Estructuras en Java

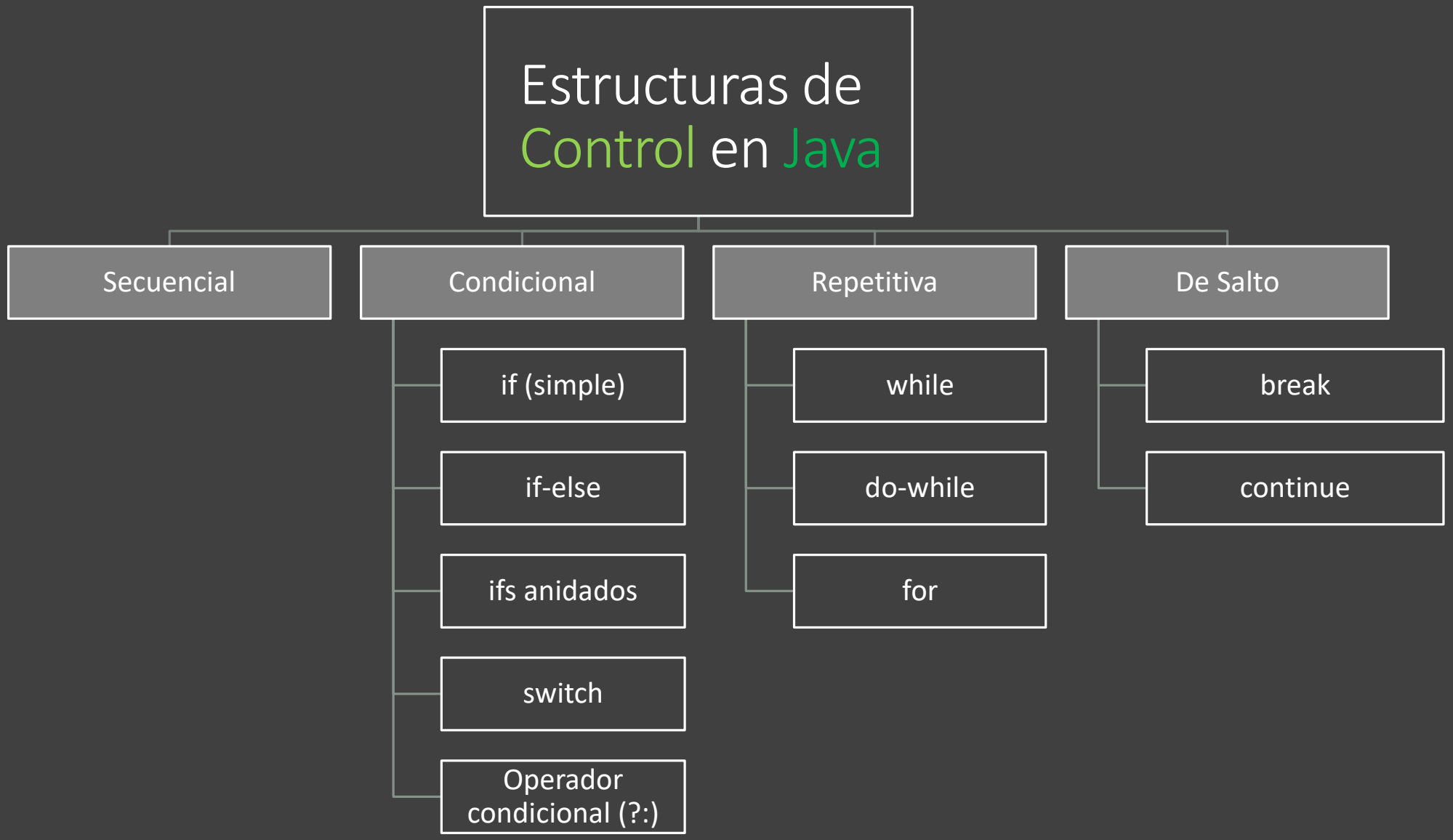
Estructuras de control en Java

Los programas contienen instrucciones que se ejecutan generalmente una a continuación de la otra según la secuencia en la que el programador ha escrito el código. Sin embargo, hay ocasiones en las que es necesario romper esta secuencia de ejecución para hacer que una serie de instrucciones se ejecuten o no dependiendo de una determinada condición o hacer que una serie de instrucciones se repitan un número determinado de veces.

