// Librerias a incluir

#include <stdio.h>

#include <stdlib.h>

// Funcion principal

int main()
```

```
{
   // Crear variables auxiliares
   int contador=0;
   // Inicializar el contador
   contador=1;
   // repetir mientras que el contador tenga
   // un valor inferior a 10
   while (contador<11) {
      // visualizar el valor de contador
      printf("\n%d",contador);
      // incrementar contador en una unidad
      contador=contador+1;
   };
      return 0;
}
```

2. Que calcule el factorial de un número cualquiera y lo muestre en pantalla.

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
  int num, num2;
  printf("Introduce número: ");
  scanf("%d",&num);
  num2=num;
```

```
while (num2!=1){
        num2=num2-1;
        num=num*num2;
}
printf("El factorial es: %d \n", num);
system("PAUSE");
return 0;
}
```

3. *Que calcule la media de X números, se dejarán de solicitar números hasta que se introduzca el cero.*

```
#include <stdio.h>
#include <stdlib.h>
#include <clocale>  // libreria para que reconozca las tildes y signos
int main(void)
{
        setlocale(LC_CTYPE,"Spanish"); // función para que reconozca las tildes y signos
   int num=1,cont=0;
   float sum=0;
   while (num!=0)
   {
      printf("Introduzca número:  ");
      scanf("%d",&num);
      sum=sum+num;
      if (num!=0)
      {
        cont=cont+1;
      }
        }
```

```
  printf("La media es:%6.2f\n",sum/cont);

 system("PAUSE");

   return 0;

}
```

4. *Programa que genere los números del 10 al 40*

```
#include <stdio.h>
main() {

        int i=0, j=0, final=40;
        while(i<final) {

i=j*10;
printf("%d\n",i);
j++;}
j=100; /* sentencia que ejecutamos cuando deje de cumplirse la condición del while. */

}
```

5. *Una compañía de seguros tiene contratados a N vendedores. Cada uno hace tres ventas a la semana. Su política de pagos es que un vendedor recibe un sueldo base, y un 10% extra por comisiones de sus ventas. El gerente de su compañía desea saber cuánto dinero obtendrá en la semana cada vendedor por concepto de comisiones por las tres ventas realizadas, y cuanto tomando en cuenta su sueldo base y sus comisiones.*

```
#include <stdio.h>

#include <math.h>

#include <conio.h>

#include <stdlib.h>

int main() {

    int sueldo,r1,t,c1,c2,c3,v1,v2,v3,sueldot,ct;

   printf("*ANALIZADOR DE SUELDOS DE LOS VENDEDORES DE SEGUROS*\n\n");

   r1=1;

            while(r1<=1){

                t=t+1;
```

```
            printf("\n\nANALIZADOR DEL VENDEDOR %d \n\n",t);
        printf("Escriba el sueldo base del vendedor %d \n",t);
        scanf("%d",&sueldo);
            printf("\nAnalisis de las ventas del vendedor %d \n\n ",t);
            printf("Digite el valor de la venta 1\n");
        scanf("%d",&v1);
        c1=v1*0.1;
            printf("\nEl valor de la comision 1 es de %d \n\n",c1);
            printf("Digite el valor de la venta 2\n");
        scanf("%d",&v2);
        c2=v2*0.1;
            printf("\nEl valor de la comision 2 es de %d\n\n",c2);
            printf("Digite el valor de la venta 3\n");
        scanf("%d", &v3);
        c3=v3*0.1;
            printf("\nEl valor de la comision 3 es de %d\n\n",c3);
        ct=c1+c2+c3;
            printf("\n\nLa ganancia en comisión del vendedor %d es de %d
\n",t,ct);
        sueldot=ct+sueldo;
            printf("\n\nEl sueldo total del vendedor %d es de %d\n ",t,sueldot);
            printf("\n\nHay otro vendedor?\n\n");
            printf("1. Si\n");
            printf("2. No\n");

        scanf("%d",&r1);
        printf("\n\n\n");

        }
```

```
                printf("Gracias por utilizar el programa");
                getch();
        return 0;
}
```

6. *Adivinar el número.*

```
#include <stdio.h>
int main(int argc, char *argv[])
{
        int num, num2, opc=0;
    printf("\n Adivinar numero");
        printf("\n 1 - Comenzar.");
        printf("\n 2 - Salir.\n");
        printf("\n Introduce una opcion:");
  scanf("%d",&opc);
    while (opc!=2)
        {
        num = rand() % 100;//Origina aleatoriamente numeros entre 0 y 99

     printf("\n Introduce numero: ");
        scanf("%d",&num2);

        while(num!=num2)
        {
        if (num>num2)
         printf("Es mayor");
        else
        printf("Es menor");

        printf("\n Introduce numero: ");
        scanf("%d",&num2);
        }
```

```
            printf("\n Has acertado! \n");
            printf("\n 1 - Jugar de nuevo.");
            printf("\n 2 - Salir.");
            printf("\n Introduce una opcion:");
            scanf("%d",&opc);
            }
            system("PAUSE");
            return 0;
      }
```

8.2 Sentencia DO...WHILE

Su sintaxis es:

do{

sentencia1;

sentencia2;

}while (condición);

Con esta sentencia se controla la condición al final del bucle. Si ésta se cumple, el programa vuelve a ejecutar las sentencias del bucle.

La única diferencia entre las sentencias while y do...while es que con la segunda el cuerpo del bucle se ejecutará por lo menos una vez.

```
/* Uso de la sentencia DO...WHILE. */
 #include <stdio.h>
 main() /* Muestra un menú si no se pulsa 4 */
{
   char seleccion;
   do{
         printf("1.- Comenzar\n");
```

```
            printf("2.- Abrir\n");
            printf("3.- Grabar\n");
            printf("4.- Salir\n");
            printf("Escoge una opción: ");
            seleccion=getchar();
            switch(seleccion){
             case '1':printf("Opción 1");
                      break;
             case '2':printf("Opción 2");
                      break;
             case '3':printf("Opción 3");
            }
    }while(seleccion!='4');
}
```

EJERCICIOS

1. Un proveedor de estéreos ofrece un descuento del 10% sobre el precio sin IVA, de algún aparato si este cuesta $200.000 o más. Además, independientemente de esto, ofrece un 5% de descuento si la marca es "NOSY". Determinar cuanto pagara, con IVA incluido, un cliente cualquiera por la compra de su aparato.

Ejercicio con el repetir (do… while)

```
#include <stdio.h>
#include <math.h>
#include <string.h>  // cadena de caracteres
int main()
{
float producto, valor_sin_iva, descuento, total, valor_apagar1, valor_apagar, descuento_marca, iva,
descuento_marc;
char marca[]="nosy";
char resp[2];
do {
printf("introduzca el valor del producto:\n");
scanf("%f", &producto);
printf("digite la marca del producto:\n");
scanf("%s",marca);
iva=(producto*19)/100 ;
```

```
printf("valor del iva es :%.2f\n",iva);
valor_sin_iva=(producto-iva) ;
printf("valor del producto sin iva:%.2f\n",valor_sin_iva);
if (producto>=200000)  {
descuento=(valor_sin_iva*10/100);
printf("el descuento del 10 por ciento del estereo es %.2f\n",descuento);
total=producto-descuento;
printf("el valor a pagar es de: %.2f\n",total);
} else{
printf("el producto no tiene descuento del 10 por ciento %.2f\n",producto);
}if (strcmp (marca,"nosy")== 0 ){
descuento_marca=(valor_sin_iva*5)/100 ;
printf("El descuento por la marca nosy es:%.2f\n",descuento_marca);
valor_apagar=(producto-descuento-descuento_marca);
printf("el valor a pagar con el descuento por ser marca nosy es %.2f\n", valor_apagar);
}
printf ("\nSi desea repetir escriba -si-, si desea terminar escriba -no-\n");
scanf("%s",& resp);
} while (strcmp (resp,"si") == 0);
}
```

2. En un juego de preguntas a las que se responde "Si" o "No" gana quien responda correctamente las tres preguntas. Si se responde mal a cualquiera de ellas ya no se pregunta la siguiente y termina el juego. Las preguntas son:

1. *Colón descubrió América?*

2. *La independencia de México fue en el año 1810?*

3. *The Doors fue un grupo de rock Americano?*

```
#include <stdio.h>
#include <string.h>
int main () {
char resp1[2], resp2[2], resp3[2], resp4[2];
do {
printf ("\nBienvenido, para poder pasar debe responder correctamente las 3 preguntas. Para
responder escriba -si- o -no- segun corresponda y luego pulse enter\n");
printf ("1. Colon descubrio America?\n");
scanf ("%s", &resp1);
if (strcmp(resp1, "no") == 0 ){
printf ("\nUsted perdio\n");
printf ("\nSi desea repetir la pegunta escriba -Si-, si desea terminar escriba -No-\n");
scanf("%s",& resp4);
} else if (strcmp(resp1,"si") == 0 ){
printf ("\nCorrecto.\n 2. La independencia de Mexico fue en 1810?\n");
scanf ("%s", &resp2);
if (strcmp(resp2,"no") == 0) {
printf ("\nUsted perdio\n");
printf ("\nSi desea repetir la pregunta escriba -Si-, si desea terminar escriba -No-\n");
scanf("%s",& resp4);
} else if (strcmp(resp2,"si") == 0) {
printf ("\nCorrecto.\n 3. The Doors fue un grupo de rock americano?\n");
scanf ("%s",& resp3);
if (strcmp(resp3,"no") == 0) {
```

```
printf ("\nUsted perdio");
printf ("\nSi desea repetir la pregunta escriba -Si-, si desea terminar escriba -No-\n");
scanf("%s",& resp4);
} else if (strcmp (resp3,"si") == 0) {
printf ("\nFelicidades. Usted gano\n");
}
printf ("\nSi desea repetir la pregunta escriba -si-, si desea terminar escriba -no-\n");
scanf("%s",& resp4);
}
}
} while (strcmp (resp4,"si") == 0);
}
```

3. Tabla de multiplicar de cualquier número

```
#include <stdio.h>

#include <math.h>

#include <string.h>

int main (){

        int multiplicando,multiplicador, producto;

        printf("Ingrese el numero al que se le va a calcular su tabla de multiplicar\n\n");

        scanf("%i", &multiplicador);

        printf("MULTIPLICANDO \t    MULTIPLICADOR      \t PRODUCTO");

        multiplicando=1;

        do{

                producto=multiplicando*multiplicador;

printf("\n\n\t %i \t x \t  %i  \t  = \t  %i",multiplicando,multiplicador,producto);

        multiplicando=multiplicando+1;

        }  while (multiplicando<=10);

return 0;

        }
```

4. Hacer un programa que al ingresar tres números diferentes imprima el número medio.

```
#include <stdio.h>

#include <string.h>

#include <stdlib.h> //system (“cls”)
```

```
int main (){
int numero1,numero2,numero3;
char resp[2];
do{
system("cls");   // limpiar pantalla
printf("Ingresar tres numeros distintos:\n");
scanf("%d",&numero1);
scanf("%d",&numero2);
scanf("%d",&numero3);
        if (numero1<numero2 && numero1>numero3){
        printf("El numero intermedio entre ellos:%d\n",numero1);
        } else if (numero2<numero3 && numero2>numero1){
        printf("El numero intermedio entre ellos:%d\n",numero2);
        } else if (numero3<numero2 && numero3>numero1){
        printf("El numero intermedio es el:%d\n",numero3);
        } printf("Desea repitir el procedimiento, digite si o no: \n");
        scanf("%s",resp);
} while(strcmp(resp,"si")==0);
}
```

5. Imprimir los números del 1 al 10 cada uno con su respectivo factorial.

```
#include <stdio.h>
#include <math.h>
#include <conio.h> //    getch()
#include <stdlib.h>
int main() {
        int nun1 = 1,factorial;
        printf("Tabla de factoriales   \n ");
        printf("NUMERO                  FACTORIAL \n\n");
```

```
   do{
                printf(" %d                    %d \n",nun1,factorial);
                nun1=nun1+1;
                factorial=factorial*nun1;
        }  while( nun1 <= 10 );
        getch();
        return 0;
}
```

6. En un supermercado un cajero captura los precios de los artículos que los clientes compran e indica a cada cliente cual es el monto de lo que deben pagar. Al final del día le indica a su supervisor cuanto fue lo que cobró en total a todos los clientes que pasaron por su caja.

```
#include<stdio.h>
#include<string.h> //cadena de caracteres
int main () {
float precio,contador,cantidad,valortot;
char reps[2];
contador=0;
do{
printf("diga el precio del producto.\n");
scanf("%f",&precio);
printf("digite el n%cmero de productos.\n",163);
scanf("%f",&cantidad);
valortot=(precio*cantidad);
contador=contador+valortot;
printf("%cdesea ingresar otro producto?.\n",168);  //%c y el número 168 imprimir el signo de interrogación de apertura
scanf("%s",&reps);
}while (strcmp(reps,"si")==0 || (reps,"Si")==0 || (reps,"SI")==0);
printf("el total de la compra es:%2.f",contador);
}
```

7. En una tienda de descuento se efectúa una promoción en la cual se hace un descuento sobre el valor de la compra total según el color de la bolita que el cliente saque al pagar en caja. Si la bolita es de color blanco no se le hará descuento alguno, si es verde se le hará un 10% de descuento, si es amarilla un 25%, si es azul un 50% y si es roja un 100%. Determinar la cantidad final que el cliente deberá pagar por su compra. Se sabe que solo hay bolitas de los colores mencionados.

```
#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <conio.h>

#include <clocale>


int main(){

		setlocale(LC_CTYPE,"Spanish");

		system ("color 0A");

		float compra, descuento, total_compra, acumulador;

		int contador_amarilla, contador_azul, contador_verde,
contador_blanca,contador_roja;

		char bolita[15];

		char respuesta [2];

		contador_blanca=0;

		contador_verde=0;

		contador_amarilla=0;

		contador_azul=0;

		contador_roja=0;

		acumulador=0;

	do{

	printf("digite el valor de su compra\n");

	scanf ("%f",&compra);

	printf ("digite el color de la bolita\n");

	scanf ("%s",&bolita);
```

```
        if  (strcmp(bolita,"blanca")==0) {
                total_compra=compra;
                printf ("el total a pagar es de:%.2f \n", total_compra, " no tiene descuento por ser
blanca \n");
                contador_blanca=contador_blanca+1;
                acumulador=acumulador+total_compra;
                printf ("el acumulador de las compras es:%.2f\n ", acumulador);
        }else
                if      (strcmp(bolita,"verde")==0) {
                        descuento=(compra*10)/100;
                        printf(" su descuento por elegir la bolita de color verde es del 10 %.2f \n",
descuento);
                        total_compra=compra-descuento;
                        printf(" su total a pagar es de :%.2f \n  ", total_compra);
                        contador_verde=contador_verde+1;
                        acumulador=acumulador+total_compra;
                        printf(" el acumulador de las compras es:%.2f \n ", acumulador);
                }else
                        if  (strcmp(bolita,"amarilla")==0) {
                                descuento=(compra*25)/100;
                                printf(" su descuento por elegir la bolita de color amarilla es del
25:%.2f \n ", descuento);
                                total_compra=compra-descuento;
                                printf(" su total a pagar es de : %.2f \n", total_compra);
                                contador_amarilla=contador_amarilla+1;
                                acumulador=acumulador+total_compra;
                                printf ("el acumulador de las compras es:%.2f \n ", acumulador);
                        }else
                                if  (strcmp(bolita,"azul")==0) {
```

```
                                        descuento=(compra*50)/100;
                                        printf (" su descuento por elegir la bolita de color azul es del
50:%.2f\n ", descuento);
                                        total_compra=compra-descuento;
                                        printf (" su total a pagar es de :%.2f\n  ", total_compra);
                                        contador_azul=contador_azul+1;
                                        acumulador=acumulador+total_compra;
                                        printf(" el acumulador de las compras es:%.2f \n ",
acumulador);
                                } else
                                        if  (strcmp(bolita,"roja")==0) {
                                                descuento=(compra*100)/100;
                                                printf (" su descuento por elegir la bolita de color
roja es del 100: %.2f \n",descuento);
                                                total_compra=compra-descuento;
                                                printf (" su total a pagar es de :%.2f \n",
total_compra);
                                                contador_roja=contador_roja+1;
                                                acumulador=acumulador+total_compra;
                                                printf(" el acumulador de las compras es: %.2f \n",
acumulador);
                                        }

        printf (" el total de las bolitas blancas son: %i \n", contador_blanca);
        printf (" el total de las bolitas verdes son :%i \n", contador_verde);
        printf (" el total de las bolitas amarillas son:%i \n", contador_amarilla);
        printf (" el total de las bolitas azul son:%i \n", contador_azul);
        printf (" el total de las bolitas rojas son:% i \n", contador_roja);

        printf ("Desea ingresar otra compra \n");
        scanf ("%s", respuesta);
```

```
} while ((strcmp(respuesta,"Si")==0) || (strcmp(respuesta,"SI")==0) || (strcmp(respuesta,"si")==0));

return 0;

}
```

8.3 Sentencia FOR

Su sintaxis es:

```
for (inicialización;condición;incremento){

        sentencia1;

        sentencia2;

}
```

La inicialización indica una variable (variable de control) que condiciona la repetición del bucle. Si hay más, van separadas por comas:

```
for (a=1,b=100;a!=b;a++,b- -){
```

El flujo del bucle FOR transcurre de la siguiente forma:

```
/* Uso de la sentencia FOR. */
 #include <stdio.h>
 main() /* Escribe la tabla de multiplicar */
{
   int num,x,result;
   printf("Introduce un número: ");
   scanf("%d",&num);
   for (x=0;x<=10;x++){
      result=num*x;
      printf("\n%d por %d = %d\n",num,x,result);
   }
}
```

Ejercicios

1. ***Programa que muestra los veinte primeros números naturales.***

```
#include <stdio.h>
int main(){
        int i=21;
        const int tope=20;
        for(i=0;i<=20;i=i+1){
                printf("%d\n",i);
        }
        printf("\nHasta pronto");
```

```
}
```

2. Programa que muestra los números pares hasta 30.

```
 #include <stdio.h>
int main(){
        int i=31;
        const int tope=30;
        for(i=0;i<=30;i=i+2){
                printf("%d\n",i);
        }
        printf("\nHasta pronto");
}
```

3. Programa que muestre los múltiplos de siete (hasta 123).

```
 #include <stdio.h>
int main(){
        int i=124;
        const int tope=123;
        for(i=0;i<=123;i=i+7){
        printf("%d\n",i);
        }
        printf("\n,Hasta pronto");
}
```

4. Programa que muestre una cuenta atrás desde diez hasta cero.

```
 #include <stdio.h>
int main(){
        int i=10;
        const int tope=0;
```

```
        for(i=10;i>=0;i=i-1){
                printf("%d\n",i);
        }
        printf("\nHasta pronto!\n");
}
```

5. Programa que muestre los números impares que haya del 1 al 100

```
 #include <stdio.h>
#include <stdlib.h>
int main()
{
          int x;
          for (x=1;x<=100;x++)
           {
           if (x%2!=0)
           {
          printf("%d\n",x);
           }
           }
           system("PAUSE");
           return 0;
}
```

6. Programa que escriba las tablas de multiplicar del 0 al 10.

```
 #include <stdio.h>
#include <stdlib.h>
int main(void)
{
```

```
        int x,y;
        for (x=0;x<=10;x++)
        {
        for (y=1;y<=10;y++)
        {
        printf("%d X %d = %d \n",x,y,x*y);
        }
        printf("\n");
        }
        printf("\n");
        system("PAUSE");
        return 0;
}
```

7. Programa que muestre la tabla de multiplicar de un número cualquiera.

```
 #include <stdlib.h>
int main(void)
{
        int x,num;
                printf("Introduce numero:\t");
                scanf("%d",&num);
        printf("\n");
        for (x=1;x<=10;x++)
        {
        printf("%d X %d = %d \n",num,x,num*x);
        }
        printf("\n");
        system("PAUSE");
```

```
        return 0;
}
```

8. Programa que calcule "x" términos de la sucesión de Fibonacci.

```
 #include <stdio.h>
#include <stdlib.h>
int main(){
      printf("Bienvenido al programa para calcular la progresión de Fibonacci.\n\n");
      int veces, primer=0,segun=1,proximo,r;
      char borrado;
      printf("Introduzca el numero de terminos: ");
   scanf("%d",&veces);
      scanf("%c",&borrado);
      system("cls");
      printf("He aqui la sucesion de %d terminos: \n",veces);
      for(int i=0;i<=veces;i++){
        r=primer+segun;
              primer=segun;
              segun=r;
          printf("\n\t\t\t\t%d",r);
     }
        printf("\n\nGracias por utilizar este programa.\n\n");
}
```

9. Programa que calcule el factorial de un número.

```
#include <stdio.h>
```

```
int main(){
	int i,num,fact=1;
	printf("Bienvenido al programa para calcular factoriales.\n");
		printf("\nEscriba un numero entero: ");
		scanf("%d",&num);
		for(i=num;i>1; i--){
		 fact=fact*i;
		}
		printf("\nEl factorial de %d es %d\n",num,fact);
}
```

10. Programa que genera apuestas de fútbol.

```
#include <stdlib.h>
#include <time.h>
int main(){
	int a;
	srand((unsigned)time(NULL));
	printf("Bienvenido, aquí tiene su apuesta de futbol: ");
	printf("\n\n");
	for(int i=1; i<=15; i++){
		a=rand()%(3);
		if(a==1){
			printf("\t\t\t\t%d - 1\n",i);
		}else if(a==2){
			printf("\t\t\t\t%d - 2\n",i);
		}else{
			printf("\t\t\t\t%d - X\n",i);
		}
```

```
    }
}
```

11. Programa que simule el lanzamiento de una moneda las veces que el usuario desee, posteriormente hará un recuento de las veces que ha salido tanto cara como cruz.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
int main(){
    int x,veces,cara=0,cruz=0;
    srand((unsigned)time(NULL));
    printf("Pruebe a lanzar la moneda.\n\n");
    printf("Cuantas veces?: ");
    scanf("%d",&veces);
    for(int i=1;i<=veces;i++){
        x=rand()%(2);
        if(x==1){
            printf("\nCara\n\n");
            cara++;
        }else{
            printf("\nCruz\n\n");
            cruz++;
        }
    }
    printf("\n\tRecuento\n\n");
    printf("La cara ha salido %d veces.\n",cara);
    printf("La cruz ha salido %d veces.\n\n",cruz);
    printf("Gracias por utilizar este programa.\n\n");
}
```

9. ARREGLOS

*Un array es un identificador que referencia un conjunto de datos del mismo tipo. Imagina un tipo de dato **int**; podremos crear un conjunto de datos de ese tipo y utilizar uno u otro con sólo cambiar el índice que lo referencia. El índice será un valor entero y positivo. En **C** los arrays comienzan por la posición **0**.*

9.1 VECTORES - UNIDIMENSIONALES

Un vector es un array unidimensional, es decir, sólo utiliza un índice para referenciar a cada uno de los elementos. Su declaración será:

tipo nombre [tamaño];

*El tipo puede ser cualquiera de los ya conocidos y el tamaño indica el número de elementos del vector (se debe indicar entre corchetes **[ ]**). En el ejemplo puedes observar que la variable **i** es utilizada como índice, el primer **for** sirve para rellenar el vector y el segundo para visualizarlo. Como ves, las posiciones van de **0** a **9** (total 10 elementos).*

```
/* Declaración de un array. */

#include <stdio.h>

main() /* Rellenamos del 0 - 9 */
{
        int vector[10],i;
        for (i=0;i<10;i++) vector[i]=i;
        for (i=0;i<10;i++) printf(" %d",vector[i]);
}
```

Podemos inicializar (asignarle valores) un vector en el momento de declararlo. Si lo hacemos así no es necesario indicar el tamaño. Su sintaxis es:

tipo nombre []={ valor 1, valor 2...}

Ejemplos:

```
int vector[]={1,2,3,4,5,6,7,8};
char vector[]="programador";
char vector[]={'p','r','o','g','r','a','m','a','d','o','r'};
```

*Una particularidad con los vectores de tipo **char** (cadena de caracteres), es que deberemos indicar en qué elemento se encuentra el fin de la cadena mediante el carácter nulo (**\0**). Esto no lo controla el compilador, y tendremos que ser nosotros los que insertemos este carácter al final de la cadena.*

*Por tanto, en un vector de 10 elementos de tipo **char** podremos rellenar un máximo de 9, es decir, hasta **vector[8]**. Si sólo rellenamos los 5 primeros, hasta **vector[4]**, debemos asignar el caracter nulo a **vector[5]**. Es muy sencillo: **vector[5]='\0';** .*

*Ahora veremos un ejemplo de cómo se rellena un vector de tipo **char**.*

```
/* Vector de tipo char. */

#include <stdio.h>
```

```
main() /* Rellenamos un vector char */
{
        char cadena[20];
        int i;
        for (i=0;i<19 && cadena[i-1]!=13;i++)
                cadena[i]=getche( );
        if (i==19) cadena[i]='\0';
        else cadena[i-1]='\0';
        printf("\n%s",cadena);
}
```

*Podemos ver que en el **for** se encuentran dos condiciones:*

***1.-** Que no se hayan rellenado todos los elementos (**i<19**).*

***2.-** Que el usuario no haya pulsado la tecla ENTER, cuyo código ASCII es **13**. (**cadena[x-i]!=13**).*

*También podemos observar una nueva función llamada **getche( )**, que se encuentra en **conio.h**. Esta función permite la entrada de un caracter por teclado. Después se encuentra un **if**, que comprueba si se ha rellenado todo el vector. Si es cierto, coloca el caracter nulo en el elemento nº20 (**cadena[19]**). En caso contrario tenemos el **else**, que asigna el caracter nulo al elemento que almacenó el caracter ENTER.*

En resumen: al declarar una cadena deberemos reservar una posición más que la longitud que queremos que tenga dicha cadena.

.- Llamadas a funciones con arrays

Como ya se comentó en el tema anterior, los arrays únicamente pueden ser enviados a una función por referencia. Para ello deberemos enviar la dirección de memoria del primer elemento del array. Por tanto, el argumento de la función deberá ser un puntero.

```
/* Envío de un array a una función. */

#include <stdio.h>

void visualizar(int []); /* prototipo */
main() /* rellenamos y visualizamos */
{
        int array[25],i;
        for (i=0;i<25;i++)
        {
                printf("Elemento nº %d",i+1);
                scanf("%d",&array[i]);
        }
        visualizar(&array[0]);
}

void visualizar(int array[]) /* desarrollo */
{
```

```
        int i;
        for (i=0;i<25;i++) printf("%d",array[i]);
}
```

En el ejemplo se puede apreciar la forma de enviar un array por referencia. La función se podía haber declarado de otra manera, aunque funciona exactamente igual:

declaración o prototipo

void visualizar(int *);

desarrollo de la función

void visualizar(int *array)

EJERCICIOS

1. *Que lea 10 números por teclado, los almacene en un vector y muestre la suma, resta, multiplicación y división de todos.*

```
#include <stdio.h>
#include <clocale>
#include <stdlib.h>

int main(){
                    setlocale(LC_CTYPE,"Spanish");
    int x,tabla[10];
    int sum,res,mul,div;

    for (x=0;x<10;x++)
    {
  printf("Introduzca un número\n");
  scanf("%d",&tabla[x]);

}

    sum=tabla[0];

res=tabla[0];

mul=tabla[0];

div=tabla[0];

    for (x=1;x<10;x++)

    {

  sum=sum+tabla[x];

  res=res-tabla[x];

  mul=mul*tabla[x];

  div=div/tabla[x];
```

```
    }

        printf("Suma: %d\n",sum);

        printf("Resta: %d\n",res);

        printf("Multiplicación: %d\n",mul);

        printf("División: %d\n",div);

    system("PAUSE");

    return 0;

}
```

2. *Que rellene un vector de dos dimensiones con números pares, que pida una posición X,Y y mostrar el número correspondiente.*

```
#include <stdio.h>
#include <clocale>
#include <stdlib.h>

int main(int argc, char *argv[])

{
    setlocale(LC_CTYPE,"Spanish");
    int x,y,num=2, numeros[3][3];

    for (x=0;x<3;x++)
    {
       for (y=0;y<3;y++)

       {
          numeros[x][y]=num;

          num=num*2;

       }

    }

    printf("Introduzca coordenada x: ");
    scanf("%d",&x);
    printf("Introduzca coordenada y: ");
    scanf("%d",&y);
    printf("El número es: %d\n",numeros[x][y]);
    system("PAUSE");
    return 0;
}
```

3. *Vamos a crear un programa que lea un vector de 10 posiciones, luego determine si la quinta posición es positiva, si la primera posición es negativa y si la ultima posición es cero.(*)*

```
#include <stdio.h>
   #include <conio.h>
   #define N 10
   main()
   {
      float  x[N];
      int  i;

      for(i=0;  i<N; i++) {
         printf("Ingrese el valor %d:\n", i);
         scanf("%f",  &x[i]);
      }
      if(x[ 4]> 0)
         printf( "La  quinta Posición es Positiva\n\n");

       if(x[ 0]< 0)
         printf( "La 1era Posición es  Negativo\n\n");

       if(x[N- 1]== 0)
         printf( "La última Posición es  Cero\n\n");

      getch();
   return  0;
   }
```

4. Desarrollar el código C para un programa que calcule la superficie de un terreno que le corresponde a un heredero después de n generaciones, partiendo de una superficie inicial en la generación cero. Se supone que hay división a partes iguales entre herederos y que el número máximo de generaciones con que trabajará el programa es 50. Los datos de superficie inicial, número de generaciones y número de herederos por generación se deben solicitar al usuario del programa.

```
#include <stdio.h>

#include <stdlib.h>

#define MAXGENERACIONES 50

int main() {

          int hGen[MAXGENERACIONES]; int n = 0; int i=0;

          double supin = 0.0; double toca = 0.0;

   printf("***Calculo superficie herederos***\n\n");

   printf("Indique el numero de generaciones: ");

   scanf("%d", &n);

   printf("Indique la superficie inicial: ");

   scanf("%lf", &supin);
```

```
        toca = supin;

        for (i=1; i<=n; i++) {

    printf("Indique el numero de herederos de la generacion %d: ", i);

    scanf("%d", &hGen[i]);

        toca = toca/hGen[i];

        }

        printf("Al heredero actual le corresponde una superficie de %.2lf m2\n", toca);

        return 0;

}
```

5. Que rellene un array con los 100 primeros números enteros y los muestre en pantalla en orden ascendente.

```
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
        int x,tabla[100];

        for (x=1;x<=100;x++)
        {
        tabla[x]=x;
  }

        for (x=1;x<=100;x++)
        {
        printf("%d\n",tabla[x]);
  }

        system("PAUSE");
        return 0;
}

}
```

7. Que rellene un array con los números primos comprendidos entre 1 y 100 y los muestre en pantalla en orden ascendente.

```
#include <stdio.h>
#include <stdlib.h>

int main(void)14
{
        int x,cont,z,i,tabla[100];
```

```
	i=0;
	for (x=1;x<=100;x++)
	{
	cont=0;
	 for (z=1;z<=x;z++)
	{
	if (x%z==0)
	{
	cont++;
	}
	}

	if (cont==2 || z==1 || z==0)
	{
	tabla[i]=x;
	i++;
	}

	}

	for (x=0;x<i;x++)
	 {
	printf("%d\n",tabla[x]);
   }

	system("PAUSE");
	return 0;
}
```

8. Que rellene un array con los números pares comprendidos entre 1 y 100 y los muestre en pantalla en orden ascendente.

```
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
	int x,cont,z,i,tabla[100];

	i=0;
	for (x=1;x<=100;x++)
	{
	cont=0;
	if (x%2==0)
	{
```

```
        tabla[i]=x;
        i++;
        }
        }

        for (x=0;x<i;x++)
         {
        printf("%d\n",tabla[x]);
   }

        system("PAUSE");
        return 0;
}
```

9. Que rellene un array con los números impares comprendidos entre 1 y 100 y los muestre en pantalla en orden ascendente.

