TEMA 2.
LENGUAJE C. CONCEPTOS BÁSICOS Y PROGRAMACIÓN ELEMENTAL.

PARTE 2: INSTRUCCIONES DE CONTROL

- **En los programas C estudiados hasta el momento.**
  - **Las instrucciones se ejecutan en el mismo orden en que aparecían en el programa.**
  - **Cada instrucción se ejecuta una vez.**
- **En general, los programas pueden requerir.**
  - **Realizar comprobaciones de condiciones que sean verdaderas y falsas y ejecutar una cosa u otra en función de si se cumple la condición o no (EJECUCIÓN CONDICIONAL).**
  - **Ejecución repetida de un grupo de sentencias mientras se cumple una condición (BUCLES).**

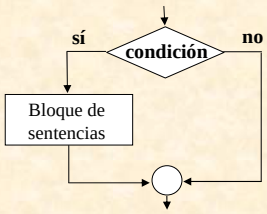
1. Ejecución condicional: IF

SENTENCIA IF

Sintaxis

```
if (expresión)
{
    sentencia o grupo de sentencias;
}
```

Se ejecuta la sentencia o grupo de sentencias sólo si expresión es cierta (distinta de cero).



EJEMPLO 1:

```
// Función que calcula el valor absoluto de un numero

main()
{
    int x;
    printf("Introduzca x:  ");
    scanf("%d", &x);

    if(x<0)
        x = -x;

    printf("Valor absoluto = %d\n", x);
}
```


1. Ejecución condicional: IF

SENTENCIA IF

EJEMPLO 2:

```
// Función que calcula la media de dos
// números sólo si son positivos.

main()
{
    float x, y, media;
    printf("Introduzca x:  ");
    scanf("%f", &x);
    printf("Introduzca y:  ");
    scanf("%f", &y);

    if((x>=0) && (y>=0))
    {
        media = (x+y) / 2;
        printf("Media = %f\n", media);
    }
}
```

EJEMPLO 3:

```
// Función que calcula la media de dos
// números con la condición de que al
// menos uno de ellos sea positivo

main()
{
    float x, y, media;
    printf("Introduzca x:  ");
    scanf("%f", &x);
    printf("Introduzca y:  ");
    scanf("%f", &y);

    if((x>=0) || (y>=0))
    {
        media = (x+y) / 2;
        printf("Media = %f\n", media);
    }
}
```

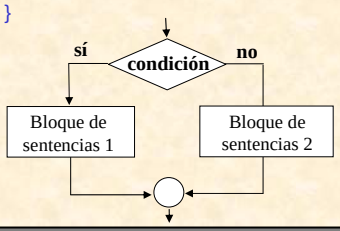
1. Ejecución condicional: IF-ELSE

SENTENCIA IF-ELSE

Si no se cumple la condición de la sentencia *if*, puede añadirse una sentencia alternativa mediante la instrucción *else*.

Sintaxis

```
if (expresión)
{
    sentencia o grupo de sentencias;
}
else
{
    sentencia o grupo de sentencias;
}
```



EJEMPLO 1:

```
// Función que indica si un número es par o impar.

main()
{
    int x;
    printf("Introduzca x:  ");
    scanf("%d", &x);

    if(x % 2 == 0)
        printf("El numero %d es par\n", x);
    else
        printf("El numero %d es impar\n", x);
}
```


1. Ejecución condicional: IF-ELSE

SENTENCIA IF-ELSE

EJEMPLO 2:

```
// Función que lee un caracter. Si es una vocal minúscula, se muestra la mayúscula correspondiente, y si
// es otro caracter, se muestra el código ASCII de dicho caracter.
main()
{
    char letra;
    printf("Introduzca un caracter: ");
    scanf("%c", &letra);

    if((letra == 'a') || (letra == 'e') || (letra == 'i') || (letra == 'o') || (letra == 'u'))
    {
        printf("Se ha introducido una vocal minuscula\n");
        printf("La mayuscula correspondiente es %c\n", letra-32);
    }
    else
    {
        printf("No se ha introducido una vocal minuscula");
        printf("El codigo ASCII de %c es %d\n", letra, letra);
    }
}
```

1. Ejecución condicional: ELSE IF

SENTENCIA ELSE IF

- Hasta ahora, si se cumple la condición se ejecuta una sentencia, y si no se cumple, se ejecuta otra.
- Mediante *else if* es posible elegir entre más de dos alternativas.
- Se ejecuta la primera sentencia cuya condición se cumple.

