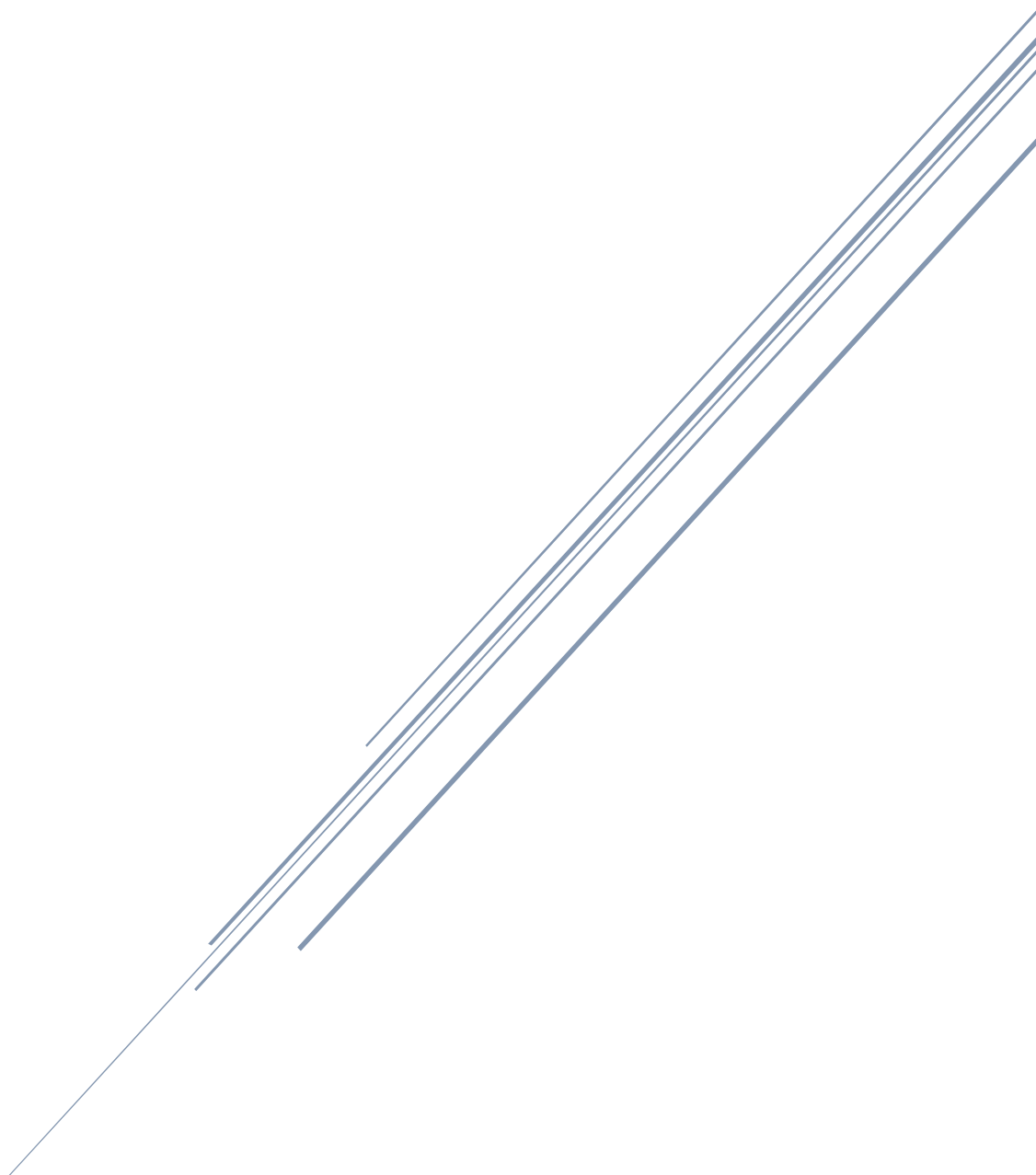


# BREAK, CONTINUE

PROGRAMACIÓN



[Escuela]  
[Título del curso]

Contenido

Break y Continue ..... 2

    Sentencia sin etiqueta (Unlabeled) ..... 3

    Sentencia con etiqueta (Labeled)..... 3

## Break y Continue

Las palabras reservadas *break* y *continue*, se utilizan en Java para detener completamente un bucle (*break*) o 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 al cumplirse o no una determinada condición.

La gran diferencia entre ambos es que, *break*, detiene la ejecución del bucle y salta a la primera línea del programa tras el bucle y *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).

NOTA: La sentencia *continue* debe estar siempre incluida dentro de un bucle, sino recibiréis un error de compilación. Mientras que la sentencia *break*, podemos encontrarla dentro de un bucle o en una sentencia *switch*. (Aunque esta regla no la aplicamos a los *break* con etiqueta, ahora lo veremos)

La sentencia *continue* provoca que únicamente la iteración actual del bucle más interno detenga su ejecución y que comience inmediatamente la siguiente iteración si el bucle todavía no ha terminado. Por ejemplo:

```
for (int i = 0; i < 10; i++) {  
    System.out.println("Dentro del bucle");  
    continue;  
    System.out.println("Nunca lo escribirá");  
}
```

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”.