Las estructuras de control permiten modificar el orden natural de ejecución de un programa. Mediante ellas podemos conseguir que el flujo de ejecución de las instrucciones sea el natural o varíe según se cumpla o no una condición o que un bloque de instrucciones se repitan dependiendo de que una condición se cumpla o no.

Las estructuras de control tienen las siguientes características:
- Tienen un único punto de entrada y un único punto de salida.
- Están compuestas por instrucciones o por otras estructuras de control.

Estructuras de Control en Java



Estructura secuencial

Las instrucciones de un programa se ejecutan por defecto en orden secuencial. Esto significa que las instrucciones se ejecutan en secuencia, una después de otra, en el orden en que aparecen escritas dentro del programa.

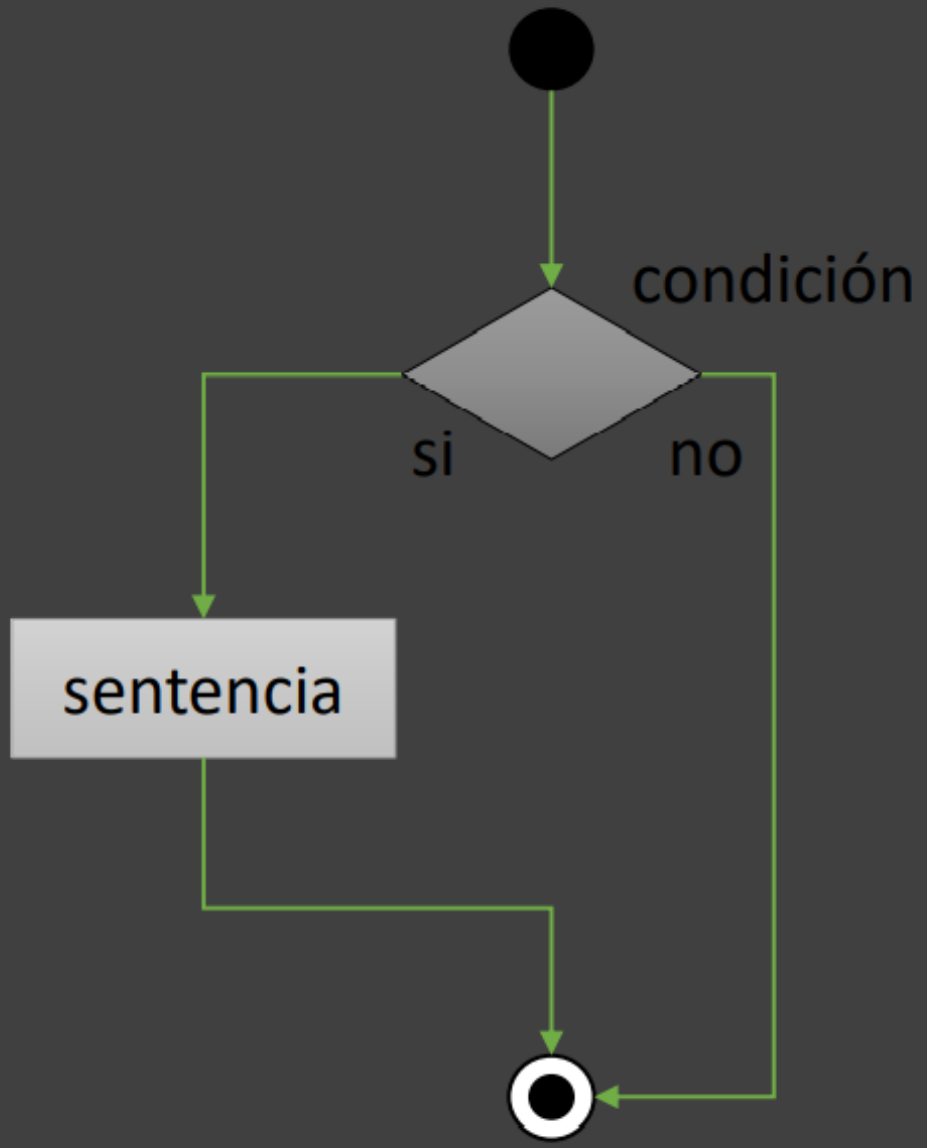
La estructura secuencial es el orden natural de ejecución.