EJEMPLO 1:

```
// Función que lee una nota y la convierte a aprobado, notable…
main()
{
    int nota;
    printf("Introduzca una nota: ");
    scanf("%d", &nota);
    if(nota < 5)
        printf("Suspenso\n");
    else if(nota < 7)
        printf("Aprobado\n");
    else if(nota < 9)
        printf("Notable\n");
    else if(nota < 10)
        printf("Sobresaliente\n");
    else
        printf("Nota incorrecta\n");
}
```


1. Ejecución condicional: ELSE IF

SENTENCIA ELSE IF

EJEMPLO 2:

```
// Función que lee un carácter e indica si es una letra, un número o ninguna de ellas
main()
{
    char letra;
    printf("Introduzca un caracter: ");
    scanf("%c", &letra);
    if((letra >= 48) && (letra <= 57))
    {
        printf("Se ha introducido un numero\n");
    }
    else if(((letra >= 65) && (letra <= 90)) || ((letra >= 97) && (letra <= 122)))
    {
        printf("Se ha introducido una letra\n");
    }
    else
    {
        printf("No se ha introducido una letra ni un numero\n");
    }
}
```

1. Ejecución condicional: ?

EL OPERADOR CONDICIONAL ?

Sintaxis:

variable = expresión1 ? expresion2 : expresión3;

- Si expresión1 es cierta, variable toma el valor de expresión2.
- Si expresión1 es falsa, variable toma el valor de expresión3.

Ejemplos: (a, b, maximo y valorAbs son variables enteras).

```
maximo = (a>b) ? a : b;
valorAbs = (a>0) ? a : -a;
```

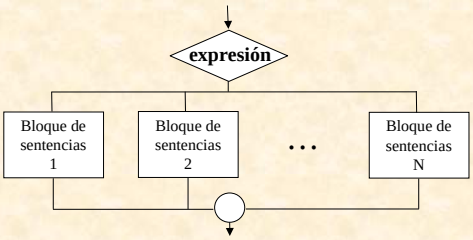

1. Ejecución condicional: SWITCH

SENTENCIA SWITCH:

Sintaxis

```
switch (expresion entera)
{
    case valor1:
        sentencias1;
        break;
    case valor2:
        sentencias2;
        break;
    …
    …
    default:
        sentenciasN;
}
```

- Sirve para seleccionar una sentencia o grupo de sentencias entre varias disponibles.
- La selección se basa en el valor de una expresión que aparece junto a *switch*.
- Cuando se ejecuta esta instrucción, se evalúa la expresión entera y se transfiere el control al grupo de instrucciones cuya etiqueta *case* tenga el mismo valor que el de expresión. Se ejecutan las instrucciones hasta que se encuentra la etiqueta *break*.
- Si ninguna etiqueta case tiene un valor coincidente, se transfiere el control a las instrucciones tras la etiqueta *default*.



1. Ejecución condicional: SWITCH

SENTENCIA SWITCH

EJEMPLO 1:

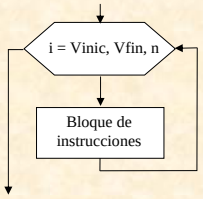
```
// Función que simula una máquina de refrescos

#define COLA 1
#define NARANJA 2
#define AGUA 3

main()
{
    int opcion;

    printf("Elija una opcion: ");
    printf("1: Cola, 2: Naranja, 3: Agua\n");
    scanf("%d", &opcion);

    switch(opcion)
    {
        case COLA:
            printf("Precio = 0.75 euros\n");
            break;
        case NARANJA:
            printf("Precio = 0.60 euros\n");
            break;
        case AGUA:
            printf("Precio = 0.45 euros\n");
            break;
        default:
            printf("Opcion incorrecta\n");
    }
}
```


2. Bucles: FOR

SENTENCIA FOR:

Sintaxis

```
for (inicialización; test; actualización)
{
    grupo de sentencias
}
```



- ✓ La sentencia *for* se usa para generar bucles, en los cuales, un grupo de instrucciones se ejecuta de forma repetida, mientras se satisface una condición.
- ✓ La sentencia *for* ejecuta repetidamente el grupo de sentencias mientras *test* sea cierto.
  - × *inicialización*: se ejecuta una sola vez al comienzo del bucle. Sirve para inicializar el índice que controla la repetición del bucle.
  - × *test*: se chequea antes de iniciar cada bucle: *condición de entrada*.
  - × *actualización*: se ejecuta tras cada bucle. Sirve para actualizar el índice del bucle.
- ✓ El bucle for se suele utilizar cuando se conocen a priori el número de pasadas a ejecutar (número de repeticiones del bucle).

2. Bucles: FOR

SENTENCIA FOR:

EJEMPLO 1:

```
// Función que calcula y saca por pantalla valores para la función y = 2x+100 para x desde 0 hasta 50

main()
{
    int x, y;
    for(x=0; x<=50; x++)
    {
        y = 2*x+100;
        printf("x=%d, y=%d\n", x, y);
    }
}
```


2. Bucles: FOR

SENTENCIA FOR:

EJEMPLO 2:

```
// Funcion que pide n valores al usuario, los suma y muestra el resultado. El valor de n también se pide.

main()
{
    int n, i;
    float x, suma=0;

    printf("Cuantos numeros vas a introducir?  ");
    scanf("%d", &n);

    for(i=1; i<=n; i++)
    {
        printf("Introduzca valor %d: ", i);
        scanf("%f", &x);
        suma += x;
    }
    printf("La suma es %f\n", suma);
}
```

2. Bucles: FOR

SENTENCIA FOR:

Observaciones:

- Pueden eliminarse uno, dos o los tres campos.
- Pueden existir varias inicializaciones y/o actualizaciones (separadas por comas).
- Cualquier expresión es válida para los campos.

