

Tema 4: Sentencias de control

Antonio J. Sierra

Índice

1. Sentencias de selección.
2. Sentencias de iteración.
3. Sentencias de salto.

Sentencias de selección

- Permite la elección de distintos caminos de ejecución en función del resultado de una expresión o del estado de una variable.
 - **if**
 - **switch**

if

- Ramificación condicional.
if (condición) {
 sentencia1;
}else{
 sentencia2;
}
- **sentenciax**: simple o un bloque
- **condición** : expresión que devuelve un **boolean**.

switch (I)

- Ramificación múltiple.

```
switch (expresión){  
    case valor1:  
        //secuencia de sentencias  
        break;  
    case valor2:  
        //secuencia de sentencias  
        break;  
    case valorN:  
        //secuencia de sentencias  
        break;  
    default:  
        //secuencia de sentencias  
}
```

switch (y II)

- **expresión** puede devolver cualquier tipo simple
- Los “valores” en **case** deben tener un tipo compatible con el de la expresión.
- cada “valor” debe ser un literal único (no una variable)
- Los “valores” no pueden estar duplicados en las sentencias **case**.

Sentencias de iteración

- Tres tipos:
 - **while**
 - **do-while**
 - **for**

while

- Sentencia de bucle básica.
- Repite una sentencia o bloque mientras su expresión de control sea verdadera.

```
while(condición) {  
    //cuerpo del bucle  
}
```
- **condición** es una expresión booleana.

do-while

- Para ejecutar el cuerpo del bucle **while** al menos una vez.

```
do{  
    //cuerpo del bucle  
}while(condición);
```

for

- La forma general de la sentencia **for** es la siguiente:

```
for(inicialización; condición; iteración) {  
    //cuerpo  
}
```

- Primero **inicialización**, luego se evalúa **condición**, se ejecuta el cuerpo y por último **iteración**.
- “La coma” se puede usar como separador.
- Bucle infinito: `for(;;);`

Sentencias de salto

- Tres tipos:
 - **break**
 - **continue**
 - **return**

break

- Tiene tres usos
 - Con sentencias **switch**
 - Para salir de un bucle (Evitar!!)
 - Una forma de “**goto**” (Evitar!!)
break etiqueta;

continue

- Continúa el bucle en el siguiente paso de la iteración

return

- Devuelve explícitamente desde un método.
- Provoca que el control del programa vuelva al método llamante.
- Aunque puede aparecer en cualquier parte de un método, existen convenios en su colocación.