La mayoría de las instrucciones están separadas por el carácter punto y coma (;).

Las instrucciones se suelen agrupar en bloques.

El bloque de sentencias se define por el carácter llave de apertura { para marcar el inicio del mismo, y el carácter llave de cierre } para marcar el final.

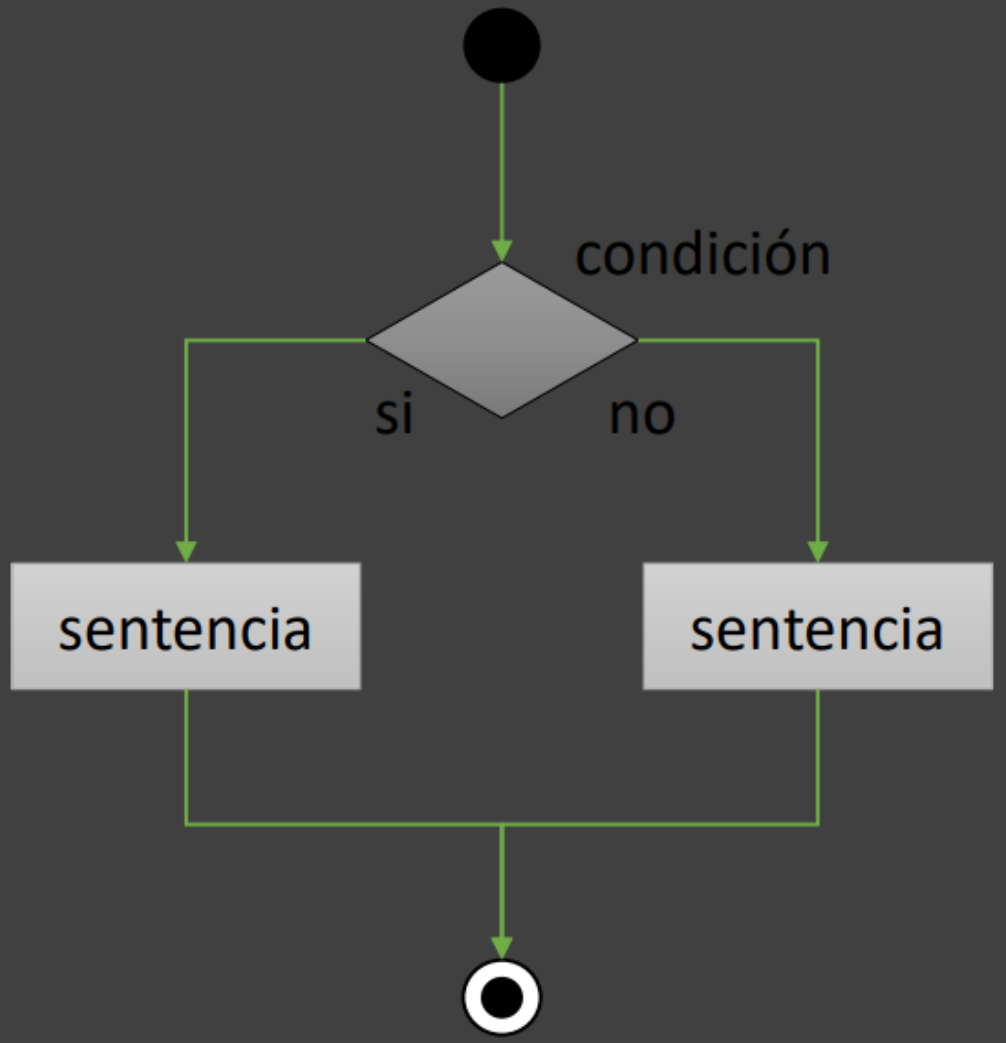
Si el bloque de instrucciones está constituido por una única instrucción no es obligatorio el uso de las llaves de apertura y cierre { }, aunque es mala práctica no hacerlo.



