EXPRESIONES LOGICAS

- Necesarias para tomar decisiones.
- Dos posibles valores: a) 0 (cero) si la expresión es falsa.

  b) 1 (uno) si la expresión es cierta.

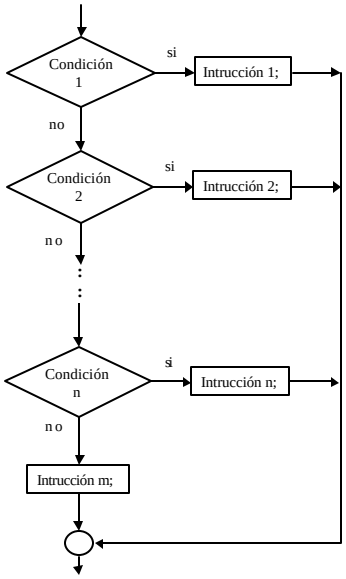
Sintaxis básica:

variable operadorconstante;

variable operadorvariable;

Operador ➔ de **Igualdad** ó de **Relación**.

Selección Múltiple.



SWITCH ➔ Otra estructura de selección múltiple. Usando en sustitución del if / else if / else.

Sintaxis:

```
switch (variable)
{
 case valor1:
      instrucción 1; .....
      break;
 case valor2:
      instrucción 2; .....
      break;
 case valorn:
      instrucción n; .....
      break;
 default:
      instrucción m;
}
```

Esta estructura está compuesta por varios comandos:

- **switch**: identifica el comienzo de la estructura, entre paréntesis esta la variable a ser evaluada.
- **case** : las alternativas, comparan el valor en la "condición".
- **break** : en la alternativa (case) correcta se hacen las instrucciones hasta que se encuentre un "break", que rompe la estructura.
- **default** : hace el trabajo del "else" final, es opcional.

ESTRUCTURA DE REPETICION

Se requieren 4 elementos:

1) Instrucción de repetición, define los límites de las instrucciones que se van a repetir. En C hay 3 tipos:
   - while
   - for
   - do / while

2) Condición o Expresión Lógica, la estructura se repite mientras la condición se cumpla o sea cierta.

3) Inicialización de la variable de control, la condición tiene que ser inicializada antes de entrar a la estructura.

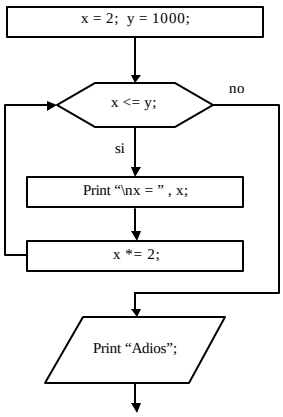
4) Negación de la condición, la variable de control tiene que provocar que la condición sea falsa en algún momento.

Además existen 2 tipos de implementación:

1) Prueba Anterior ➔ la condición se verifica antes de entrar a la estructura (entrada controlada).

2) Prueba Posterior ➔ la condición se verifica después de ejecutar las instrucciones una vez (salida controlada).

ESTRUCTURA DE REPETICION (WHILE)



Sintaxis:

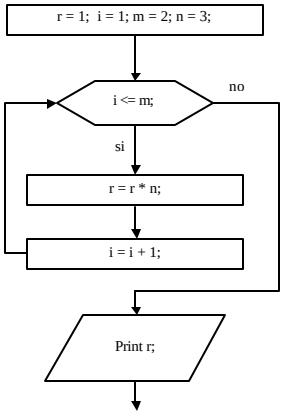
```
while (condición)
{
  instrucción 0;
  instrucción 1;
  instrucción 2;
      :
  instrucción n;
}
```

Ejemplos:

```
x = 2;  y = 1000;
while (x <= y)
 {
   cout << " \nx = " << x;
   x *= 2;
 }
cout << "Adios";
```

```
contador = 1;
while (contador < 10)
 {
   cout <<  contador <<" \n";
   contador ++;
 }
```

ESTRUCTURA DE REPETICION (FOR)



Sintaxis:

```
for (inicialización ; condición ; control )
{
  instrucción 0;
  instrucción 1;
  instrucción 2;
      :
  instrucción n;
}
```

1. **Inicialización**: inicializamos la variable de control o el contador de la estructura.
2. **Condición**: expresión lógica.
3. **Control**: se incrementa o decrementa la variable de control.

Ejemplos:

```
r = 1;  m = 3;  n = 3;
for (i = 1; i <= m ; i++)
 {
   r = r * n;
 }
cout << r;
```

```
for (j = 5; j <= 25 ; j += 5)
 {
   if (j <= 23)
   cout << "prueba";
 }
```

Existen 2 instrucciones que se pueden usar en las estructuras de repetición.

1) break; ➔ obliga a el procesador a salir de la estructura.

```
contador = 1;
while (contador <= 10)
 {
   cout << "Entre un número:";
   cin >> num;
   if (num < 76)
   {
    cout << "Perdiste, ja, ja, ja \n";
    break;
   }
   contador ++;
 }
```

2) continue; ➔ cuando el procesador ve un "continue", la próxima iteracción comienza inmediatamente, y se obvian las instrucciones que restan del ciclo.

```
contador = 1;
while (contador <= 10)
 {
   cout << "Entre la nota del examen:";
   cin >> nota;
   if (nota < 0 || nota > 100)
   {
    cout << "La nota no es válida\n";
    continue;
   }
   total = nota * 0.20;
   cout << total;
   contador ++;
 }
```

ESTRUCTURA DE REPETICION (DO/WHILE)

Ejecuta las instrucciones encerradas entre {}, una vez antes de verificar la condición ("Post-test structure").

Sintaxis:

```
do
{
  instrucción 0;
  instrucción 1;
  instrucción 2;
      :
  instrucción n;
} while (condición)
```

Ejemplos:

```
do{
   cout << " \n Entre su contraseña: ";
   cin >> pin;
 }  while (pin != card_pin)
```

EL while y el for son estructuras casi siempre intercambiables. Mayormente es cuestion de estilo.

for ➔ más usado para condición de cuenta fija ("fixed count").

While ➔ más usado para condición de cuenta variable. ("variable count").

Ambas son estructuras de prueba anterior ("Pre-test structure").

ESTRUCTURA DE REPETICION ANIDADAS

Las combinaciones de ciclos ("loops") son arbitrarias. Se usan las que hagan falta, de los tipos que hagan falta.

Sintaxis:

```
while (condición)
{
  instrucción 0;
  instrucción 1;
 for (inicialización ; condición ; control )
  {
    instrucción 2;
    instrucción 3;
    instrucción 4;
        :
    instrucción n;
  }
  instrucción n+1;
      :
  instrucción m;
}
```