Tema 1: Introducción a la programacción

Estructuras de control en C/C++

Este material ha sido desarrollado en su totalidad por Román Ginés Martínez Ferrández (rgmf@riseup.net) salvo referencias al pie de página.

Sumario

1. Estructuras de control

Los programas constan de una secuencia de instrucciones que el procesador ejecuta en el orden en que aparecen escritas.

Esta secuencia puede ser interrumpida, modificada, pueden aparecer bifurcaciones, repeticiones, etc. Es aquí donde aparecen las denominadas **estructuras de control** que rompen con dicha secuencia y aparecen en todos los programas por simples que sean.

1.1. Estructuras de selección

Estas estructuras ejecutan un bloque de instrucciones u otro según se cumpla o no una condición.

a) Selección if simple

Permite redirigir la acción a un bloque u otro en función de una condición: si es verdadera se ejecuta un bloque y si no se ejecuta otro bloque. **El bloque *else* es opcional**.

```
if (<condición>)
{
   <instrucción 1.1>;
   <instrucción 1.2>;
   ...
   <instrucción 1.N>;
}
else
{
   <instrucción 2.1>;
   <instrucción 2.2>;
   ...
   <instrucción 2.N>;
}
```

b) Selección if múltiple

Se trata de múltiples *if-else* anidados.

Se emplea cuando se necesita hacer varias comprobaciones de manera que hay varios bloques de instrucciones de las cuales se ejecutará una en función de la condición que sea verdadera.

Opcionalmente podemos acabar con un *else* por si ninguna de las condiciones fuera verdadera.

A continuación se muestra la sintaxis que sigue una selección múltiple:

```
if (<condición 1>)
{
   <instrucción 1.1>;
   <instrucción 1.2>;
   ...
   <instrucción 1.N>;
}
else if (<condición 2>)
{
   <instrucción 2.1>;
   <instrucción 2.2>;
   ...
   <instrucción 2.N>;
}
else if (<condición 3>)
{
   <instrucción 3.1>;
   <instrucción 3.2>;
   ...
   <instrucción 3.N>;
}
...
else
{
   <instrucción M.1>;
   <instrucción M.2>;
   ...
   <instrucción M.N>;
}
```

Solo uno de los bloques de arriba se ejecutará, en función de la condición que sea verdadera.

1.2. Estructuras de control iterativas

Estas estructuras permiten repetir un bloque de instrucciones:

- un número determinado de veces (con contador),
- o un número indeterminado de veces, es decir, mientras sea verdadera una condición o hasta que se de una condición.

a) Repeticiones con contador

Estas estructuras de control se repiten un número de veces para lo cual se inicializa un contador que se va incrementando al final de cada iteración y se detiene cuando llega a un cierto número.

A estas estructuras se las llama **estructuras for** y esta es su sintaxis:

```
for (<inicialización>; <condición>; <incremento>)
{
   <instrucción 1>;
   <instrucción 2>;
   ...
   <instrucción N>;
}
```

Veamos un ejemplo para aclarar esta estructura. El siguiente ejemplo de código muestra por pantalla los números del 0 al 9:

```
for (int i = 0; i < 10; i++)
{
   cout << i;
}
```

b) Repeticiones con condición inicial

Con estas estructuras conseguimos que se repita un bloque de instrucciones mientras se cumpla la condición. A estas estructuras se las conoce también como **estructuras while** y su sintaxis es la siguiente:

```
while (<condición>)
{
   <instrucción 1>;
   <instrucción 2>;
   ...
   <instrucción N>;
}
```

Veamos un ejemplo en el que se pide al usuario un número positivo y lo vuelve a pedir si no es positivo:

```
int numero;

cout << “Dame un número positivo (mayor que 0): “;
cin >> numero;

while (numero <= 0)
{
   cout << “Insisto, dame un númeor mayor que 0: “;
   cin >> numero;
}
```

c) Repeticiones con condición final

Estas se diferencia de la anterior en que siempre se ejecuta su bloque de instrucciones una vez y luego dependerá de la condición que siga repitiendo este bloque o salga.

```
do
{
   <instrucción 1>;
   <instrucción 2>;
   ...
   <instrucción N>;
} while (<condición>);
```

Veamos el mismo ejemplo que arriba usando la repetición con condición final:

```
int numero;

do
{
   cout << “Dame un númeor mayor que 0: “;
   cin >> numero;
} while (numero <= 0);
```

2. Condiciones – operadores relacionales

Como has podido ver muchas estructuras de control requieren trabajar con condiciones. Estas condiciones se construyen con operadores lógicos que en C/C++ son:

Operador	Significado
<	Menor
<=	Menor o igual
>	Mayor
>=	Mayor o igual
==	Igual
!=	Distinto

Estos operadores devuelve **verdadero** o **falso** de la misma manera que una suma de números enteros da como resultado un número entero.

3. Condiciones múltiples – operadores lógicos y tabla de verdad

Podemos concatenar condiciones con los operadores **&&** (y lógico) y **||** (o lógico). Además, también podemos utilizar el operador lógico **!** (no lógico, niega una condición).

Para entender estos operadores vamos a ver sus tablas de verdad:

Tabla de verdad del && (y lógico)		
Condición1	Condición2	Condición1 && Condición2
V	V	V
V	F	F
F	V	F
F	F	F

Tabla de verdad del || (o lógico)		
Condición1	Condición2	Condición1 || Condición2
V	V	V
V	F	V
F	V	V
F	F	F

Tabla de verdad del ! (no lógico)	
Condición	!Condición
V	F
F	V