Estructura condicional simple if

Ejecuta el bloque de código si la condición es verdadera.

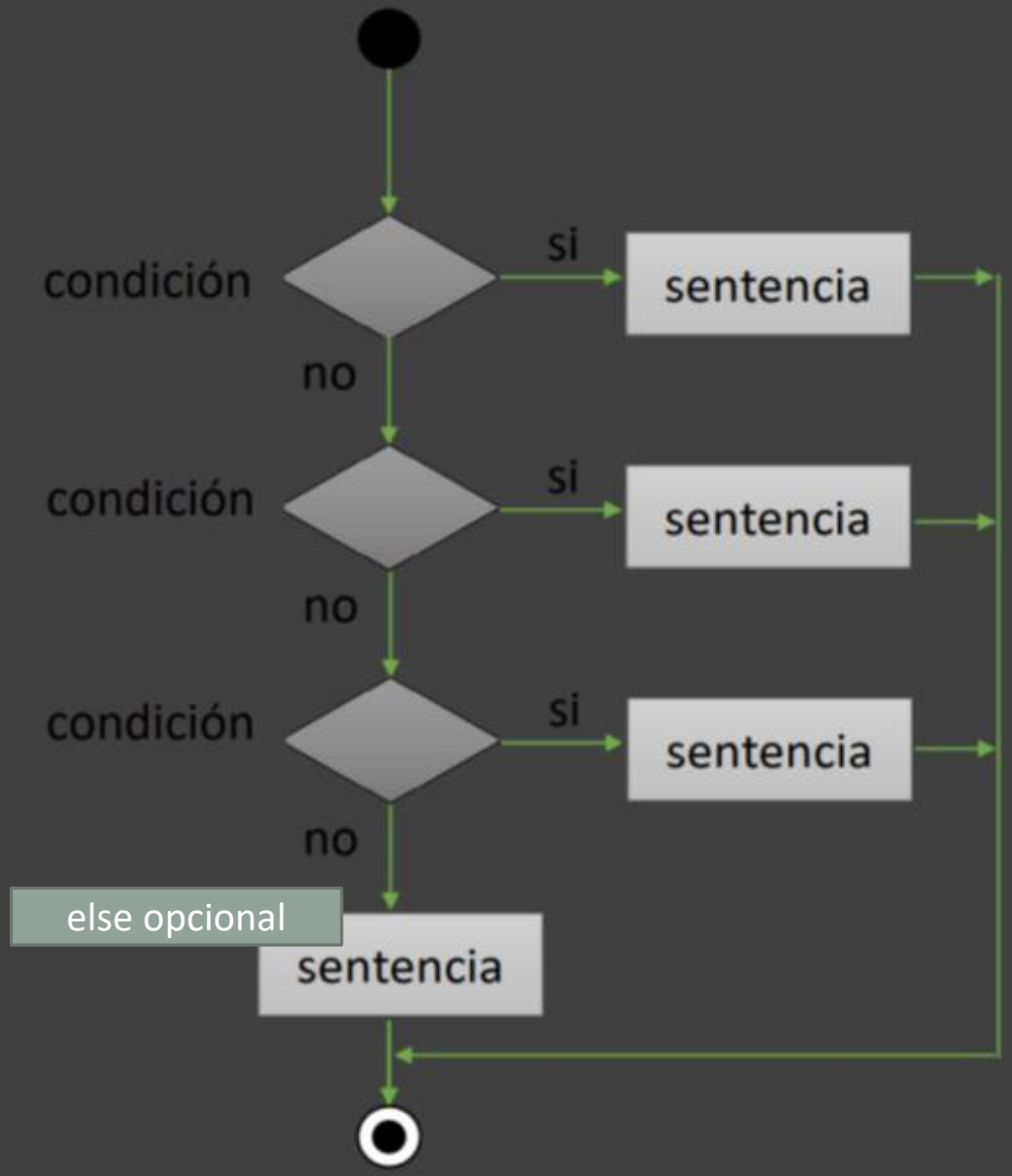
```
if (condición){
Sentencias (sólo se ejecutan si cumple la condición)
}
```



