CICLO BREAK Y CONTINUE

En java hay dos sentencias que nos permiten forzar la salida de un bucle, y estas sentencias son break (para salir completamente del bucle) y continue (para salir de la iteración actual y saltar directamente a la siguiente).

Estas instrucciones siempre irán dentro de algún if (o else) porque carece de sentido si se va a ejecutar siempre pues en ese caso está claro que hay algo que estamos haciendo mal porque una parte de nuestro código no llegará a ejecutarse nunca.

BREAK

La sentencia break hace que se salga del bucle inmediatamente por lo que no se ejecutara ni el código que se encuentre después del break en esa misma iteración ni ninguna de las posibles iteraciones restantes.

Esta sentencia no es exclusiva para los bucles y de hecho su uso más conocido es dentro de un switch.

Para ilustrar el funcionamiento de un break dentro de un bucle vamos a poner un ejemplo muy sencillo en el que obtenemos en que día de la semana nos encontramos (del 1 al 7) y si el día coincide con el contador del bucle se ejecuta el break.

```
Calendar cal = Calendar.getInstance();

cal.setTimeInMillis(System.currentTimeMillis());

int dia = cal.get(Calendar.DAY_OF_WEEK);

for (int i = 1; i <= 7; i++) {

  if (dia == i){

    System.out.println("Hoy es el " + i + "º dia de la semana.");

    break;

  }

  System.out.println("Dia " + i);

}

System.out.println("Seguimos...");
```

El resultado de la ejecución para un jueves como hoy sería el siguiente: (Por defecto el primer día de la semana es el domingo por eso nos dice que el jueves es el 5º día y no el 4º)

```
Dia 1

Dia 2

Dia 3

Dia 4

Hoy es el 5º dia de la semana.

Seguimos...
```

Como puedes ver en el momento que se encuentra la sentencia break automáticamente se sale del bucle y la ejecución continua en la línea siguiente al bucle.

CONTINUE

Con continue se puede detener la ejecución de la iteración actual y saltar a la siguiente. El caso normal de uso de continue es cuando dentro de un bucle estamos haciendo muchas comprobaciones y en caso de que se cumpla alguna no hace falta seguir comprobando el resto pero no queremos meter unas comprobaciones anidadas dentro de otras para una mayor legibilidad.

La sentencia continue siempre tiene que estar dentro de un bucle porque si no producirá un error de compilación, en cualquier caso, no tiene sentido ponerla fuera de un bucle.

Para ver el funcionamiento de continue vamos a ver un ejemplo similar al usado para el break, pero en este caso en lugar de detenerse el bucle al llegar al coincidir el día como pasaba con break, lo que se hace es pasar al día siguiente sin hacer nada.

```
Calendar cal = Calendar.getInstance();
cal.setTimeInMillis(System.currentTimeMillis());
int dia = cal.get(Calendar.DAY_OF_WEEK);
for (int i = 1; i <= 7; i++) {
  if (dia == i){
    continue;
  }
  System.out.println("Dia " + i);
}
System.out.println("Seguimos...");
```

El resultado de la ejecución para un miércoles como hoy sería el siguiente:

```
Dia 1
```

Dia 2

Dia 3

Dia 4

Dia 6

Dia 7

Seguimos...

En este ejemplo se ve que la sentencia continue es muy fácilmente sustituible porque hacer que no se ejecute una parte de código dentro de un bucle siempre se puede hacer con un if-else, pero compara los 2 siguientes códigos.

```
for (int i = 1; i < 7; i++) {
    if (dia == i && dia % 2 == 0) {
        ...
    } else {
        if (hora == i && dia == i) {
            ...
        } else {
            if (hora == i && dia == i) {
                ...
            } else {
                if ((minuto == i && dia != i) && hora != i - 1) {
                    ...
                } else {
                    ...
                }
            }
```

```
        }
    }
}
for (int i = 1; i < 7; i++) {
    if (dia == i && dia % 2 == 0){
        ...
        continue;
    }
    if (hora == i && dia == i) {
        ...
        continue;
    }
    if (hora == i && dia == i) {
        ...
        continue;
    }

    if ((minuto == i && dia != i) && hora != i - 1) {
        ...
        continue;
    }
    ...
}
```

El código haría exactamente lo mismo pero el segundo es mucho más fácil de leer, de mantener y de modificar.

BREAK Y CONTINUE CON ETIQUETA

Tanto break como continue se pueden usar con etiquetas para hacer que en lugar de simplemente salir del bucle o saltar a la siguiente iteración como ocurre cuando los usamos sin ninguna etiqueta (como en los ejemplos anteriores) salten a la parte del código donde indique la etiqueta.

La etiqueta no puede estar en cualquier parte de nuestro código por lo que no vamos a poder hacer saltos condicionales puros tipo goto, sino que solo podremos hacer saltos a bucles dentro de los que nos encontremos. ¿Y eso no es lo mismo que hemos estado haciendo? La respuesta es... depende con los ejemplos anteriores podemos salir del bucle que contiene la palabra continue o break, pero si usamos etiquetas podemos salir a otro bucle superior que contenga el bucle que si que contiene el break o continue.

Mejor verlo en un ejemplo que seguro que queda más claro.

```
descansar:
        for (int i = 1; i < 8; i++) {
            for (int j = 1; j < 9; j++) {
                if (i % 2 == 0){
                    System.out.println("Dia "+ i + ": Descansando");
                    continue descansar;
                }
                System.out.println("Dia "+ i + ": Trabajando " + j + " horas");
            }
        }
        System.out.println("Salimos del bucle...");
```

Lo que hace el código de este ejemplo es... bueno mejor que lo mires un poco para comprenderlo bien y que lo pruebes y le quites la etiqueta para ver cómo cambia y si hay algo que no entiendes pues cualquier pregunta es bienvenida.