```
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
        int x,cont,z,i,tabla[100];

        i=0;
        for (x=1;x<=100;x++)
        {
        cont=0;
        if (x%2==1)
        {
        tabla[i]=x;
        i++;
        }
        }

        for (x=0;x<i;x++)
         {
        printf("%d\n",tabla[x]);
   }

        system("PAUSE");
        return 0;
}
```

10. Que lea 10 números por teclado, los almacene en un array y muestre la suma, resta, multiplicación y división de todos.

```
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
        int x,tabla[10];
        int sum,res,mul,div;

        for (x=0;x<10;x++)
         {
        printf("Introduzca número\n");
        scanf("%d",&tabla[x]);
        }

        sum=tabla[0];
        res=tabla[0];
        mul=tabla[0];
        div=tabla[0];

         for (x=1;x<10;x++)
        {
        sum=sum+tabla[x];
      res=res-tabla[x];
        mul=mul*tabla[x];
        div=div/tabla[x];
        }

        printf("Suma: %d\n",sum);
        printf("Resta: %d\n",res);
        printf("Multiplicación: %d\n",mul);
        printf("División: %d\n",div);

   system("PAUSE");
        return 0;
}
```

11. Que lea 10 números por teclado, los almacene en un array y los ordene de forma ascendente.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
  float aux, numeros[10];
```

```
 int i,j,n=10;

 for (i=0;i<n;i++){
                printf("Escriba un número");
                scanf("%f",&numeros[i]);
        }

 for(i=0;i<n-1;i++)
 {
  for(j=i+1;j<n;j++)
  {
        if(numeros[i]<numeros[j])
        {
        aux=numeros[i];
        numeros[i]=numeros[j];
        numeros[j]=aux;
     }
  }
 }

 for (i=n-1;i>=0;i--){
        printf("%f\n",numeros[i]);
 }

 system("PAUSE");
 return 0;
}
```

12. Que lea 10 números por teclado, 5 para un array y 5 para otro array distinto. Mostrar los 10 números en pantalla mediante un solo array.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
 int aux, numeros1[5],numeros2[5],numeros3[10];
 int i,j;

 for (i=0;i<5;i++){
              printf("Escriba un número");
                scanf("%d",&numeros1[i]);
        }

 for (i=0;i<5;i++){
                printf("Escriba un número");
```

```
            scanf("%d",&numeros2[i]);
        }


 for(i=0;i<5;i++)
 {
  numeros3[i]=numeros1[i];
 }

 for(i=0;i<5;i++)
 {
  numeros3[5+i]=numeros2[i];
 }
```

6. Que rellene un array con los 100 primeros números enteros y los muestre en pantalla en orden descendente.

```
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
        int x,tabla[100];

        for (x=1;x<=100;x++)
        {
        tabla[x]=x;
   }

        for (x=100;x>=1;x--)
        {
        printf("%d\n",tabla[x]);
   }

        system("PAUSE");
        return 0;

  for (i=0;i<10;i++){
        printf("%d\n",numeros3[i]);
 }

 system("PAUSE");
 return 0;
}
```

13. Que lea 5 números por teclado, los copie a otro array multiplicados por 2 y muestre el segundo array.

```
#include <stdio.h>
#include <stdlib.h>

int main()
```

```
{
  int aux, numeros1[5],numeros2[5];
  int i,j;

  for (i=0;i<5;i++){
                printf("Escriba un número");
                scanf("%d",&numeros1[i]);
          }

  for(i=0;i<5;i++)
  {
   numeros2[i]=numeros1[i]*2;
  }

  for (i=0;i<5;i++){
          printf("%d\n",numeros2[i]);
  }

  system("PAUSE");
  return 0;
}
```

14. Que lea 5 números por teclado, los copie a otro array multiplicados por 2 y los muestre todos ordenados usando un tercer array.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
  int aux, numeros1[5],numeros2[5],numeros3[10];
  int i,j;

  for (i=0;i<5;i++){
                printf("Escriba un número");
                scanf("%d",&numeros1[i]);
          }

  for(i=0;i<5;i++)
  {
   numeros2[i]=numeros1[i]*2;
  }

  for(i=0;i<5;i++)
  {
   numeros3[i]=numeros1[i];
  }

  for(i=0;i<5;i++)
  {
   numeros3[5+i]=numeros2[i];
  }

  for (i=0;i<10;i++){
          printf("%d\n",numeros3[i]);
```

```
 }

 system("PAUSE");
 return 0;
}
```

15. Que rellene un array con los 100 primeros números pares y muestre su suma.

```
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
        int x,cont,sum,i,tabla[100];

        i=0;
        sum=0;
        for (x=1;x<=100;x++)
        {
        cont=0;
        if (x%2==0)
        {
        tabla[i]=x;
        i++;
        }
        }

        for (x=0;x<i;x++)
        {
        sum=sum+tabla[x];
        }

        printf("%d\n",sum);

        system("PAUSE");
   return 0;
}
```

16. Que lea 10 números por teclado, los almacene en un array y muestre la media.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
 float sum, numeros1[10];
 int i;

 sum=0;
 for (i=0;i<10;i++){
                printf("Escriba un número");
                scanf("%f",&numeros1[i]);
        }

 for(i=0;i<10;i++)
```

```
 {
  sum=sum+numeros1[i];
 }

 printf("%f\n",sum/10);

 system("PAUSE");
 return 0;
}
```

17. Que mediante un array almacene números tanto positivos como negativos y los muestre ordenados.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
 float aux, numeros[10];
 int i,j,n=10;

 for (i=0;i<n;i++){
            printf("Escriba un número");
            scanf("%f",&numeros[i]);
        }

 for(i=0;i<n-1;i++)
 {
  for(j=i+1;j<n;j++)
  {
        if(numeros[i]<numeros[j])
        {
        aux=numeros[i];
        numeros[i]=numeros[j];
        numeros[j]=aux;
    }
  }
 }

 for (i=n-1;i>=0;i--){
        printf("%f\n",numeros[i]);
 }

 system("PAUSE");
 return 0;
}
```

18. Que rellene un array con 20 números y luego busque un número concreto.

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
```

```
        int  i,x=0,vector[20], n=20, dato, centro,inf=0,sup=n-1;

        for (i=0;i<20;i++){
            printf("Escriba un número");
     scanf("%d",&vector[i]);
   }

        printf("Escriba el número a buscar");
   scanf("%d",&dato);

        while(inf<=sup)
        {
        centro=(sup+inf)/2;
        if (vector[centro]==dato)
        {
        printf("Existe\n");
        x=1;
        break;
        }
        else if(dato < vector [centro] )
        {
        sup=centro-1;
        }
        else
        {
        inf=centro+1;
        }
        }

        if (x==0)
   {
        printf("No existe\n");
   }

        system("PAUSE");
        return 0;

}
```

19. Almacenar n números en un vector, almacenarlos en otro vector en orden inverso al vector original e imprimir el vector resultante.

```
#include <stdio.h>
int main(){
  int i=0,n;
  int temp;
  int vectorA[i];



  printf("Digite la cantidad de numeros a calcular\n");
  scanf("%d", &n);

  for(i=0; i<n; i++){
  printf("Digite el valor %d del vector A\n", i+1);
```

```
 scanf("%d", &vectorA[i]);
 }

 printf("\n");
 printf("VECTOR ORIGINAL\n\n");
 for(i=0; i<n; i++){
 printf("posicion %d: %d \n", i+1, vectorA[i]);
 }

 for(i=0; i<n/2; i++){
   int temp=vectorA[i];
    }

  for (int i=0; i<n/2; i++) {
    int temp=vectorA[i];
    vectorA[i]=vectorA[n-1-i];
    vectorA[n-1-i]=temp;
    }

   printf("\n");
 printf("VECTOR RESULTANTE\n\n");
 for(int i=0; i<n; i++){
   printf("posicion %d: %d \n", i+1, vectorA[i]);
 }
 return 0;

}
```

EJERCICIO 12 TALLER. Desarrollar un algoritmo que permita almacenar la cédula y el nombre de n estudiantes. 2. El usuario puede ingresar un numero de cedula a buscar en el vector y el algoritmo debe mostrar el nombre que corresponde al número de documento ingresado, siempre y cuando haya sido almacenado previamente.

```
#include <stdio.h>
#include<clocale>
#define N 3

struct registro{
        char nombre;
   int cedula;
}personas[N];

int main()
{
        setlocale(LC_CTYPE,"Spanish");

   int opcion, i;

   printf("Por favor ingrese los datos a consultar\n\n");
```

```
        printf("Por favor ingrese el nombre: ");
        scanf("%s",&personas[i].nombre) ;
        fflush(stdin);

                printf("Por favor ingrese la cédula: ");
        scanf("%i",&personas[i].cedula) ;
        fflush(stdin);

             printf("\n\n");

        printf("Ingrese el numero del registro realizado:\n ");
        scanf("%d",&opcion);
        printf("\n\n");

        if(opcion >=0 && opcion<N){
           printf( "Los datos consultados son:\n");
           printf("\n\n");
           printf("Nombre:%s",personas[opcion].nombre);
           printf("Cédula:%i",personas[opcion].cedula);
        }else{
           printf("\n\n");
           printf("Indice errado\n");
                   }
    return 0;
}
```

DE OTRA FORMA

```
#include <iostream>

#include <conio.h>

#include <stdio.h>

#include<clocale>

#define N 3


using namespace std;

 int main()

{

          setlocale(LC_CTYPE,"Spanish");

          struct {
```

```
        string nombre;
        string cedula;
        }personas[N];

        int opcion, i;
        printf("Por favor ingrese los datos a consultar\n\n");
   for(i=0;i<N;i++){
        cout << "Por favor ingrese el nombre [" << i << "]: ";
        getline (cin,personas[i].nombre) ;
        cout << "Por favor ingrese la cedula [" << i << "]: " ;
        getline (cin, personas[i].cedula) ;
        cout << endl;
        }
     printf("Ingrese el numero del registro realizado:\n ");
    scanf("%d",&opcion);
        cout << endl << endl;

        if(opcion >= 0 && opcion < N){
             cout << "Los datos consultados son:";
        cout << endl << endl;
        cout << "Nombre:" << personas[opcion].nombre << endl;
        cout << "Cédula:" << personas[opcion].cedula << endl;
        }else{
        cout << endl << endl;
        cout << "Indice erroneo." << endl;
                        }
        return 0;
}
```

EJERCICIO 5 TALLER VECTORES:

Almacenar n números en un vector, imprimir cuantos son ceros, cuantos son negativos, cuantos positivos. Imprimir además la suma de los negativos y la suma de los positivos.

```
#include <stdio.h>
int main(){
        int i, datos;
        int suma_neutro=0;
        int suma_negativos=0;
        int suma_positivos=0;
        int acumulador_negativos=0;
        int acumulador_positivos=0;
        int vector[100];

        printf("ingrese el numero de datos del vector\n");
        scanf("%d", &datos);

        for (i=0;i<datos;i++){
                        printf("Digite los valores posicion %d\t",i);
                        scanf("%d",&vector[i]);
        }
        printf("  \n ");

   printf ("LOS NUMEROS NEUTROS SON:\n");
   for (i=0;i<datos;i++){
        if (vector[i]==0){
                printf("%d\n",vector[i]);
                suma_neutro=suma_neutro+1;
                }
}
        printf("EL TOTAL DE LOS NUMEROS NEUTROS SON:%d \n",suma_neutro);

        printf("  \n ");

        printf ("LOS NUMEROS POSITIVOS SON:\n");
   for (i=0;i<datos;i++){
        if (vector[i]>0){
                printf("%d\n",vector[i]);
                        suma_positivos=suma_positivos+1;
                        acumulador_positivos=acumulador_positivos+vector[i];
                        }
}
        printf("EL TOTAL DE LOS NUMEROS POSITIVOS SON:%d \n",suma_positivos);
        printf("EL TOTAL DE LA SUMA DE LOS POSITIVOS SON:%d \n",acumulador_positivos);

        printf("  \n ");

        printf ("LOS NUMEROS NEGATIVOS SON:\n");
   for (i=0;i<datos;i++){
        if (vector[i]<0){
                printf("%d\n",vector[i]);
                        suma_negativos=suma_negativos+1;
                        acumulador_negativos=acumulador_negativos+vector[i];
                        }
```

```
}
        printf("EL TOTAL DE LOS NUMEROS NEGATIVOS SON:%d \n",suma_negativos);
        printf("EL TOTAL DE LA SUMA DE LOS NEGATIVOS SON:%d \n",acumulador_negativos);

return 0;

}
```

9.2. MATRICES - BIDIMENSIONAL o MULTIDIMENSIONAL

Una matriz es un array multidimensional. Se definen igual que los vectores excepto que se requiere un índice por cada dimensión.

Su sintaxis es la siguiente:

tipo nombre [tamaño 1][tamaño 2]...;

Una matriz bidimensional se podría representar gráficamente como una tabla con filas y columnas.

*La matriz tridimensional se utiliza, por ejemplo, para trabajos gráficos con objetos **3D**.*

En el ejemplo puedes ver cómo se rellena y visualiza una matriz bidimensional. Se necesitan dos bucles para cada una de las operaciones. Un bucle controla las filas y otro las columnas.

```
/* Matriz bidimensional. */

#include <stdio.h>

main() /* Rellenamos una matriz */
{
        int x,i,numeros[3][4];
        /* rellenamos la matriz */
        for (x=0;x<3;x++)
                for (i=0;i<4;i++)

        scanf("%d",&numeros[x][i]);
        /* visualizamos la matriz */
        for (x=0;x<3;x++)
                for (i=0;i<4;i++)

        printf("%d",numeros[x][i]);
}
```

Si al declarar una matriz también queremos inicializarla, habrá que tener en cuenta el orden en el que los valores son asignados a los elementos de la matriz. Veamos algunos ejemplos:

int numeros[3][4]={1,2,3,4,5,6,7,8,9,10,11,12};

quedarían asignados de la siguiente manera:

numeros[0][0]=1 numeros[0][1]=2 numeros[0][2]=3 numeros[0][3]=4
numeros[1][0]=5 numeros[1][1]=6 numeros[1][2]=7 numeros[1][3]=8
numeros[2][0]=9 numeros[2][1]=10 numeros[2][2]=11 numeros[2][3]=12

También se pueden inicializar cadenas de texto:

char dias[7][10]={"lunes","martes","miércoles","jueves","viernes","sábado","domingo"};

Para referirnos a cada palabra bastaría con el primer índice:

printf("%s",dias[i]);

EJERCICIOS

1. Que muestre los primeros 100 números de izquierda a derecha usando un array de dos dimensiones.

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
        int x,y, numeros[10][10];
    for (x=0;x<10;x++)
  {
        for (y=0;y<10;y++)
        {
        numeros[x][y]=(x*10)+1+y;
    }
        }
        for (x=0;x<10;x++)
  {
        for (y=0;y<10;y++)
        {
        printf("%d ",numeros[x][y]);
    }
        printf("\n");
        }
        system("PAUSE");
  return 0;
}
```

2. Que muestre los primeros 100 números de izquierda a derecha usando un array de dos dimensiones, la última fila a mostrará la suma de sus respectivas columnas.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
        int x,y,sum, numeros[11][10];
    for (y=0;y<10;y++)
        {
        sum=0;
        for (x=0;x<10;x++)
     {
        numeros[x][y]=(x*10)+1+y;
        sum=sum+numeros[x][y];
        }
        numeros[10][y]=sum;
  }
        for (x=0;x<11;x++)
  {
        for (y=0;y<10;y++)
        {
        printf("%d ",numeros[x][y]);
     }
         printf("\n");
        }
        system("PAUSE");
  return 0;
}
```

3. Que rellene un array de dos dimensiones con números pares, lo pinte y después que pida una posición X,Y y mostrar el número correspondiente.

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
        int x,y,num=2, numeros[3][3];
    for (x=0;x<3;x++)
  {
        for (y=0;y<3;y++)
        {
        numeros[x][y]=num;
        num=num*2;
        }
        }
        printf("Introduzca coordenada x: ");
  scanf("%d",&x);
  printf("Introduzca coordenada y: ");
        scanf("%d",&y);
        printf("El número es: %d\n",numeros[x][y]);
    system("PAUSE");
        return 0;
}
```

4. Que rellene una matriz de 3x3 y muestre su traspuesta (la traspuesta se consigue intercambiando filas por columnas y viceversa).

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
        int x,y,num=0, numeros[4][4];
    for (x=0;x<3;x++)
   {
        for (y=0;y<3;y++)
        {
        numeros[x][y]=num;
        num++;
        }
        }
        printf("El array original es: \n\n\n");

   for(x = 0;x < 3;x++)
   {
        for(y = 0;y < 3;y++)
        {
        printf(" %d       ", numeros[x][y]);
    }
        printf("\n\n\n");
   }
        printf("La traspuesta es: \n\n\n");
    for(x = 0;x < 3;x++)
   {
        for(y = 0;y < 3;y++)
        {
        printf(" %d       ", numeros[y][x]);
    }
        printf("\n\n\n");
        }
        system("PAUSE");
        return 0;
}
```

5. • Hacer un programa para rellenar una matriz poniendo el usuario el número de filas y columnas, posteriormente mostrar la matriz en pantalla.

```
#include<stdio.h>

#include<clocale>

int main(){

        setlocale(LC_CTYPE,"Spanish");

        int matriz[100][100],filas,columnas;

        int i,j;
```

```
printf("por favor digite el número de filas: \n");
scanf ("%i",&filas);
printf ("por favor digite el número de columnas: \n");
scanf ("%i",&columnas);
for (i=0;i<filas;i++){
        for (j=0;j<columnas;j++){
            printf ("digite un número [%d,%d]: ",i,j);
            scanf ("%i",&matriz[i][j]);
        }
}
printf ("\n");
for (i=0;i<filas;i++){
        for (j=0;j<columnas;j++)
            printf ("%i\t",matriz[i][j]);
            printf ("\n");
        }
}
```

6. programa para rellenar una matriz 4x4 o el tamaño que deseemos, posteriormente mostrar la matriz en pantalla.

```
#include<stdio.h>
#include<clocale>
int main(){
    setlocale(LC_CTYPE,"Spanish");
    int matriz[4][4];
    int i,j;

    for (i=0;i<4;i++){
```

```
                        for (j=0;j<4;j++){
                                printf ("digite un número [%d,%d]: ",i,j);
                                scanf ("%i",&matriz[i][j]);
                        }
        }
        printf ("\n");
        for (i=0;i<4;i++){
                        for (j=0;j<4;j++)
                                printf ("%i\t",matriz[i][j]);
                                printf ("\n");
                        }
return 0;
}
```

7. sumar los extremos de una matriz el total de cada fila y cada columna.

```
#include <stdio.h>
#include <conio.h>
#define TAM 3
 int main()
{
 int matriz[TAM+1][TAM+1],i,j;

  // INICIALIZAR MATRIZ
 for(i=0;i<TAM+1;i++)
  for(j=0;j<TAM+1;j++)
            matriz[i][j] = 0;

 // INGRESAR LOS DATOS
```

```
for(i=0;i<TAM;i++)
{
 for(j=0;j<TAM;j++)
 {
  printf("Ingresar dato entero en ( fila [%d] columna [%d] ):",i,j);
  scanf("%d",&matriz[i][j]);
 }
 printf("\n");
}

// MOSTRAR LOS DATOS QUE SE INGRESE EN LA MATRIZ
for(i=0;i<TAM;i++)
{
 for(j=0;j<TAM;j++)
  printf("%d\t",matriz[i][j]);

 printf("\n");
}

// ACUMULAR EN LA ULTIMA POSICION DE LA MATRIZ
for(i=0;i<TAM;i++)
{
 for(j=0;j<TAM;j++)
 {
  matriz[i][TAM]+= matriz[i][j];
  matriz[TAM][j]+= matriz[i][j];
 }
}
```

```
 printf("\n");


 // MOSTRAR LAS SUMAS
 for(i=0;i<TAM+1;i++)
        {
  for(j=0;j<TAM+1;j++)
   printf("%d\t",matriz[i][j]);
  printf("\n");
        }
        getch();
}
```

8. Obtener el valor de una casilla específica

```
#include<stdio.h>
 int main(){
int tabla[3][4]= {{0,3,4,10},{3,-1,10,-10},{10,100,3,1}};
 for (int i=0;i<3;i++){
        for (int j=0;j<4;j++){
                printf("%i",tabla[i][j]);
                }
        printf("\n");
}
printf("fila1casilla1: %i \n",tabla [1][1]);
return 0;
}
```

*9. · Hacer un algoritmo que almacene números en una matriz de 5 * 6. Imprimir la suma de los números almacenados en la matriz.*

```
#include <stdio.h>
int main(){
```

```
    int filas,columnas;
    int suma=0;
    int matriz[5][6];
            //ingresar datos
    for (filas=0;filas<5;filas++){
            for(columnas=0;columnas<6;columnas++){
                    printf("Ingrese el valor para la celda [%d,%d]: ",filas,columnas);
                    scanf("%d",&matriz[filas][columnas]);
    }
            }
                    printf("\n");
  //imprimir matriz
            for (filas=0;filas<5;filas++){
                    for(columnas=0;columnas<6;columnas++)
               printf(" %d \t ",matriz[filas][columnas]);
                     printf("\n");
            }
                    printf ("\n\n\n");
    //operacion  suma
            for (filas=0;filas<5;filas++)
            {
                    for(columnas=0;columnas<6;columnas++)
                            suma+=matriz[filas][columnas];
                    }
                    printf("la suma de la matriz es:%d\n",suma);
return 0;
}
```

10. FUNCIONES

10.1.- Tiempo de vida de los datos

Según el lugar donde son declaradas puede haber dos tipos de variables.

Globales: *las variables permanecen activas durante todo el programa. Se crean al iniciarse éste y se destruyen de la memoria al finalizar. Pueden ser utilizadas en cualquier función.*

Locales: *las variables son creadas cuando el programa llega a la función en la que están definidas. Al finalizar la función desaparecen de la memoria.*

Si dos variables, una global y una local, tienen el mismo nombre, la local prevalecerá sobre la global dentro de la función en que ha sido declarada.

Dos variables locales pueden tener el mismo nombre siempre que estén declaradas en funciones diferentes.

```
/* Variables globales y locales. */

#include <stdio.h>

int num1=1;
main() /* Escribe dos cifras */
{
        int num2=10;
        printf("%d\n",num1);
        printf("%d\n",num2);
}
```

10.2.- Funciones

Las funciones son bloques de código utilizados para dividir un programa en partes más pequeñas, cada una de las cuáles tendrá una tarea determinada.

Su sintaxis es:

tipo_función nombre_función (tipo y nombre de argumentos)
{
bloque de sentencias
}

tipo_función*: puede ser de cualquier tipo de los que conocemos. El valor devuelto por la función será de este tipo. Por defecto, es decir, si no indicamos el tipo, la función devolverá un valor de tipo entero (* ***int*** *). Si no queremos que retorne ningún valor deberemos indicar el tipo vacío (* ***void*** *).*

nombre_función*: es el nombre que le daremos a la función.*

tipo y nombre de argumentos: son los parámetros que recibe la función. Los argumentos de una función no son más que variables locales que reciben un valor. Este valor se lo enviamos al hacer la llamada a la función. Pueden existir funciones que no reciban argumentos.

bloque de sentencias: es el conjunto de sentencias que serán ejecutadas cuando se realice la llamada a la función.

Las funciones pueden ser llamadas desde la función ***main*** *o desde otras funciones. Nunca se debe llamar a la función* ***main*** *desde otro lugar del programa. Por último recalcar que los argumentos de la función y sus variables locales se destruirán al finalizar la ejecución de la misma.*

10.3.- Declaración de las funciones

Al igual que las variables, las funciones también han de ser declaradas. Esto es lo que se conoce como prototipo de una función. Para que un programa en C sea compatible entre distintos compiladores es imprescindible escribir los prototipos de las funciones.

Los prototipos de las funciones pueden escribirse antes de la función main o bién en otro fichero. En este último caso se lo indicaremos al compilador mediante la directiva #include.

En el ejemplo adjunto podremos ver la declaración de una función (prototipo). Al no recibir ni retornar ningún valor, está declarada como void en ambos lados. También vemos que existe una variable global llamada num. Esta variable es reconocible en todas las funciones del programa. Ya en la función main encontramos una variable local llamada num. Al ser una variable local, ésta tendrá preferencia sobre la global. Por tanto la función escribirá los números 10 y 5.

```
/* Declaración de funciones. */

#include <stdio.h>

void funcion(void); /* prototipo */
int num=5; /* variable global */
```

```
main() /* Escribe dos números */
{
        int num=10; /* variable local */
        printf("%d\n",num);
        funcion(); /* llamada */
}

void funcion(void)
{
        printf("%d\n",num);
}
```

10.4.- Paso de parámetros a una función

Como ya hemos visto, las funciones pueden retornar un valor. Esto se hace mediante la instrucción ***return****, que finaliza la ejecución de la función, devolviendo o no un valor.*

En una misma función podemos tener más de una instrucción return. La forma de retornar un valor es la siguiente:

return (valor o expresión);

El valor devuelto por la función debe asignarse a una variable. De lo contrario, el valor se perderá.

En el ejemplo puedes ver lo que ocurre si no guardamos el valor en una variable. Fíjate que a la hora de mostrar el resultado de la suma, en el ***printf****, también podemos llamar a la función.*

```
/* Paso de parámetros. */

#include <stdio.h>

int suma(int,int); /* prototipo */
main() /* Realiza una suma */
{
        int a=10,b=25,t;
        t=suma(a,b); /* guardamos el valor */
        printf("%d=%d",suma(a,b),t);
        suma(a,b); /* el valor se pierde */
}

int suma(int a,int b)
{
        return (a+b);
}
```

Ahora veremos lo que se conoce como paso de parámetros.

Existen dos formas de enviar parámetros a una función:

Por valor: *cualquier cambio que se realice dentro de la función en el argumento enviado,* ***NO*** *afectará al valor original de las variables utilizadas en la llamada. Es como si trabajaramos con una copia, no con el original. No es posible enviar por valor* ***arrays****, deberemos hacerlo por referencia.*

Por referencia: *lo que hacemos es enviar a la función la dirección de memoria donde se encuentra la variable o dato. Cualquier modificación* ***SI*** *afectará a las variables utilizadas en la llamada. Trabajamos directamente con el original.*

```
/* Paso por valor. */

#include <stdio.h>

void intercambio(int,int);
main() /* Intercambio de valores */
{
        int a=1,b=2;
        printf("a=%d y b=%d",a,b);
        intercambio(a,b); /* llamada */
        printf("a=%d y b=%d",a,b);
}

void intercambio (int x,int y)
{
        int aux;
        aux=x;
        x=y;
        y=aux;
        printf("a=%d y b=%d",x,y);
}
```

Para enviar un valor por referencia se utiliza el símbolo ***&*** *(ampersand) delante de la variable enviada. Esto le indica al compilador que la función que se ejecutará tendra que obtener la dirección de memoria en que se encuentra la variable.*

Vamos a fijarnos en los ejemplos. En el ejemplo anterior podrás comprobar que antes y después de la llamada, las variables mantienen su valor. Solamente se modifica en la función ***intercambio*** *(paso por valor).*

En el siguiente ejemplo podrás ver cómo las variables intercambian su valor tras la llamada de la función (paso por referencia).

Las variables con un ******* *son conocidas como* ***punteros****, el único dato en 'C' que puede almacenar una dirección de memoria.*

```
/* Paso por referencia. */

#include <stdio.h>

void intercambio(int *,int *);
main() /* Intercambio de valores */
{
        int a=1,b=2;
        printf("a=%d y b=%d",a,b);
        intercambio(&a,&b); /* llamada */
        printf("a=%d y b=%d",a,b);
}

void intercambio (int *x,int *y)
{
        int aux;
        aux=*x;
        *x=*y;
        *y=aux;
        printf("a=%d y b=%d",*x,*y);
}
```

.- Los argumentos de la función main

Ya hemos visto que las funciones pueden recibir argumentos. Pues bién, la función ***main*** *no podía ser menos y también puede recibir argumentos, en este caso desde el exterior.*

Los argumentos que puede recibir son:

argc: *es un contador. Su valor es igual al número de argumentos escritos en la línea de comandos, contando el nombre del programa que es el primer argumento.*

argv: *es un puntero a un array de cadenas de caracteres que contiene los argumentos, uno por cadena.*

En este ejemplo vamos a ver un pequeño programa que escribirá un saludo por pantalla. El programa ***FUNCION6.EXE****.*

```
/* Argumentos de la main. */
```

```
#include <stdio.h>

main(int argc,char *argv[]) /* argumentos */
{
        printf("\nCurso de Programación en C - Copyright (c) 1997-2001, Sergio Pacho\n");
        printf("Programa de ejemplo.\n\n");
        if (argc<2)
        {
                printf("Teclee: funcion6 su_nombre");
                exit(1); /* fin */
        }
        printf("Hola %s",argv[1]);
}
```

EJERCICIOS

1.

```
#include <stdio.h>
int arreglo[10] = {3,10,1,8,15,5,12,6,5,4}; /*Declaracion e inicialización
del arreglo. */


/*imprimearreglo - Funcion que muestra por pantalla
el contenido de un arreglo.*/
void imprimearreglo() {
   int i; for (i=0;i<10;i++)
      printf("Elemento %d: %d \n",i,arreglo[i]);
}

int main() /*Funcion Principal del Programa*/
{
   int i,j,k;
   imprimearreglo();
   for (i=1; i<10; i++) {
      j=i;
         while (j>=0 && arreglo[j]<arreglo[j-1]) {
            k=arreglo[j];
            arreglo[j]=arreglo[j-1];
            arreglo[j-1]=k;
            j--;
         }
   }
   printf("\n\nArreglo ordenado \n\n");
   imprimearreglo();
}
```

2. programa pide al usuario ingresar las notas de uno o más alumnos, y va mostrando los promedios de cada uno de ellos:

```
#include <stdio.h>
float promedio(int valores[], int cantidad) {
```

```
    int i;
    float suma = 0.0;

    for (i = 0; i < cantidad; ++i)
        suma += valores[i];

    return suma / (float) cantidad;
}

int main() {

    int notas[10];
    char nombre[20];
    char opcion[3];
    int n, i;

    do {
        printf("Ingrese nombre del alumno: ");
        scanf("%s", nombre);

        printf("Cuantas notas tiene %s? ", nombre);
        scanf("%d", &n);

        for (i = 0; i < n; ++i) {
            printf("  Nota %d: ", i + 1);
            scanf("%d", &notas[i]);
        }

        printf("El promedio de %s es %.1f\n", nombre, promedio(notas, n));

        printf("Desea calcular mas promedios (si/no)? ");
        scanf("%s", opcion);

    } while (opcion[0] == 's' || opcion[0] == 'S');

    return 0;
}
```

EJERCICIO FUNCION VECTOR

Ingresar 10 números de tipo entero y los almacena en un arreglo; después le pide que introduzca un número para que busque su posición dentro del arreglo..

El programa utiliza una función llamada BuscaNumero que recibe como parámetros el arreglo con los 10 números capturados, el número de elementos del arreglo (en este caso 10) y el número del cual se desea saber su posición dentro del arreglo.. La función regresa -1 si el número que se busca no se encuentra en el arreglo y en caso contrario, regresa la primera posición del arreglo que contiene dicho número.

El programa también utiliza una función llamada MuestraArreglo que no regresa valor alguno, sólo recibe como parámetros el arreglo y el número de elementos. Esta función imprime en pantalla los elementos del arreglo separados por un tabulador.