Estructura condicional doble if-else

Ejecuta el bloque de código si la condición es verdadera o ejecuta el otro bloque de código si la condición es falsa.

```
if (condición){
Sentencias (sólo se ejecutan si cumple la condición)
}else{
Sentencias (sólo se ejecutan si no se cumple la condición)
}
```



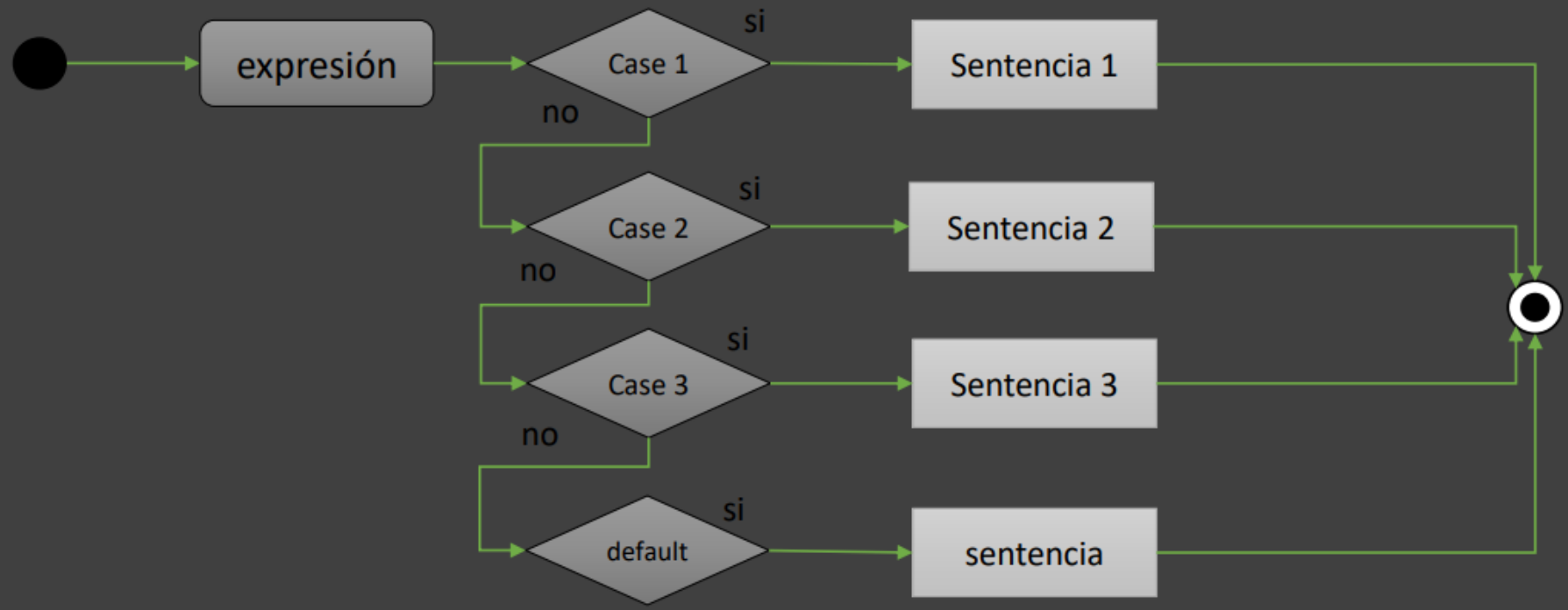
Estructura condicional múltiples if-else

Ejecuta el primer bloque de código si la condición es verdadera, si no es así ejecuta el siguiente bloque de código…hasta llegar al último bloque "else" (opcional) que se ejecutaría como última opción.

```
if (condición){
Sentencias (sólo se ejecutan si cumple la condición)
}else if (otra condición){
Sentencias (sólo se ejecutan si no se cumple la condición)
}else{
Sentencias (sólo se ejecutan si no se cumple ninguna condición previa)
}
```