Sin embargo, la sentencia *break* provoca que el programa detenga la ejecución del bucle más interno y continúe la ejecución desde la primera línea de código que se encuentre tras el bucle. Por ejemplo:

```
for (int i = 0; i < 10; i++) {  
    System.out.println("Dentro del bucle");  
    break;  
    System.out.println("Nunca lo escribirá");  
}  
System.out.println("Tras el bucle");
```

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"  
"Tras el bucle"
```

El bucle *for*, sólo ejecuta una iteración y después termina, continuando el programa desde la primera línea tras el bucle.

Las sentencias *break* y *continue* pueden llevar asociada una etiqueta (Labeled) o no (Unlabeled). Vamos a ver ambos casos:

### Sentencia sin etiqueta (Unlabeled)

Este es el uso más común que solemos hacer de las etiquetas *break* y *continue*. Los ejemplos que hemos visto hasta ahora son ejemplos claros de sentencias sin etiqueta. Veamos un par de ejemplos más:

Break:

```
boolean problema = true;
while (true) {
    if (problema) {
        System.out.println("There was a problem");
        break;
    }
}
// Siguiente línea de código
```

Continue:

```
while (!EOF) {
    //leemos un campo de un fichero
    if (wrongField) {
        continue;    // vamos al siguiente campo
    }
    // procesamos el campo
}
```

### Sentencia con etiqueta (Labeled)

Hay muchas sentencias en un programa en Java que pueden ser etiquetadas, pero su uso más común es con los bucles *for* y *while*, asociadas a *break* y *continue*. Una etiqueta debe ir colocada justo antes del bucle al que va etiquetando y se forma con un identificador válido seguido de dos puntos (:).

Hay que tener muy clara la diferencia entre usar sentencias *break* y *continue* con etiquetas a usar sentencias *break* y *continue* sin etiquetar. El uso de las sentencias *break* y *continue* con etiquetas, será necesario cuando tengamos varios bucles anidados y necesitemos indicar cuál es el bucle anidado que debe continuar su ejecución tras romper el flujo normal de ejecución del programa. Por ejemplo:

```
bucle1:
for (int i = 0; i < 10; i++) {
    System.out.println("Bucle i. i = "+i);
    for (int j = 0; j < 10; j++) {
        System.out.println("Bucle j. j = "+j);
        for (int k = 0; k < 10; k++) {
            System.out.println("Bucle k. k = "+k);
            break bucle1;
        }
    }
}
```

```
}
```

En esta ocasión la salida del programa sería:

```
"Bucle i. i = 0"  
"Bucle j. j = 0"  
"Bucle k. k = 0"
```

El programa comienza la ejecución en el primer bucle for. Ejecuta la primera iteración y entra en el bucle for anidado de “j”. Realiza la primera iteración y entra en el bucle for anidado de “k”. Aquí nuevamente ejecuta la primera iteración y se encuentra con una sentencia *break*. (Recordad que *break*, detiene la ejecución del bucle y provoca que el programa continúe desde la primera línea después del bucle). Pero en este caso es una sentencia *break* con etiqueta, por tanto, continuará con la primera línea tras el bucle que está indicado por la etiqueta. Y en esta ocasión es el bucle for de “i”; de esta forma, al ejecutar esta sentencia *break*, provocamos el que programa deje de ejecutar el resto de iteraciones de los tres bucles for anidados.

Veamos un ejemplo con una sentencia *continue*.

```
fuera:  
    for (int i=0; i<5; i++) {  
        for (int j=0; j<5; j++) {  
            System.out.println("Dentro "+i+" "+j);  
            continue fuera;  
        } // fin del bucle anidado  
        System.out.println("fuera"); // nunca lo escribe  
    }  
System.out.println("Fuera del bucle");
```

La salida que obtenemos tras la ejecución del código anterior es:

```
Dentro 0 0  
Dentro 1 0  
Dentro 2 0  
Dentro 3 0  
Dentro 4 0  
Fuera del bucle
```

Veamos porqué.

Después de que se ejecute la sentencia *continue*, el flujo de ejecución continúa ejecutando la siguiente iteración del bucle for identificado por la etiqueta, hasta que éste alcance la condición de parada. Finalmente, cuando la condición de parada del bucle se alcanza, el bucle termina y se escribirá “Fuera del Bucle”. Si en lugar de poner una sentencia *continue*, hubieramos puesto un *break*, el resultado hubiera sido:

```
Dentro  
Fuera del bucle
```

NOTA: Las sentencias *break* y *continue* con etiquetas, deben estar dentro del bucle que tiene la misma etiqueta, de otra forma, el código no compilará.

Ejercicio. (Entregar)

Realizar un programa que dado un entero por pantalla devuelva un listado de los numero primos desde el cero hasta el dato introducido, usar bucles y break o continue.