```
int main()
{
 int x=0, numero=0, posicion=0;
 int ar_numeros[10] = {0};

 printf("Introduzca los 10 numeros enteros que se almacenaran en el arreglo\n");
 for (x=0; x<10; ++x)
    {
     printf("Valor para el elemento [%d]: ", x);
     scanf("%d",&ar_numeros[x]);
    }
 printf("\n");

 printf("Introduzca el número que desea buscar en el arreglo\n");
 scanf("%d",&numero);
 printf("\n");
 MuestraArreglo(ar_numeros,10);

 posicion=BuscaNumero(ar_numeros,10,numero);
 if (posicion != -1)
    printf("El número %d está en la posición %d del arreglo\n",numero, posicion);
 else
    printf("El número %d no está en el arreglo\n",numero);

 return 0;
}
```

EJERCICIO FUNCION MATRIZ

Para pasar una función un arreglo de dos dimensiones, debemos indicar el tamaño de la segunda dimensión del arreglo

Ejemplo:int MiFuncion(int mi_arreglo[][5], int num_elementos);

El siguiente programa le pide al usuario que introduzca 9 números y los almacena en un arreglo de dos dimensiones, en este caso una matriz de 3×3; posteriormente utiliza una función llamada ImprimeMatriz para mostrar como quedaron almacenados los números en la matriz. Dicha función recibe como parámetros, la matriz de 3×3 y el tamaño de la primera dimensión (normalmente la primera dimensión son filas y la segunda dimensión columnas).

```
#include <stdio.h>

 void ImprimeMatriz(int m[][3], int filas)

{

 int i=0,j=0;

  for (i=0; i<filas; ++i)    {

     for (j=0; j<3; ++j)
```

```
        {
          printf("%d ",m[i][j]);
        }
     printf("\n");
    }
}
 int main()
{
 int x=0,y=0;
 int matriz[3][3] = {{0},{0},{0}};
  printf("Introduzca los valores para la matriz\n");
 for (x=0; x<3; ++x)     {
     for (y=0; y<3; ++y)  {
          printf("Valor para el elemento [%d][%d]: ", x, y);
         scanf("%d",&matriz[x][y]);
          }
     printf("\n");
    }
  printf("Matriz\n");
 ImprimeMatriz(matriz, 3);
  return 0;
}
```

Función Vector Ejercicio 5 taller

2. *Almacenar 300 números en un vector, imprimir cuantos son ceros, cuantos son negativos, cuantos positivos. Imprimir además la suma de los negativos y la suma de los positivos.*

```
Include <stdio.h>
#include <clocale>
void postivos(int vector[], int datos )
{
 int  i;
```

```
 int suma_positivos=0;
 int acumulador_positivos=0;
        printf ("\n LOS NÚMEROS POSITIVOS SON: \n");
        for (i=0;i<datos;i++){
                if (vector[i]>0){

                        printf ("%i\n",vector[i]);
                        suma_positivos=suma_positivos+1;
                        acumulador_positivos=acumulador_positivos+vector[i];
                }
        }

        printf ("\n el total de los números positivos son:\n %i ",suma_positivos);
        printf ("\n el total de la suma de los positivos es: \n %i",acumulador_positivos);
        printf ("\n");
}

int main(){
setlocale(LC_CTYPE,"Spanish");
        int i, datos;
        int suma_neutro=0;
        int suma_negativos=0;
        int acumulador_negativos=0;
        int vector [datos];
        printf ("por favor ingrese la cantidad de datos que tendrá el vector: \n");
        scanf ("%i",&datos);
                for (i=0;i<datos;i++){
                printf ("\n digite el número para la posición %i\t",i);
                scanf ("%i",&vector[i]);
        }
        printf ("\n LOS NÚMEROS NEUTROS SON: \n");
        for (i=0;i<datos;i++){
                if (vector[i]==0){
                        printf ("%i\n",vector[i]);
                        suma_neutro=suma_neutro+1;
                }
        }
        printf ("\n el total de los números neutros son:\n %i ",suma_neutro);
        printf ("\n");

        postivos (vector,datos);
                printf ("\n LOS NÚMEROS NEGATIVOS SON: \n");

        for (i=0;i<datos;i++){
                if (vector[i]<0){
```

```
                printf ("%i\n",vector[i]);
                suma_negativos=suma_negativos+1;
                acumulador_negativos=acumulador_negativos+vector[i];
            }
        }
        printf ("\n el total de los números negativos son:\n %i ",suma_negativos);
        printf ("\n el total de la suma de los negativos es: \n %i",acumulador_negativos);
        printf ("\n");
        return 0;
}
```

11. PUNTEROS

Un puntero es una variable que contiene la dirección de memoria de otra variable. Se utilizan para pasar información entre una función y sus puntos de llamada.

11.1.- Declaración

Su sintaxis es la siguiente:

tipo *nombre;

Donde nombre es, naturalmente, el nombre de la variable, y tipo es el tipo del elemento cuya dirección almacena el puntero.

11.2.- Operadores

Existen dos operadores especiales para trabajar con punteros: ***&*** *y* *******.*

El primero devuelve la dirección de memoria de su operando. Por ejemplo, si queremos guardar en el puntero ***x*** *la dirección de memoria de la variable* ***num****, deberemos hacer lo siguiente:*

x=#

El segundo devuelve el valor de la variable cuya dirección es contenida por el puntero. Este ejemplo sitúa el contenido de la variable apuntada por ***x****, es decir* ***num****, en la variable* ***a****:*

a=*x;

11.3.- Asignación

Los punteros se asignan igual que el resto de las variables. El programa ejemplo mostrará las direcciones contenidas en ***p1*** *y* ***p2****, que será la misma en ambos punteros.*

```
/* Asignaciones de punteros. */
#include <stdio.h>
main() /* Asignamos direcciones */
{
        int a;
        int *p1,*p2;
        p1=&a;
        p2=p1;
        printf("%p %p",p1,p2);
}
```

11.4.- Aritmética de direcciones

Es posible desplazar un puntero recorriendo posiciones de memoria. Para ello podemos usar los operadores de suma, resta, incremento y decremento (+, -, ++, - -). Si tenemos un puntero (p1) de tipo ***int*** *(2 bytes), apuntando a la posición 30000 y hacemos:* ***p1=p1+5;*** *el puntero almacenará la posición 30010, porque apunta 5 enteros por encima (10 bytes más).*

Ejercicios

1. *Escribir un programa que calcule y visualice la media aritmética de un vector de 10 elementos numéricos, utilizando una variable puntero que apunte a dicho vector.*

```
# include <stdio.h>
int main ()
{
float med, suma=0;
int tabla[10], j;
int *pt=tabla;
for(j=0;j<10;j++)
{
printf("\nIntroducir el elemento %d: ",j+1);
scanf("%d",pt+j);
}
for (j=0;j<10;j++)
suma+=*(pt+j);
med=suma/10;
printf("\n\n")
printf("La media vale:%f",med);
}
```

2. ***Escribir un programa que ponga en mayúsculas el primer carácter de una cadena de caracteres y todo aquel carácter que siga a un punto, utilizando un puntero a dicha cadena.***

```
# include <stdio.h>
# include <ctype.h>
void main (void)
{
char texto[80];
char *ptext=texto;
printf("\nIntroduccir un texto menor de 80 caracteres:\n");
gets(texto);
ptext=toupper(*ptext);
while(*ptext!='\0')
{
ptext++;
if(*ptext=='.')
*(ptext+2)=toupper(*(ptext+2));
}
puts(texto);
}
```

3. ***Escribir un programa que mediante el empleo de un menú realice operaciones aritméticas, llamando a las funciones correspondiente. Desarrollar este programa utilizando una tabla de punteros a funciones.***

```
# include <stdio.h>
# include <math.h>
int main (void)
{
void suma(float x, float y);
void resta(float x, float y);
void prod(float x, float y);
void divi(float x, float y);
```

```
void(*p[4])(float x, float y)={suma, resta, prod, divi};
float num1, num2;
int opcion;
printf ("\nIntroducir dos números \n");
scanf("%f%f",&num1,&num2);
printf("1.-Suma\n");
printf("2.-Resta\n");
printf("3.-Producto\n");
printf("4.-División\n");
do
{
printf("Seleccionar una opción\n");
scanf("%d",&opcion);
}
while(opcion<1 || opcion >4);
(*p[opcion-1])(num1,num2);
}
void suma(float x, float y)
{
printf("La suma vale:%f\n",x+y);
}
void resta(float x, float y)
{
printf("La resta vale:%f\n",x-y);
}
void prod(float x, float y)
{
printf("El producto vale:%f\n",x*y);
}
void divi(float x, float y)
{
if(y==0)
printf("La división no es posible. Divisor=0\n");
else
printf("La división vale:%f\n",x/y);
}
```

4. ***Escribir un programa que llame a tres funciones de nombre 'Pepe', 'Ana' y 'Maria' mediante el uso de una tabla de punteros a funciones.***

```
# include <stdio.h>
int Pepe(), Ana(), Maria();
int (*mf[])()={Pepe, Ana, Maria};
void main (void)
{
int j;
for (j=0;j<3;j++)
(*nf[j])();
/*Las dos lineas anteriores pueden simplificarse en:
for (j=0;j<3;(*nf[j++])());*/
}
Pepe()
{
printf("soy Pepe\n"):
```

```
}
Ana ()
{
printf("soy Ana\n");
}
Maria()
{
printf("soy Maria\n");
}
```

5. ***Programa que define una tabla de proveedores teniendo asignados cada proveedor un nombre, cantidad vendida del artículo, precio unitario (introducidos por teclado) e importe (calculado a partir de los datos anteriores). Se pretende visualizar los datos de cada proveedor, el importe total de compra, así como el nombre del proveedor más barato y el del más caro.***

```
# include <stdio.h>
# define NUM 3
# define DIM 16
void main (void)
{
struct proveedor
{
char prov[DIM];
int cant_p;
float precio;
float importe;
};
struct proveedor provee[NUM];
int cont, k=0, j=0;
float preciomin, preciomax;
float total=0.0;
/* Carga de la tabla de datos de proveedores*/
for(cont=0;cont<NUM:cont++)
{
 printf("\nIntroducir nombre del proveedor %d:",
cont+1);
gets(provee[cont].prov);
printf("Introducir cantidad de piezas: ");
scanf("%d", &provee[cont].cant_p);
printf("Introducir precio unitario:";
scanf("%d",&provee[cont].precio);
provee[cont].importe=provee[cont].cant_p*provee[cont]
.precio;
while(getchar()!='\n');
}
/*Visualizar datos de proveedores*/
for(cont=0;cont<NUM;cont++)
printf("\n%s %s %s %s", provee[cont].prov,
provee[cont.cant_p, provee[cont].precio,
provee[cont].importe);
total += provee[cont].importe;
printf("\nEl importe total es:%f",total);
/*Calcular el proveedor más barato y el más caro*/
preciomin=provee[0].precio;
preciomax=provee[0].precio;
```

```
for(cont=1;cont<NUM;cont++)
{
if(preciomin>provee[cont].precio)
{
preciomin=provee[cont].precio;
k=cont;
}
if(preciomax<provee[cont].precio
{
preciomax<provee[cont].precio;
j=cont;
}
}
printf("\n El proveedor mas barato es: %s",
provee[k].prov);
printf("\n El proveedor más caro es: %s", provee[j].prov);
}
```

6. *Programa que define una tabla de proveedores empleando una estructura que anida los datos del proveedor (nombre, dirección , y teléfono), cantidad vendida, precio unitario e importe (calculado). Los datos no calculados se introducen por teclado. Se pretende visualizar en pantalla los datos de cada proveedor, el importe total de las compras y el nombre y teléfono del proveedor más barato.*

```
# include <stdio.h>
# define NUM 3
struct dat_p
{
char nom_p[16];
char dir_p[30];
char telef_p[10];
};
struct entrada
{
struct dat_p datos;
int cant;
float precio;
float importe;
};
void main (void)
{
struct entrada sumin[NUM];
int cont, j=0;
float total=0.0;
float preciomin;
printf("\nIntroducir datos de proveedores\n");
for(cont=0;cont<NUM;cont++)
{
printf("Introducir proveedor num:%d",cont+1);
printf("\nNombre:");
gets(sumin[cont].datos.nom_p);
printf("\nDirección:");
gets(sumin[cont].datos.dir_p);
printf("\nTelefono:"");
gets(sumin[cont].datos.telef_p);
```

```
}
printf("\n*****************************************");
printf("\nIntroducir cantidades y precios de cada
proveedor");
for(cont=0;cont<NUM;cont++);
{
printf("\nProveedor: %s",sumin[cont]datos.nom_p);
printf("\nCantidad suministrada:");
scanf("%d",&sumin[cont].cant);
printf("Precio unitario:");
scanf("%f",&sumin[cont].precio);
while(getchar()!='\n');
sumin[cont].importe=sumin[cont].cant*
*sumin[cont].precio;
}
printf("%s %s %s %s\n","Proveedor","Cant","Precio",
"Importe");
for(cont=0;cont<NUM;cont++)
{
printf ("%s %d %f %f\n",sumin[cont].datos.nom_p,
sumin[cont].cant, sumin[cont].precio, sumin[cont].importe);
total += sumin[cont].importe;
}
printf("\nEl importe total es:%f", total);
preciomin=sumin[0].precio;
for(cont=0;cont<NUM;cont++)
{
if (preciomin>sumin[cont].precio)
{
preciomin=sumin[cont].precio;
j=cont;
}
}
printf("\nEL proveedor mas barato es: %s,
sumin[j].datos.nom_p);
printf("\nY su teléfono es: %s", sumin[j].datos.telef_p);
}
```

12. MANEJO DE CADENA DE CARACTERES

Cadenas

Una cadena o cadena de caracteres nos es más que una serie de caracteres manipulados como una unidad. Si asemejamos una cadena al lenguaje castellano sería como una palabra, que es un conjunto de sílabas y vocales en donde cada una de estas viene a ser un carácter.

Visto desde otro punto vendría a ser un arreglo de caracteres.

Una cadena puede contener cualquier carácter, puede almacenar un nombre propio una dirección, es decir, lo que nosotros precisemos.

Declaración
*Una cadena se la **define** de la siguiente manera*

char cadena[20];

La cadena anterior puede contener un máximo de 20 caracteres.

Inicialización
Se puede inicializar una cadena de la siguiente manera:

cadena = "Hola" ;

Cualquier valor que se le asigne a una cadena va entre comillas dobles " ", como en el ejemplo anterior "Hola" está entre comillas dobles.

*Una cadena siempre finaliza con el carácter de fin de cadena **'\0'**, que siempre se añade al final automáticamente, en el ejemplo anterior se añade al final de "Hola" el carácter de fin de cadena.*

También podemos considerar a una cadena como un arreglo de caracteres, y se puede inicializar de la siguiente manera:

cadena = { 'H', 'o', 'l', 'a' } ;

El arreglo de caracteres se vería de esta forma:

'H' 'o' 'l' 'a' '\0'

0 1 2 3 4

Nótese que en la posición 4 se aumenta el fin de cadena

Ejemplo
Se desea tener un programa que sea amable con el usuario, el programa deberá conocer el nombre del usuario y responderle con un mensaje amigable.

```
#include <string.h>

#include <stdio.h>

#include <clocale>

int main()

{

        setlocale(LC_CTYPE,"spanish");
```

```
        char nombre[30];

        printf("¿Cuál es tu nombre?\n");

        scanf("%s",nombre);

        printf("Que tengas un buen día %s",nombre);

}
```

En el ejemplo anterior el mensaje "¿Cuál es tu nombre?" es una cadena pues está entre comillas. También es una cadena la variable nombre que recibirá un valor desde teclado.

Operaciones con Cadenas

Existen muchas operaciones que se pueden realizar utilizando cadenas, la mayoría de la operación que podemos requerir se encuentran ya a nuestra disposición dentro de la librería string.h

Longitud

La longitud de una cadena la podemos conocer utilizando la función strlen.

Sintaxis

```
strlen( cadena ) ;
```

Ejemplo

```
#include <string.h>
#include <stdio.h>
#include <clocale>
int main()
{
        setlocale(LC_CTYPE,"spanish");
   char nombre[30];
   int tamano;
   printf("¿Cuál es tu nombre?\n");
        scanf("%s",nombre);
        tamano = strlen(nombre );
   printf("Tu nombre tiene %d",tamano,"letras");
}
```

Comparación

Para saber si dos cadenas son exactamente iguales utilizamos la función strcmp.

Sintaxis

```
strcmp ( cadena1, cadena2 );
```

Esta función devuelve un valor de acuerdo al resultado de la comparación.

Devuelve:

0 *si la dos cadenas son exactamente iguales*

Mayor a 0 *si la cadena1 es mayor a la cadena2*

Menor a 0 *si la cadena1 es menor que la cadena2*

Ejemplo

```
#include <string.h>

#include <stdio.h>

#include <clocale>

int main()

{

        setlocale(LC_CTYPE,"spanish");

   char contrasena[30], reContrasena[30];

   int resultado;

   printf("Escribe tu contraseña\n");

   scanf("%s",contrasena);

   printf("Re escribe tu contraseña\n");

   scanf("%s",reContrasena);

   resultado = strcmp(contrasena, reContrasena);

   if ( resultado == 0 )

      printf("La contraseña es aceptada\n");

   else

      printf("La contraseña no coincide\n");

}
```

Copia

Podemos reflejar todo el contenido de una cadena a otra, en otras palabras la copiamos tal cual, para esto utilizamos la función strcpy.

Sintaxis

```
        strcpy( cadenaDestino, cadenaOrigen );
```

Todo el contenido de la cadenaOrigen se copia a la cadenaDestino, si esta última tuviera algún valor este se borra.

Ejemplo

```
#include <string.h>
#include <stdio.h>
#include <clocale>
int main()
{
        setlocale(LC_CTYPE,"spanish");
        char nombre[30], apellido[30];
        printf("¿Cuál es tu nombre? \n");
   scanf("%s",nombre);
        printf("¿Cuál es tu apellido paterno\n");
   scanf("%s",apellido);
   strcat(nombre, " "); //Se le añade un espacio en blanco
   strcat(nombre, apellido);
   printf("Tu nombre completo es:%s",nombre);
}
```

Concatenación

Podemos juntar o concatenar dos cadenas una a continuación de la otra. Utilizamos la función strcat.

Sintaxis

```
        strcat( cadenaDestino, cadenaOrigen );
```

Todo el contenido de la cadenaOrigen se añade a continuación de la cadenaDestino, si esta última contiene algo entonces al final contendrá lo que contenía más el contenido de la cadenaOrigen.

Ejemplo

```
#include <iostream>
#include <string.h>
#include <stdio.h>
#include <clocale>
int main()
{
        setlocale(LC_CTYPE,"spanish");
   char origen[30], copia[30];
   printf("¿Qué día es hoy? \n");
   scanf("%s",origen);
   strcpy(copia, origen);
   printf("Hoy es %s",copia);
}
```

/*Longitud de una cadena en C ++
*primer nivel */*

```
#include <stdio.h>
#include <string.h>
#include <clocale>

int main()
 {
  setlocale(LC_CTYPE,"spanish");
```

```
  char* cadena = "Hola, buen día. Estudiantes de la USCO";
  int contador = 0;
  // Recorrer la cadena hasta encontrar el carácter NUL o de terminación
  while (cadena[contador] != 0) {
    contador++;
  }
  printf("La longitud de '%s' es %d", cadena, contador);
  /*   Salida:
  La longitud de 'Hola, buen día. Estudiantes de la USCO' es 38  */
  return 0;
}
```

Una versión más simple

La versión presentada anteriormente es una versión explicativa. Podemos ahorrarnos el cuerpo del ciclo while simplemente preincrementando el valor de contador.

```
/*Longitud de una cadena en C ++
  primer nivel
*/

#include <stdio.h>
#include <string.h>
#include <clocale>

int main()
 {
  setlocale(LC_CTYPE,"spanish");
  char* cadena = "Hola, buen día. Estudiantes de la USCO";
  int contador = 0;
  while (cadena[++contador] != 0);
  printf("La longitud de '%s' es %d", cadena, contador);
          /*        Salida:
          La longitud de 'Hola, buen día. Estudiantes de la USCO' es 38 */
  return 0;
}
```

Lo que hacemos es usar ++contador, que incrementará la variable, regresará el valor después de incrementarla y a su vez servirá como índice del while.

Funciona de la misma manera para obtener la longitud de una cadena en C

CONTAR VOCALES UTILIZANDO FUNCIONES

```
#include <stdio.h>

#include <ctype.h>

// Definición
```

```
int contarVocales(char *cadena);

int main(int argc, char const *argv[])

{

	// Un arreglo de longitud de 1000

	// porque no podemos tener arreglos de longitud dinámica ni strings

	char entrada[1000];

	printf("Escribe una cadena:\n");

	gets(entrada);

	int vocales = contarVocales(entrada);

	printf("El numero de vocales que tiene la cadena es: %d\n", vocales);

	return 0;

}

//Cuerpo de la función

int contarVocales(char *cadena){

	int vocales = 0;

	// Recorrer toda la cadena

	for (int indice = 0; cadena[indice] != '\0'; ++indice){



		// Obtener el char de la posición en donde está el índice

		// y por otro lado convertirla a minúscula



		// Así no importa si ponen 'A' o 'a', ambas letras serán convertidas a 'a'
```

```
            char letraActual = tolower(cadena[indice]);

            if (
                letraActual == 'a' ||
                letraActual == 'e' ||
                letraActual == 'i' ||
                letraActual == 'o' ||
                letraActual == 'u'
                )
            {
                vocales++;
            }
        }
        return vocales;
}
#include <stdio.h> // Para printf
#include <ctype.h> // Para toupper y isalpha
#include <clocale>

// Función que indica si un carácter es vocal, ¿hace falta más explicación?
int esVocal(char letra) {
    // Convertir a mayúscula para evitar hacer más comparaciones
```

```
    char letraEnMayuscula = (char) toupper(letra);
    return letraEnMayuscula == 'A' ||
            letraEnMayuscula == 'E' ||
            letraEnMayuscula == 'I' ||
            letraEnMayuscula == 'O' ||
            letraEnMayuscula == 'U';
}
```

contar consonantes de una cadena

```
int contarConsonantes(char cadena[]) {
    int consonantes = 0;// Almacenar la cantidad de consonantes
    int i = 0;// El índice para recorrer la cadena
    while (cadena[i]) {
        // Si es del alfabeto pero no es vocal
        if (isalpha(cadena[i]) && !esVocal(cadena[i])) {
            consonantes++;
        }
        i++;
    }
    return consonantes;
}


int main() {
```

```
    setlocale(LC_CTYPE,"spanish");
    char cadena[] = "Hola chicos vamos a contar las consonantes en lenguaje C";
    int consonantes = contarConsonantes(cadena);
    printf("La cadena '%s' \ntiene %d consonantes", cadena, consonantes);
    return 0;
}
```

1. ***EJERCICIOS***

```
#include<stdio.h>
#include<string.h>
#include<locale.h>
 int main(){
   setlocale(LC_CTYPE, "SPANISH");
                    char operacion[15];
          float numero_1, numero_2, resultado;
          printf("Programa para realizar operaciones matemáticas con dos números\n");

          printf("Digite el nombre de la operación que quiere realizar (suma, resta, multiplicacion, division):\n");
          scanf("%s", &operacion);

          printf("Digite el primer número:\n");
          scanf("%f", &numero_1);
```

```
printf("Digite el segundo número:\n");

scanf("%f", &numero_2);


if((strcmp(operacion, "suma")==0)||(strcmp(operacion, "SUMA")==0)){


                resultado=numero_1+numero_2;

                printf("La suma de los números es: %.2f", resultado);

}

else if((strcmp(operacion, "resta")==0)||(strcmp(operacion, "RESTA")==0)){


                resultado=numero_1-numero_2;

                printf("El resultado de la resta es: %.2f", resultado);

}

else if((strcmp(operacion, "multiplicacion")==0)||(strcmp(operacion,
"MULTIPLICACION")==0)){

                resultado=numero_1*numero_2;

                printf("El resultado de la multiplicación es: %.2f", resultado);

}

else if((strcmp(operacion, "division")==0)||(strcmp(operacion, "DIVISION")==0)){

                resultado=numero_1/numero_2;

                printf("La división de los números es: %.2f", resultado);
```

}

}

2. EJERCICIO

```
#include<stdio.h>

#include<string.h>

#include<locale.h>


int main(){


setlocale(LC_CTYPE, "SPANISH");


        char tipo[1];


        printf("Programa para definir el incremento del límite de su tarjeta de crédito\n");


        printf("Digite el tipo de su tarjeta de credito (A,B,C). Si su tarjeta es de otro tipo digite el número 0\n");


        scanf("%s", tipo);



        if ((strcmp(tipo, "A")==0)||(strcmp(tipo, "a")==0)){
```

```
                printf("El aumento del límite de su crédito será del 28 porcentaje");
        }
        else if((strcmp(tipo, "B")==0)||(strcmp(tipo, "b")==0)){
                printf("El aumento del límite de su crédito será del 37 porcentaje");
        }
        else if((strcmp(tipo, "C")==0)||(strcmp(tipo, "c")==0)){
                printf("El aumento del límite de su crédito será del 55 porcentaje");
        }
        else if(strcmp(tipo, "0")==0){
                printf("El aumento del límite de su crédito será del 60 porcentaje");
        }
        else if(strcmp(tipo, "")!=0) {
                printf("Lo siento, te has equivocado");
        }
}
```

Copiar el contenido de una cadena en otra cadena. Vea el código del Cuadro de Código 14.4. Mientras no se llega al carácter fin de cadena original, se van copiando uno a uno los valores de las variables de la cadena en copia. Al final, cuando ya se ha llegado al final en original, habrá que cerrar también la cadena en copia, mediante un carácter ASCII 0.

1. Que lea una cadena y la muestre al revés.

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{

        int indice,x;
   char frase[50];

        printf("Introduzca una frase: ");
   gets(frase);

        for(x = 0;x < 50;x++)
        {
        if (frase[x]=='\0')
        {
        indice=x;
        break;
    }
        }
        printf("La frase al reves es: \n\n");
     for(x = indice-1;x >=0;x--)
   {
        printf("%c",frase[x]);
   }
        printf("\n\n");
        system("PAUSE");
        return 0;
}
```

2. Que lea una cadena y diga cuantas vocales hay.

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
        int sum=0,x;
   char frase[50];

        printf("Introduzca una frase: ");
   gets(frase);
        for(x = 0;x < 50;x++)
        {
        switch (frase[x])
        {
        case 'a':
        sum++;
        break;
        case 'e':
        sum++;
        break;
        case 'i':
        sum++;
        break;
        case 'o':
```

```
            sum++;
        break;
        case 'u':
        sum++;
        break;
        default:
          break;
        }
        }

        printf("\n\nEn la frase hay %d vocales\n\n",sum);

   printf("\n\n");

        system("PAUSE");
        return 0;
}
```

3. Que lea una cadena y diga cuantas mayúsculas hay.

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
        int sum=0,x;
   char frase[50];

        printf("Introduzca una frase: ");
   gets(frase);

        for(x = 0;x < 50;x++)
   {
        if (frase[x]>=65 && frase[x]<=90)
        {
        sum++;
        }
        }

        printf("\n\nEn la frase hay %d mayúsculas\n\n",sum);

   printf("\n\n");

        system("PAUSE");
        return 0;
}
```

4. Que lea una cadena y la encripte sumando 3 al código ASCII de cada carácter. Mostrar por pantalla.

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
```

```
{
        int sum=0,x;
   char frase[50];

        printf("Introduzca una frase: ");
   gets(frase);

        for(x = 0; x < 50;x++)
        {
        if (frase[x]!='\0')
    {
        frase[x]=frase[x]+3;
        }
        }

        printf("\n\nLa nueva frase es:\n\n",sum);
   printf("\n\n%s\n\n",frase);
        printf("\n\n");

        system("PAUSE");
   return 0;
}
```

5. Que gestione los datos de stock de una tienda de comestibles, la información a recoger será: nombre del producto, precio, cantidad en stock. La tienda dispone de 10 productos distintos. El programa debe ser capaz de:

1. *Dar de alta un producto nuevo.*
2. *Buscar un producto por su nombre.*
3. *Modificar el stock y precio de un producto dado.*

```
#include <stdio.h>
#include <stdlib.h>

struct producto {
        char nombre[50];
        float precio;
        int cantidad;
};

int main(int argc, char *argv[])
{
        struct producto prod,productos[10];

        int x,opcion=1;

        for (x=0;x<10;x++)
     {
        strcpy(productos[x].nombre,"X");
        productos[x].precio=0;
        productos[x].cantidad=0;
```

```
    }

    while ((opcion==1 || opcion==2 || opcion==3) && (opcion!=4))
    {

    printf("1- Alta de producto\n");
    printf("2- Buscar por nombre\n");
    printf("3- Modificar stock y precio\n");
    printf("4- Salir\n");
    printf("Introduzca una opción: ");
scanf("%d",&opcion);

    if (opcion==1)
{
    printf("Introduzca un nombre: ");
    gets(prod.nombre);
    gets(prod.nombre);
    printf("Introduzca un precio: ");
    scanf("%f",&prod.precio);
    printf("Introduzca un stock: ");
    scanf("%d",&prod.cantidad);

  for(x = 9; x >=0; x--)
  {
    if (x!=0)
    {
    strcpy(productos[x].nombre,productos[x-1].nombre);
    productos[x].precio=productos[x-1].precio;
    productos[x].cantidad=productos[x-1].cantidad;
    }
    else
    {
    strcpy(productos[x].nombre,prod.nombre);
    productos[x].precio=prod.precio;
    productos[x].cantidad=prod.cantidad;
    }
    }
    printf("\nProducto creado. \n\n");
    }
    else if (opcion==2)
{
    printf("Introduzca un nombre: ");
  gets(prod.nombre);
    gets(prod.nombre);

    for(x = 0; x < 10;x++)
  {

    if (strcmp(productos[x].nombre,prod.nombre)==0)
    {
    printf("\nNombre: %s\n",productos[x].nombre);
          printf("Precio: %f\n",productos[x].precio);
    printf("Cantidad en Stock: %d\n",productos[x].cantidad);
    }
    }
```

```
        printf("\n\n");
        }
        else if (opcion==3)
    {
        printf("Introduzca un nombre: ");
       gets(prod.nombre);
        gets(prod.nombre);

        for(x = 0; x < 10;x++)
       {
        if (strcmp(productos[x].nombre,prod.nombre)==0)
        {
        printf("Introduzca un precio: ");
              scanf("%f",&productos[x].precio);
        printf("Introduzca un stock: ");
        scanf("%d",&productos[x].cantidad);
        printf("\nProducto modificado.");
        }
        }
        printf("\n\n");
        }
        }

        system("PAUSE");
        return 0;
}
```

6. Que gestiona las notas de una clase de 20 alumnos de los cuales sabemos el nombre y la nota. El programa debe ser capaz de:

1. *Buscar un alumno.*
2. *Modificar su nota.*
3. *Realizar la media de todas las notas.*
4. *Realizar la media de las notas menores de 5.*
5. *Mostrar el alumno que mejores notas ha sacado.*
6. *Mostrar el alumno que peores notas ha sacado.*

```
#include <stdio.h>
#include <stdlib.h>

struct alumno {
        char nombre[50];
   float nota;
};

int main(int argc, char *argv[])
{
        struct alumno alum,alumnos[5];

   int x,opcion=1;
        float sum=0,cont=0,mejor,peor;
```

```
        for (x=0;x<5;x++)
{
        printf("Introduzca nombre alumno:");
        gets(alumnos[x].nombre);
        gets(alumnos[x].nombre);
        printf("Introduzca nota:");
        scanf("%f",&alumnos[x].nota);
        }

        while ((opcion==1 || opcion==2 ||
        opcion==3 || opcion==4 ||
        opcion==5 || opcion==6) && (opcion!=7))
        {

        printf("1- Buscar un alumno\n");
        printf("2- Modificar nota\n");
        printf("3- Media de todas las notas\n");
        printf("4- Media de todas las notas inferiores a 5\n");
        printf("5- Alumno con mejores notas\n");
        printf("6- Alumno con peores notas\n");
        printf("7- Salir\n");
        printf("Introduzca una opción: ");
    scanf("%d",&opcion);

        if (opcion==1)
    {
        printf("Introduzca un nombre: ");
        gets(alum.nombre);
        gets(alum.nombre);

      for(x = 0; x < 5;x++)
       {
        if (strcmp(alumnos[x].nombre,alum.nombre)==0)
        {
        printf("\nNombre: %s\n",alumnos[x].nombre);
        printf("Nota: %f\n",alumnos[x].nota);
        }
        }
        printf("\n\n");
        }
        else if (opcion==2)
    {
        printf("Introduzca un nombre: ");
        gets(alum.nombre);
        gets(alum.nombre);

      for(x = 0; x < 5;x++)
       {
        if (strcmp(alumnos[x].nombre,alum.nombre)==0)
        {
        printf("Introduzca una nota: ");
        scanf("%f",&alumnos[x].nota);
        printf("\nNota modificada.");
        }
```