Ejemplos:

```
for (x=0; x<100; )
  printf("%d\n",x++);
```

```
for (x=0, y=0; x<100; x++, y+=2)
  printf("x:%d, y:%d\n", x, y);
```

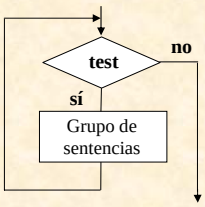
```
for (x=0, printf("Comienzo del bucle\n"); x<100; x++)
  printf("%d\n",x);
```


2. Bucles: WHILE Y DO-WHILE

SENTENCIA WHILE:

Sintaxis

```
while (test)
{
    grupo de sentencias
}
```

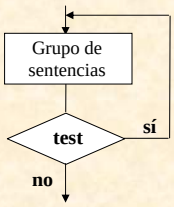


- ✓ Ejecuta repetidamente las sentencias mientras *test* sea verdadero (valor diferente a 0).
- ✓ *test*: se chequea antes de iniciar cada bucle: *condición de entrada*.
- ✓ Se suele utilizar cuando no se conoce de antemano el número de repeticiones.

SENTENCIA DO-WHILE:

Sintaxis

```
do
{
    grupo de sentencias
}
while (test)
```



- ✓ ejecuta repetidamente las sentencias mientras *test* sea cierto.
- ✓ *test*: se chequea después de ejecutar cada bucle: *condición de salida*.

2. Bucles: WHILE Y DO-WHILE

✓ **Ejemplo *while***

```
x=0;
while (x<MAXIMO)
  printf("%d\n",x++);
```

✓ **Ejemplo *do-while***

```
x=0;
do
  printf("%d\n",x++);
while (x<MAXIMO);
```

- Con 'do-while' el bucle siempre se ejecuta al menos una vez.
- Con 'for' o 'while' puede que no se ejecute ninguna vez.

2. Bucles: WHILE Y DO-WHILE

SENTENCIAS WHILE Y DO-WHILE:

EJEMPLO 1:

```
// Funcion que pide n valores al usuario, los suma y muestra el resultado. El valor de n también se pide.

main()
{
    int n, i=0;
    float x, suma=0;

    printf("Cuantos numeros vas a introducir?  ");
    scanf("%d", &n);

    while(i<n)
    {
        printf("Introduzca valor %d: ", i+1);
        scanf("%f", &x);
        suma += x;
        i++;
    }
    printf("La suma es %f\n", suma);
}
```

2. Bucles: WHILE Y DO-WHILE

SENTENCIAS WHILE Y DO-WHILE:

EJEMPLO 2:

```
// Funcion que pide valores al usuario hasta que se introduce uno negativo. Al final devuelve el valor medio.

main()
{
    int i=0;
    float x, media=0;

    printf("Introduzca valor:  ");
    scanf("%f", &x);

    while(x>0)
    {
        media += x;
        i++;
        printf("Introduzca valor: ");
        scanf("%f", &x);
    }
    media /= i;
    printf("La media es %f\n", media);
}
```


2. Bucles: BUCLES ANIDADOS

BUCLES ANIDADOS:

```
// Función que calcula los divisores de los valores introducidos por el usuario.
// Antes de comenzar, el programa pregunta cuántos números se van a pedir

main()
{
    int n, x, i, j;

    printf("Cuantos numeros se van a introducir?  ");
    scanf("%d", &n);

    for(i=0; i<n; i++)
    {
        printf("Introduzca numero: ");
        scanf("%d", &x);
        printf("Los divisores de %d son: ", x);
        for(j=1; j<=x; j++)
            if(x%j == 0)
                printf("%d ", j);
        printf("\n");
    }
}
```

2. Bucles: BUCLES ANIDADOS

SENTENCIA BREAK:

- Hace finalizar el bucle.

```
// Muestra el código ASCII de la tecla pulsada
// hasta que se pulsa el cero

main()
{
    while (1)
    {
        printf("Pulse una tecla (0 para acabar)\n");
        scanf("%c",&tecla);
        if (tecla=='0')
        {
            printf("bucle finalizado por usuario\n");
            break;
        }
        printf("%c: código ASCII %d\n",tecla,tecla);
    }
}
```

SENTENCIA CONTINUE:

- No finaliza la iteración actual. Salta a la siguiente iteración.

```
// Traduce a mayúsculas 10 letras minúsculas
// que introduce el usuario.
// Si no se introduce una letra minúscula,
// no hace nada.

main()
{
    for (i=0;i<10;i++)
    {
        printf("introduzca una letra minúscula\n");
        scanf("%c",&tecla);
        if (tecla<97 || tecla>122)
            continue;
        printf("%c mayúscula: %c\n", tecla, tecla-32);
    }
}
```