

## 4.D. Estructuras de salto.

✓ Hecho

### 1. Estructuras de salto.

#### 1.1. Sentencias break y continue.

Se trata de dos instrucciones que permiten modificar el comportamiento de otras estructuras o sentencias de control, simplemente por el hecho de estar incluidas en algún punto de su secuencia de instrucciones.

La sentencia **break** incidirá sobre las estructuras de control **switch, while, for** y **do-while** del siguiente modo:

- Si aparece una sentencia **break** dentro de la secuencia de instrucciones de cualquiera de las estructuras mencionadas anteriormente, dicha estructura terminará inmediatamente.
- Si aparece una sentencia **break** dentro de un bucle anidado sólo finalizará la sentencia de iteración más interna, el resto se ejecuta de forma normal.

Es decir, que **break** sirve para romper el flujo de control de un bucle, aunque no se haya cumplido la condición del bucle. Si colocamos un **break** dentro del código de un bucle, cuando se alcance el **break**, automáticamente se saldrá del bucle pasando a ejecutarse la siguiente instrucción inmediatamente después de él.

En la siguiente imagen, puedes apreciar cómo se utilizaría la sentencia **break** dentro de un bucle **for**.

```

6 public class sentencia_break {
7     public static void main(String[] args) {
8         // Declaración de variables
9         int contador;
10
11
12         //Procesamiento y salida de información
13
14         for (contador=1;contador<=10;contador++)
15         {
16             if (contador==7)
17                 break;
18             System.out.println ("Valor: " + contador);
19         }
20         System.out.println ("Fin del programa");
21         /* El bucle sólo se ejecutará en 6 ocasiones, ya que cuando
22          * la variable contador sea igual a 7 encontraremos un break que
23          * romperá el flujo del bucle, transfiriéndonos a la sentencia que
24          * imprime el mensaje de Fin del programa.
25          */
26     }
  
```

La sentencia **continue** incidirá sobre las sentencias o estructuras de control **while, for** y **do-while** del siguiente modo:

- Si aparece una sentencia **continue** dentro de la secuencia de instrucciones de cualquiera de las sentencias anteriormente indicadas, dicha sentencia dará por terminada la iteración actual y se ejecuta una nueva iteración, evaluando de nuevo la expresión condicional del bucle.
- Si aparece en el interior de un bucle anidado solo afectará a la sentencia de iteración más interna, el resto se ejecutaría de forma normal.

Es decir, la sentencia **continue** forzará a que se ejecute la siguiente iteración del bucle, sin tener en cuenta las instrucciones que pudiera haber después del **continue**, y hasta el final del código del bucle.

En la siguiente imagen, puedes apreciar cómo se utiliza la sentencia **continue** en un bucle **for** para imprimir por pantalla sólo los números pares.

```

4  * Uso de la sentencia continue
5  */
6  public class sentencia_continue {
7  public static void main(String[] args) {
8      // Declaración de variables
9      int contador;
10
11     System.out.println ("Imprimiendo los números pares que hay del 1 al 10... ");
12     //Procesamiento y salida de información
13
14     for (contador=1;contador<=10;contador++)
15     {
16         if (contador % 2 != 0) continue;
17         System.out.print(contador + " ");
18     }
19     System.out.println ("\nFin del programa");
20     /* Las iteraciones del bucle que generarán la impresión de cada uno
21     * de los números pares, serán aquellas en las que el resultado de
22     * calcular el resto de la división entre 2 de cada valor de la variable
23     * contador, sea igual a 0.
24     */
25 }
26
27 }
28

```

Para clarificar algo más el funcionamiento de ambas sentencias de salto, te ofrecemos a continuación un diagrama representativo.

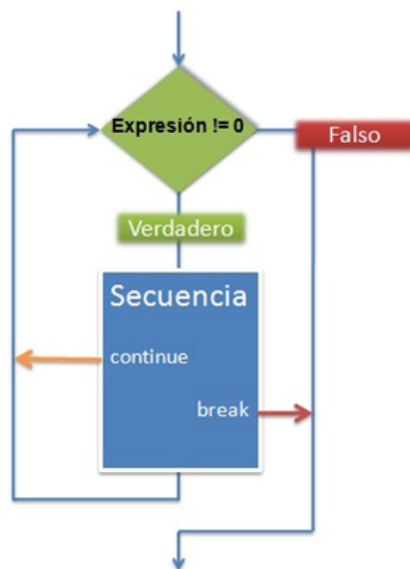


Imagen extraída de curso Programación del MECD.

### Autoevaluación

La instrucción `break` puede utilizarse en las estructuras de control `switch`, `while`, `for` y `do-while`, no siendo imprescindible utilizarla en la cláusula `default` de la estructura `switch`. ¿Verdadero o Falso?

- ☒ Verdadero.
- ☐ Falso.

### ◀ 4.C. Estructuras de repetición.

Ir a...