```
    }
    printf("\n\n");
    }
    else if (opcion==3)
    {
    sum=0;
    for(x = 0; x < 5;x++)
  {
    sum=sum+alumnos[x].nota;
    }
    printf("\nLa media de las notas es de: %f \n",(sum/5));
}
    else if (opcion==4)
    {
    sum=0;
    cont=0;
    for(x = 0; x < 5;x++)
  {
    if (alumnos[x].nota<5)
    {
    sum=sum+alumnos[x].nota;
    cont++;
    }
    }
      printf("\nLa media de las notas inferiores a 5 es: %f \n",sum/cont);
}
    else if (opcion==5)
{
    mejor=0;
    for(x = 0; x < 5;x++)
    {
    if (alumnos[x].nota>mejor)
    {
    mejor=alumnos[x].nota;
    alum.nota=alumnos[x].nota;
    strcpy(alum.nombre,alumnos[x].nombre);
    }
    }
    printf("\nEl alumno con mejores notas es: %s \n",alum.nombre);
}
    else if (opcion==6)
       {
    peor=10;
    for(x = 0; x < 5;x++)
    {
    if (alumnos[x].nota<peor)
    {
    peor=alumnos[x].nota;
    alum.nota=alumnos[x].nota;
    strcpy(alum.nombre,alumnos[x].nombre);
    }
    }
    printf("\nEl alumno con peores notas es: %s \n",alum.nombre);
}
    }
```

```
        system("PAUSE");
        return 0;
}
```

7. Escribir un nombre y contar los caracteres

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main()
        {
        char nombre[20]; //string de 20 caracteres
        int longitud_string;

        printf("Escribe tu nombre: ");
        scanf("%s", &nombre);
        printf("Te llamas: %s \n", nombre); //muestro nombre por pantalla

        longitud_string = strlen(nombre); //strlen cuenta caracteres
        printf( "Tu nombre: %s, tiene %i caracteres. \n", nombre, longitud_string );

          system("PAUSE");
        }
```

8. El ahorcado

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
int main() {
        char frase[60],rep[100],temporal[100];
   char pal;
        int longitud,i,j,inicial,acertado=0,temp=0,oportunidades=5;
        int repetido=0,gano=0;

        printf("\t Juego del Ahorcado\n");
        printf("Introduzca la palabra a adivinar: ");
   gets(frase);

        system("cls");

   longitud = 0;
        inicial = 0;
        j = 0;

   rep[0] = ' ';
        rep[1] = '\0';


        do {
                system("cls");
        temp=0;
```

```
    if(inicial == 0) {
    for(i=0;i<strlen(frase);i++) {
    if(frase[i] == ' ') {
    temporal[i] = ' ';
    longitud++;
    }
    else {
    temporal[i] = '_';
    longitud++;
    }
    }
    }

    inicial = 1;

    temporal[longitud] = '\0';

    for(i=0;i<strlen(rep);i++) {
    if(rep[i] == pal) {
    repetido = 1;
    break;
    }
    else {
   repetido = 0;
    }
    }

    if(repetido == 0) {
 for(i=0;i<strlen(frase);i++) {
                if(frase[i] == pal) {
                 temporal[i] = pal;
                  acertado++;
                  temp=1;
                }
    }
    }

    if(repetido == 0) {
    if(temp == 0) {
    oportunidades = oportunidades - 1;
    }
    }
    else {
    printf("Ya se ha introducido este caracter");
  printf("\n\n");
    }

    printf("\n");

    for(i=0;i<strlen(temporal);i++) {
    printf(" %c ",temporal[i]);
}

    printf("\n");
```

```
        if(strcmp(frase,temporal) == 0) {
                gano = 1;
                break;
        }

        printf("\n");
     printf("Letras Acertadas: %d",acertado);
        printf("\n");
        printf("Oportunidades Restantes: %d",oportunidades);
     printf("\n");

        rep[j] = pal;
     j++;

        if (oportunidades==0)
        {
        break;
        }

        printf("Introduzca una letra:");
        scanf("\n%c",&pal);

        }while(oportunidades != 0);

        if(gano) {
                printf("\n\n");
     printf("Enhorabuena, has ganado.");
   }
        else {
                printf("\n\n");
        printf("Has perdido.");
        }

        printf("\n\n");
        system("PAUSE");
   return 0;

}
```

EJERCICIO PROMEDIO DE NOTAS

El siguiente programa pide al usuario ingresar las notas de uno o más alumnos, y va mostrando los promedios de cada uno de ellos:

```
#include <stdio.h>

float promedio(int valores[], int cantidad) {
   int i;
   float suma = 0.0;

   for (i = 0; i < cantidad; ++i)
      suma += valores[i];

   return suma / (float) cantidad;
```

```
}


int main() {

   int notas[10];
   char nombre[20];
   char opcion[3];
   int n, i;

   do {
      printf("Ingrese nombre del alumno: ");
      scanf("%s", nombre);

      printf("Cuantas notas tiene %s? ", nombre);
      scanf("%d", &n);

      for (i = 0; i < n; ++i) {
         printf("  Nota %d: ", i + 1);
         scanf("%d", &notas[i]);
      }

      printf("El promedio de %s es %.1f\n", nombre, promedio(notas, n));

      printf("Desea calcular mas promedios (si/no)? ");
      scanf("%s", opcion);

   } while (opcion[0] == 's' || opcion[0] == 'S');

   return 0;
}
```

LENGUAJE C++

¿Qué es C? C es un lenguaje de programación de alto nivel desarrollado en el año 1972 por Dennos Ritchie en AT&T Bell Labs. La legibilidad, facilidad de mantenimiento y la portabilidad son algunas de las ventajas de este lenguaje, además que permite descender a nivel de hardware.

1. ESTRUCTURA DE UN PROGRAMA EN C

1.1 Estructura

Todo programa en **C** consta de una o más funciones, una de las cuales se llama **main**. El programa comienza en la función main, desde la cual es posible llamar a otras funciones.

Cada función estará formada por la cabecera de la función, compuesta por el nombre de la misma y la lista de argumentos (si los hubiese), la declaración de las variables a utilizar y la secuencia de sentencias a ejecutar.

Ejemplo:

```
#include <stdio.h>    libreria o biblioteca

#include <math.h>
```

```
#include <conio.h>

declaraciones globales

main( ) {
        variables locales
        bloque
}

funcion1( ) {
        variables locales
        bloque
}
```

1.2 Zona de Declaraciones y Cuerpo del Programa

Zona de declaraciones contiene las librerías y las variables que se van a utilizar en la realización del programa y la función principal además de la declaración de constantes.

Nota: Las variables pueden ir antes de iniciar las instrucciones del programa.

El cuerpo del programa se divide en inicio, instrucciones y fin.

La estructura quedaría de la siguiente manera:

****Zona de Declaraciones ****

#include <stdio.h>

main() *FUNCIÓN PRINCIPAL*

CUERPO DEL PROGRAMA

int Tipo de variables nombre; Variables

{ Inicio del programa

INSTRUCCIONES

} Fin del programa

Nota: En el lenguaje C no es lo mismo una variable en minúsculas (ejemplo nom) a una en mayúsculas (ejemplo NOM), por lo que te recomiendo que todo lo hagas en minúsculas.

1.3 Comentarios

A la hora de programar es conveniente añadir comentarios (cuantos más mejor) para poder saber que función tiene cada parte del código, en caso de que no lo utilicemos durante algún tiempo.

Además facilitaremos el trabajo a otros programadores que puedan utilizar nuestro archivo fuente.

Para poner comentarios en un programa escrito en **C** usamos los símbolos **/*** y ***/**:

```
/* Este es un ejemplo de comentario */

/* Un comentario también puede
estar escrito en varias líneas */
```

El símbolo **/*** se coloca al principio del comentario y el símbolo ***/** al final.

El comentario, contenido entre estos dos símbolos, no será tenido en cuenta por el compilador

1.4 Palabras clave

Existen una serie de indicadores reservados, con una finalidad determinada, que no podemos utilizar como identificadores.

A continuación vemos algunas de estas palabras clave:

char	**int**	**float**	**double**	**if**
else	**do**	**while**	**for**	**switch**
short	**long**	**extern**	**static**	**default**
continue	**break**	**register**	**sizeof**	**typedef**

1.5 Identificadores

Un identificador es el nombre que damos a las variables y funciones. Está formado por una secuencia de letras y dígitos, aunque también acepta el carácter de subrayado **_**. Por contra no acepta los acentos ni la **ñ**/**Ñ**.

El primer carácter de un identificador no puede ser un número, es decir que debe ser una letra o el símbolo **_**.

Se diferencian las mayúsculas de las minúsculas, así **num**, **Num** y **nuM** son distintos identificadores.

A continuación vemos algunos ejemplos de identificadores válidos y no válidos:

Válidos	**No válidos**
_num	1num
var1	número2
fecha_nac	año_nac

2. TIPOS DE DATOS

2.1 Tipos

En 'C' existen básicamente cuatro tipos de datos, aunque como se verá después, podremos definir nuestros propios tipos de datos a partir de estos cuatro. A continuación se detalla su nombre, el tamaño que ocupa en memoria y el rango de sus posibles valores.

TIPO	Tamaño	Rango de valores
char	1 byte	-128 a 127
int	2 bytes	-32768 a 32767
float	4 bytes	3'4 E-38 a 3'4 E+38
double	8 bytes	1'7 E-308 a 1'7 E+308

Tipos de Datos

TIPO		TAMAÑO EN BITS	RANGO MÍNIMO
CARACTER	char	8	-128 a 127
	unsigned char	8	0 a 255
	signed char	8	-128 a 127
ENTERO	int	16	-32768 a 32767
	unsigned int	16	0 a 65535
	signed int	16	Igual que **int**
	short int	16	Igual que **int**
	unsigned short int	16	0 a 65535
	signed short int	16	Igual que **short int**
	long int	32	-2147483648 a 2147483647
	signed long int	32	Igual a **long int**
	unsigned long int	32	0 a 4294967295
PUNTO FLOTANTE	float	32	±(1.8e-38 a 3.4e+38) 6 dígitos de precisión
	double	64	±(2.2e-308 a 1.8e+308) 15 dígitos de precisión
	long double	80	±(3.4e-4932 a 1.2e+4932) 18 dígitos de precisión
SIN TIPO	void	0	Sin valor

2.2. Calificadores de tipo

Los calificadores de tipo tienen la misión de modificar el rango de valores de un determinado tipo de variable. Estos calificadores son cuatro:

- **signed**

Le indica a la variable que va a llevar signo. Es el utilizado por defecto.

	tamaño	**rango de valores**
signed char	1 byte	-128 a 127
signed int	2 bytes	-32768 a 32767

- **unsigned**

Le indica a la variable que no va a llevar signo (valor absoluto).

	tamaño	**rango de valores**
unsigned char	1 byte	0 a 255
unsigned int	2 bytes	0 a 65535

- **short**

  Rango de valores en formato corto (limitado). Es el utilizado por defecto.

	tamaño	rango de valores
short char	1 byte	-128 a 127
short int	2 bytes	-32768 a 32767

- **long**

  Rango de valores en formato largo (ampliado).

	tamaño	rango de valores
long int	4 bytes	-2.147.483.648 a 2.147.483.647
long double	10 bytes	-3'36 E-4932 a 1'18 E+4932

También es posible combinar calificadores entre sí:

signed long int = long int = long

unsigned long int = unsigned long 4 bytes 0 a 4.294.967.295 (El mayor entero permitido en 'C')

2.3 LAS VARIABLES

Una variable es un tipo de dato, referenciado mediante un identificador (que es el nombre de la variable). Su contenido podrá ser modificado a lo largo del programa.

Una variable sólo puede pertenecer a un tipo de dato. Para poder utilizar una variable, primero tiene que ser declarada:

[calificador] <tipo> <nombre>

Es posible inicializar y declarar más de una variable del mismo tipo en la misma sentencia:

[calificador] <tipo> <nombre1>,<nombre2>=<valor>,<nombre3>=<valor>,<nombre4>

Ejemplo

```
/* Uso de las variables */

#include <stdio.h>

main() /* Suma dos valores */
{
        int num1=4,num2,num3=6;
```

```
        printf("El valor de num1 es
%d",num1);
        printf("\nEl valor de num3 es
%d",num3);
        num2=num1+num3;
        printf("\nnum1 + num3 =
%d",num2);
}
```

printf (“digite un numero”);

scanf(“%i”, &num1)

¿ Dónde se declaran ?

Las variables pueden ser de dos tipos según el lugar en que las declaremos: *globales* o *locales*.

La variable global se declara antes de la **main( )**. Puede ser utilizada en cualquier parte del programa y se destruye al finalizar éste.

La variable local se declara después de la **main( )**, en la función en que vaya a ser utilizada. Sólo existe dentro de la función en que se declara y se destruye al finalizar dicha función.

El identificador (nombre de la variable) no puede ser una **palabra clave** y los caracteres que podemos utilizar son las letras: **a-z** y **A-Z** (-ojo! la **ñ** o **Ñ** no está permitida), los números: **0-9** y el símbolo de subrayado _. Además hay que tener en cuenta que el primer carácter no puede ser un número.

Ejemplo

```
/* Declaración de variables */

#include <stdio.h>

int a;
main() /* Muestra dos valores */
{
        int b=4;
        printf("b es local y vale
%d",b);
        a=5;
        printf("\na es global y vale
%d",a);
}
```

2.4.- CONSTANTES

Al contrario que las **variables**, las constantes mantienen su valor a lo largo de todo el programa.

Para indicar al compilador que se trata de una constante, usaremos la directiva **#define**:

```
#define <identificador> <valor>
```

Observa que no se indica el punto y coma de final de sentencia ni tampoco el tipo de dato.

La directiva **#define** no sólo nos permite sustituir un nombre por un valor numérico, sino también por una cadena de caracteres.

El valor de una constante no puede ser modificado de ninguna manera.

Ejemplo

```
/* Uso de las constantes */

#include <stdio.h>
#define pi 3.1416
#define escribe printf
main() /* Calcula el perímetro */
{
        int radio;
        printf("Introduce el radio: ");
        scanf("%d",&radio);
        printf("El perímetro es:
%f",2*pi*radio);
}
```

2.5.- SECUENCIAS DE ESCAPE

Ciertos caracteres no representados gráficamente se pueden representar mediante lo que se conoce como secuencia de escape.

A continuación vemos una tabla de las más significativas:

\n	salto de línea \n\n\n\n
\b	retroceso
\t	tabulación horizontal
\v	tabulación vertical
****	contrabarra
\f	salto de página
\'	apóstrofe
\"	comillas dobles
\0	fin de una cadena de caracteres

Ejemplo

```
/* Uso de las secuencias de escape */

#include <stdio.h>

main() /* Escribe diversas sec. de escape */
{
        printf("Me llamo \"Nemo\" el
grande");
        printf("\n")
        printf("\nDirección: C\\ Mayor 25");
        printf("\nHa salido la letra \'L\'");
        printf("\nRetroceso\b");
        printf("\n\tEsto ha sido todo");
}
```

2.6- INCLUSIÓN DE FICHEROS

En la programación en C es posible utilizar funciones que no estén incluídas en el propio programa. Para ello utilizamos la directiva **#include**, que nos permite añadir librerías o funciones que se encuentran en otros ficheros a nuestro programa.

Para indicar al compilador que vamos a incluir ficheros externos podemos hacerlo de dos maneras (siempre antes de las declaraciones).

1. Indicando al compilador la ruta donde se encuentra el fichero.

```
#include "misfunc.h"
#include "c:\includes\misfunc.h"
```

2. Indicando que se encuentran en el directorio por defecto del compilador.

```
#include <misfunc.h>
```

3.- OPERADORES ARITMÉTICOS Y DE ASIGNACIÓN

A continuación se explican los tipos de operadores (aritméticos y de asignación) que permiten realizar operaciones matemáticas en lenguaje C.

3.1.- Operadores aritméticos

Existen dos tipos de operadores aritméticos:

Los binarios: nro=nro+1

+	Suma
-	Resta
*****	Multiplicación
/	División
%	Módulo (resto)

y los unarios:

++	Incremento (suma 1)
- -	Decremento (resta 1)
-	Cambio de signo

Su sintaxis es:

binarios:
<variable1><operador><variable2>

unarios:
<variable><operador> y al revés, **<operador><variable>.**

Ejemplo

```
/* Uso de los operadores aritméticos */

#include <stdio.h>

main() /* Realiza varias operaciones */
{
        int a=1,b=2,c=3,r;
        r=a+b;
        printf("%d + %d = %d\n",a,b,r);
        r=c-a;
        printf("%d - %d = %d\n",c,a,r);
        b++;
        printf("b + 1 = %d",b);
}
```

3.2.- Operadores de asignación

La mayoría de los operadores aritméticos binarios explicados en el capítulo anterior tienen su correspondiente operador de asignación:

=	Asignación simple suma=a+b
+=	Suma
-=	Resta
*=	Multiplicación
/=	División
%=	Módulo (resto)

Con estos operadores se pueden escribir, de forma más breve, expresiones del tipo:

n=n+3 se puede escribir **n+=3**

k=k*(x-2) lo podemos sustituir por **k*=x-2**

Ejemplo

```
/* Uso de los operadores de asignación */

#include <stdio.h>

main() /* Realiza varias operaciones */
{
        int a=1,b=2,c=3,r;
        a+=5;
        printf("a + 5 = %d\n",a);
        c-=1;
        printf("c - 1 = %d\n",c);
        b*=3;
        printf("b * 3 = %d",b);
}
```

3.3.- JERARQUÍA DE LOS OPERADORES

Será importante tener en cuenta la precedencia de los operadores a la hora de trabajar con ellos:

( ) Mayor precedencia
++, - -
***, /, %**
+, - Menor precedencia

Las operaciones con mayor precedencia se realizan antes que las de menor precedencia.

Si en una operación encontramos signos del mismo nivel de precedencia, dicha operación se realiza de izquierda a derecha. A continuación se muestra un ejemplo sobre ello:

a*b+c/d-e

1. a*b resultado = x
2. c/d resultado = y
3. x+y resultado = z
4. z-e

Fijarse que la multiplicación se resuelve antes que la división ya que está situada más a la izquierda en la operación. Lo mismo ocurre con la suma y la resta.

Ejemplo

```
/* Jerarquía de los operadores */

#include <stdio.h>

main() /* Realiza una operación */
{
        int a=6,b=5,c=4,d=2,e=1,x,y,z,r;
        x=a*b;
        printf("%d * %d = %d\n",a,b,x);
        y=c/d;
        printf("%d / %d = %d\n",c,d,y);
        z=x+y;
        printf("%d + %d = %d\n",x,y,z);
        r=z-e;
        printf("%d = %d",r,a*b+c/d-e);
}
```

4. SALIDA Y ENTRADA DE DATOS

4.1.- Sentencia printf( )

La rutina printf permite la aparición de valores numéricos, caracteres y cadenas de texto por pantalla.

El prototipo de la sentencia *printf* es el siguiente:

printf(control,arg1,arg2...);

En la cadena de control indicamos la forma en que se mostrarán los argumentos posteriores. También podemos introducir una cadena de texto (sin necesidad de argumentos), o combinar ambas posibilidades, así como **secuencias de escape**.

En el caso de que utilicemos argumentos deberemos indicar en la cadena de control tantos modificadores como argumentos vayamos a presentar.

El modificador está compuesto por el carácter **%** seguido por un carácter de conversión, que indica de qué tipo de dato se trata.

Ejemplo

```
/* Uso de la sentencia printf() 1. */
```

```
#include <stdio.h>

main() /* Saca por pantalla una suma */
{
        int a=20,b=10;
        printf("El valor de a es %d\n",a);
        printf("El valor de b es %d\n",b);
        printf("Por tanto %d+%d=%d",a,b,a+b);
}
```

Los modificadores más utilizados son:

%c	Un único carácter
%d	Un entero con signo, en base decimal
%u	Un entero sin signo, en base decimal
%o	Un entero en base octal
%x	Un entero en base hexadecimal
%e	Un número real en coma flotante, con exponente
%f	Un número real en coma flotante, sin exponente
%s	Una cadena de caracteres
%p	Un puntero o dirección de memoria

Ejemplo

```
/* Uso de la sentencia printf() 2. */

#include <stdio.h>

main() /* Modificadores 1 */
{
        char cad[]="El valor de";
        int a=-15;
        unsigned int b=3;
        float c=932.5;
        printf("%s a es %d\n",cad,a);
        printf("%s b es %u\n",cad,b);
        printf("%s c es %e o %f",cad,c,c);
}
```

El formato completo de los modificadores es el siguiente:

% [signo] [longitud] [.precisión] [l/L] conversión

Signo: indicamos si el valor se ajustará a la izquierda, en cuyo caso utilizaremos el signo menos, o a la derecha (por defecto).

Longitud: especifica la longitud máxima del valor que aparece por pantalla. Si la longitud es menor que el número de dígitos del valor, éste aparecerá ajustado a la izquierda.

***Precisión*: indicamos el número máximo de decimales que tendrá el valor.**

l/L: utilizamos l cuando se trata de una variable de tipo long y L cuando es de tipo double.

Ejemplo

```
/* Uso de la sentencia printf() 3. */

#include <stdio.h>

main() /* Modificadores 2 */
{
        char cad[ ]="El valor de";
        int a=25986;
        long int b=1976524;
        float c=9.57645;
        printf("%s a es %9d\n",cad,a);
        printf("%s b es %ld\n",cad,b);
        printf("%s c es %.3f",cad,c);
}
```

4.2.- Sentencia scanf( )

La rutina scanf permite entrar datos en la memoria del ordenador a través del teclado.

El prototipo de la sentencia *scanf* es el siguiente:

scanf(control,arg1,arg2...);

En la cadena de control indicaremos, por regla general, los modificadores que harán referencia al tipo de dato de los argumentos. Al igual que en la sentencia *printf* los **modificadores** estarán formados por el carácter **%** seguido de un carácter de conversión. Los argumentos indicados serán, nuevamente, las variables.

La principal característica de la sentencia *scanf* es que necesita saber la posición de la memoria del ordenador en que se encuentra la variable para poder almacenar la información obtenida. Para indicarle esta posición utilizaremos el símbolo *ampersand* (**&**), que colocaremos delante del nombre de cada variable. (Esto no será necesario en los arrays).

Ejemplo

```
/* Uso de la sentencia scanf(). */

#include <stdio.h>

main() /* Solicita dos datos */
{
        char nombre[10];
        int edad;
        printf("Introduce tu nombre:
");
        scanf("%s",nombre);
        printf("Introduce tu edad: ");
        scanf("%d",&edad);
}
```

5. OPERADORES RELACIONALES

Los operadores relacionales se utilizan para comparar el contenido de dos variables.

En C existen seis operadores relacionales básicos:

>	Mayor que	
<	Menor que	
>=	Mayor o igual que	
<=	Menor o igual que	
==	Igual que	nro1==nro2
!=	Distinto que	

El resultado que devuelven estos operadores es **1** para Verdadero y **0** para Falso.

Si hay más de un operador se evalúan de izquierda a derecha. Además los operadores == y != están por debajo del resto en cuanto al orden de precedencia.

Ejemplo

```
/* Uso de los operadores relacionales.  */

#include <stdio.h>

main() /* Compara dos números entre ellos */
{
   int a,b;
   printf("Introduce el valor de A: ");
   scanf("%d",&a);
   printf("Introduce el valor de B: ");
   scanf("%d",&b);
   if(a>b)
      printf("A es mayor que B");
   else if(a<b)
      printf("B es mayor que A");
   else
      printf("A y B son iguales");
}
```

6. ESTRUCTURAS DE PROGRAMACIÓN EN LENGUAJE C++

6.1. ESTRUCTURAS SECUENCIALES

Programas

1. Programa que calcule la suma de dos números

```
#include <stdio.h>
  int main()
{
        int num1, num2, resultado;
        printf("Por favor, introduzca un numero:  ");
        scanf("%d",&num1);
       printf("Ahora, inserte otro:  ");
        scanf("%d",&num2);
        resultado=num1+num2;
        printf("\nEl resultado es %d\n",resultado);
}
```

2. Programa que calcula longitudes de circunferencia

```
#include <stdio.h>
  int main(){
        float radio,sol1;
        printf("Bienvenido, calcularemos la longitud de su circunferencia.\n\n");
        printf("Lo unico que debe hacer es introducir el radio: ");
        scanf("%f",&radio);
        longitud=2*3.141592*radio;
        printf("\n\nEl resultado es %f\n\n",longitud);
}
```

3. Programa que calcula la media aritmética de tres números cualesquiera.

```
#include <stdio.h>
       int main(){
```

```
        float num1,num2,num3,media;
        printf("Bienvenido, calcularemos la media aritmetica de tres numeros.\n\n");
        printf("Por favor, introduzca el primero: ");
        scanf("%f",&num1);
        printf("Ahora, inserte el segundo de ellos: ");
        scanf("%f",&num2);
        printf("Por ultimo, teclee el numero final: ");
        scanf("%f",&num3);
        media=(num1+num2+num3)/3;
        printf("\nEl resultado es %f\n\n",media);
}
```

4. Programa que calcula el área de un triángulo (Fórmula de Herón).

```
 #include <stdio.h>
#include <math.h>
  int main(){
       float lado1,lado2,lado3,sp,area;
        printf("Bienvenido. Calcularemos el area del triangulo.\n\n");
        printf("Introduce el primer lado: ");
        scanf("%f",&lado1);
        printf("Ahora, inserta el segundo lado: ");
        scanf("%f",&lado2);
        printf("Por ultimo, escribe el tercer lado: ");
         scanf("%f",&lado3);
        resultado=(lado1+lado2+lado3)/2;
        area=sqrt(resultado*(resultado-lado1)*(resultado-lado2)*(resultado-lado3));
         printf("\nEl area obtenida es %f\n\n",area);
        printf("Muchas gracias por utilizar este programa.\n\n");
}
```

5. Programa que calcula el capital final de un interés simple.

```
#include <stdio.h>
 int main()
 {
       float capital,interes,tiempo,capital_final;
       printf("Bienvenido. Calcularemos el capital final de un interes simple.\n\n");
       printf("Por favor, introduce el capital inicial: ");
       scanf("%f",&capital);
       printf("Ahora, escribe el interes al que esta colocado: ");
       scanf("%f",&interes);
       printf("Por ultimo, inserta el tiempo al que se deja el capital: ");
       scanf("%f",&tiempo);
       capital_final=capital+capital*(interes/100)*tiempo;
       printf("\n\nEl capital final es de %f\n\n",capital_final);
}
```

6. Programa que calcule raíces cuadradas enteras.

```
#include <stdio.h>
#include <math.h>
 int main(){
       int num,resultado;
       printf("Por favor, inserte un numero");
       scanf("%d",&num);
      resultado=sqrt(num);
       printf("\nSu raiz cuadrada es %d\n\n",resultado);
}
```

Otro símbolo que necesitaremos a partir de ahora será el "%", que sirve para calcular el resto de una división. Podemos verlo en el siguiente ejemplo.

7. Programa que calcule el resto de cualquier división entera.

```
#include <stdio.h>
```

```
int main()
{
	int dividendo,divisor,resto;
	printf("Hola, obtendremos el resto de cualquier division entera.\n\n");
	printf("Inserte el dividendo: ");
	scanf("%d",&dividendo);
	printf("Bien, escriba el divisor: ");
	scanf("%d",&divisor);
	resto=dividendo%divisor;
	resto=dividendo mod divisor
	printf("\nEl resto de la division es %d\n\n",resto);
}
```

8) Leer un número y escribir el valor absoluto del mismo.

```
#include<stdio.h>
#include<math.h>
 int main()
{
int num,valor_absoluto;
printf("ingrese un numero para obtener su valor absoluto\n");
scanf("%d",&num);
valor_absoluto=abs(num);
printf("\nvalor dado es\n %d",valor_absoluto);
}
```

9. Programa que imprima en pantalla: HOLA COMO ESTAS.

```
  #include <stdio.h>
  #include <conio.h>
int main()
  {
	printf(“HOLA\n”);
```

```
        printf("COMO
        ESTAS\n");
        getch();
        return 0;   /*finaliza la ejecución de una función y devuelve el control a la función de
        llamada*/
  }
```

Notaras que en el programa anterior está incluida la librería <conio.h> esta librería es utilizada para getch() que se encuentra al final de las instrucciones; getch(); te obliga a presionar una tecla antes de finalizar tu programa.

10. Programa que lee 2 números, los suma, imprime el resultado de la suma y lo multiplica por 2.

```
#include <stdio.h>

#include <conio.h>

int main()

{

 int num1, num2, suma, multiplicacion;

 printf("Digite el primer numero\n");

 scanf("%i",&num1);

 printf("Digite el segundo numero\n");

 scanf("%i",&num2);

 suma=num1+num2;

 multiplicacion=res1*2;

 printf("El resultado de la suma es: %i \n",suma);

 printf("El resultado de la multiplicación es: %i\n", multplicacion);

 getch();

return 0;

}
```

7. SENTENCIAS CONDICIONALES

Este tipo de sentencias permiten variar el flujo del programa en base a unas determinadas condiciones.

Existen varias estructuras diferentes:

7.1.- Estructura IF...ELSE

Sintaxis:

```
if (condición) sentencia;
```

La sentencia solo se ejecuta si se cumple la condición. En caso contrario el programa sigue su curso sin ejecutar la sentencia.

Otro formato:

```
if (condición) sentencia1;
else sentencia2;
```

Si se cumple la condición ejecutará la sentencia1, sinó ejecutará la sentencia2. En cualquier caso, el programa continuará a partir de la sentencia2.

```
/* Uso de la sentencia condicional IF. */

#include <stdio.h>

main() /* Simula una clave de acceso */
{
  int usuario,clave=18276;
  printf("Introduce tu clave: ");
  scanf("%d",&usuario);
```

```
  if(usuario==clave)
    printf("Acceso permitido");
  else
    printf("Acceso denegado");
}
```

Otro formato:

```
if (condición) sentencia1;
else if (condición) sentencia2;
else if (condición) sentencia3;
else sentencia4;
```

Con este formato el flujo del programa únicamente entra en una de las condiciones. Si una de ellas se cumple, se ejecuta la sentencia correspondiente y salta hasta el final de la estructura para continuar con el programa.

Existe la posibilidad de utilizar llaves para ejecutar más de una sentencia dentro de la misma condición.

```
/* Uso de la sentencia condicional ELSE...IF. */

#include <stdio.h>

main() /* Escribe bebé, niño o adulto */
{
  int edad;
  printf("Introduce tu edad: ");
  scanf("%d",&edad);
  if (edad<1)
    printf("Lo siento, te has equivocado.");
  else if (edad<3) printf("Eres un bebé");
  else if (edad<13) printf("Eres un niño");
  else printf("Eres adulto");
}
```

OPERADORES LÓGICOS

Los operadores lógicos básicos son tres:

&&	AND
||	OR
!	NOT (El valor contrario)

Estos operadores actúan sobre expresiones lógicas. Permiten unir expresiones lógicas simples formando otras más complejas.

OPERANDOS		AND	OR
V	V	V	V
V	F	F	V
F	V	F	V
F	F	F	F

V = Verdadero F = Falso

Ejemplo

```
/* Uso de los op. lógicos AND,OR,NOT. */

#include <stdio.h>

main() /* Compara un número introducido */
{
  int numero;
  printf("Introduce un número: ");
  scanf("%d",&numero);
  if(!(numero>=0))
    printf("El número es negativo");
  else if((numero<=100)&&(numero>=25))
    printf("El número está entre 25 y 100");
  else if((numero<25)||(numero>100))
    printf("El número no está entre 25 y 100");
}
```

Problemas Condicionales

Ejemplo:

1. Programa para determinar si un número es par o impar.

