Una instrucción if identifica qué instrucción se debe ejecutar dependiendo del valor de una expresión booleana. En el ejemplo siguiente, la variable bool condition se establece en true y, a continuación, se comprueba en la instrucción if . El resultado es The variable is set to true.

```
bool condition = true;

if (condition){

    Console.WriteLine("The variable is set to true.");

}

else{

    Console.WriteLine("The variable is set to false.");

}
```

Una instrucción if en C# puede tener dos formas, tal como se muestra en el ejemplo siguiente.

```
// if-else statement

if (condition){

    then-statement;

}

else{

    else-statement;

}

// Next statement in the program.

// if statement without an else

if (condition){

    then-statement;

}

// Next statement in the program.
```

En una instrucción if-else , si condición se evalúa como true, se ejecuta then-statement . Si condición es false, se ejecuta else-statement . Puesto que condición no puede ser simultáneamente true y false, la then-statement y la else-statement de una instrucción if-else nunca se podrán ejecutar a la vez. Después de ejecutar la then-statement o la else-statement , el control se transfiere a la siguiente instrucción situada después de la instrucción if .

En una instrucción if que no incluya una instrucción else , si condición es true, se ejecutará la then-statement . Si condición es false, el control se transfiere a la siguiente instrucción situada después de la instrucción if .

Tanto la then-statement como la else-statement pueden constar de una única instrucción o de varias instrucciones entre llaves ({}). Para una única instrucción, las llaves son opcionales, pero se recomiendan.

La instrucción o las instrucciones en la then-statement y la else-statement pueden ser de cualquier tipo, incluida otra instrucción if anidada dentro de la instrucción if . En instrucciones if anidadas, cada cláusula else pertenece a la última if que no tiene su else correspondiente. En el ejemplo siguiente, Result1 aparece si m > 10 y n > 20 se evalúan como true. Si m > 10 es true, pero n > 20 es false, aparece Result2 .

```
// Try with m = 12 and then with m = 8.

int m = 12;

int n = 18;

if (m > 10)

   if (n > 20)    {

      Console.WriteLine("Result1");

   }

 else

   {

      Console.WriteLine("Result2");

   }
```

Result2 aparece si la condición (m > 10) se evalúa como false.Result2 aparece si la condición (m > 10) se evalúa como false.

Puesto que puede ser válida tanto una instrucción del bloque else como del bloque then, puede usar cualquier expresión booleana válida para la condición.

Puede usar operadores lógicos como !, &&, ||, &, | y ^ para hacer condiciones compuestas. En el código siguiente, se muestran algunos ejemplos:

```
// NOT

bool result = true;

if (!result)

{

   Console.WriteLine("The condition is true (result is false).");

}

else

{

   Console.WriteLine("The condition is false (result is true).");

}



// Short-circuit AND

int m = 9;

int n = 7;

int p = 5;

if (m >= n && m >= p)

{

   Console.WriteLine("Nothing is larger than m.");

}



// AND and NOT

if (m >= n && !(p > m))

{

   Console.WriteLine("Nothing is larger than m.");
```

```
}

// Short-circuit OR
if (m > n || m > p)
{
    Console.WriteLine("m isn't the smallest.");
}

// NOT and OR
m = 4;
if (!(m >= n || m >= p))
{
    Console.WriteLine("Now m is the smallest.");
}
// Output:
// The condition is false (result is true).
// Nothing is larger than m.
// Nothing is larger than m.
// m isn't the smallest.
// Now m is the smallest.
```