

CARRERA PROFESIONAL

DESARROLLO DE SISTEMAS DE INFORMACIÓN

FUNDAMENTOS DE PROGRAMACIÓN

Tema

**FUNDAMENTOS DE LAS ESTRUCTURAS DE
CONTROL CONDICIONAL**

CONCEPTOS BÁSICOS

Las **estructuras de control** controlan el flujo de ejecución de un programa o función. Las estructuras de control permiten combinar instrucciones o sentencias individuales en una simple unidad lógica con un punto de entrada y un punto de salida.

Las instrucciones o sentencias se organizan en tres tipos de estructuras de control que sirven para controlar el flujo de la ejecución: *secuencia*, *selección (decisión)* y *repetición*. Hasta este momento sólo se ha utilizado el flujo secuencial. Una **sentencia compuesta** es un conjunto de sentencias encerradas entre llaves ({ y }) que se utiliza para especificar un flujo secuencial.

```
{  
    sentencia1;  
    sentencia2;  
    .sentencian;  
}
```

El control fluye de la *sentencia₁* a la *sentencia₂*, y así sucesivamente. Sin embargo, existen problemas que requieren etapas con dos o más opciones o alternativas a elegir en función del valor de una condición o expresión¹.

¹ <https://www.programarya.com/Cursos/Fundamentacion/Estructuras-de-Control>

1. Tipos de Estructura Condicionales²

Hay tres tipos básicos de estructuras condicionales

a. Estructuras Condicionales Simples

La estructura de control de selección principal es una sentencia if. La sentencia if tiene dos alternativas o formatos posibles. El formato más sencillo tiene la sintaxis siguiente:

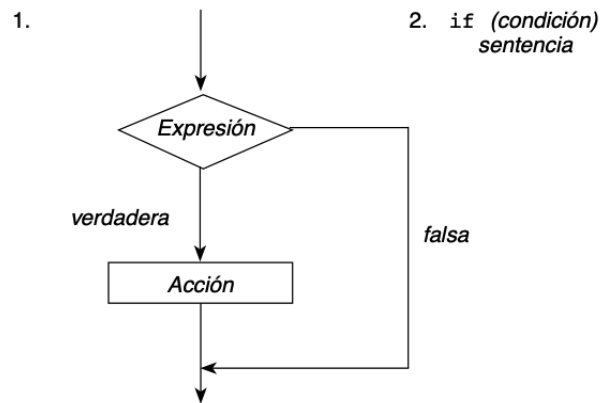
if (Expresión) Acción

Expresión lógica que determina si la acción se ha de ejecutar

Acción se ejecuta si la expresión lógica es verdadera

La sentencia if funciona de la siguiente manera. Cuando se alcanza la sentencia if dentro de un programa, se evalúa la expresión entre paréntesis que viene a continuación de if. Si Expresión es verdadera, se ejecuta Acción; en caso contrario no se ejecuta Acción (en su formato más simple, Acción es una sentencia simple, y en los restantes formatos, es una sentencia compuesta). En cualquier caso la ejecución del programa continúa con la siguiente sentencia del programa. La Figura 4.1 muestra un diagrama de flujo que indica el flujo de ejecución del programa.

² <https://desarrolloweb.com/articulos/2225.php>



Otro sistema de representar la sentencia if es:

```
if (condición) sentencia;
```

condición es una expresión entera

sentencia es cualquier sentencia ejecutable, que se ejecutará sólo si la condición toma un valor distinto de cero.

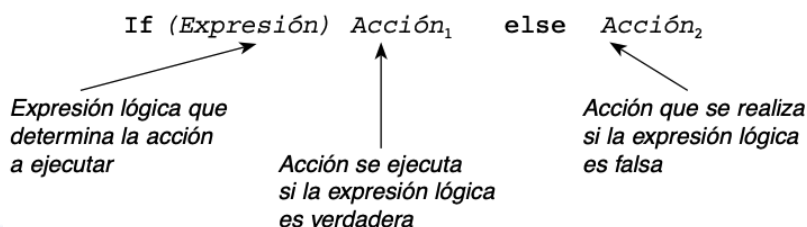
Ejemplo

Prueba de divisibilidad.

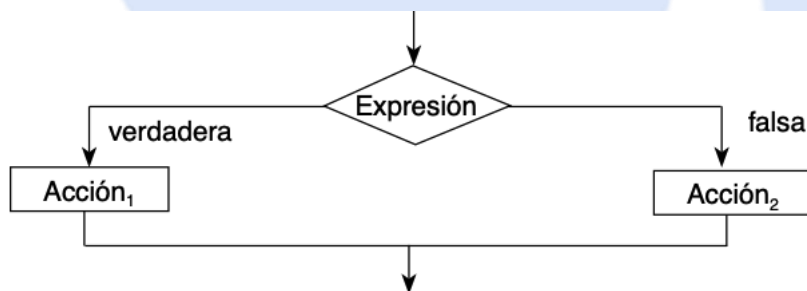
```
void main() {  
  
    int n, d;  
  
    cout << "Introduzca dos enteros:";  
  
    cin >> n >> d;  
  
    if (n%d == 0) cout << n << "es divisible por" << d << endl;  
  
}
```

b. Estructuras Condicionales Dobles

Un segundo formato de la sentencia if es la sentencia if-else. Este formato de la sentencia if tiene la siguiente sintaxis:



En este formato Acción₁ y Acción₂ son individualmente o bien una única sentencia que termina en un punto y coma (;) o un grupo de sentencias encerrado entre llaves. Cuando se ejecuta la sentencia if- else, se evalúa Expresión. Si Expresión es verdadera, se ejecuta Acción₁ y en caso contrario se ejecuta Acción₂.


Ejemplo

1. if (salario >= 100.000)

 salario_net = salario — impuestos;

else

```
salario_neto = salario;
```

Si salario es mayor que 100.000, se calcula el salario neto, restándole los impuestos; en caso contrario (else), el salario neto es igual al salario (bruto).

```
2. if (Nota >= 5)
```

```
    cout << "Aprobado" << endl;
```

```
else
```

```
    cout << "Suspenso" << endl;
```

c. Estructuras condicionales Múltiples

Hasta este momento, las sentencias if implementan decisiones que implican una o dos alternativas. En esta sección se mostrará cómo se puede utilizar la sentencia if para implementar decisiones que impliquen diferentes alternativas.

Una sentencia if es anidada cuando la sentencia de la rama verdadera o la rama falsa es a su vez una sentencia if. Una sentencia if anidada se puede utilizar para implementar decisiones con varias alternativas o multi-alternativas.

Sintaxis

```
if (condición1)
```

```
    sentencia1
```

else if (condición₂)

sentencia₂

.

.

.

else if (condición_n)

sentencia_n

else

sentencia_e

Ejemplo

Incrementar contadores de números positivos, números negativos o ceros

```
if (x > 0)
    num_pos = num_pos + 1;
else
    if (x < 0)
        num_neg = num_neg + 1;
    else
        num_ceros = num_ceros + 1;
```



INSTITUTO
KHIPU