```
#include <stdio.h>

int main() {

  int num; int res;

 printf("Programa para determinar naturaleza par o impar de un numero\n\n");

  printf ("Introduzca un numero entero: ");
```

```
  scanf ("%d", &num);
        res = num%2;
        if (res==0) {
      printf ("El numero es par\n");
        } else {
      printf ("El numero es impar\n");
        }
}
```

2. Programa que pida un número del 1 al 5 y diga si es primo o no.

```
 #include <stdio.h>
#include <stdlib.h>  // librería para utilizar system("PAUSE)
int main(){
        int num;
        printf("Introduzca número del 1 al 5:\n");
   scanf("%d",&num);
        if (num!=4) {
     printf("Es primo\n\n");
        }
        else
        {
     printf("No es primo\n\n");
        }
   system("PAUSE");
        return 0;
}
```

3. Programa que lee dos números y dice cual es el mayor y cual es el menor.

```
#include <stdio.h>
#include <conio.h>
int main()
{
int num1, num2;
```

```
printf("Escriba un numero\n");
scanf("%i",&num1);
printf("Escriba otro numero\n");
scanf("%i",&num2);
if (num1>num2)
        printf("El numero %i es mayor que %i\n",num1, num2);
else
        printf("El numero %i es menor que %i\n",num1, num2);
getch();
return 0;
}
```

4. Programa que lee un número e imprime si es positivo o negativo.

```
#include <stdio.h>
#include <conio.h>
int main()
{
int num;
printf("Escribe un numero\n");
scanf("%i",&num);
if (num>=0)
        printf("EL NUMERO %i ES POSITIVO",num);
else
        printf("EL NUMERO %i ES NEGATIVO",num);
getch();
return 0;
}
```

5. Programa que lee dos números e imprime si es divisible.

```
#include<stdio.h>

int main(){

        int num1, num2;

        printf("Por favor digite el primer numero\n");

        scanf("%i",&num1);

        printf("Por favor digite segundo numero\n");

        scanf("%i",&num2);

        if (num1%num2==0){

                printf("el numero %i es divisible entre %i\n",num1,num2);

        }

return 0;

}
```

ELSE-IF

Básicamente el else-if se utiliza para escribir una decisión múltiple y su forma general es de la siguiente manera:

```
if(condición)
        {
        Instrucciones;
        }
 else if (condición)
        }
        Instrucciones;
 else
        {
        Instrucciones;
        }
```

Es importante remarcar que no importa la cantidad de else-if que pongas no existe un límite. El último else se maneja en caso de que ninguno de los anteriores cumpla con la condición.

Programas Decisiones Múltiples

1. Programa que muestra en pantalla una serie de opciones a elegir e imprime el costo del producto elegido.

```
#include <stdio.h>
#include <conio.h>
int main()
{
int p;
printf("SELECCIONE UN PRODUCTO\n\n");
printf("1. REFRESCO\n");
printf("2. PAPAS FRITAS\n");
printf("3. HAMBURGUESA\n");
printf("4. JUGO\n");
scanf("%i",&p);
if (p == 1)
printf("EL COSTO ES: $5.00");
else if(p == 2)
printf("EL COSTO ES: $10.00");
else if(p == 3)
printf("EL COSTO ES: $20.00");
else
printf("EL COSTO ES: $8.00");
getch();
return 0;
}
```

2. Programa que escriba en pantalla un comentario con respecto a la temperatura del día.

```
#include <stdio.h>
int main(){
	int temperatura,Y;
	printf("Bienvenido. Introduzca la temperatura");
	scanf("%d",&temperatura);
	if(temperatura<15){
		printf("\nBrrr... Que frio!\n");
	}else if(temperatura<25){
		printf("\nClima templado\n");
	}else{
		printf("\nBuf!..Que calor!\n");
	}
}
```

3. Programa que resuelve ecuaciones de segundo grado.

```
#include <stdio.h>
#include <math.h>
int main(){
	float A,B,C,D,resultado,sol1,sol2;
	printf("Bienvenido, Resolveremos su ecuacion de segundo grado.\n\n");
	printf("\nPor favor, introduzca el coeficiente A: ");
	scanf("%f",&A);
	printf("\nAhora, escriba el coeficiente B: ");
	scanf("%f",&B);
	printf("\nPor ultimo, inserte el coeficiente C: ");
	scanf("%f",&C);
```

```
        D=B*B-4*A*C;
        if(D<0){
                printf("\n\nDisculpe, no tiene solucion real\n\n");
        }else if(D==0){
                resultado=-B/2*A;
                printf("\n\nEl resultado de la ecuacion es %f\n\n",resultado);
        }else{
                sol1=(-B+sqrt(D))/(2*A);
                sol2=(-B-sqrt(D))/(2*A);
                printf("\n\nLos resultados de la ecuacion son %f y %f\n\n",sol1,sol2);
        }
        printf("\nGracias por utilizar este programa\n\n");
}
```

4. Programa que resuelva la ecuación cuadrática tipo $ax^2 + bx + c$

```
 #include <stdio.h>
#include <math.h>
int main() {
  //Declaración de variables
          double a, b, c, d, e;
   printf("Programa para calcular ecuacion cuadratica tipo a*x^2 + b*x + c = 0\n\n");
   //Obtención de datos
          printf ("Introduzca valor parametro a: ");          scanf ("%lf", &a);
          printf ("Introduzca valor parametro b: ");          scanf ("%lf", &b);
          printf ("Introduzca valor parametro c: ");          scanf ("%lf", &c);
//Cálculo y muestra de resultados
          d = pow (b,2) - 4 * a * c;          e = 2 * a;
          if (d == 0) { printf ("El resultado es x1 = x2 =%lf", - b / e);
          } else {
```

```
if (d > 0) {
   printf ("El resultado es:\n");
   printf ("x1 = %lf \n", (-b + sqrt(d)) / e);
   printf ("x2 = %lf \n", (-b - sqrt(d)) / e);
    } else {
   printf ("El resultado es: \n");
   printf ("x1 = %lf + %lf * i \n", -b / e, sqrt(-d)/e);
   printf ("x2 = %lf - %lf * i \n", -b / e, sqrt(-d)/e);
    }
    }
return 0;
    }
```

5. visualizar la tarifa de la luz según el gasto de corriente eléctrica. Para un gasto menor de 1000 kwxh la tarifa es de 1.2, entre 1000 y 1850 kwxh es de 1.0 y mayor de 1850 kwxh la tarifa es de 0.9.

```
#include<stdio.h>
#define tarifa1 1.2
#define tarifa2 1
#define tarifa3 0.9
int main(){
    float gasto, tasa;
    printf("digite el gasto de energia ");
    scanf("%f",&gasto);
    if (gasto<1000){
        tasa = tarifa1;
    }
    if(gasto>1000 && gasto<1850){
        tasa = tarifa2;
    }
        if(gasto>1850){
```

```
            tasa = tarifa3;

        }

        printf("la tarifa a pagar es :%.3f",tasa);

}
```

6. *Un proveedor de estéreos ofrece un descuento del 10% sobre el precio sin IVA, de algún aparato si este cuesta $200.000 o más. Además, independientemente de esto, ofrece un 5% de descuento si la marca es "NOSY". Determinar cuanto pagara, con IVA incluido, un cliente cualquiera por la compra de su aparato.*

```
#include <stdio.h>
#include <math.h>
#include <string.h>
int main()
{
float producto, valor_sin_iva, descuento, total, valor_apagar1, valor_apagar, descuento_marca, iva,
descuento_marc;
char marca[]="nosy";

printf("introduzca el valor del producto:\n");
scanf("%f", &producto);

printf("digite la marca del producto:\n");
scanf("%s",marca);

        iva=(producto*19)/100 ;
printf("valor del iva es :%.2f\n",iva);
        valor_sin_iva=(producto-iva) ;
printf("valor del producto sin iva:%.2f\n",valor_sin_iva);

if (producto>=200000)  {

descuento=(valor_sin_iva*10/100);
printf("el descuento del 10 por ciento del estereo es %.2f\n",descuento);
total=producto-descuento;
printf("el valor a pagar es de: %.2f\n",total);

} else{

printf("el producto no tiene descuento del 10 por ciento %.2f\n",producto);
}if (strcmp (marca,"nosy")== 0 ){

descuento_marca=(valor_sin_iva*5)/100 ;

printf("El descuento por la marca nosy es:%.2f\n",descuento_marca);
valor_apagar=(producto-descuento-descuento_marca);

printf("el valor a pagar con el descuento por ser marca nosy es %.2f\n", valor_apagar);
}
}
```

7.2. LA ESTRUCTURA CONDICIONAL SWITCH … CASE

Esta estructura se suele utilizar en los menús, de manera que según la opción seleccionada se ejecuten una serie de sentencias.

Su sintaxis es:

```
switch (variable){
        case contenido_variable1:
                sentencias;
                break;
        case contenido_variable2:
                sentencias;
                break;
        default:
                sentencias;
}
```

Cada case puede incluir una o más sentencias sin necesidad de ir entre llaves, ya que se ejecutan todas hasta que se encuentra la sentencia **BREAK**. La variable evaluada sólo puede ser de tipo **entero** o **carácter**. **default** ejecutará las sentencias que incluya, en caso de que la opción escogida no exista.

```
/* Uso de la sentencia condicional SWITCH. */
#include <stdio.h>

main() /* Escribe el día de la semana */
{
   int dia;
   printf("Introduce el día: ");
   scanf("%d",&dia);
   switch(dia){
      case 1: printf("Lunes"); break;
      case 2: printf("Martes"); break;
      case 3: printf("Miércoles"); break;
      case 4: printf("Jueves"); break;
      case 5: printf("Viernes"); break;
      case 6: printf("Sábado"); break;
```

```
        case 7: printf("Domingo"); break;
    }
}
```

El condicional switch permite elegir entre varias opciones posibles. En realidad, este condicional puede ser expresado también usando condicionales if, por lo que podríamos decir que no es estrictamente necesario. Sin embargo, en algunos casos resulta más claro usar el condicional switch y resulta útil.

La sintaxis a emplear con C es la siguiente:

```
switch (selector) {

case [valor selector 1]:
Instrucción 1;
Instrucción 2;

case [valor selector 2]:
Instrucción 3;
Instrucción 4;
.
.
.
case [valor selector n]:
Instrucción j;
Instrucción k;

default:
Instrucción x;
Instrucción y;
}
```

La variable a evaluar o selector ha de ser un valor con equivalencia ordinal (equivalencia numérica entera). Por ejemplo puede ser una variable tipo int ó char (ya que cada char lleva un número entero asociado), pero no puede ser un double ni un float ni una cadena de caracteres (string). Sólo se puede evaluar una expresión y no múltiples expresiones.

Las sentencias break; hacen que una vez verificado que se cumple una opción dentro del switch, se salga del mismo y se continúe la ejecución en la siguiente instrucción después del switch. El uso de

break; es opcional, pero es habitual incluir un break; después de cada evaluación. En caso de no hacerlo, se ejecutarán todas las instrucciones dentro de cualquier case (incluso aunque no se verifique) después del que ha verificado la condición hasta encontrar una sentencia break; o terminar la ejecución del switch. Esto puede generar resultados contradictorios o no esperados, de ahí que en general siempre se incluyan sentencias break;

La cláusula default hace que se ejecuten las instrucciones a continuación de ella en caso de que la ejecución del programa llegue a alcanzarla. Si se usan las sentencias break; en cada evaluación, sólo se ejecutará lo indicado en default si no se ha encontrado una coincidencia previa.

La evaluación de expresiones es de coincidencia entre la variable evaluada y el valor indicado en case. Por ejemplo, case 12 indicaría que si la expresión evaluada vale 12 se ejecutarán las instrucciones anexas. También se puede usar case empleando constantes simbólicas (pero no con constantes definidas con const).

Veamos un ejemplo: ***programa que permita calcular el salario de un trabajador, dependiendo del nivel de antigüedad,***

1. *//ejemplo con la condición switch*

```
#include <stdio.h>

#include <stdlib.h>

#include <math.h>

int main()

{

        int nivel;

        float salario, salario_nuevo;

        printf("Introduce el nivel de antiguedad del trabajador:\n");

        scanf("%d",&nivel);

        printf("Introduce tu salario:\n");

        scanf("%f",&salario);

        switch (nivel) {

        case 5:

          salario_nuevo=salario+(salario*.035);

        printf("\nTu salario es:%.2f\n",salario_nuevo);

        break;

        case 6:

        salario_nuevo=salario+(salario*.041);
```

```
        printf("\nTu salario es: %.2f\n",salario_nuevo);
        break;
        case 7:
          salario_nuevo=salario+(salario*.048);
        printf("\nTu salario es: %.2f\n",salario_nuevo);
            break;
        case 8:
          salario_nuevo=salario+(salario*.053);
        printf("\nTu salario es: %.2f\n",salario_nuevo);
        break;
        default:
        printf("\nTu salario es:%.2f\n",salario);
}
system("PAUSE");
return 0;
}
```

2. Programa que pida por teclado un número de un día de la semana y muestre por pantalla el nombre correspondiente a dicho día.

```
#include <stdio.h>
int main()
{
  int dia;
  printf( "\n   Introduzca día de la semana en número: " );
  scanf( "%d", &dia );
  switch ( dia )
  {
        case 1 : printf( "\n   Lunes" );
        break;
```

```
        case 2 : printf( "\n   Martes" );
        break;
        case 3 : printf( "\n   Miercoles" );
        break;
        case 4 : printf( "\n   Jueves" );
        break;
        case 5 : printf( "\n   Viernes" );
        break;
        case 6 : printf( "\n   Sabado" );
        break;
        case 7 : printf( "\n   Domingo" );
        break;
  default : printf( "\n   ERROR: Dia incorrecto." );
  }
  return 0;
}
```

3. Programa que realice el cálculo del día del año.

```
#include <stdio.h>
#include <stdlib.h>
 #define MAXDIA 31
#define MAXMES 12
int main() {
        int nDia = 0; int nMes = 0; int diaDelAnyo = 0;
        printf("Calculo del dia del a%co (no bisiesto)\n", 164);
        print ("");
        printf ("Introduzca el numero del dia: ");
        scanf("%d", &nDia);
```

```
printf ("Introduzca el numero del mes: ");
scanf("%d", &nMes);
if ( nDia >= 1 && nDia <= MAXDIA && nMes >= 1 && nMes <= MAXMES) {
switch (nMes) {
case 1: diaDelAnyo = nDia; break;
case 2: diaDelAnyo = nDia + 31; break;
case 3: diaDelAnyo = nDia + 59; break;
case 4: diaDelAnyo = nDia + 90; break;
case 5: diaDelAnyo = nDia + 120; break;
case 6: diaDelAnyo = nDia + 151; break;
case 7: diaDelAnyo = nDia + 181; break;
case 8: diaDelAnyo = nDia + 212; break;
case 9: diaDelAnyo = nDia + 243; break;
case 10: diaDelAnyo = nDia + 273; break;
case 11: diaDelAnyo = nDia + 304; break;
case 12: diaDelAnyo = nDia + 334; break;
default: puts ("Datos no validos");
}
}
if (diaDelAnyo !=0) {
printf ("El %d del %d es el dia %d del a%co\n", nDia, nMes, diaDelAnyo, 164);
} else {
puts ("Los datos no son validos");
}
return 0;
}
```

4. Se quiere escribir un programa que:

1º) Muestre el listado de los signos del zodíaco, con sus números asociados.

2º) Pida por teclado un número (dato entero) asociado a un signo del zodíaco.

3º) Muestre la categoría a la que pertenece el signo del zodíaco seleccionado.

Nota: Si el número introducido por el usuario, no está asociado a ningún signo del zodíaco, se mostrará el mensaje: "ERROR: <número> no está asociado a ningún signo.".

```
#include <stdio.h>

int main()

{

  int numero;

  printf( "\n   Listado de signos del zodiaco:" );

  printf( "\n\n   1. Aries" );

  printf( "\n   2. Tauro" );

  printf( "\n   3. Geminis" );

  printf( "\n   4. Cancer" );

  printf( "\n   5. Leo" );

  printf( "\n   6. Virgo" );

  printf( "\n   7. Libra" );

  printf( "\n   8. Escorpion" );

  printf( "\n   9. Sagitario" );

  printf( "\n   10. Capricornio" );

  printf( "\n   11. Acuario" );

  printf( "\n   12. Piscis" );

  printf( "\n\n   Introduzca numero de signo: " );

  scanf( "%d", &numero );

  switch ( numero )

  {

        case  1 :

        case  5 :

        case  9 : printf( "\n   Es un signo de Fuego." );
```

```
        break;
        case  2 :
        case  6 :
        case 10 : printf( "\n   Es un signo de Tierra." );
        break;
        case  3 :
        case  7 :
        case 11 : printf( "\n   Es un signo de Aire." );
        break;
        case  4 :
        case  8 :
        case 12 : printf( "\n   Es un signo de Agua." );
        break;
        default : printf( "\n   ERROR: %d no está asociado a ningún signo.", numero );
  }
  return 0;
}
```

5. *Realizar un programa con un menú de 4 opciones:*

- ***1. Calcular el doble de un número entero.***
- ***2. Calcular la mitad de un número entero.***
- ***3. Calcular el cuadrado de un número entero.***
- ***4. Salir.***

2º) Pida por teclado la opción deseada (dato entero).

3º) Ejecute la opción del menú seleccionada.

4º) Repita los pasos 1º, 2º y 3º, mientras que, el usuario no seleccione la opción 4 (Salir) del menú.

```
#include <math.h>
```

```
#include <stdio.h>
int main()
{
        int n, opcion;
        do
        {
        printf( "\n   1. Calcular el doble de un n%cmero entero.", 163 );
        printf( "\n   2. Calcular la mitad de un n%cmero entero.", 163 );
        printf( "\n   3. Calcular el cuadrado de un n%cmero entero.", 163 );
        printf( "\n   4. Salir." );
        printf( "\n\n   Introduzca opci%cn (1-4): ", 162 );
        scanf( "%d", &opcion );
        /* Inicio del anidamiento */
        switch ( opcion )
        {
        case 1: printf( "\n   Introduzca un n%cmero entero: ", 163 );
                scanf( "%d", &n );
                printf( "\n   El doble de %d es %d\n\n", n, n * 2 );
                break;
        case 2: printf( "\n   Introduzca un n%cmero entero: ", 163 );
                scanf( "%d", &n );
                printf( "\n   La mitad de %d es %f\n\n", n, ( float ) n / 2 );
                break;
        case 3: printf( "\n   Introduzca un n%cmero entero: ", 163 );
                scanf( "%d", &n );
                printf( "\n   El cuadrado de %d es %d\n\n", n, ( int ) pow( n, 2 ) );
        }
        /* Fin del anidamiento */
```

```
    } while ( opcion != 4 );

    return 0;

}
```

- *Véase que, se han realizado dos castings, (float) y (int), para cambiar, respectivamente, los tipos de datos de los valores resultantes de las expresiones n / 2 y pow(n, 2).*
- *·Por otra parte, fíjese que el bucle do...while iterará, mientras que, opcion sea distinto del valor 4. Normalmente, en un menú, la opción de salir (opción 4 en este caso) no se debe contemplar en la alternativa múltiple, es decir, si el usuario introduce un 4, no se debe hacer nada. Pero, ¿qué ocurre si el usuario teclea un número mayor que 4 ó menor que 1?.*
- *Al introducir un número menor que 1 ó mayor que 4, se muestra de nuevo el menú. Para evitar que ocurra esto, es conveniente utilizar un filtro al leer la opción que introduce el usuario.*

Solución 2: filtrando la opción introducida por el usuario

```
#include <math.h>
#include <stdio.h>
int main()
{
   int n, opcion;
   do
   {
        printf( "\n   1. Calcular el doble de un n%cmero entero.", 163 );
        printf( "\n   2. Calcular la mitad de un n%cmero entero.", 163 );
        printf( "\n   3. Calcular el cuadrado de un n%cmero entero.", 163 );
        printf( "\n   4. Salir." );

        /* Filtramos la opción elegida por el usuario */
        do
        {
        printf( "\n   Introduzca opci%cn (1-4): ", 162 );
        scanf( "%d", &opcion );

        } while ( opcion < 1 || opcion > 4 );
```

```
        /* La opción sólo puede ser 1, 2, 3 ó 4 */

        switch ( opcion )
        {
        case 1: printf( "\n   Introduzca un n%cmero entero: ", 163 );
                scanf( "%d", &n );
                printf( "\n   El doble de %d es %d\n\n", n, n * 2 );
                break;

        case 2: printf( "\n   Introduzca un n%cmero entero: ", 163 );
                scanf( "%d", &n );
                printf( "\n   La mitad de %d es %f\n\n", n, ( float ) n / 2 );
                break;

        case 3: printf( "\n   Introduzca un n%cmero entero: ", 163 );
              scanf( "%d", &n );
                printf( "\n   El cuadrado de %d es %d\n\n", n, ( int ) pow( n, 2 ) );
        }

    } while ( opcion != 4 );
    return 0;
}
```

- La variable opción, también puede ser un dato de tipo carácter, en vez de tipo entero.

Solución 3: declarando la variable opcion de tipo carácter

```
/* Programa: Menú de opciones (Solución 3) */
#include <stdio.h>
#include <math.h>
int main()
{
        char opcion;
        int n;
        do
```

```
    {
    printf( "\n   1. Calcular el doble de un n%cmero entero.", 163 );
    printf( "\n   2. Calcular la mitad de un n%cmero entero.", 163 );
    printf( "\n   3. Calcular el cuadrado de un n%cmero entero.", 163 );
     printf( "\n   4. Salir." );
    do
    {
    printf( "\n   Introduzca opci%cn (1-4): ", 162 );
    fflush( stdin );   // limpieza del buffer
    scanf( "%c", &opcion );
    } while ( opcion < '1' || opcion > '4' );
    switch ( opcion )
    {
    case '1': printf( "\n   Introduzca un n%cmero entero: ", 163 );
              scanf( "%d", &n );
              printf( "\n   El doble de %d es %d\n\n", n, n * 2 );
              break;

    case '2': printf( "\n   Introduzca un n%cmero entero: ", 163 );
              scanf( "%d", &n );
              printf( "\n   La mitad de %d es %f\n\n", n, ( float ) n / 2 );
              break;
    case '3': printf( "\n   Introduzca un n%cmero entero: ", 163 );
              scanf( "%d", &n );
              printf( "\n   El cuadrado de %d es %d\n\n", n, ( int ) pow( n, 2 ) );
    }
    } while ( opcion != '4' );
return 0;
```

```
}
```

6. Programa que pida un número del 1 al 7 y diga el día de la semana correspondiente.

```
 #include <stdio.h>
#include <stdlib.h>
int main()
{
        int i;
        printf("Introduzca numero del 1 al 7:\t"); // \t tabulador – espacio
   scanf("%d",&i);
    switch(i){
          case 1:
              printf ("Lunes\n\n");
              break;
          case 2:
              printf ("Martes\n\n");
              break;
          case 3:
              printf ("Miercoles\n\n");
              break;
          case 4:
              printf ("Jueves\n\n");
              break;
          case 5:
              printf ("Viernes\n\n");
              break;
          case 6:
              printf ("Sabado\n\n");
              break;
```

```
        case 7:
            printf ("Domingo\n\n");
                break;
        default:
            printf ("Opción no valida\n\n");
            break;
        }
   system("PAUSE");
        return 0;
}
```

7. Programa que pida una letra y detecte si es una vocal

```
 #include <stdio.h>
#include <stdlib.h>
 int main()
{
        char c;
        printf("Introduzca un carácter:\t");
          scanf("%c",&c);
        switch (c)
        {
       case 'a':
           printf ("Es vocal\n\n");
           break;
       case 'e':
             printf ("Es vocal\n\n");
           break;
       case 'i':
```

```
            printf ("Es vocal\n\n");
            break;
        case 'o':
            printf ("Es vocal\n\n");
            break;
        case 'u':
              printf ("Es vocal\n\n");
            break;
        default:
            printf ("No es vocal\n\n");
            break;
          }
   system("PAUSE");
          return 0;
}
```

8. Programa que muestre un menú donde las opciones sean "Equilátero", "Isósceles" y "Escaleno", pida una opción y calcule el perímetro del triángulo seleccionado.

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
          int lado, base, opcion;
          printf("Introduzca lado del triangulo:\t");
          scanf("%d",&lado);
           printf("Introduzca base del triangulo:\t");
           scanf("%d",&base);
           printf ("\n") ;
           printf("Seleccione opcion:\n\n");
           printf("1 - Equilatero\n");
```

```
        printf("2 - Isosceles\n");
        printf("3 - Escaleno\n");
        scanf("%d",&opcion);
switch (opcion)
{
    case 1:
        printf("El perímetro es:%d\n",3*lado);
        break;
    case 2:
        printf("El perímetro es:%d\n",(2*lado)+base);
        break;
    case 3:
        printf("El perímetro es:%d\n",lado + lado + lado);
        break;
    default:
    printf("Opcion no valida.");
    break;
}
system("PAUSE");
return 0;
}
```

8. BUCLES

Los bucles son estructuras que permiten ejecutar partes del código de forma repetida mientras se cumpla una condición.

Esta condición puede ser simple o compuesta de otras condiciones unidas por operadores lógicos.

8.1 Sentencia WHILE

Su sintaxis es:

while (condición) sentencia;

Con esta sentencia se controla la condición antes de entrar en el bucle. Si ésta no se cumple, el programa no entrará en el bucle.

Naturalmente, si en el interior del bucle hay más de una sentencia, éstas deberán ir entre llaves para que se ejecuten como un bloque.

```
/* Uso de la sentencia WHILE. */
#include <stdio.h>
main() /* Escribe los números del 1 al 10 */
{
  int numero=1;
  while(numero<=10)
  {
        printf("%d\n",numero);
        numero++;
  }
}
```

La diferencia básica entre el bucle while y el bucle do…while, es que en el segundo se ejecutan como mínimo una vez las acciones contenidas dentro del bucle.

1. *Programa que visualice los números del 1 al 10.*

```
// Librerias a incluir

#include <stdio.h>

#include <stdlib.h>

// Funcion principal

int main()
```

```
{
   // Crear variables auxiliares
   int contador=0;
   // Inicializar el contador
   contador=1;
   // repetir mientras que el contador tenga
   // un valor inferior a 10
   while (contador<11) {
      // visualizar el valor de contador
      printf("\n%d",contador);
      // incrementar contador en una unidad
      contador=contador+1;
   };
      return 0;
}
```

2. Que calcule el factorial de un número cualquiera y lo muestre en pantalla.

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
  int num, num2;
  printf("Introduce número: ");
  scanf("%d",&num);
  num2=num;
```

```
while (num2!=1){
        num2=num2-1;
        num=num*num2;
}
printf("El factorial es: %d \n", num);
system("PAUSE");
return 0;
}
```

3. *Que calcule la media de X números, se dejarán de solicitar números hasta que se introduzca el cero.*

```
#include <stdio.h>
#include <stdlib.h>
#include <clocale>  // libreria para que reconozca las tildes y signos
int main(void)
{
        setlocale(LC_CTYPE,"Spanish"); // función para que reconozca las tildes y signos
   int num=1,cont=0;
   float sum=0;
   while (num!=0)
   {
      printf("Introduzca número:  ");
      scanf("%d",&num);
      sum=sum+num;
      if (num!=0)
      {
        cont=cont+1;
      }
        }
```

```
  printf("La media es:%6.2f\n",sum/cont);

 system("PAUSE");

   return 0;

}
```

4. *Programa que genere los números del 10 al 40*

```
#include <stdio.h>
main() {

        int i=0, j=0, final=40;
        while(i<final) {

i=j*10;
printf("%d\n",i);
j++;}
j=100; /* sentencia que ejecutamos cuando deje de cumplirse la condición del while. */

}
```

5. *Una compañía de seguros tiene contratados a N vendedores. Cada uno hace tres ventas a la semana. Su política de pagos es que un vendedor recibe un sueldo base, y un 10% extra por comisiones de sus ventas. El gerente de su compañía desea saber cuánto dinero obtendrá en la semana cada vendedor por concepto de comisiones por las tres ventas realizadas, y cuanto tomando en cuenta su sueldo base y sus comisiones.*

```
#include <stdio.h>

#include <math.h>

#include <conio.h>

#include <stdlib.h>

int main() {

    int sueldo,r1,t,c1,c2,c3,v1,v2,v3,sueldot,ct;

   printf("*ANALIZADOR DE SUELDOS DE LOS VENDEDORES DE SEGUROS*\n\n");

   r1=1;

            while(r1<=1){

                t=t+1;
```

```
            printf("\n\nANALIZADOR DEL VENDEDOR %d \n\n",t);
        printf("Escriba el sueldo base del vendedor %d \n",t);
        scanf("%d",&sueldo);
            printf("\nAnalisis de las ventas del vendedor %d \n\n ",t);
            printf("Digite el valor de la venta 1\n");
        scanf("%d",&v1);
        c1=v1*0.1;
            printf("\nEl valor de la comision 1 es de %d \n\n",c1);
            printf("Digite el valor de la venta 2\n");
        scanf("%d",&v2);
        c2=v2*0.1;
            printf("\nEl valor de la comision 2 es de %d\n\n",c2);
            printf("Digite el valor de la venta 3\n");
        scanf("%d", &v3);
        c3=v3*0.1;
            printf("\nEl valor de la comision 3 es de %d\n\n",c3);
        ct=c1+c2+c3;
            printf("\n\nLa ganancia en comisión del vendedor %d es de %d
\n",t,ct);
        sueldot=ct+sueldo;
            printf("\n\nEl sueldo total del vendedor %d es de %d\n ",t,sueldot);
            printf("\n\nHay otro vendedor?\n\n");
            printf("1. Si\n");
            printf("2. No\n");

        scanf("%d",&r1);
        printf("\n\n\n");

        }
```

```
                printf("Gracias por utilizar el programa");
                getch();
        return 0;
}
```

6. *Adivinar el número.*

```
#include <stdio.h>
int main(int argc, char *argv[])
{
        int num, num2, opc=0;
    printf("\n Adivinar numero");
        printf("\n 1 - Comenzar.");
        printf("\n 2 - Salir.\n");
        printf("\n Introduce una opcion:");
  scanf("%d",&opc);
    while (opc!=2)
        {
        num = rand() % 100;//Origina aleatoriamente numeros entre 0 y 99

     printf("\n Introduce numero: ");
        scanf("%d",&num2);

        while(num!=num2)
        {
        if (num>num2)
         printf("Es mayor");
        else
        printf("Es menor");

        printf("\n Introduce numero: ");
        scanf("%d",&num2);
        }
```

```
            printf("\n Has acertado! \n");
            printf("\n 1 - Jugar de nuevo.");
            printf("\n 2 - Salir.");
            printf("\n Introduce una opcion:");
            scanf("%d",&opc);
            }
            system("PAUSE");
            return 0;
      }
```

8.2 Sentencia DO...WHILE

Su sintaxis es:

do{

sentencia1;

sentencia2;

}while (condición);

Con esta sentencia se controla la condición al final del bucle. Si ésta se cumple, el programa vuelve a ejecutar las sentencias del bucle.

La única diferencia entre las sentencias while y do...while es que con la segunda el cuerpo del bucle se ejecutará por lo menos una vez.