Estructura condicional switch

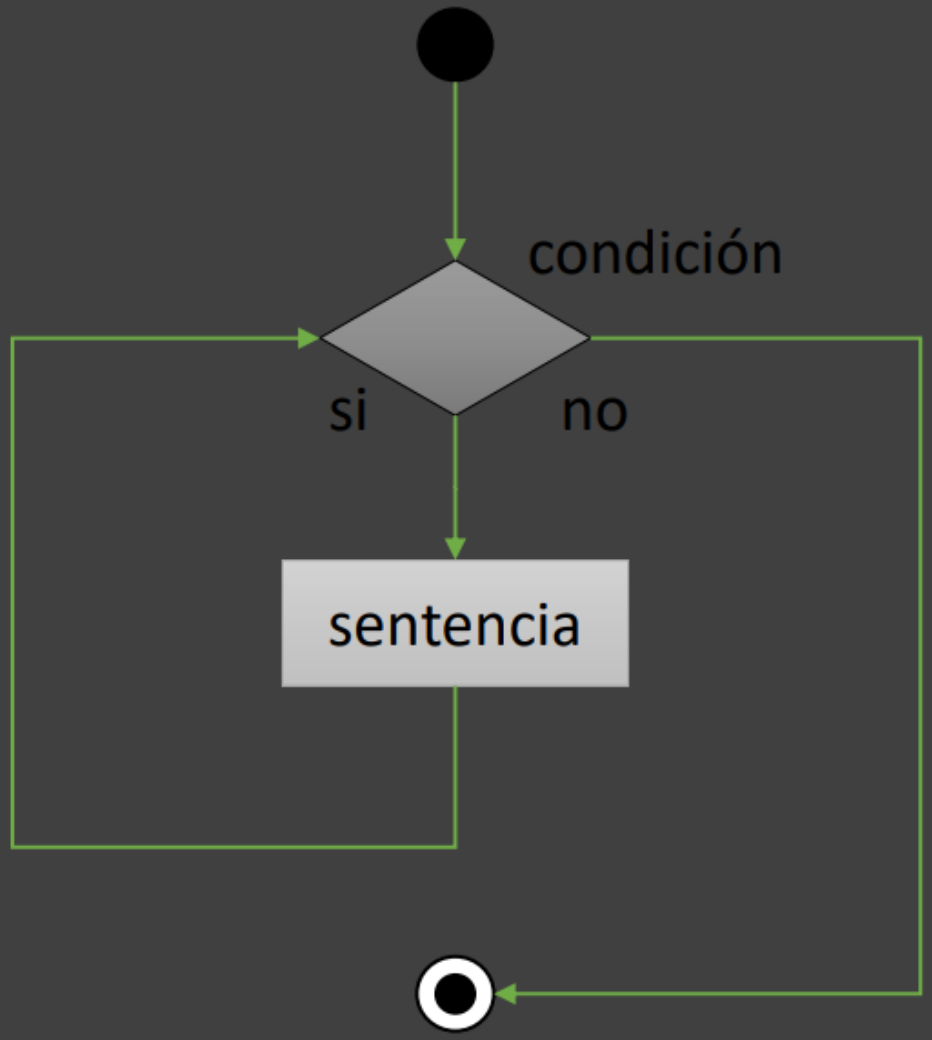
Evalúa la expresión y la compara con los distintos “case”. En el caso de coincidir “expresión” con “case” ejecuta el código correspondiente.



Estructura condicional switch

Con “break” conseguimos salir del switch y que las demás líneas de código dentro de él dejen de evaluarse.

