

# Instrucciones break y continue



# Fundamentos

➤ Provocan un salida forzada.

- `break`. Abandona el bucle


- `continue`. Pasa a la siguiente iteración

➤ Ambas instrucciones pueden utilizarse tanto en `for` como en `while`

# Instrucción break

➤ Provoca la salida de la instrucción repetitiva, pasando el control del programa a la siguiente instrucción


```
int n=leerNumero();  
int s=0;  
for(int i=1;i<n;i++){  
    s+=i;  
    if(s>100){  
        break;  
    }  
}
```



# Instrucción continue

➤ Pasa a la siguiente iteración del bucle. En el caso de un for, la llamada a continue nos llevaría directamente a la instrucción de incremento

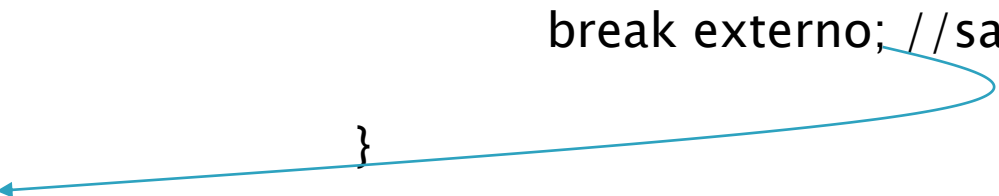
```
//muestra los números del 1 al 10, //menos el 5
for(int i=1;i<10;i++){
    if(i==5){
        continue;
    }
    System.out.println(i);
}
```



# Bucles etiquetados

- Es posible asignar una etiqueta a una instrucción for o while
- En bucles anidados, permite a las instrucciones break o continue indicar el bucle que se quiere abandonar:

```
externo: for(..){  
    while(..){  
        break externo; //sale del bucle principal  
    }  
}
```



- Si no se indica etiqueta después de break/continue, afectará al bucle más interno.

# Ejemplo

➤ ¿Qué se muestra al ejecutar el siguiente programa?

```
public class TestBuclesAnidados {  
    public static void main(String[] args) {  
        int a=0,s=0,i=1;  
        principal:  
        for(;i<10;i++){  
            while(a<5){  
                a=i++;  
                s=a+i;  
                if(s>=10){  
                    break principal;  
                }  
            }  
        }  
        System.out.println(i+":"+a+":"+s);  
    }  
}
```

→ { a=i;  
i=i+1;

El resultado es  
6:5:11