```
/* Uso de la sentencia DO...WHILE. */
 #include <stdio.h>
 main() /* Muestra un menú si no se pulsa 4 */
{
   char seleccion;
   do{
         printf("1.- Comenzar\n");
```

```
            printf("2.- Abrir\n");
            printf("3.- Grabar\n");
            printf("4.- Salir\n");
            printf("Escoge una opción: ");
            seleccion=getchar();
            switch(seleccion){
             case '1':printf("Opción 1");
                      break;
             case '2':printf("Opción 2");
                      break;
             case '3':printf("Opción 3");
            }
    }while(seleccion!='4');
}
```

EJERCICIOS

1. Un proveedor de estéreos ofrece un descuento del 10% sobre el precio sin IVA, de algún aparato si este cuesta $200.000 o más. Además, independientemente de esto, ofrece un 5% de descuento si la marca es "NOSY". Determinar cuanto pagara, con IVA incluido, un cliente cualquiera por la compra de su aparato.

Ejercicio con el repetir (do… while)

```
#include <stdio.h>
#include <math.h>
#include <string.h>  // cadena de caracteres
int main()
{
float producto, valor_sin_iva, descuento, total, valor_apagar1, valor_apagar, descuento_marca, iva,
descuento_marc;
char marca[]="nosy";
char resp[2];
do {
printf("introduzca el valor del producto:\n");
scanf("%f", &producto);
printf("digite la marca del producto:\n");
scanf("%s",marca);
iva=(producto*19)/100 ;
```

```
printf("valor del iva es :%.2f\n",iva);
valor_sin_iva=(producto-iva) ;
printf("valor del producto sin iva:%.2f\n",valor_sin_iva);
if (producto>=200000)  {
descuento=(valor_sin_iva*10/100);
printf("el descuento del 10 por ciento del estereo es %.2f\n",descuento);
total=producto-descuento;
printf("el valor a pagar es de: %.2f\n",total);
} else{
printf("el producto no tiene descuento del 10 por ciento %.2f\n",producto);
}if (strcmp (marca,"nosy")== 0 ){
descuento_marca=(valor_sin_iva*5)/100 ;
printf("El descuento por la marca nosy es:%.2f\n",descuento_marca);
valor_apagar=(producto-descuento-descuento_marca);
printf("el valor a pagar con el descuento por ser marca nosy es %.2f\n", valor_apagar);
}
printf ("\nSi desea repetir escriba -si-, si desea terminar escriba -no-\n");
scanf("%s",& resp);
} while (strcmp (resp,"si") == 0);
}
```

2. En un juego de preguntas a las que se responde "Si" o "No" gana quien responda correctamente las tres preguntas. Si se responde mal a cualquiera de ellas ya no se pregunta la siguiente y termina el juego. Las preguntas son:

1. *Colón descubrió América?*

2. *La independencia de México fue en el año 1810?*

3. *The Doors fue un grupo de rock Americano?*

```
#include <stdio.h>
#include <string.h>
int main () {
char resp1[2], resp2[2], resp3[2], resp4[2];
do {
printf ("\nBienvenido, para poder pasar debe responder correctamente las 3 preguntas. Para
responder escriba -si- o -no- segun corresponda y luego pulse enter\n");
printf ("1. Colon descubrio America?\n");
scanf ("%s", &resp1);
if (strcmp(resp1, "no") == 0 ){
printf ("\nUsted perdio\n");
printf ("\nSi desea repetir la pegunta escriba -Si-, si desea terminar escriba -No-\n");
scanf("%s",& resp4);
} else if (strcmp(resp1,"si") == 0 ){
printf ("\nCorrecto.\n 2. La independencia de Mexico fue en 1810?\n");
scanf ("%s", &resp2);
if (strcmp(resp2,"no") == 0) {
printf ("\nUsted perdio\n");
printf ("\nSi desea repetir la pregunta escriba -Si-, si desea terminar escriba -No-\n");
scanf("%s",& resp4);
} else if (strcmp(resp2,"si") == 0) {
printf ("\nCorrecto.\n 3. The Doors fue un grupo de rock americano?\n");
scanf ("%s",& resp3);
if (strcmp(resp3,"no") == 0) {
```

```
printf ("\nUsted perdio");
printf ("\nSi desea repetir la pregunta escriba -Si-, si desea terminar escriba -No-\n");
scanf("%s",& resp4);
} else if (strcmp (resp3,"si") == 0) {
printf ("\nFelicidades. Usted gano\n");
}
printf ("\nSi desea repetir la pregunta escriba -si-, si desea terminar escriba -no-\n");
scanf("%s",& resp4);
}
}
} while (strcmp (resp4,"si") == 0);
}
```

3. Tabla de multiplicar de cualquier número

```
#include <stdio.h>

#include <math.h>

#include <string.h>

int main (){

        int multiplicando,multiplicador, producto;

        printf("Ingrese el numero al que se le va a calcular su tabla de multiplicar\n\n");

        scanf("%i", &multiplicador);

        printf("MULTIPLICANDO \t    MULTIPLICADOR      \t PRODUCTO");

        multiplicando=1;

        do{

                producto=multiplicando*multiplicador;

printf("\n\n\t %i \t x \t  %i  \t  = \t  %i",multiplicando,multiplicador,producto);

        multiplicando=multiplicando+1;

        }  while (multiplicando<=10);

return 0;

        }
```

4. Hacer un programa que al ingresar tres números diferentes imprima el número medio.

```
#include <stdio.h>

#include <string.h>

#include <stdlib.h> //system (“cls”)
```

```
int main (){
int numero1,numero2,numero3;
char resp[2];
do{
system("cls");   // limpiar pantalla
printf("Ingresar tres numeros distintos:\n");
scanf("%d",&numero1);
scanf("%d",&numero2);
scanf("%d",&numero3);
        if (numero1<numero2 && numero1>numero3){
        printf("El numero intermedio entre ellos:%d\n",numero1);
        } else if (numero2<numero3 && numero2>numero1){
        printf("El numero intermedio entre ellos:%d\n",numero2);
        } else if (numero3<numero2 && numero3>numero1){
        printf("El numero intermedio es el:%d\n",numero3);
        } printf("Desea repitir el procedimiento, digite si o no: \n");
        scanf("%s",resp);
} while(strcmp(resp,"si")==0);
}
```

5. Imprimir los números del 1 al 10 cada uno con su respectivo factorial.

```
#include <stdio.h>
#include <math.h>
#include <conio.h> //    getch()
#include <stdlib.h>
int main() {
        int nun1 = 1,factorial;
        printf("Tabla de factoriales   \n ");
        printf("NUMERO                  FACTORIAL \n\n");
```

```
   do{
                printf(" %d                    %d \n",nun1,factorial);
                nun1=nun1+1;
                factorial=factorial*nun1;
        }  while( nun1 <= 10 );
        getch();
        return 0;
}
```

6. En un supermercado un cajero captura los precios de los artículos que los clientes compran e indica a cada cliente cual es el monto de lo que deben pagar. Al final del día le indica a su supervisor cuanto fue lo que cobró en total a todos los clientes que pasaron por su caja.

```
#include<stdio.h>
#include<string.h> //cadena de caracteres
int main () {
float precio,contador,cantidad,valortot;
char reps[2];
contador=0;
do{
printf("diga el precio del producto.\n");
scanf("%f",&precio);
printf("digite el n%cmero de productos.\n",163);
scanf("%f",&cantidad);
valortot=(precio*cantidad);
contador=contador+valortot;
printf("%cdesea ingresar otro producto?.\n",168);  //%c y el número 168 imprimir el signo de interrogación de apertura
scanf("%s",&reps);
}while (strcmp(reps,"si")==0 || (reps,"Si")==0 || (reps,"SI")==0);
printf("el total de la compra es:%2.f",contador);
}
```

7. En una tienda de descuento se efectúa una promoción en la cual se hace un descuento sobre el valor de la compra total según el color de la bolita que el cliente saque al pagar en caja. Si la bolita es de color blanco no se le hará descuento alguno, si es verde se le hará un 10% de descuento, si es amarilla un 25%, si es azul un 50% y si es roja un 100%. Determinar la cantidad final que el cliente deberá pagar por su compra. Se sabe que solo hay bolitas de los colores mencionados.

```
#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <conio.h>

#include <clocale>


int main(){

		setlocale(LC_CTYPE,"Spanish");

		system ("color 0A");

		float compra, descuento, total_compra, acumulador;

		int contador_amarilla, contador_azul, contador_verde,
contador_blanca,contador_roja;

		char bolita[15];

		char respuesta [2];

		contador_blanca=0;

		contador_verde=0;

		contador_amarilla=0;

		contador_azul=0;

		contador_roja=0;

		acumulador=0;

	do{

	printf("digite el valor de su compra\n");

	scanf ("%f",&compra);

	printf ("digite el color de la bolita\n");

	scanf ("%s",&bolita);
```

```
        if  (strcmp(bolita,"blanca")==0) {
                total_compra=compra;
                printf ("el total a pagar es de:%.2f \n", total_compra, " no tiene descuento por ser
blanca \n");
                contador_blanca=contador_blanca+1;
                acumulador=acumulador+total_compra;
                printf ("el acumulador de las compras es:%.2f\n ", acumulador);
        }else
                if      (strcmp(bolita,"verde")==0) {
                        descuento=(compra*10)/100;
                        printf(" su descuento por elegir la bolita de color verde es del 10 %.2f \n",
descuento);
                        total_compra=compra-descuento;
                        printf(" su total a pagar es de :%.2f \n  ", total_compra);
                        contador_verde=contador_verde+1;
                        acumulador=acumulador+total_compra;
                        printf(" el acumulador de las compras es:%.2f \n ", acumulador);
                }else
                        if  (strcmp(bolita,"amarilla")==0) {
                                descuento=(compra*25)/100;
                                printf(" su descuento por elegir la bolita de color amarilla es del
25:%.2f \n ", descuento);
                                total_compra=compra-descuento;
                                printf(" su total a pagar es de : %.2f \n", total_compra);
                                contador_amarilla=contador_amarilla+1;
                                acumulador=acumulador+total_compra;
                                printf ("el acumulador de las compras es:%.2f \n ", acumulador);
                        }else
                                if  (strcmp(bolita,"azul")==0) {
```

```
                                        descuento=(compra*50)/100;
                                        printf (" su descuento por elegir la bolita de color azul es del
50:%.2f\n ", descuento);
                                        total_compra=compra-descuento;
                                        printf (" su total a pagar es de :%.2f\n  ", total_compra);
                                        contador_azul=contador_azul+1;
                                        acumulador=acumulador+total_compra;
                                        printf(" el acumulador de las compras es:%.2f \n ",
acumulador);
                                } else
                                        if  (strcmp(bolita,"roja")==0) {
                                                descuento=(compra*100)/100;
                                                printf (" su descuento por elegir la bolita de color
roja es del 100: %.2f \n",descuento);
                                                total_compra=compra-descuento;
                                                printf (" su total a pagar es de :%.2f \n",
total_compra);
                                                contador_roja=contador_roja+1;
                                                acumulador=acumulador+total_compra;
                                                printf(" el acumulador de las compras es: %.2f \n",
acumulador);
                                        }

        printf (" el total de las bolitas blancas son: %i \n", contador_blanca);
        printf (" el total de las bolitas verdes son :%i \n", contador_verde);
        printf (" el total de las bolitas amarillas son:%i \n", contador_amarilla);
        printf (" el total de las bolitas azul son:%i \n", contador_azul);
        printf (" el total de las bolitas rojas son:% i \n", contador_roja);

        printf ("Desea ingresar otra compra \n");
        scanf ("%s", respuesta);
```

```
} while ((strcmp(respuesta,"Si")==0) || (strcmp(respuesta,"SI")==0) || (strcmp(respuesta,"si")==0));

return 0;

}
```

8.3 Sentencia FOR

Su sintaxis es:

```
for (inicialización;condición;incremento){

        sentencia1;

        sentencia2;

}
```

La inicialización indica una variable (variable de control) que condiciona la repetición del bucle. Si hay más, van separadas por comas:

```
for (a=1,b=100;a!=b;a++,b- -){
```

El flujo del bucle FOR transcurre de la siguiente forma:

```
/* Uso de la sentencia FOR. */
 #include <stdio.h>
 main() /* Escribe la tabla de multiplicar */
{
   int num,x,result;
   printf("Introduce un número: ");
   scanf("%d",&num);
   for (x=0;x<=10;x++){
      result=num*x;
      printf("\n%d por %d = %d\n",num,x,result);
   }
}
```

Ejercicios

1. ***Programa que muestra los veinte primeros números naturales.***

```
#include <stdio.h>
int main(){
        int i=21;
        const int tope=20;
        for(i=0;i<=20;i=i+1){
                printf("%d\n",i);
        }
        printf("\nHasta pronto");
```

```
}
```

2. Programa que muestra los números pares hasta 30.

```
 #include <stdio.h>
int main(){
        int i=31;
        const int tope=30;
        for(i=0;i<=30;i=i+2){
                printf("%d\n",i);
        }
        printf("\nHasta pronto");
}
```

3. Programa que muestre los múltiplos de siete (hasta 123).

```
 #include <stdio.h>
int main(){
        int i=124;
        const int tope=123;
        for(i=0;i<=123;i=i+7){
        printf("%d\n",i);
        }
        printf("\n,Hasta pronto");
}
```

4. Programa que muestre una cuenta atrás desde diez hasta cero.

```
 #include <stdio.h>
int main(){
        int i=10;
        const int tope=0;
```

```
        for(i=10;i>=0;i=i-1){
                printf("%d\n",i);
        }
        printf("\nHasta pronto!\n");
}
```

5. Programa que muestre los números impares que haya del 1 al 100

```
 #include <stdio.h>
#include <stdlib.h>
int main()
{
          int x;
          for (x=1;x<=100;x++)
           {
           if (x%2!=0)
           {
          printf("%d\n",x);
           }
           }
           system("PAUSE");
           return 0;
}
```

6. Programa que escriba las tablas de multiplicar del 0 al 10.

```
 #include <stdio.h>
#include <stdlib.h>
int main(void)
{
```

```
        int x,y;
        for (x=0;x<=10;x++)
        {
        for (y=1;y<=10;y++)
        {
        printf("%d X %d = %d \n",x,y,x*y);
        }
        printf("\n");
        }
        printf("\n");
        system("PAUSE");
        return 0;
}
```

7. Programa que muestre la tabla de multiplicar de un número cualquiera.

```
 #include <stdlib.h>
int main(void)
{
        int x,num;
                printf("Introduce numero:\t");
                scanf("%d",&num);
        printf("\n");
        for (x=1;x<=10;x++)
        {
        printf("%d X %d = %d \n",num,x,num*x);
        }
        printf("\n");
        system("PAUSE");
```

```
        return 0;
}
```

8. Programa que calcule "x" términos de la sucesión de Fibonacci.

```
 #include <stdio.h>
#include <stdlib.h>
int main(){
      printf("Bienvenido al programa para calcular la progresión de Fibonacci.\n\n");
      int veces, primer=0,segun=1,proximo,r;
      char borrado;
      printf("Introduzca el numero de terminos: ");
   scanf("%d",&veces);
      scanf("%c",&borrado);
      system("cls");
      printf("He aqui la sucesion de %d terminos: \n",veces);
      for(int i=0;i<=veces;i++){
        r=primer+segun;
              primer=segun;
              segun=r;
          printf("\n\t\t\t\t%d",r);
     }
        printf("\n\nGracias por utilizar este programa.\n\n");
}
```

9. Programa que calcule el factorial de un número.

```
#include <stdio.h>
```

```
int main(){
	int i,num,fact=1;
	printf("Bienvenido al programa para calcular factoriales.\n");
		printf("\nEscriba un numero entero: ");
		scanf("%d",&num);
		for(i=num;i>1; i--){
		 fact=fact*i;
		}
		printf("\nEl factorial de %d es %d\n",num,fact);
}
```

10. Programa que genera apuestas de fútbol.

```
#include <stdlib.h>
#include <time.h>
int main(){
	int a;
	srand((unsigned)time(NULL));
	printf("Bienvenido, aquí tiene su apuesta de futbol: ");
	printf("\n\n");
	for(int i=1; i<=15; i++){
		a=rand()%(3);
		if(a==1){
			printf("\t\t\t\t%d - 1\n",i);
		}else if(a==2){
			printf("\t\t\t\t%d - 2\n",i);
		}else{
			printf("\t\t\t\t%d - X\n",i);
		}
```

```
    }
}
```

11. Programa que simule el lanzamiento de una moneda las veces que el usuario desee, posteriormente hará un recuento de las veces que ha salido tanto cara como cruz.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
int main(){
    int x,veces,cara=0,cruz=0;
    srand((unsigned)time(NULL));
    printf("Pruebe a lanzar la moneda.\n\n");
    printf("Cuantas veces?: ");
    scanf("%d",&veces);
    for(int i=1;i<=veces;i++){
        x=rand()%(2);
        if(x==1){
            printf("\nCara\n\n");
            cara++;
        }else{
            printf("\nCruz\n\n");
            cruz++;
        }
    }
    printf("\n\tRecuento\n\n");
    printf("La cara ha salido %d veces.\n",cara);
    printf("La cruz ha salido %d veces.\n\n",cruz);
    printf("Gracias por utilizar este programa.\n\n");
}
```

9. ARREGLOS

*Un array es un identificador que referencia un conjunto de datos del mismo tipo. Imagina un tipo de dato **int**; podremos crear un conjunto de datos de ese tipo y utilizar uno u otro con sólo cambiar el índice que lo referencia. El índice será un valor entero y positivo. En **C** los arrays comienzan por la posición **0**.*

9.1 VECTORES - UNIDIMENSIONALES

Un vector es un array unidimensional, es decir, sólo utiliza un índice para referenciar a cada uno de los elementos. Su declaración será:

tipo nombre [tamaño];

*El tipo puede ser cualquiera de los ya conocidos y el tamaño indica el número de elementos del vector (se debe indicar entre corchetes **[ ]**). En el ejemplo puedes observar que la variable **i** es utilizada como índice, el primer **for** sirve para rellenar el vector y el segundo para visualizarlo. Como ves, las posiciones van de **0** a **9** (total 10 elementos).*

```
/* Declaración de un array. */

#include <stdio.h>

main() /* Rellenamos del 0 - 9 */
{
        int vector[10],i;
        for (i=0;i<10;i++) vector[i]=i;
        for (i=0;i<10;i++) printf(" %d",vector[i]);
}
```

Podemos inicializar (asignarle valores) un vector en el momento de declararlo. Si lo hacemos así no es necesario indicar el tamaño. Su sintaxis es:

tipo nombre []={ valor 1, valor 2...}

Ejemplos:

```
int vector[]={1,2,3,4,5,6,7,8};
char vector[]="programador";
char vector[]={'p','r','o','g','r','a','m','a','d','o','r'};
```

*Una particularidad con los vectores de tipo **char** (cadena de caracteres), es que deberemos indicar en qué elemento se encuentra el fin de la cadena mediante el carácter nulo (**\0**). Esto no lo controla el compilador, y tendremos que ser nosotros los que insertemos este carácter al final de la cadena.*

*Por tanto, en un vector de 10 elementos de tipo **char** podremos rellenar un máximo de 9, es decir, hasta **vector[8]**. Si sólo rellenamos los 5 primeros, hasta **vector[4]**, debemos asignar el caracter nulo a **vector[5]**. Es muy sencillo: **vector[5]='\0';** .*

*Ahora veremos un ejemplo de cómo se rellena un vector de tipo **char**.*

```
/* Vector de tipo char. */

#include <stdio.h>
```

```
main() /* Rellenamos un vector char */
{
        char cadena[20];
        int i;
        for (i=0;i<19 && cadena[i-1]!=13;i++)
                cadena[i]=getche( );
        if (i==19) cadena[i]='\0';
        else cadena[i-1]='\0';
        printf("\n%s",cadena);
}
```

*Podemos ver que en el **for** se encuentran dos condiciones:*

***1.-** Que no se hayan rellenado todos los elementos (**i<19**).*

***2.-** Que el usuario no haya pulsado la tecla ENTER, cuyo código ASCII es **13**. (**cadena[x-i]!=13**).*

*También podemos observar una nueva función llamada **getche( )**, que se encuentra en **conio.h**. Esta función permite la entrada de un caracter por teclado. Después se encuentra un **if**, que comprueba si se ha rellenado todo el vector. Si es cierto, coloca el caracter nulo en el elemento nº20 (**cadena[19]**). En caso contrario tenemos el **else**, que asigna el caracter nulo al elemento que almacenó el caracter ENTER.*

En resumen: al declarar una cadena deberemos reservar una posición más que la longitud que queremos que tenga dicha cadena.

.- Llamadas a funciones con arrays

Como ya se comentó en el tema anterior, los arrays únicamente pueden ser enviados a una función por referencia. Para ello deberemos enviar la dirección de memoria del primer elemento del array. Por tanto, el argumento de la función deberá ser un puntero.

```
/* Envío de un array a una función. */

#include <stdio.h>

void visualizar(int []); /* prototipo */
main() /* rellenamos y visualizamos */
{
        int array[25],i;
        for (i=0;i<25;i++)
        {
                printf("Elemento nº %d",i+1);
                scanf("%d",&array[i]);
        }
        visualizar(&array[0]);
}

void visualizar(int array[]) /* desarrollo */
{
```

```
        int i;
        for (i=0;i<25;i++) printf("%d",array[i]);
}
```

En el ejemplo se puede apreciar la forma de enviar un array por referencia. La función se podía haber declarado de otra manera, aunque funciona exactamente igual:

declaración o prototipo

void visualizar(int *);

desarrollo de la función

void visualizar(int *array)

EJERCICIOS

1. *Que lea 10 números por teclado, los almacene en un vector y muestre la suma, resta, multiplicación y división de todos.*

```
#include <stdio.h>
#include <clocale>
#include <stdlib.h>

int main(){
                    setlocale(LC_CTYPE,"Spanish");
    int x,tabla[10];
    int sum,res,mul,div;

    for (x=0;x<10;x++)
    {
  printf("Introduzca un número\n");
  scanf("%d",&tabla[x]);

}

    sum=tabla[0];

res=tabla[0];

mul=tabla[0];

div=tabla[0];

    for (x=1;x<10;x++)

    {

  sum=sum+tabla[x];

  res=res-tabla[x];

  mul=mul*tabla[x];

  div=div/tabla[x];
```

```
    }

        printf("Suma: %d\n",sum);

        printf("Resta: %d\n",res);

        printf("Multiplicación: %d\n",mul);

        printf("División: %d\n",div);

    system("PAUSE");

    return 0;

}
```

2. *Que rellene un vector de dos dimensiones con números pares, que pida una posición X,Y y mostrar el número correspondiente.*

```
#include <stdio.h>
#include <clocale>
#include <stdlib.h>

int main(int argc, char *argv[])

{
    setlocale(LC_CTYPE,"Spanish");
    int x,y,num=2, numeros[3][3];

    for (x=0;x<3;x++)
    {
       for (y=0;y<3;y++)

       {
          numeros[x][y]=num;

          num=num*2;

       }

    }

    printf("Introduzca coordenada x: ");
    scanf("%d",&x);
    printf("Introduzca coordenada y: ");
    scanf("%d",&y);
    printf("El número es: %d\n",numeros[x][y]);
    system("PAUSE");
    return 0;
}
```

3. *Vamos a crear un programa que lea un vector de 10 posiciones, luego determine si la quinta posición es positiva, si la primera posición es negativa y si la ultima posición es cero.(*)*

```
#include <stdio.h>
   #include <conio.h>
   #define N 10
   main()
   {
      float  x[N];
      int  i;

      for(i=0;  i<N; i++) {
         printf("Ingrese el valor %d:\n", i);
         scanf("%f",  &x[i]);
      }
      if(x[ 4]> 0)
         printf( "La  quinta Posición es Positiva\n\n");

       if(x[ 0]< 0)
         printf( "La 1era Posición es  Negativo\n\n");

       if(x[N- 1]== 0)
         printf( "La última Posición es  Cero\n\n");

      getch();
   return  0;
   }
```

4. Desarrollar el código C para un programa que calcule la superficie de un terreno que le corresponde a un heredero después de n generaciones, partiendo de una superficie inicial en la generación cero. Se supone que hay división a partes iguales entre herederos y que el número máximo de generaciones con que trabajará el programa es 50. Los datos de superficie inicial, número de generaciones y número de herederos por generación se deben solicitar al usuario del programa.

```
#include <stdio.h>

#include <stdlib.h>

#define MAXGENERACIONES 50

int main() {

          int hGen[MAXGENERACIONES]; int n = 0; int i=0;

          double supin = 0.0; double toca = 0.0;

   printf("***Calculo superficie herederos***\n\n");

   printf("Indique el numero de generaciones: ");

   scanf("%d", &n);

   printf("Indique la superficie inicial: ");

   scanf("%lf", &supin);
```

```
        toca = supin;

        for (i=1; i<=n; i++) {

    printf("Indique el numero de herederos de la generacion %d: ", i);

    scanf("%d", &hGen[i]);

        toca = toca/hGen[i];

        }

        printf("Al heredero actual le corresponde una superficie de %.2lf m2\n", toca);

        return 0;

}
```

5. Que rellene un array con los 100 primeros números enteros y los muestre en pantalla en orden ascendente.

```
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
        int x,tabla[100];

        for (x=1;x<=100;x++)
        {
        tabla[x]=x;
  }

        for (x=1;x<=100;x++)
        {
        printf("%d\n",tabla[x]);
  }

        system("PAUSE");
        return 0;
}

}
```

7. Que rellene un array con los números primos comprendidos entre 1 y 100 y los muestre en pantalla en orden ascendente.

```
#include <stdio.h>
#include <stdlib.h>

int main(void)14
{
        int x,cont,z,i,tabla[100];
```

```
	i=0;
	for (x=1;x<=100;x++)
	{
	cont=0;
	 for (z=1;z<=x;z++)
	{
	if (x%z==0)
	{
	cont++;
	}
	}

	if (cont==2 || z==1 || z==0)
	{
	tabla[i]=x;
	i++;
	}

	}

	for (x=0;x<i;x++)
	 {
	printf("%d\n",tabla[x]);
   }

	system("PAUSE");
	return 0;
}
```

8. Que rellene un array con los números pares comprendidos entre 1 y 100 y los muestre en pantalla en orden ascendente.

```
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
	int x,cont,z,i,tabla[100];

	i=0;
	for (x=1;x<=100;x++)
	{
	cont=0;
	if (x%2==0)
	{
```

```
        tabla[i]=x;
        i++;
        }
        }

        for (x=0;x<i;x++)
         {
        printf("%d\n",tabla[x]);
   }

        system("PAUSE");
        return 0;
}
```

9. Que rellene un array con los números impares comprendidos entre 1 y 100 y los muestre en pantalla en orden ascendente.

```
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
        int x,cont,z,i,tabla[100];

        i=0;
        for (x=1;x<=100;x++)
        {
        cont=0;
        if (x%2==1)
        {
        tabla[i]=x;
        i++;
        }
        }

        for (x=0;x<i;x++)
         {
        printf("%d\n",tabla[x]);
   }

        system("PAUSE");
        return 0;
}
```

10. Que lea 10 números por teclado, los almacene en un array y muestre la suma, resta, multiplicación y división de todos.

```
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
        int x,tabla[10];
        int sum,res,mul,div;

        for (x=0;x<10;x++)
         {
        printf("Introduzca número\n");
        scanf("%d",&tabla[x]);
        }

        sum=tabla[0];
        res=tabla[0];
        mul=tabla[0];
        div=tabla[0];

         for (x=1;x<10;x++)
        {
        sum=sum+tabla[x];
      res=res-tabla[x];
        mul=mul*tabla[x];
        div=div/tabla[x];
        }

        printf("Suma: %d\n",sum);
        printf("Resta: %d\n",res);
        printf("Multiplicación: %d\n",mul);
        printf("División: %d\n",div);

   system("PAUSE");
        return 0;
}
```

11. Que lea 10 números por teclado, los almacene en un array y los ordene de forma ascendente.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
  float aux, numeros[10];
```

```
 int i,j,n=10;

 for (i=0;i<n;i++){
                printf("Escriba un número");
                scanf("%f",&numeros[i]);
        }

 for(i=0;i<n-1;i++)
 {
  for(j=i+1;j<n;j++)
  {
        if(numeros[i]<numeros[j])
        {
        aux=numeros[i];
        numeros[i]=numeros[j];
        numeros[j]=aux;
     }
  }
 }

 for (i=n-1;i>=0;i--){
        printf("%f\n",numeros[i]);
 }

 system("PAUSE");
 return 0;
}
```

12. Que lea 10 números por teclado, 5 para un array y 5 para otro array distinto. Mostrar los 10 números en pantalla mediante un solo array.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
 int aux, numeros1[5],numeros2[5],numeros3[10];
 int i,j;

 for (i=0;i<5;i++){
              printf("Escriba un número");
                scanf("%d",&numeros1[i]);
        }

 for (i=0;i<5;i++){
                printf("Escriba un número");
```

```
            scanf("%d",&numeros2[i]);
        }


 for(i=0;i<5;i++)
 {
  numeros3[i]=numeros1[i];
 }

 for(i=0;i<5;i++)
 {
  numeros3[5+i]=numeros2[i];
 }
```

6. Que rellene un array con los 100 primeros números enteros y los muestre en pantalla en orden descendente.

```
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
        int x,tabla[100];

        for (x=1;x<=100;x++)
        {
        tabla[x]=x;
   }

        for (x=100;x>=1;x--)
        {
        printf("%d\n",tabla[x]);
   }

        system("PAUSE");
        return 0;

  for (i=0;i<10;i++){
        printf("%d\n",numeros3[i]);
 }

 system("PAUSE");
 return 0;
}
```

13. Que lea 5 números por teclado, los copie a otro array multiplicados por 2 y muestre el segundo array.

```
#include <stdio.h>
#include <stdlib.h>

int main()
```

```
{
  int aux, numeros1[5],numeros2[5];
  int i,j;

  for (i=0;i<5;i++){
                printf("Escriba un número");
                scanf("%d",&numeros1[i]);
          }

  for(i=0;i<5;i++)
  {
   numeros2[i]=numeros1[i]*2;
  }

  for (i=0;i<5;i++){
          printf("%d\n",numeros2[i]);
  }

  system("PAUSE");
  return 0;
}
```

14. Que lea 5 números por teclado, los copie a otro array multiplicados por 2 y los muestre todos ordenados usando un tercer array.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
  int aux, numeros1[5],numeros2[5],numeros3[10];
  int i,j;

  for (i=0;i<5;i++){
                printf("Escriba un número");
                scanf("%d",&numeros1[i]);
          }

  for(i=0;i<5;i++)
  {
   numeros2[i]=numeros1[i]*2;
  }

  for(i=0;i<5;i++)
  {
   numeros3[i]=numeros1[i];
  }

  for(i=0;i<5;i++)
  {
   numeros3[5+i]=numeros2[i];
  }

  for (i=0;i<10;i++){
          printf("%d\n",numeros3[i]);
```

```
 }

 system("PAUSE");
 return 0;
}
```

15. Que rellene un array con los 100 primeros números pares y muestre su suma.

