OPERADORES RELACIONALES

Estos operadores se utilizan para verificar relaciones como igualdad, mayor que, menor que. Devuelven el resultado booleano después de la comparación y se usan ampliamente en las instrucciones de bucle, así como en las sentencias condicionales if/else. El formato general es, variable operador_relacion valor

Algunos de los operadores relacionales son:

Mayor que (>)

El operador mayor que evalúa los dos operandos implicados y devuelve true si el valor del primero es mayor exclusivo que el segundo. Devuelve false en caso contrario, es decir, cuando el valor del primero sea menor o igual que el segundo.

Ejemplos:

En el siguiente ejemplo la variable a tiene valor 5, la variable b tiene valor 2. El valor de a es mayor que el de b, por lo que el resultado de evaluar a > b será true:

```
int a = 5;

int b = 2;

a > b; //true
```

Si ahora hacemos la evaluación al revés, es decir b > a, el resultado será el opuesto (false), ya que 2 (variable b) no es mayor que 5 (variable a):

```
int a = 5;

int b = 2;

b > a; //false
```

¿Qué ocurriría si la variable a tuviese el mismo valor que la variable b? En el siguiente ejemplo la variable a toma valor 5 y la variable b también toma valor 5. Tanto a > b, como b > a devolverán false, puesto que en ambos casos la evaluación será 5 > 5:

```
int a = 5;

int b = 5;

b > a; //false

a > b; //false
```

Mayor o igual que (>=)

El operador mayor o igual que evalúa los dos operandos implicados y devuelve true si el valor del primero es mayor o igual que el segundo. Devuelve false en caso contrario, es decir, cuando el valor del primero sea menor exlusivo. La diferencia con el operador anterior es que, cuando los valores de los operandos sean iguales también devolverá true.

Ejemplos:

En el siguiente ejemplo la variable a tiene valor 20, la variable b tiene valor 7. El valor de a es mayor que el de b, por lo que el resultado de evaluar a >= b será true:

```
int a = 20;

int b = 7;

a >= b; //true
```

Si ahora hacemos la evaluación al revés, es decir b >= a, el resultado será el opuesto (false), ya que 7 (variable b) no es mayor ni igual que 20 (variable a):

```
int a = 20;

int b = 7;

b >= a; //false
```

En el siguiente ejemplo ambas variables toman valor 20. Tanto a >= b, como b >= a devolverán true, puesto que en ambos casos se cumple que los valores de a y b son iguales:

```
int a = 20;

int b = 20;

b >= a; //true

a >= b; //true
```

Menor que (<)

El operador menor que evalúa los dos operandos implicados y devuelve true si el valor del primero es menor exclusivo que el segundo. Devuelve false en caso contrario, es decir, cuando el valor del primero sea mayor o igual que el del segundo.

Ejemplos:

En el siguiente ejemplo la variable a tiene valor 5, la variable b tiene valor 2. El valor de a no es menor que el de b, por lo que el resultado de evaluar a < b será false:

```
int a = 5;

int b = 2;

a < b; //false
```

Si ahora hacemos la evaluación al revés, es decir b < a, el resultado será el opuesto (true), ya que 2 (variable b) es menor que 5 (variable a):

```
int a = 5;

int b = 2;

b < a; //true
```

En el siguiente ejemplo ambas variables toman valor 5. Tanto a <= b, como b <= a devolverán false, puesto que en ambos casos la evaluación será 5 < 5:

```
int a = 5;

int b = 5;

b < a; //false

a < b; //false
```

Menor o igual que (<=)

El operador menor o igual que evalúa los dos operándos implicados y devuelve true si el valor del primero es menor o igual que el segundo. Devuelve false en caso contrario, es decir, cuando el primero sea menor exlusivo. La diferencia con el operador anterior es que, cuando los valores de los operándos sean iguales también devolverá true.

Ejemplos:

En el siguiente ejemplo la variable a tiene valor 20, la variable b tiene valor 7. El valor de a no es menor ni igual que el de b, por lo que el resultado de evaluar a <= b será false:

```
int a = 20;

int b = 7;

a <= b; //false
```

Si ahora hacemos la evaluación al revés, es decir b <= a, el resultado será el opuesto (false) ya que 7 (variable b) es menor que 20 (variable a):

```
int a = 20;

int b = 7;

b <= a; //true
```

En el siguiente ejemplo ambas variables toman valor 20. Tanto a <= b, como b <= a devolverán true, puesto que en ambos casos se cumple que el valor de a y b son iguales:

```
int a = 20;

int b = 20;

b <= a; //true

a <= b; //true
```

Igual que (==)

El operador igual que evalúa los dos operandos implicados y si sus valores son iguales, devolverá valor true. En cualquier otro caso devolverá false.

Ejemplos:

En este primer ejemplo la variable a toma valor 3 y la variable b toma valor 7. La evaluación de igualdad devolverá false, ya que 3 y 7 no son iguales.

```
int a = 3;

int b = 7;

a == b; //false
```

Solo cuando ambos valores sean iguales la evaluación devolverá true. En el siguiente ejemplo tanto a como b tienen valor 3, por lo que 3 == 3 se cumple:

```
int a = 3;

int b = 3;
```

```
a == b; //true
```

Distinto de (!=)

El operador distinto de es el opuesto al anterior. Evalúa los dos operandos implicados y devuelve true cuando sus valores son diferentes. Por lo tanto, devolverá false cuando sean iguales.

Ejemplos:

En este ejemplo la variable a toma valor 3 y la variable b toma valor 7. La evaluación devolverá true, ya que 3 y 7 son distintos.

```
int a = 3;

int b = 7;

a != b; //true
```

Cuando ambos valores sean iguales la evaluación será false. En este ejemplo a y b tienen valor 3. Puesto que son iguales, la evaluación devolverá false:

```
int a = 3;

int b = 3;

a != b; //false
```