



BUCLAS O SENTENCIAS REPETITIVAS

Septiembre / 2019

INTRODUCCIÓN A LA PROGRAMACIÓN
INGENIERÍA EN INFORMÁTICA

Objetivos

Caracterizar los bucles o sentencias repetitivas como parte de la construcción de un programa computacional.

Objetivos

Identificar el tipo de bucle o sentencia repetitiva a utilizar en la construcción de un programa computacional.

Sumario

- ◇ Sentencia for
- ◇ Sentencia while
- ◇ Sentencia do ... while
- ◇ Sentencia break y continue

Bibliografía

- ◇ *Como programar en Java.*
- ◇ *Aprenda Java como si estuviera en primero.*

Bucle

Un bucle se utiliza para realizar un proceso repetidas veces. Se denomina también lazo o *loop*.

El código incluido entre las llaves {} (opcionales si el proceso repetitivo consta de una sola línea), se ejecutará mientras se cumpla unas determinadas condiciones.

Bucle

Una instrucción de repetición (también llamada instrucción de ciclo, o un ciclo) permite al programador especificar que un programa debe repetir una acción mientras cierta condición sea verdadera.

Bucle while

Es una instrucción de repetición que permite repetir ciertas instrucciones mientras cierta condición sea verdadera.

Seudocódigo

Mientras tenga artículos en la lista de compras

- Comprar el artículo

- Quitarlo de mi lista

Bucle while

Sintaxis

```
while (booleanExpression) {  
    statements;  
}
```

Donde si la *booleanExpression* devuelve verdadero se ejecutará el grupo de instrucciones (*statements;*) dentro de las llaves `{}` .

Bucle while

Ejemplo

```
int producto = 3;  
while ( producto <= 100 ){  
    producto = 3 * producto;  
}
```

Un segmento de programa diseñado para encontrar la primera potencia de 3 que sea mayor a 100.

Bucle for

Java también cuenta con la instrucción de repetición *for*, que especifica los detalles de la repetición controlada por contador en una sola línea de código.

Seudocódigo

Asignar variable factorial el valor 0

Para cada $x \in [1, 100]$ con paso 1

Asignar a factorial el valor de factorial * x

Bucle for

Sintaxis

```
for(initialization; booleanExpression; increment){  
    statements;  
}
```

que es equivalente a utilizar **while** en la siguiente forma,

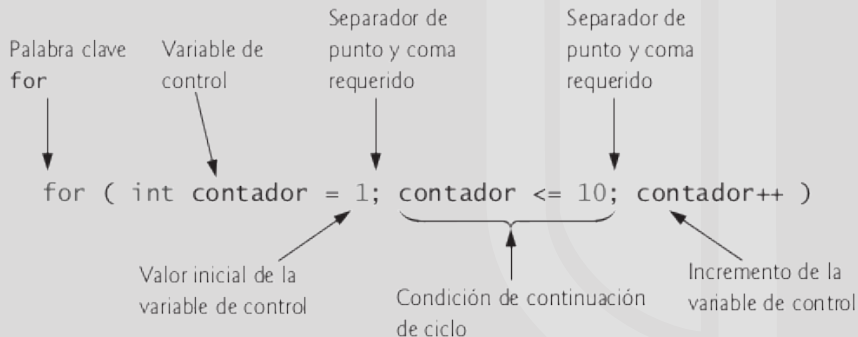
```
initialization;  
while (booleanExpression) {  
    statements;  
    increment;  
}
```

Bucle for

La sentencia o sentencias *initialization* se ejecuta al comienzo del **for**, e *increment* después de *statements*. La *booleanExpression* se evalúa al comienzo de cada iteración; el bucle termina cuando la expresión de comparación toma el valor **false**.

Bucle for

Componentes del encabezado de la instrucción for



Bucle for

Cualquiera de las tres partes puede estar vacía. La *initialization* y el *increment* pueden tener varias expresiones separadas por comas.

Código

```
for(int i=1,j=i+10;i<5;i++,j=2*i){  
    System.out.println("i="+i+" j= "+j)  
    ;  
}
```

Salida

```
i=1 j=11  
i=2 j=4  
i=3 j=6  
i=4 j=8
```

Bucle do ... while

Es similar al bucle while pero con la particularidad de que el control está al final del bucle (lo que hace que el bucle se ejecute al menos una vez, independientemente de que la condición se cumpla o no).

Bucle do ... while

Una vez ejecutados los *statements*, se evalúa la condición: si resulta **true** se vuelven a ejecutar las sentencias incluidas en el bucle, mientras que si la condición se evalúa a **false** finaliza el bucle.

Bucle do ... while

Seudocódigo

Hacer:

- Comprar el artículo

- Quitarlo de mi lista

Mientras tenga artículos en la lista de compras

Bucle do ... while

Sintaxis

```
do {  
    statements;  
} while (booleanExpression);
```

Ejemplo

```
int contador = 1; // inicializa contador  
do {  
    System.out.printf("%d ", contador);  
    ++contador;  
} while ( contador <= 10 ); // fin de do...while  
System.out.println(); // imprime una nueva linea
```

Sentencia *break*

La sentencia **break** es válida tanto para las bifurcaciones como para los bucles. Hace que se salga inmediatamente del bucle o bloque que se está ejecutando, sin realizar la ejecución del resto de las sentencias.

Sentencia *continue*

La sentencia **continue** se utiliza en los bucles. Finaliza la iteración “i” que en ese momento se está ejecutando (no ejecuta el resto de sentencias que hubiera hasta el final del bucle). Vuelve al comienzo del bucle y comienza la siguiente iteración (i+1).

Conclusiones

- ◇ El **do ... while** se utiliza con frecuencia para controlar la satisfacción de una determinada condición de error o de convergencia.

Conclusiones

- ◇ Prestar especial atención a los bucles infinitos, hecho que ocurre cuando la condición de finalizar el bucle no se llega a cumplir nunca.

Conclusiones

- ◇ Java sólo tiene tres tipos de estructuras de control, a las cuales nos referiremos de aquí en adelante como instrucciones de control:
 1. La instrucción de secuencia.
 2. Las instrucciones de selección (4 tipos)
 3. Las instrucciones de repetición (3 tipos)

Conclusiones

- ◇ Cada programa se forma combinando tantas instrucciones de secuencia, selección y repetición como sea apropiado para el algoritmo que implemente el programa.

UNIVERSIDAD DE MATANZAS

cosechando el saber

FIN