```
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
        int x,cont,sum,i,tabla[100];

        i=0;
        sum=0;
        for (x=1;x<=100;x++)
        {
        cont=0;
        if (x%2==0)
        {
        tabla[i]=x;
        i++;
        }
        }

        for (x=0;x<i;x++)
        {
        sum=sum+tabla[x];
        }

        printf("%d\n",sum);

        system("PAUSE");
   return 0;
}
```

16. Que lea 10 números por teclado, los almacene en un array y muestre la media.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
 float sum, numeros1[10];
 int i;

 sum=0;
 for (i=0;i<10;i++){
                printf("Escriba un número");
                scanf("%f",&numeros1[i]);
        }

 for(i=0;i<10;i++)
```

```
 {
  sum=sum+numeros1[i];
 }

 printf("%f\n",sum/10);

 system("PAUSE");
 return 0;
}
```

17. Que mediante un array almacene números tanto positivos como negativos y los muestre ordenados.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
 float aux, numeros[10];
 int i,j,n=10;

 for (i=0;i<n;i++){
            printf("Escriba un número");
            scanf("%f",&numeros[i]);
        }

 for(i=0;i<n-1;i++)
 {
  for(j=i+1;j<n;j++)
  {
        if(numeros[i]<numeros[j])
        {
        aux=numeros[i];
        numeros[i]=numeros[j];
        numeros[j]=aux;
    }
  }
 }

 for (i=n-1;i>=0;i--){
        printf("%f\n",numeros[i]);
 }

 system("PAUSE");
 return 0;
}
```

18. Que rellene un array con 20 números y luego busque un número concreto.

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
```

```
        int  i,x=0,vector[20], n=20, dato, centro,inf=0,sup=n-1;

        for (i=0;i<20;i++){
            printf("Escriba un número");
     scanf("%d",&vector[i]);
   }

        printf("Escriba el número a buscar");
   scanf("%d",&dato);

        while(inf<=sup)
        {
        centro=(sup+inf)/2;
        if (vector[centro]==dato)
        {
        printf("Existe\n");
        x=1;
        break;
        }
        else if(dato < vector [centro] )
        {
        sup=centro-1;
        }
        else
        {
        inf=centro+1;
        }
        }

        if (x==0)
   {
        printf("No existe\n");
   }

        system("PAUSE");
        return 0;

}
```

19. Almacenar n números en un vector, almacenarlos en otro vector en orden inverso al vector original e imprimir el vector resultante.

```
#include <stdio.h>
int main(){
  int i=0,n;
  int temp;
  int vectorA[i];



  printf("Digite la cantidad de numeros a calcular\n");
  scanf("%d", &n);

  for(i=0; i<n; i++){
  printf("Digite el valor %d del vector A\n", i+1);
```

```
 scanf("%d", &vectorA[i]);
 }

 printf("\n");
 printf("VECTOR ORIGINAL\n\n");
 for(i=0; i<n; i++){
 printf("posicion %d: %d \n", i+1, vectorA[i]);
 }

 for(i=0; i<n/2; i++){
   int temp=vectorA[i];
    }

  for (int i=0; i<n/2; i++) {
    int temp=vectorA[i];
    vectorA[i]=vectorA[n-1-i];
    vectorA[n-1-i]=temp;
    }

   printf("\n");
 printf("VECTOR RESULTANTE\n\n");
 for(int i=0; i<n; i++){
   printf("posicion %d: %d \n", i+1, vectorA[i]);
 }
 return 0;

}
```

EJERCICIO 12 TALLER. Desarrollar un algoritmo que permita almacenar la cédula y el nombre de n estudiantes. 2. El usuario puede ingresar un numero de cedula a buscar en el vector y el algoritmo debe mostrar el nombre que corresponde al número de documento ingresado, siempre y cuando haya sido almacenado previamente.

```
#include <stdio.h>
#include<clocale>
#define N 3

struct registro{
        char nombre;
   int cedula;
}personas[N];

int main()
{
        setlocale(LC_CTYPE,"Spanish");

   int opcion, i;

   printf("Por favor ingrese los datos a consultar\n\n");
```

```
        printf("Por favor ingrese el nombre: ");
        scanf("%s",&personas[i].nombre) ;
        fflush(stdin);

                printf("Por favor ingrese la cédula: ");
        scanf("%i",&personas[i].cedula) ;
        fflush(stdin);

             printf("\n\n");

        printf("Ingrese el numero del registro realizado:\n ");
        scanf("%d",&opcion);
        printf("\n\n");

        if(opcion >=0 && opcion<N){
           printf( "Los datos consultados son:\n");
           printf("\n\n");
           printf("Nombre:%s",personas[opcion].nombre);
           printf("Cédula:%i",personas[opcion].cedula);
        }else{
           printf("\n\n");
           printf("Indice errado\n");
                   }
    return 0;
}
```

DE OTRA FORMA

```
#include <iostream>

#include <conio.h>

#include <stdio.h>

#include<clocale>

#define N 3


using namespace std;

 int main()

{

          setlocale(LC_CTYPE,"Spanish");

          struct {
```

```
        string nombre;
        string cedula;
        }personas[N];

        int opcion, i;
        printf("Por favor ingrese los datos a consultar\n\n");
   for(i=0;i<N;i++){
        cout << "Por favor ingrese el nombre [" << i << "]: ";
        getline (cin,personas[i].nombre) ;
        cout << "Por favor ingrese la cedula [" << i << "]: " ;
        getline (cin, personas[i].cedula) ;
        cout << endl;
        }
     printf("Ingrese el numero del registro realizado:\n ");
    scanf("%d",&opcion);
        cout << endl << endl;

        if(opcion >= 0 && opcion < N){
             cout << "Los datos consultados son:";
        cout << endl << endl;
        cout << "Nombre:" << personas[opcion].nombre << endl;
        cout << "Cédula:" << personas[opcion].cedula << endl;
        }else{
        cout << endl << endl;
        cout << "Indice erroneo." << endl;
                        }
        return 0;
}
```

EJERCICIO 5 TALLER VECTORES:

Almacenar n números en un vector, imprimir cuantos son ceros, cuantos son negativos, cuantos positivos. Imprimir además la suma de los negativos y la suma de los positivos.

```
#include <stdio.h>
int main(){
        int i, datos;
        int suma_neutro=0;
        int suma_negativos=0;
        int suma_positivos=0;
        int acumulador_negativos=0;
        int acumulador_positivos=0;
        int vector[100];

        printf("ingrese el numero de datos del vector\n");
        scanf("%d", &datos);

        for (i=0;i<datos;i++){
                        printf("Digite los valores posicion %d\t",i);
                        scanf("%d",&vector[i]);
        }
        printf("  \n ");

   printf ("LOS NUMEROS NEUTROS SON:\n");
   for (i=0;i<datos;i++){
        if (vector[i]==0){
                printf("%d\n",vector[i]);
                suma_neutro=suma_neutro+1;
                }
}
        printf("EL TOTAL DE LOS NUMEROS NEUTROS SON:%d \n",suma_neutro);

        printf("  \n ");

        printf ("LOS NUMEROS POSITIVOS SON:\n");
   for (i=0;i<datos;i++){
        if (vector[i]>0){
                printf("%d\n",vector[i]);
                        suma_positivos=suma_positivos+1;
                        acumulador_positivos=acumulador_positivos+vector[i];
                        }
}
        printf("EL TOTAL DE LOS NUMEROS POSITIVOS SON:%d \n",suma_positivos);
        printf("EL TOTAL DE LA SUMA DE LOS POSITIVOS SON:%d \n",acumulador_positivos);

        printf("  \n ");

        printf ("LOS NUMEROS NEGATIVOS SON:\n");
   for (i=0;i<datos;i++){
        if (vector[i]<0){
                printf("%d\n",vector[i]);
                        suma_negativos=suma_negativos+1;
                        acumulador_negativos=acumulador_negativos+vector[i];
                        }
```

```
}
        printf("EL TOTAL DE LOS NUMEROS NEGATIVOS SON:%d \n",suma_negativos);
        printf("EL TOTAL DE LA SUMA DE LOS NEGATIVOS SON:%d \n",acumulador_negativos);

return 0;

}
```

9.2. MATRICES - BIDIMENSIONAL o MULTIDIMENSIONAL

Una matriz es un array multidimensional. Se definen igual que los vectores excepto que se requiere un índice por cada dimensión.

Su sintaxis es la siguiente:

tipo nombre [tamaño 1][tamaño 2]...;

Una matriz bidimensional se podría representar gráficamente como una tabla con filas y columnas.

*La matriz tridimensional se utiliza, por ejemplo, para trabajos gráficos con objetos **3D**.*

En el ejemplo puedes ver cómo se rellena y visualiza una matriz bidimensional. Se necesitan dos bucles para cada una de las operaciones. Un bucle controla las filas y otro las columnas.

```
/* Matriz bidimensional. */

#include <stdio.h>

main() /* Rellenamos una matriz */
{
        int x,i,numeros[3][4];
        /* rellenamos la matriz */
        for (x=0;x<3;x++)
                for (i=0;i<4;i++)

        scanf("%d",&numeros[x][i]);
        /* visualizamos la matriz */
        for (x=0;x<3;x++)
                for (i=0;i<4;i++)

        printf("%d",numeros[x][i]);
}
```

Si al declarar una matriz también queremos inicializarla, habrá que tener en cuenta el orden en el que los valores son asignados a los elementos de la matriz. Veamos algunos ejemplos:

int numeros[3][4]={1,2,3,4,5,6,7,8,9,10,11,12};

quedarían asignados de la siguiente manera:

numeros[0][0]=1 numeros[0][1]=2 numeros[0][2]=3 numeros[0][3]=4
numeros[1][0]=5 numeros[1][1]=6 numeros[1][2]=7 numeros[1][3]=8
numeros[2][0]=9 numeros[2][1]=10 numeros[2][2]=11 numeros[2][3]=12

También se pueden inicializar cadenas de texto:

char dias[7][10]={"lunes","martes","miércoles","jueves","viernes","sábado","domingo"};

Para referirnos a cada palabra bastaría con el primer índice:

printf("%s",dias[i]);

EJERCICIOS

1. Que muestre los primeros 100 números de izquierda a derecha usando un array de dos dimensiones.

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
        int x,y, numeros[10][10];
    for (x=0;x<10;x++)
  {
        for (y=0;y<10;y++)
        {
        numeros[x][y]=(x*10)+1+y;
    }
        }
        for (x=0;x<10;x++)
  {
        for (y=0;y<10;y++)
        {
        printf("%d ",numeros[x][y]);
    }
        printf("\n");
        }
        system("PAUSE");
  return 0;
}
```

2. Que muestre los primeros 100 números de izquierda a derecha usando un array de dos dimensiones, la última fila a mostrará la suma de sus respectivas columnas.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
        int x,y,sum, numeros[11][10];
    for (y=0;y<10;y++)
        {
        sum=0;
        for (x=0;x<10;x++)
     {
        numeros[x][y]=(x*10)+1+y;
        sum=sum+numeros[x][y];
        }
        numeros[10][y]=sum;
  }
        for (x=0;x<11;x++)
  {
        for (y=0;y<10;y++)
        {
        printf("%d ",numeros[x][y]);
     }
         printf("\n");
        }
        system("PAUSE");
  return 0;
}
```

3. Que rellene un array de dos dimensiones con números pares, lo pinte y después que pida una posición X,Y y mostrar el número correspondiente.

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
        int x,y,num=2, numeros[3][3];
    for (x=0;x<3;x++)
  {
        for (y=0;y<3;y++)
        {
        numeros[x][y]=num;
        num=num*2;
        }
        }
        printf("Introduzca coordenada x: ");
  scanf("%d",&x);
  printf("Introduzca coordenada y: ");
        scanf("%d",&y);
        printf("El número es: %d\n",numeros[x][y]);
    system("PAUSE");
        return 0;
}
```

4. Que rellene una matriz de 3x3 y muestre su traspuesta (la traspuesta se consigue intercambiando filas por columnas y viceversa).

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
        int x,y,num=0, numeros[4][4];
    for (x=0;x<3;x++)
   {
        for (y=0;y<3;y++)
        {
        numeros[x][y]=num;
        num++;
        }
        }
        printf("El array original es: \n\n\n");

   for(x = 0;x < 3;x++)
   {
        for(y = 0;y < 3;y++)
        {
        printf(" %d       ", numeros[x][y]);
    }
        printf("\n\n\n");
   }
        printf("La traspuesta es: \n\n\n");
    for(x = 0;x < 3;x++)
   {
        for(y = 0;y < 3;y++)
        {
        printf(" %d       ", numeros[y][x]);
    }
        printf("\n\n\n");
        }
        system("PAUSE");
        return 0;
}
```

5. • Hacer un programa para rellenar una matriz poniendo el usuario el número de filas y columnas, posteriormente mostrar la matriz en pantalla.

```
#include<stdio.h>

#include<clocale>

int main(){

        setlocale(LC_CTYPE,"Spanish");

        int matriz[100][100],filas,columnas;

        int i,j;
```

```
printf("por favor digite el número de filas: \n");
scanf ("%i",&filas);
printf ("por favor digite el número de columnas: \n");
scanf ("%i",&columnas);
for (i=0;i<filas;i++){
        for (j=0;j<columnas;j++){
            printf ("digite un número [%d,%d]: ",i,j);
            scanf ("%i",&matriz[i][j]);
        }
}
printf ("\n");
for (i=0;i<filas;i++){
        for (j=0;j<columnas;j++)
            printf ("%i\t",matriz[i][j]);
            printf ("\n");
        }
}
```

6. programa para rellenar una matriz 4x4 o el tamaño que deseemos, posteriormente mostrar la matriz en pantalla.

```
#include<stdio.h>
#include<clocale>
int main(){
    setlocale(LC_CTYPE,"Spanish");
    int matriz[4][4];
    int i,j;

    for (i=0;i<4;i++){
```

```
                        for (j=0;j<4;j++){
                                printf ("digite un número [%d,%d]: ",i,j);
                                scanf ("%i",&matriz[i][j]);
                        }
        }
        printf ("\n");
        for (i=0;i<4;i++){
                        for (j=0;j<4;j++)
                                printf ("%i\t",matriz[i][j]);
                                printf ("\n");
                        }
return 0;
}
```

7. sumar los extremos de una matriz el total de cada fila y cada columna.

```
#include <stdio.h>
#include <conio.h>
#define TAM 3
 int main()
{
 int matriz[TAM+1][TAM+1],i,j;

  // INICIALIZAR MATRIZ
 for(i=0;i<TAM+1;i++)
  for(j=0;j<TAM+1;j++)
            matriz[i][j] = 0;

 // INGRESAR LOS DATOS
```

```
for(i=0;i<TAM;i++)
{
 for(j=0;j<TAM;j++)
 {
  printf("Ingresar dato entero en ( fila [%d] columna [%d] ):",i,j);
  scanf("%d",&matriz[i][j]);
 }
 printf("\n");
}

// MOSTRAR LOS DATOS QUE SE INGRESE EN LA MATRIZ
for(i=0;i<TAM;i++)
{
 for(j=0;j<TAM;j++)
  printf("%d\t",matriz[i][j]);

 printf("\n");
}

// ACUMULAR EN LA ULTIMA POSICION DE LA MATRIZ
for(i=0;i<TAM;i++)
{
 for(j=0;j<TAM;j++)
 {
  matriz[i][TAM]+= matriz[i][j];
  matriz[TAM][j]+= matriz[i][j];
 }
}
```

```
 printf("\n");


 // MOSTRAR LAS SUMAS
 for(i=0;i<TAM+1;i++)
        {
  for(j=0;j<TAM+1;j++)
   printf("%d\t",matriz[i][j]);
  printf("\n");
        }
        getch();
}
```

8. Obtener el valor de una casilla específica

```
#include<stdio.h>
 int main(){
int tabla[3][4]= {{0,3,4,10},{3,-1,10,-10},{10,100,3,1}};
 for (int i=0;i<3;i++){
        for (int j=0;j<4;j++){
                printf("%i",tabla[i][j]);
                }
        printf("\n");
}
printf("fila1casilla1: %i \n",tabla [1][1]);
return 0;
}
```

*9. · Hacer un algoritmo que almacene números en una matriz de 5 * 6. Imprimir la suma de los números almacenados en la matriz.*

```
#include <stdio.h>
int main(){
```

```
    int filas,columnas;
    int suma=0;
    int matriz[5][6];
            //ingresar datos
    for (filas=0;filas<5;filas++){
            for(columnas=0;columnas<6;columnas++){
                    printf("Ingrese el valor para la celda [%d,%d]: ",filas,columnas);
                    scanf("%d",&matriz[filas][columnas]);
    }
            }
                    printf("\n");
  //imprimir matriz
            for (filas=0;filas<5;filas++){
                    for(columnas=0;columnas<6;columnas++)
               printf(" %d \t ",matriz[filas][columnas]);
                     printf("\n");
            }
                    printf ("\n\n\n");
    //operacion  suma
            for (filas=0;filas<5;filas++)
            {
                    for(columnas=0;columnas<6;columnas++)
                            suma+=matriz[filas][columnas];
                    }
                    printf("la suma de la matriz es:%d\n",suma);
return 0;
}
```

10. FUNCIONES

10.1.- Tiempo de vida de los datos

Según el lugar donde son declaradas puede haber dos tipos de variables.

Globales: *las variables permanecen activas durante todo el programa. Se crean al iniciarse éste y se destruyen de la memoria al finalizar. Pueden ser utilizadas en cualquier función.*

Locales: *las variables son creadas cuando el programa llega a la función en la que están definidas. Al finalizar la función desaparecen de la memoria.*

Si dos variables, una global y una local, tienen el mismo nombre, la local prevalecerá sobre la global dentro de la función en que ha sido declarada.

Dos variables locales pueden tener el mismo nombre siempre que estén declaradas en funciones diferentes.

```
/* Variables globales y locales. */

#include <stdio.h>

int num1=1;
main() /* Escribe dos cifras */
{
        int num2=10;
        printf("%d\n",num1);
        printf("%d\n",num2);
}
```

10.2.- Funciones

Las funciones son bloques de código utilizados para dividir un programa en partes más pequeñas, cada una de las cuáles tendrá una tarea determinada.

Su sintaxis es:

```
tipo_función nombre_función (tipo y nombre de argumentos)
{
        bloque de sentencias
}
```

***tipo_función**: puede ser de cualquier tipo de los que conocemos. El valor devuelto por la función será de este tipo. Por defecto, es decir, si no indicamos el tipo, la función devolverá un valor de tipo entero (**int**). Si no queremos que retorne ningún valor deberemos indicar el tipo vacío (**void**).*

***nombre_función**: es el nombre que le daremos a la función.*

tipo y nombre de argumentos: son los parámetros que recibe la función. Los argumentos de una función no son más que variables locales que reciben un valor. Este valor se lo enviamos al hacer la llamada a la función. Pueden existir funciones que no reciban argumentos.

bloque de sentencias: es el conjunto de sentencias que serán ejecutadas cuando se realice la llamada a la función.

*Las funciones pueden ser llamadas desde la función **main** o desde otras funciones. Nunca se debe llamar a la función **main** desde otro lugar del programa. Por último recalcar que los argumentos de la función y sus variables locales se destruirán al finalizar la ejecución de la misma.*

10.3.- Declaración de las funciones

Al igual que las variables, las funciones también han de ser declaradas. Esto es lo que se conoce como prototipo de una función. Para que un programa en C sea compatible entre distintos compiladores es imprescindible escribir los prototipos de las funciones.

Los prototipos de las funciones pueden escribirse antes de la función main o bién en otro fichero. En este último caso se lo indicaremos al compilador mediante la directiva #include.

En el ejemplo adjunto podremos ver la declaración de una función (prototipo). Al no recibir ni retornar ningún valor, está declarada como void en ambos lados. También vemos que existe una variable global llamada num. Esta variable es reconocible en todas las funciones del programa. Ya en la función main encontramos una variable local llamada num. Al ser una variable local, ésta tendrá preferencia sobre la global. Por tanto la función escribirá los números 10 y 5.

```
/* Declaración de funciones. */

#include <stdio.h>

void funcion(void); /* prototipo */
int num=5; /* variable global */
```

```
main() /* Escribe dos números */
{
        int num=10; /* variable local */
        printf("%d\n",num);
        funcion(); /* llamada */
}

void funcion(void)
{
        printf("%d\n",num);
}
```

10.4.- Paso de parámetros a una función

Como ya hemos visto, las funciones pueden retornar un valor. Esto se hace mediante la instrucción ***return****, que finaliza la ejecución de la función, devolviendo o no un valor.*

En una misma función podemos tener más de una instrucción return. La forma de retornar un valor es la siguiente:

return (valor o expresión);

El valor devuelto por la función debe asignarse a una variable. De lo contrario, el valor se perderá.

En el ejemplo puedes ver lo que ocurre si no guardamos el valor en una variable. Fíjate que a la hora de mostrar el resultado de la suma, en el ***printf****, también podemos llamar a la función.*

```
/* Paso de parámetros. */

#include <stdio.h>

int suma(int,int); /* prototipo */
main() /* Realiza una suma */
{
        int a=10,b=25,t;
        t=suma(a,b); /* guardamos el valor */
        printf("%d=%d",suma(a,b),t);
        suma(a,b); /* el valor se pierde */
}

int suma(int a,int b)
{
        return (a+b);
}
```

Ahora veremos lo que se conoce como paso de parámetros.

Existen dos formas de enviar parámetros a una función:

***Por valor:** cualquier cambio que se realice dentro de la función en el argumento enviado, **NO** afectará al valor original de las variables utilizadas en la llamada. Es como si trabajaramos con una copia, no con el original. No es posible enviar por valor **arrays**, deberemos hacerlo por referencia.*

***Por referencia:** lo que hacemos es enviar a la función la dirección de memoria donde se encuentra la variable o dato. Cualquier modificación **SI** afectará a las variables utilizadas en la llamada. Trabajamos directamente con el original.*

```
/* Paso por valor. */

#include <stdio.h>

void intercambio(int,int);
main() /* Intercambio de valores */
{
        int a=1,b=2;
        printf("a=%d y b=%d",a,b);
        intercambio(a,b); /* llamada */
        printf("a=%d y b=%d",a,b);
}

void intercambio (int x,int y)
{
        int aux;
        aux=x;
        x=y;
        y=aux;
        printf("a=%d y b=%d",x,y);
}
```

*Para enviar un valor por referencia se utiliza el símbolo **&** (ampersand) delante de la variable enviada. Esto le indica al compilador que la función que se ejecutará tendra que obtener la dirección de memoria en que se encuentra la variable.*

*Vamos a fijarnos en los ejemplos. En el ejemplo anterior podrás comprobar que antes y después de la llamada, las variables mantienen su valor. Solamente se modifica en la función **intercambio** (paso por valor).*

En el siguiente ejemplo podrás ver cómo las variables intercambian su valor tras la llamada de la función (paso por referencia).

*Las variables con un ***** son conocidas como **punteros**, el único dato en 'C' que puede almacenar una dirección de memoria.*

```
/* Paso por referencia. */

#include <stdio.h>

void intercambio(int *,int *);
main() /* Intercambio de valores */
{
        int a=1,b=2;
        printf("a=%d y b=%d",a,b);
        intercambio(&a,&b); /* llamada */
        printf("a=%d y b=%d",a,b);
}

void intercambio (int *x,int *y)
{
        int aux;
        aux=*x;
        *x=*y;
        *y=aux;
        printf("a=%d y b=%d",*x,*y);
}
```

.- Los argumentos de la función main

Ya hemos visto que las funciones pueden recibir argumentos. Pues bién, la función ***main*** *no podía ser menos y también puede recibir argumentos, en este caso desde el exterior.*

Los argumentos que puede recibir son:

argc: *es un contador. Su valor es igual al número de argumentos escritos en la línea de comandos, contando el nombre del programa que es el primer argumento.*

argv: *es un puntero a un array de cadenas de caracteres que contiene los argumentos, uno por cadena.*

En este ejemplo vamos a ver un pequeño programa que escribirá un saludo por pantalla. El programa ***FUNCION6.EXE****.*

```
/* Argumentos de la main. */
```

```
#include <stdio.h>

main(int argc,char *argv[]) /* argumentos */
{
        printf("\nCurso de Programación en C - Copyright (c) 1997-2001, Sergio Pacho\n");
        printf("Programa de ejemplo.\n\n");
        if (argc<2)
        {
                printf("Teclee: funcion6 su_nombre");
                exit(1); /* fin */
        }
        printf("Hola %s",argv[1]);
}
```

EJERCICIOS

1.

```
#include <stdio.h>
int arreglo[10] = {3,10,1,8,15,5,12,6,5,4}; /*Declaracion e inicialización
del arreglo. */


/*imprimearreglo - Funcion que muestra por pantalla
el contenido de un arreglo.*/
void imprimearreglo() {
   int i; for (i=0;i<10;i++)
      printf("Elemento %d: %d \n",i,arreglo[i]);
}

int main() /*Funcion Principal del Programa*/
{
   int i,j,k;
   imprimearreglo();
   for (i=1; i<10; i++) {
      j=i;
         while (j>=0 && arreglo[j]<arreglo[j-1]) {
            k=arreglo[j];
            arreglo[j]=arreglo[j-1];
            arreglo[j-1]=k;
            j--;
         }
   }
   printf("\n\nArreglo ordenado \n\n");
   imprimearreglo();
}
```

2. programa pide al usuario ingresar las notas de uno o más alumnos, y va mostrando los promedios de cada uno de ellos:

```
#include <stdio.h>
float promedio(int valores[], int cantidad) {
```

```
    int i;
    float suma = 0.0;

    for (i = 0; i < cantidad; ++i)
        suma += valores[i];

    return suma / (float) cantidad;
}

int main() {

    int notas[10];
    char nombre[20];
    char opcion[3];
    int n, i;

    do {
        printf("Ingrese nombre del alumno: ");
        scanf("%s", nombre);

        printf("Cuantas notas tiene %s? ", nombre);
        scanf("%d", &n);

        for (i = 0; i < n; ++i) {
            printf("  Nota %d: ", i + 1);
            scanf("%d", &notas[i]);
        }

        printf("El promedio de %s es %.1f\n", nombre, promedio(notas, n));

        printf("Desea calcular mas promedios (si/no)? ");
        scanf("%s", opcion);

    } while (opcion[0] == 's' || opcion[0] == 'S');

    return 0;
}
```

EJERCICIO FUNCION VECTOR

Ingresar 10 números de tipo entero y los almacena en un arreglo; después le pide que introduzca un número para que busque su posición dentro del arreglo..

El programa utiliza una función llamada BuscaNumero que recibe como parámetros el arreglo con los 10 números capturados, el número de elementos del arreglo (en este caso 10) y el número del cual se desea saber su posición dentro del arreglo.. La función regresa -1 si el número que se busca no se encuentra en el arreglo y en caso contrario, regresa la primera posición del arreglo que contiene dicho número.

El programa también utiliza una función llamada MuestraArreglo que no regresa valor alguno, sólo recibe como parámetros el arreglo y el número de elementos. Esta función imprime en pantalla los elementos del arreglo separados por un tabulador.

```
int main()
{
 int x=0, numero=0, posicion=0;
 int ar_numeros[10] = {0};

 printf("Introduzca los 10 numeros enteros que se almacenaran en el arreglo\n");
 for (x=0; x<10; ++x)
    {
     printf("Valor para el elemento [%d]: ", x);
     scanf("%d",&ar_numeros[x]);
    }
 printf("\n");

 printf("Introduzca el número que desea buscar en el arreglo\n");
 scanf("%d",&numero);
 printf("\n");
 MuestraArreglo(ar_numeros,10);

 posicion=BuscaNumero(ar_numeros,10,numero);
 if (posicion != -1)
    printf("El número %d está en la posición %d del arreglo\n",numero, posicion);
 else
    printf("El número %d no está en el arreglo\n",numero);

 return 0;
}
```

EJERCICIO FUNCION MATRIZ

Para pasar una función un arreglo de dos dimensiones, debemos indicar el tamaño de la segunda dimensión del arreglo

Ejemplo:int MiFuncion(int mi_arreglo[][5], int num_elementos);

El siguiente programa le pide al usuario que introduzca 9 números y los almacena en un arreglo de dos dimensiones, en este caso una matriz de 3×3; posteriormente utiliza una función llamada ImprimeMatriz para mostrar como quedaron almacenados los números en la matriz. Dicha función recibe como parámetros, la matriz de 3×3 y el tamaño de la primera dimensión (normalmente la primera dimensión son filas y la segunda dimensión columnas).

```
#include <stdio.h>

 void ImprimeMatriz(int m[][3], int filas)

{

 int i=0,j=0;

  for (i=0; i<filas; ++i)    {

     for (j=0; j<3; ++j)
```

```
        {
          printf("%d ",m[i][j]);
        }
     printf("\n");
    }
}
 int main()
{
 int x=0,y=0;
 int matriz[3][3] = {{0},{0},{0}};
  printf("Introduzca los valores para la matriz\n");
 for (x=0; x<3; ++x)     {
     for (y=0; y<3; ++y)  {
          printf("Valor para el elemento [%d][%d]: ", x, y);
         scanf("%d",&matriz[x][y]);
          }
     printf("\n");
    }
  printf("Matriz\n");
 ImprimeMatriz(matriz, 3);
  return 0;
}
```

Función Vector Ejercicio 5 taller

2. *Almacenar 300 números en un vector, imprimir cuantos son ceros, cuantos son negativos, cuantos positivos. Imprimir además la suma de los negativos y la suma de los positivos.*

```
Include <stdio.h>
#include <clocale>
void postivos(int vector[], int datos )
{
 int  i;
```

```
 int suma_positivos=0;
 int acumulador_positivos=0;
        printf ("\n LOS NÚMEROS POSITIVOS SON: \n");
        for (i=0;i<datos;i++){
                if (vector[i]>0){

                        printf ("%i\n",vector[i]);
                        suma_positivos=suma_positivos+1;
                        acumulador_positivos=acumulador_positivos+vector[i];
                }
        }

        printf ("\n el total de los números positivos son:\n %i ",suma_positivos);
        printf ("\n el total de la suma de los positivos es: \n %i",acumulador_positivos);
        printf ("\n");
}

int main(){
setlocale(LC_CTYPE,"Spanish");
        int i, datos;
        int suma_neutro=0;
        int suma_negativos=0;
        int acumulador_negativos=0;
        int vector [datos];
        printf ("por favor ingrese la cantidad de datos que tendrá el vector: \n");
        scanf ("%i",&datos);
                for (i=0;i<datos;i++){
                printf ("\n digite el número para la posición %i\t",i);
                scanf ("%i",&vector[i]);
        }
        printf ("\n LOS NÚMEROS NEUTROS SON: \n");
        for (i=0;i<datos;i++){
                if (vector[i]==0){
                        printf ("%i\n",vector[i]);
                        suma_neutro=suma_neutro+1;
                }
        }
        printf ("\n el total de los números neutros son:\n %i ",suma_neutro);
        printf ("\n");

        postivos (vector,datos);
                printf ("\n LOS NÚMEROS NEGATIVOS SON: \n");

        for (i=0;i<datos;i++){
                if (vector[i]<0){
```

```
                printf ("%i\n",vector[i]);
                suma_negativos=suma_negativos+1;
                acumulador_negativos=acumulador_negativos+vector[i];
            }
        }
        printf ("\n el total de los números negativos son:\n %i ",suma_negativos);
        printf ("\n el total de la suma de los negativos es: \n %i",acumulador_negativos);
        printf ("\n");
        return 0;
}
```

11. PUNTEROS

Un puntero es una variable que contiene la dirección de memoria de otra variable. Se utilizan para pasar información entre una función y sus puntos de llamada.

11.1.- Declaración

Su sintaxis es la siguiente:

tipo *nombre;

Donde nombre es, naturalmente, el nombre de la variable, y tipo es el tipo del elemento cuya dirección almacena el puntero.

