

## 5. Estructuras de salto

---

Las instrucciones especiales de salto son:

**break**

**continue**

**return**

Las sentencias **break** y **continue** se utilizan en las estructuras de repetición. Permiten interrumpir la ejecución —mediante el **break**— o volver al principio y evaluar la condición de salida —en el caso de **continue**—. Además, el **break** también se utiliza para interrumpir la ejecución de un **switch**.

Por su parte, la sentencia **return** se utiliza para finalizar una función o método, indicando el valor a devolver, si es el caso. De hecho, puede haber funciones o métodos que no devuelvan nada (en inglés, *void*).

Las palabras reservadas *break* y *continue* se utilizan en Java para detener completamente un bucle (**break**) o para detener únicamente la iteración actual y saltar a la siguiente (**continue**). Normalmente, si usamos **break** o **continue**, lo haremos dentro de una sentencia **if** que indicará cuándo debemos detener el bucle —habitualmente del tipo infinito— al cumplirse o no una determinada condición.

La gran **diferencia** entre ambas sentencias es que *break* detiene la ejecución del bucle y salta a la primera línea del programa tras el bucle, es decir, a la siguiente instrucción por ejecutar, mientras que *continue* detiene la iteración actual y pasa a la siguiente iteración del bucle sin salir de él (a menos que el propio bucle haya llegado al límite de sus iteraciones).

### 5.1. Break

La sentencia **break** se utiliza para interrumpir la ejecución de una estructura de repetición o de un **switch**. Cuando se ejecuta el **break**, el flujo del programa continúa por la sentencia inmediatamente posterior a la estructura de repetición o al **switch**.

Ejemplo:

```
for (int i = 0; i < 10; i++) {  
    System.out.println("Dentro del bucle, en la primera y última  
iteración.");  
    break;  
    System.out.println("Nunca lo escribiré ya que se ha roto el bucle  
for");  
}  
System.out.println("Sigüiente instrucción tras el bucle que se  
ejecutara tras el break.");
```

Cuando el programa alcance la sentencia *break*, detendrá la ejecución del bucle y saldrá de él. Por tanto, el resultado de la ejecución del programa será:

- Dentro del bucle, en la primera y última iteración.
- Siguiente instrucción tras el bucle que se ejecutará tras el *break*.

El bucle *for* solo ejecuta una iteración y después termina, de tal modo que continúa el programa desde la primera línea ubicada tras el bucle.

## 5.2. Continue

La sentencia *continue* únicamente puede aparecer dentro de una estructura de repetición. El efecto que produce es que se deja de ejecutar el resto del bucle para volver a evaluar la condición del bucle, continuando con la siguiente iteración si el bucle lo permite.

Ejemplo:

```
for (int i = 0; i < 10; i++) {
    System.out.println("Dentro del bucle, por donde pasará diez
veces.");
    continue;
    System.out.println("Nunca lo escribirá esta instrucción, ya que
no llegará nunca.");
}
```

Cada vez que el programa alcance la sentencia *continue*, saltará a una nueva iteración del bucle y, por tanto, la última sentencia *System.out.println* nunca llegará a escribirse.

El resultado del programa será escribir 10 veces «Dentro del bucle, por donde pasará diez veces», pudiendo poner también la variable *i* que hace de índice para numerarlas.

## 5.3. Return

La sentencia *return* se utiliza para terminar la ejecución de un método o función. De esta forma volvemos a la instrucción que hizo la llamada a dicho método.

Con la instrucción *return* podemos devolver un valor o no:

- Devuelvo valor: *return* variable;
- No devuelvo nada: *return*;