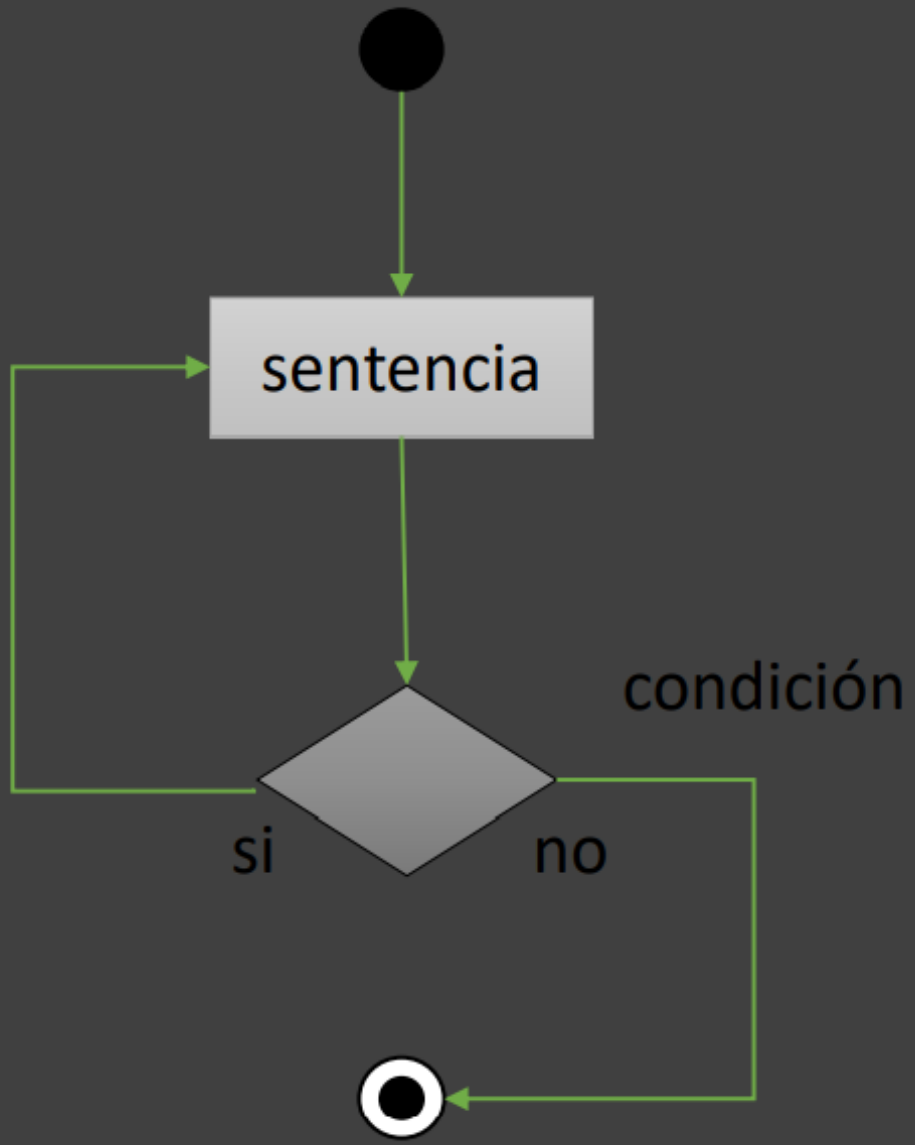
```
switch (variable){

case valor1:
      Sentencias (ejecuta las sentencias si valor1 == variable);
      break; (salimos de la ejecución del switch. Los demás “case” no se llegan a evaluar)
case valor2:
      Sentencias (ejecuta las sentencias si valor2 == variable);
      break; (salimos de la ejecución del switch. Los demás “case” no se llegan a evaluar)
case valor3:
      Sentencias (ejecuta las sentencias si valor3 == variable);
      break; (salimos de la ejecución del switch. Los demás “case” no se llegan a evaluar)
default:
      Sentencias (ejecuta las sentencias si no existe coincidencia previa);
}
```



Estructura repetitiva while

Ejecuta un bloque de código mientras se cumpla la condición. Es importante resaltar que antes de ejecutar las sentencias evalúa la condición. Es una estructura pre-condición.

```
while (condición){
Sentencias (sólo se ejecutan si cumple la condición)
}
```