11.2.- Operadores

Existen dos operadores especiales para trabajar con punteros: ***&*** *y* *******.*

El primero devuelve la dirección de memoria de su operando. Por ejemplo, si queremos guardar en el puntero ***x*** *la dirección de memoria de la variable* ***num****, deberemos hacer lo siguiente:*

x=#

El segundo devuelve el valor de la variable cuya dirección es contenida por el puntero. Este ejemplo sitúa el contenido de la variable apuntada por ***x****, es decir* ***num****, en la variable* ***a****:*

a=*x;

11.3.- Asignación

Los punteros se asignan igual que el resto de las variables. El programa ejemplo mostrará las direcciones contenidas en ***p1*** *y* ***p2****, que será la misma en ambos punteros.*

```
/* Asignaciones de punteros. */
#include <stdio.h>
main() /* Asignamos direcciones */
{
        int a;
        int *p1,*p2;
        p1=&a;
        p2=p1;
        printf("%p %p",p1,p2);
}
```

11.4.- Aritmética de direcciones

Es posible desplazar un puntero recorriendo posiciones de memoria. Para ello podemos usar los operadores de suma, resta, incremento y decremento (+, -, ++, - -). Si tenemos un puntero (p1) de tipo ***int*** *(2 bytes), apuntando a la posición 30000 y hacemos:* ***p1=p1+5;*** *el puntero almacenará la posición 30010, porque apunta 5 enteros por encima (10 bytes más).*

Ejercicios

1. *Escribir un programa que calcule y visualice la media aritmética de un vector de 10 elementos numéricos, utilizando una variable puntero que apunte a dicho vector.*

```
# include <stdio.h>
int main ()
{
float med, suma=0;
int tabla[10], j;
int *pt=tabla;
for(j=0;j<10;j++)
{
printf("\nIntroducir el elemento %d: ",j+1);
scanf("%d",pt+j);
}
for (j=0;j<10;j++)
suma+=*(pt+j);
med=suma/10;
printf("\n\n")
printf("La media vale:%f",med);
}
```

2. ***Escribir un programa que ponga en mayúsculas el primer carácter de una cadena de caracteres y todo aquel carácter que siga a un punto, utilizando un puntero a dicha cadena.***

```
# include <stdio.h>
# include <ctype.h>
void main (void)
{
char texto[80];
char *ptext=texto;
printf("\nIntroduccir un texto menor de 80 caracteres:\n");
gets(texto);
ptext=toupper(*ptext);
while(*ptext!='\0')
{
ptext++;
if(*ptext=='.')
*(ptext+2)=toupper(*(ptext+2));
}
puts(texto);
}
```

3. ***Escribir un programa que mediante el empleo de un menú realice operaciones aritméticas, llamando a las funciones correspondiente. Desarrollar este programa utilizando una tabla de punteros a funciones.***

```
# include <stdio.h>
# include <math.h>
int main (void)
{
void suma(float x, float y);
void resta(float x, float y);
void prod(float x, float y);
void divi(float x, float y);
```

```
void(*p[4])(float x, float y)={suma, resta, prod, divi};
float num1, num2;
int opcion;
printf ("\nIntroducir dos números \n");
scanf("%f%f",&num1,&num2);
printf("1.-Suma\n");
printf("2.-Resta\n");
printf("3.-Producto\n");
printf("4.-División\n");
do
{
printf("Seleccionar una opción\n");
scanf("%d",&opcion);
}
while(opcion<1 || opcion >4);
(*p[opcion-1])(num1,num2);
}
void suma(float x, float y)
{
printf("La suma vale:%f\n",x+y);
}
void resta(float x, float y)
{
printf("La resta vale:%f\n",x-y);
}
void prod(float x, float y)
{
printf("El producto vale:%f\n",x*y);
}
void divi(float x, float y)
{
if(y==0)
printf("La división no es posible. Divisor=0\n");
else
printf("La división vale:%f\n",x/y);
}
```

4. ***Escribir un programa que llame a tres funciones de nombre 'Pepe', 'Ana' y 'Maria' mediante el uso de una tabla de punteros a funciones.***

```
# include <stdio.h>
int Pepe(), Ana(), Maria();
int (*mf[])()={Pepe, Ana, Maria};
void main (void)
{
int j;
for (j=0;j<3;j++)
(*nf[j])();
/*Las dos lineas anteriores pueden simplificarse en:
for (j=0;j<3;(*nf[j++])());*/
}
Pepe()
{
printf("soy Pepe\n"):
```

```
}
Ana ()
{
printf("soy Ana\n");
}
Maria()
{
printf("soy Maria\n");
}
```

5. ***Programa que define una tabla de proveedores teniendo asignados cada proveedor un nombre, cantidad vendida del artículo, precio unitario (introducidos por teclado) e importe (calculado a partir de los datos anteriores). Se pretende visualizar los datos de cada proveedor, el importe total de compra, así como el nombre del proveedor más barato y el del más caro.***

```
# include <stdio.h>
# define NUM 3
# define DIM 16
void main (void)
{
struct proveedor
{
char prov[DIM];
int cant_p;
float precio;
float importe;
};
struct proveedor provee[NUM];
int cont, k=0, j=0;
float preciomin, preciomax;
float total=0.0;
/* Carga de la tabla de datos de proveedores*/
for(cont=0;cont<NUM:cont++)
{
 printf("\nIntroducir nombre del proveedor %d:",
cont+1);
gets(provee[cont].prov);
printf("Introducir cantidad de piezas: ");
scanf("%d", &provee[cont].cant_p);
printf("Introducir precio unitario:";
scanf("%d",&provee[cont].precio);
provee[cont].importe=provee[cont].cant_p*provee[cont]
.precio;
while(getchar()!='\n');
}
/*Visualizar datos de proveedores*/
for(cont=0;cont<NUM;cont++)
printf("\n%s %s %s %s", provee[cont].prov,
provee[cont.cant_p, provee[cont].precio,
provee[cont].importe);
total += provee[cont].importe;
printf("\nEl importe total es:%f",total);
/*Calcular el proveedor más barato y el más caro*/
preciomin=provee[0].precio;
preciomax=provee[0].precio;
```

```
for(cont=1;cont<NUM;cont++)
{
if(preciomin>provee[cont].precio)
{
preciomin=provee[cont].precio;
k=cont;
}
if(preciomax<provee[cont].precio
{
preciomax<provee[cont].precio;
j=cont;
}
}
printf("\n El proveedor mas barato es: %s",
provee[k].prov);
printf("\n El proveedor más caro es: %s", provee[j].prov);
}
```

6. *Programa que define una tabla de proveedores empleando una estructura que anida los datos del proveedor (nombre, dirección , y teléfono), cantidad vendida, precio unitario e importe (calculado). Los datos no calculados se introducen por teclado. Se pretende visualizar en pantalla los datos de cada proveedor, el importe total de las compras y el nombre y teléfono del proveedor más barato.*

```
# include <stdio.h>
# define NUM 3
struct dat_p
{
char nom_p[16];
char dir_p[30];
char telef_p[10];
};
struct entrada
{
struct dat_p datos;
int cant;
float precio;
float importe;
};
void main (void)
{
struct entrada sumin[NUM];
int cont, j=0;
float total=0.0;
float preciomin;
printf("\nIntroducir datos de proveedores\n");
for(cont=0;cont<NUM;cont++)
{
printf("Introducir proveedor num:%d",cont+1);
printf("\nNombre:");
gets(sumin[cont].datos.nom_p);
printf("\nDirección:");
gets(sumin[cont].datos.dir_p);
printf("\nTelefono:"");
gets(sumin[cont].datos.telef_p);
```

```
}
printf("\n*****************************************");
printf("\nIntroducir cantidades y precios de cada
proveedor");
for(cont=0;cont<NUM;cont++);
{
printf("\nProveedor: %s",sumin[cont]datos.nom_p);
printf("\nCantidad suministrada:");
scanf("%d",&sumin[cont].cant);
printf("Precio unitario:");
scanf("%f",&sumin[cont].precio);
while(getchar()!='\n');
sumin[cont].importe=sumin[cont].cant*
*sumin[cont].precio;
}
printf("%s %s %s %s\n","Proveedor","Cant","Precio",
"Importe");
for(cont=0;cont<NUM;cont++)
{
printf ("%s %d %f %f\n",sumin[cont].datos.nom_p,
sumin[cont].cant, sumin[cont].precio, sumin[cont].importe);
total += sumin[cont].importe;
}
printf("\nEl importe total es:%f", total);
preciomin=sumin[0].precio;
for(cont=0;cont<NUM;cont++)
{
if (preciomin>sumin[cont].precio)
{
preciomin=sumin[cont].precio;
j=cont;
}
}
printf("\nEL proveedor mas barato es: %s,
sumin[j].datos.nom_p);
printf("\nY su teléfono es: %s", sumin[j].datos.telef_p);
}
```

12. MANEJO DE CADENA DE CARACTERES

Cadenas

Una cadena o cadena de caracteres nos es más que una serie de caracteres manipulados como una unidad. Si asemejamos una cadena al lenguaje castellano sería como una palabra, que es un conjunto de sílabas y vocales en donde cada una de estas viene a ser un carácter.

Visto desde otro punto vendría a ser un arreglo de caracteres.

Una cadena puede contener cualquier carácter, puede almacenar un nombre propio una dirección, es decir, lo que nosotros precisemos.

Declaración
*Una cadena se la **define** de la siguiente manera*

char cadena[20];

La cadena anterior puede contener un máximo de 20 caracteres.

Inicialización
Se puede inicializar una cadena de la siguiente manera:

cadena = "Hola" ;

Cualquier valor que se le asigne a una cadena va entre comillas dobles " ", como en el ejemplo anterior "Hola" está entre comillas dobles.

*Una cadena siempre finaliza con el carácter de fin de cadena **'\0'**, que siempre se añade al final automáticamente, en el ejemplo anterior se añade al final de "Hola" el carácter de fin de cadena.*

También podemos considerar a una cadena como un arreglo de caracteres, y se puede inicializar de la siguiente manera:

cadena = { 'H', 'o', 'l', 'a' } ;

El arreglo de caracteres se vería de esta forma:

'H' 'o' 'l' 'a' '\0'

0 1 2 3 4

Nótese que en la posición 4 se aumenta el fin de cadena

Ejemplo
Se desea tener un programa que sea amable con el usuario, el programa deberá conocer el nombre del usuario y responderle con un mensaje amigable.

```
#include <string.h>

#include <stdio.h>

#include <clocale>

int main()

{

        setlocale(LC_CTYPE,"spanish");
```

```
        char nombre[30];

        printf("¿Cuál es tu nombre?\n");

        scanf("%s",nombre);

        printf("Que tengas un buen día %s",nombre);

}
```

En el ejemplo anterior el mensaje "¿Cuál es tu nombre?" es una cadena pues está entre comillas. También es una cadena la variable nombre que recibirá un valor desde teclado.

Operaciones con Cadenas

Existen muchas operaciones que se pueden realizar utilizando cadenas, la mayoría de la operación que podemos requerir se encuentran ya a nuestra disposición dentro de la librería string.h

Longitud

La longitud de una cadena la podemos conocer utilizando la función strlen.

Sintaxis

```
strlen( cadena ) ;
```

Ejemplo

```
#include <string.h>
#include <stdio.h>
#include <clocale>
int main()
{
        setlocale(LC_CTYPE,"spanish");
   char nombre[30];
   int tamano;
   printf("¿Cuál es tu nombre?\n");
        scanf("%s",nombre);
        tamano = strlen(nombre );
   printf("Tu nombre tiene %d",tamano,"letras");
}
```

Comparación

Para saber si dos cadenas son exactamente iguales utilizamos la función strcmp.

Sintaxis

```
strcmp ( cadena1, cadena2 );
```

Esta función devuelve un valor de acuerdo al resultado de la comparación.

Devuelve:

0 *si la dos cadenas son exactamente iguales*

Mayor a 0 *si la cadena1 es mayor a la cadena2*

Menor a 0 *si la cadena1 es menor que la cadena2*

Ejemplo

```
#include <string.h>

#include <stdio.h>

#include <clocale>

int main()

{

        setlocale(LC_CTYPE,"spanish");

   char contrasena[30], reContrasena[30];

   int resultado;

   printf("Escribe tu contraseña\n");

   scanf("%s",contrasena);

   printf("Re escribe tu contraseña\n");

   scanf("%s",reContrasena);

   resultado = strcmp(contrasena, reContrasena);

   if ( resultado == 0 )

      printf("La contraseña es aceptada\n");

   else

      printf("La contraseña no coincide\n");

}
```

Copia

Podemos reflejar todo el contenido de una cadena a otra, en otras palabras la copiamos tal cual, para esto utilizamos la función strcpy.

Sintaxis

```
        strcpy( cadenaDestino, cadenaOrigen );
```

Todo el contenido de la cadenaOrigen se copia a la cadenaDestino, si esta última tuviera algún valor este se borra.

Ejemplo

```
#include <string.h>
#include <stdio.h>
#include <clocale>
int main()
{
        setlocale(LC_CTYPE,"spanish");
        char nombre[30], apellido[30];
        printf("¿Cuál es tu nombre? \n");
   scanf("%s",nombre);
        printf("¿Cuál es tu apellido paterno\n");
   scanf("%s",apellido);
   strcat(nombre, " "); //Se le añade un espacio en blanco
   strcat(nombre, apellido);
   printf("Tu nombre completo es:%s",nombre);
}
```

Concatenación

Podemos juntar o concatenar dos cadenas una a continuación de la otra. Utilizamos la función strcat.

Sintaxis

```
        strcat( cadenaDestino, cadenaOrigen );
```

Todo el contenido de la cadenaOrigen se añade a continuación de la cadenaDestino, si esta última contiene algo entonces al final contendrá lo que contenía más el contenido de la cadenaOrigen.

Ejemplo

```
#include <iostream>
#include <string.h>
#include <stdio.h>
#include <clocale>
int main()
{
        setlocale(LC_CTYPE,"spanish");
   char origen[30], copia[30];
   printf("¿Qué día es hoy? \n");
   scanf("%s",origen);
   strcpy(copia, origen);
   printf("Hoy es %s",copia);
}
```

/*Longitud de una cadena en C ++
*primer nivel */*

```
#include <stdio.h>
#include <string.h>
#include <clocale>

int main()
 {
  setlocale(LC_CTYPE,"spanish");
```

```
  char* cadena = "Hola, buen día. Estudiantes de la USCO";
  int contador = 0;
  // Recorrer la cadena hasta encontrar el carácter NUL o de terminación
  while (cadena[contador] != 0) {
    contador++;
  }
  printf("La longitud de '%s' es %d", cadena, contador);
  /*   Salida:
  La longitud de 'Hola, buen día. Estudiantes de la USCO' es 38  */
  return 0;
}
```

Una versión más simple

La versión presentada anteriormente es una versión explicativa. Podemos ahorrarnos el cuerpo del ciclo while simplemente preincrementando el valor de contador.

```
/*Longitud de una cadena en C ++
  primer nivel
*/

#include <stdio.h>
#include <string.h>
#include <clocale>

int main()
 {
  setlocale(LC_CTYPE,"spanish");
  char* cadena = "Hola, buen día. Estudiantes de la USCO";
  int contador = 0;
  while (cadena[++contador] != 0);
  printf("La longitud de '%s' es %d", cadena, contador);
          /*        Salida:
          La longitud de 'Hola, buen día. Estudiantes de la USCO' es 38 */
  return 0;
}
```

Lo que hacemos es usar ++contador, que incrementará la variable, regresará el valor después de incrementarla y a su vez servirá como índice del while.

Funciona de la misma manera para obtener la longitud de una cadena en C

CONTAR VOCALES UTILIZANDO FUNCIONES

```
#include <stdio.h>

#include <ctype.h>

// Definición
```

```
int contarVocales(char *cadena);

int main(int argc, char const *argv[])

{

	// Un arreglo de longitud de 1000

	// porque no podemos tener arreglos de longitud dinámica ni strings

	char entrada[1000];

	printf("Escribe una cadena:\n");

	gets(entrada);

	int vocales = contarVocales(entrada);

	printf("El numero de vocales que tiene la cadena es: %d\n", vocales);

	return 0;

}

//Cuerpo de la función

int contarVocales(char *cadena){

	int vocales = 0;

	// Recorrer toda la cadena

	for (int indice = 0; cadena[indice] != '\0'; ++indice){



		// Obtener el char de la posición en donde está el índice

		// y por otro lado convertirla a minúscula



		// Así no importa si ponen 'A' o 'a', ambas letras serán convertidas a 'a'
```

```
            char letraActual = tolower(cadena[indice]);

            if (
                letraActual == 'a' ||
                letraActual == 'e' ||
                letraActual == 'i' ||
                letraActual == 'o' ||
                letraActual == 'u'
                )
            {
                vocales++;
            }
        }
        return vocales;
}
#include <stdio.h> // Para printf
#include <ctype.h> // Para toupper y isalpha
#include <clocale>

// Función que indica si un carácter es vocal, ¿hace falta más explicación?
int esVocal(char letra) {
    // Convertir a mayúscula para evitar hacer más comparaciones
```

```
    char letraEnMayuscula = (char) toupper(letra);
    return letraEnMayuscula == 'A' ||
            letraEnMayuscula == 'E' ||
            letraEnMayuscula == 'I' ||
            letraEnMayuscula == 'O' ||
            letraEnMayuscula == 'U';
}
```

contar consonantes de una cadena

```
int contarConsonantes(char cadena[]) {
    int consonantes = 0;// Almacenar la cantidad de consonantes
    int i = 0;// El índice para recorrer la cadena
    while (cadena[i]) {
        // Si es del alfabeto pero no es vocal
        if (isalpha(cadena[i]) && !esVocal(cadena[i])) {
            consonantes++;
        }
        i++;
    }
    return consonantes;
}


int main() {
```

```
    setlocale(LC_CTYPE,"spanish");
    char cadena[] = "Hola chicos vamos a contar las consonantes en lenguaje C";
    int consonantes = contarConsonantes(cadena);
    printf("La cadena '%s' \ntiene %d consonantes", cadena, consonantes);
    return 0;
}
```

1. ***EJERCICIOS***

```
#include<stdio.h>
#include<string.h>
#include<locale.h>
 int main(){
   setlocale(LC_CTYPE, "SPANISH");
                    char operacion[15];
          float numero_1, numero_2, resultado;
          printf("Programa para realizar operaciones matemáticas con dos números\n");
                    printf("Digite el nombre de la operación que quiere realizar (suma, resta,
multiplicacion, division):\n");
          scanf("%s", &operacion);
                    printf("Digite el primer número:\n");
          scanf("%f", &numero_1);
          printf("Digite el segundo número:\n");
          scanf("%f", &numero_2);
```

```
        if((strcmp(operacion, "suma")==0)||(strcmp(operacion, "SUMA")==0)){

                                        resultado=numero_1+numero_2;

                        printf("La suma de los números es: %.2f", resultado);

        }

        else if((strcmp(operacion, "resta")==0)||(strcmp(operacion, "RESTA")==0)){

                         resultado=numero_1-numero_2;

                        printf("El resultado de la resta es: %.2f", resultado);

        }

        else if((strcmp(operacion, "multiplicacion")==0)||(strcmp(operacion,
"MULTIPLICACION")==0)){

                        resultado=numero_1*numero_2;

                        printf("El resultado de la multiplicación es: %.2f", resultado);

        }

        else if((strcmp(operacion, "division")==0)||(strcmp(operacion, "DIVISION")==0)){

                        resultado=numero_1/numero_2;

                        printf("La división de los números es: %.2f", resultado);


}

}
```

2. EJERCICIO

```
#include<stdio.h>

#include<string.h>
```

```
#include<locale.h>

 int main(){

        setlocale(LC_CTYPE, "SPANISH");

        char tipo[1];

                printf("Programa para definir el incremento del límite de su tarjeta de crédito\n");

                printf("Digite el tipo de su tarjeta de credito (A,B,C). Si su tarjeta es de otro tipo
digite el número 0\n");

                scanf("%s", tipo);

                        if ((strcmp(tipo, "A")==0)||(strcmp(tipo, "a")==0)){

                        printf("El aumento del límite de su crédito será del 28 porcentaje");

        }

        else if((strcmp(tipo, "B")==0)||(strcmp(tipo, "b")==0)){

                        printf("El aumento del límite de su crédito será del 37 porcentaje");

        }

        else if((strcmp(tipo, "C")==0)||(strcmp(tipo, "c")==0)){

                        printf("El aumento del límite de su crédito será del 55 porcentaje");

        }

        else if(strcmp(tipo, "0")==0){

                        printf("El aumento del límite de su crédito será del 60 porcentaje");

        }

        else if(strcmp(tipo, "")!=0) {

                        printf("Lo siento, te has equivocado");
```

```
        }

}
```

Copiar el contenido de una cadena en otra cadena. Vea el código del Cuadro de Código 14.4. Mientras no se llega al carácter fin de cadena original, se van copiando uno a uno los valores de las variables de la cadena en copia. Al final, cuando ya se ha llegado al final en original, habrá que cerrar también la cadena en copia, mediante un carácter ASCII 0.

1. Que lea una cadena y la muestre al revés.

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{

        int indice,x;
   char frase[50];

        printf("Introduzca una frase: ");
   gets(frase);

        for(x = 0;x < 50;x++)
        {
        if (frase[x]=='\0')
        {
        indice=x;
        break;
    }
        }
        printf("La frase al reves es: \n\n");
     for(x = indice-1;x >=0;x--)
   {
        printf("%c",frase[x]);
   }
        printf("\n\n");
        system("PAUSE");
        return 0;
}
```

2. Que lea una cadena y diga cuantas vocales hay.

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
        int sum=0,x;
   char frase[50];

        printf("Introduzca una frase: ");
   gets(frase);
        for(x = 0;x < 50;x++)
        {
        switch (frase[x])
```

```
        {
        case 'a':
        sum++;
        break;
        case 'e':
        sum++;
        break;
        case 'i':
        sum++;
        break;
        case 'o':
             sum++;
        break;
        case 'u':
        sum++;
        break;
        default:
          break;
        }
        }

        printf("\n\nEn la frase hay %d vocales\n\n",sum);

   printf("\n\n");

        system("PAUSE");
        return 0;
}
```

3. Que lea una cadena y diga cuantas mayúsculas hay.

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
        int sum=0,x;
   char frase[50];

        printf("Introduzca una frase: ");
   gets(frase);

        for(x = 0;x < 50;x++)
   {
        if (frase[x]>=65 && frase[x]<=90)
        {
        sum++;
        }
        }

        printf("\n\nEn la frase hay %d mayúsculas\n\n",sum);

   printf("\n\n");
```

```
	system("PAUSE");
	return 0;
}
```

4. Que lea una cadena y la encripte sumando 3 al código ASCII de cada carácter. Mostrar por pantalla.

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
	int sum=0,x;
   char frase[50];

	printf("Introduzca una frase: ");
   gets(frase);

	for(x = 0; x < 50;x++)
	{
	if (frase[x]!='\0')
    {
	frase[x]=frase[x]+3;
	}
	}

	printf("\n\nLa nueva frase es:\n\n",sum);
   printf("\n\n%s\n\n",frase);
	printf("\n\n");

	system("PAUSE");
   return 0;
}
```

5. Que gestione los datos de stock de una tienda de comestibles, la información a recoger será: nombre del producto, precio, cantidad en stock. La tienda dispone de 10 productos distintos. El programa debe ser capaz de:

1. *Dar de alta un producto nuevo.*
2. *Buscar un producto por su nombre.*
3. *Modificar el stock y precio de un producto dado.*

```
#include <stdio.h>
#include <stdlib.h>

struct producto {
	char nombre[50];
	float precio;
	int cantidad;
};
```

```
int main(int argc, char *argv[])
{
        struct producto prod,productos[10];

        int x,opcion=1;

        for (x=0;x<10;x++)
    {
        strcpy(productos[x].nombre,"X");
        productos[x].precio=0;
        productos[x].cantidad=0;
        }

        while ((opcion==1 || opcion==2 || opcion==3) && (opcion!=4))
        {

        printf("1- Alta de producto\n");
        printf("2- Buscar por nombre\n");
        printf("3- Modificar stock y precio\n");
        printf("4- Salir\n");
        printf("Introduzca una opción: ");
      scanf("%d",&opcion);

        if (opcion==1)
    {
        printf("Introduzca un nombre: ");
        gets(prod.nombre);
        gets(prod.nombre);
        printf("Introduzca un precio: ");
        scanf("%f",&prod.precio);
        printf("Introduzca un stock: ");
        scanf("%d",&prod.cantidad);

      for(x = 9; x >=0; x--)
       {
        if (x!=0)
        {
        strcpy(productos[x].nombre,productos[x-1].nombre);
        productos[x].precio=productos[x-1].precio;
        productos[x].cantidad=productos[x-1].cantidad;
        }
        else
        {
        strcpy(productos[x].nombre,prod.nombre);
        productos[x].precio=prod.precio;
        productos[x].cantidad=prod.cantidad;
        }
        }
        printf("\nProducto creado. \n\n");
        }
        else if (opcion==2)
    {
        printf("Introduzca un nombre: ");
      gets(prod.nombre);
        gets(prod.nombre);
```

```
        for(x = 0; x < 10;x++)
       {

        if (strcmp(productos[x].nombre,prod.nombre)==0)
        {
        printf("\nNombre: %s\n",productos[x].nombre);
              printf("Precio: %f\n",productos[x].precio);
        printf("Cantidad en Stock: %d\n",productos[x].cantidad);
        }
        }
        printf("\n\n");
        }
        else if (opcion==3)
    {
        printf("Introduzca un nombre: ");
       gets(prod.nombre);
        gets(prod.nombre);

        for(x = 0; x < 10;x++)
       {
        if (strcmp(productos[x].nombre,prod.nombre)==0)
        {
        printf("Introduzca un precio: ");
              scanf("%f",&productos[x].precio);
        printf("Introduzca un stock: ");
        scanf("%d",&productos[x].cantidad);
        printf("\nProducto modificado.");
        }
        }
        printf("\n\n");
        }
        }

        system("PAUSE");
        return 0;
}
```

6. Que gestiona las notas de una clase de 20 alumnos de los cuales sabemos el nombre y la nota. El programa debe ser capaz de:

1. *Buscar un alumno.*
2. *Modificar su nota.*
3. *Realizar la media de todas las notas.*
4. *Realizar la media de las notas menores de 5.*
5. *Mostrar el alumno que mejores notas ha sacado.*
6. *Mostrar el alumno que peores notas ha sacado.*

```
#include <stdio.h>
#include <stdlib.h>
```

```
struct alumno {
        char nombre[50];
   float nota;
};

int main(int argc, char *argv[])
{
        struct alumno alum,alumnos[5];

   int x,opcion=1;
        float sum=0,cont=0,mejor,peor;

        for (x=0;x<5;x++)
   {
        printf("Introduzca nombre alumno:");
        gets(alumnos[x].nombre);
        gets(alumnos[x].nombre);
        printf("Introduzca nota:");
        scanf("%f",&alumnos[x].nota);
        }

        while ((opcion==1 || opcion==2 ||
        opcion==3 || opcion==4 ||
        opcion==5 || opcion==6) && (opcion!=7))
        {

        printf("1- Buscar un alumno\n");
        printf("2- Modificar nota\n");
        printf("3- Media de todas las notas\n");
        printf("4- Media de todas las notas inferiores a 5\n");
        printf("5- Alumno con mejores notas\n");
        printf("6- Alumno con peores notas\n");
        printf("7- Salir\n");
        printf("Introduzca una opción: ");
      scanf("%d",&opcion);

        if (opcion==1)
     {
        printf("Introduzca un nombre: ");
        gets(alum.nombre);
        gets(alum.nombre);

       for(x = 0; x < 5;x++)
        {
        if (strcmp(alumnos[x].nombre,alum.nombre)==0)
        {
        printf("\nNombre: %s\n",alumnos[x].nombre);
        printf("Nota: %f\n",alumnos[x].nota);
        }
        }
        printf("\n\n");
        }
        else if (opcion==2)
     {
        printf("Introduzca un nombre: ");
```

```
    gets(alum.nombre);
    gets(alum.nombre);

   for(x = 0; x < 5;x++)
   {
    if (strcmp(alumnos[x].nombre,alum.nombre)==0)
    {
    printf("Introduzca una nota: ");
    scanf("%f",&alumnos[x].nota);
    printf("\nNota modificada.");
    }
    }
    printf("\n\n");
    }
    else if (opcion==3)
    {
    sum=0;
    for(x = 0; x < 5;x++)
  {
    sum=sum+alumnos[x].nota;
    }
    printf("\nLa media de las notas es de: %f \n",(sum/5));
}
    else if (opcion==4)
    {
    sum=0;
    cont=0;
    for(x = 0; x < 5;x++)
  {
    if (alumnos[x].nota<5)
    {
    sum=sum+alumnos[x].nota;
    cont++;
    }
    }
      printf("\nLa media de las notas inferiores a 5 es: %f \n",sum/cont);
}
    else if (opcion==5)
{
    mejor=0;
    for(x = 0; x < 5;x++)
    {
    if (alumnos[x].nota>mejor)
    {
    mejor=alumnos[x].nota;
    alum.nota=alumnos[x].nota;
    strcpy(alum.nombre,alumnos[x].nombre);
    }
    }
    printf("\nEl alumno con mejores notas es: %s \n",alum.nombre);
}
    else if (opcion==6)
       {
    peor=10;
    for(x = 0; x < 5;x++)
```

```
        {
        if (alumnos[x].nota<peor)
        {
        peor=alumnos[x].nota;
        alum.nota=alumnos[x].nota;
        strcpy(alum.nombre,alumnos[x].nombre);
        }
        }
        printf("\nEl alumno con peores notas es: %s \n",alum.nombre);
    }
        }

        system("PAUSE");
        return 0;
}
```

7. Escribir un nombre y contar los caracteres

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main()
        {
      char nombre[20]; //string de 20 caracteres
      int longitud_string;

      printf("Escribe tu nombre: ");
      scanf("%s", &nombre);
      printf("Te llamas: %s \n", nombre); //muestro nombre por pantalla

      longitud_string = strlen(nombre); //strlen cuenta caracteres
      printf( "Tu nombre: %s, tiene %i caracteres. \n", nombre, longitud_string );

        system("PAUSE");
        }
```

8. El ahorcado

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
int main() {
        char frase[60],rep[100],temporal[100];
  char pal;
        int longitud,i,j,inicial,acertado=0,temp=0,oportunidades=5;
        int repetido=0,gano=0;

        printf("\t Juego del Ahorcado\n");
        printf("Introduzca la palabra a adivinar: ");
  gets(frase);

        system("cls");

  longitud = 0;
```

```
        inicial = 0;
        j = 0;

rep[0] = ' ';
        rep[1] = '\0';


        do {
                system("cls");
        temp=0;

        if(inicial == 0) {
        for(i=0;i<strlen(frase);i++) {
        if(frase[i] == ' ') {
        temporal[i] = ' ';
        longitud++;
        }
        else {
        temporal[i] = '_';
        longitud++;
        }
        }
        }

        inicial = 1;

        temporal[longitud] = '\0';

        for(i=0;i<strlen(rep);i++) {
        if(rep[i] == pal) {
        repetido = 1;
        break;
        }
        else {
       repetido = 0;
        }
        }

        if(repetido == 0) {
    for(i=0;i<strlen(frase);i++) {
                if(frase[i] == pal) {
                 temporal[i] = pal;
                  acertado++;
                  temp=1;
                }
        }
        }

        if(repetido == 0) {
        if(temp == 0) {
        oportunidades = oportunidades - 1;
        }
        }
        else {
        printf("Ya se ha introducido este caracter");
```

```
        printf("\n\n");
            }

            printf("\n");

            for(i=0;i<strlen(temporal);i++) {
            printf(" %c ",temporal[i]);
        }

            printf("\n");

            if(strcmp(frase,temporal) == 0) {
                    gano = 1;
                    break;
            }

            printf("\n");
        printf("Letras Acertadas: %d",acertado);
            printf("\n");
            printf("Oportunidades Restantes: %d",oportunidades);
        printf("\n");

            rep[j] = pal;
        j++;

            if (oportunidades==0)
            {
            break;
            }

            printf("Introduzca una letra:");
            scanf("\n%c",&pal);

            }while(oportunidades != 0);

            if(gano) {
                    printf("\n\n");
        printf("Enhorabuena, has ganado.");
    }
            else {
                    printf("\n\n");
            printf("Has perdido.");
            }

            printf("\n\n");
            system("PAUSE");
    return 0;

}
```

EJERCICIO PROMEDIO DE NOTAS

El siguiente programa pide al usuario ingresar las notas de uno o más alumnos, y va mostrando los promedios de cada uno de ellos:

```
#include <stdio.h>

float promedio(int valores[], int cantidad) {
    int i;
    float suma = 0.0;

    for (i = 0; i < cantidad; ++i)
        suma += valores[i];

    return suma / (float) cantidad;
}


int main() {

    int notas[10];
    char nombre[20];
    char opcion[3];
    int n, i;

    do {
        printf("Ingrese nombre del alumno: ");
        scanf("%s", nombre);

        printf("Cuantas notas tiene %s? ", nombre);
        scanf("%d", &n);

        for (i = 0; i < n; ++i) {
            printf("  Nota %d: ", i + 1);
            scanf("%d", &notas[i]);
        }

        printf("El promedio de %s es %.1f\n", nombre, promedio(notas, n));

        printf("Desea calcular mas promedios (si/no)? ");
        scanf("%s", opcion);

    } while (opcion[0] == 's' || opcion[0] == 'S');

    return 0;
}
```