Estructura repetitiva do while

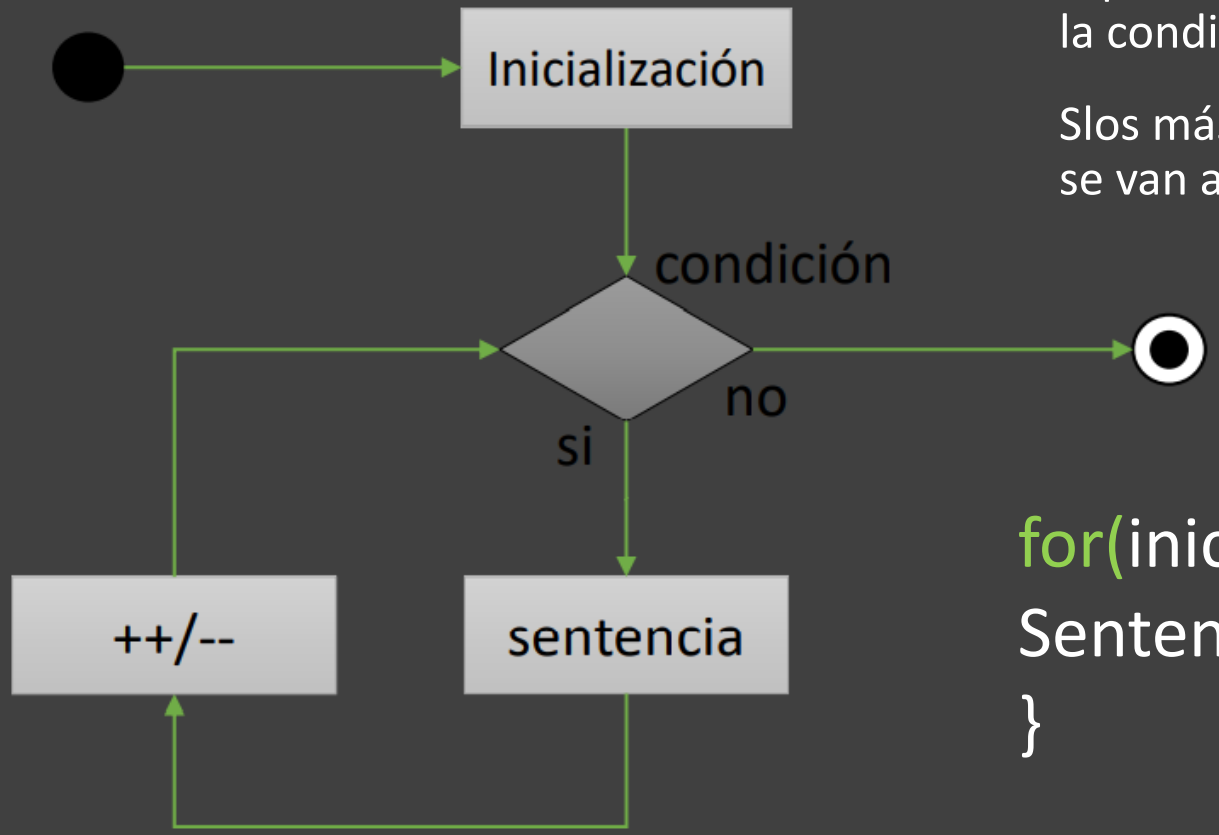
Al contrario que la estructura while es una estructura post-condición. Esto conlleva que el código es ejecutado al menos una vez. Una vez ejecutado el código por vez primera entraremos en el bucle hasta que la condición deje de cumplirse.

```
do{
Sentencias (se ejecutan al menos una vez)
}while (condición)
```

Estructura repetitiva for

Un for hace que una instrucción o bloque de instrucciones se repitan un número determinado de veces mientras se cumpla la condición.

Slos más adecuados cuando se conoce el número de veces que se van a repetir las instrucciones.




```
for(inicialización; condición; incremento){
Sentencias (se ejecutan mientras se cumpla la condición)
}
```

Estructura de salto: break

Esta instrucción provoca la finalización de una instrucción switch, while, do-while o for. En aquellos casos en los que existan estructuras de control repetitivas anidadas, un break produce la salida inmediata de aquel bucle en el que se encuentre incluida pero no de los demás.

Estructura de salto: continue

Esta instrucción provoca la ejecución de la siguiente iteración en el bucle, es decir, se salta las instrucciones que quedan hasta el final del bucle, y vuelve al inicio del mismo. Si se trata de un bucle for vuelve a la zona de incremento/decremento.