

1. Estructuras de repetición.

Nuestros programas ya son capaces de controlar su ejecución teniendo en cuenta determinadas condiciones, pero aún hemos de aprender un conjunto de estructuras que nos permita repetir una secuencia de instrucciones determinada. La función de estas estructuras es repetir la ejecución de una serie de instrucciones teniendo en cuenta una condición.



A este tipo de estructuras se las denomina **estructuras de repetición**, estructuras repetitivas, **bucles** o estructuras iterativas. En Java existen cuatro clases de bucles:

- Bucle `for` (repite para)
- Bucle `for/in` (repite para cada)
- Bucle `while` (repite mientras)
- Bucle `do while` (repite hasta)

Los bucles `for` y `for/in` se consideran bucles **controlados por contador**.

Por el contrario, los bucles `while` y `do...while` se consideran bucles **controlados por sucesos**.

La utilización de unos bucles u otros para solucionar un problema dependerá en gran medida de las siguientes preguntas:

- ¿Sabemos a priori cuántas veces necesitamos repetir un conjunto de instrucciones?
- ¿Sabemos si hemos de repetir un conjunto de instrucciones si una condición satisface un conjunto de valores?
- ¿Sabemos hasta cuándo debemos estar repitiendo un conjunto de instrucciones?
- ¿Sabemos si hemos de estar repitiendo un conjunto de instrucciones mientras se cumpla una condición?

Recomendación

Estudia cada tipo de estructura repetitiva, conoce su funcionamiento y podrás llegar a la conclusión de que algunos de estos bucles son equivalentes entre sí.

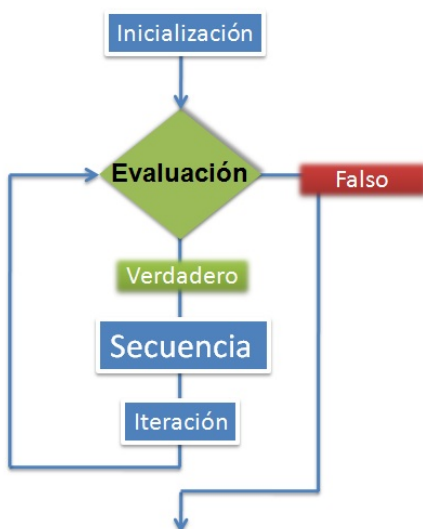
Un mismo problema, podrá ser resuelto empleando diferentes tipos de bucles y obtener los mismos resultados.

Estas y otras preguntas tendrán su respuesta en cuanto analicemos cada una de estructuras repetitivas en detalle.

1.1. Estructura `for`.

Hemos indicado anteriormente que el bucle `for` es un bucle controlado por contador. Este tipo de bucle tiene las siguientes características:

- Se ejecuta un número determinado de veces.
- Utiliza una variable contadora que controla las iteraciones del bucle.



En general, existen tres operaciones que se llevan a cabo en este tipo de bucles:

- Se inicializa la variable contadora.
- Se evalúa el valor de la variable contador, por medio de una comparación de su valor con el número de iteraciones especificado.
- Se modifica o actualiza el valor del contador a través de incrementos o decrementos de éste, en cada una de las iteraciones.

Recomendación

La inicialización de la variable contadora debe realizarse correctamente para garantizar que el bucle se lleve a cabo, **al menos, la primera repetición** de su código interno.

La **condición de terminación** del bucle debe variar en el interior del mismo, de no ser así, podemos caer en la creación de un **bucle infinito**. Cuestión que se debe evitar por todos los medios.

Es necesario *estudiar el número de veces que se repite el bucle*, pues debe ajustarse al número de veces estipulado.

En la siguiente tabla, podemos ver la especificación de la estructura `for`:

Estructura repetitiva <code>for</code>	
<p>Sintaxis:</p> <pre>for (inicialización; condición; iteración)</pre> <pre>sentencia;</pre> <p>(estructura <code>for</code> con una única sentencia)</p>	<p>Donde inicialización es una expresión en la que se inicializa una variable de control, que será la encargada de controlar el final del bucle.</p> <p>Donde condición es una expresión que evaluará la variable de control. Mientras la condición sea cierta, el cuerpo del bucle estará repitiéndose. Cuando la condición no se cumpla, terminará la ejecución del bucle.</p> <p>Donde iteración indica la manera en la que la variable de control va cambiando en cada iteración del bucle. Podrá ser mediante incremento o decremento, y no solo de uno en uno.</p>
<p>Sintaxis:</p> <pre>for (inicialización; condición; iteración)</pre> <pre>{</pre> <pre>sentencia1;</pre> <pre>sentencia2;</pre> <pre>...</pre> <pre>sentenciaN;</pre> <pre>}</pre> <p>(estructura <code>for</code> con un bloque de sentencias)</p>	

Debes conocer

Como venimos haciendo para el resto de estructuras, visualiza el siguiente programa Java y podrás analizar un ejemplo de utilización del bucle `for` para la impresión por pantalla de la tabla de multiplicar del siete. Lee atentamente los comentarios incluidos en el código, pues aclaran algunas cuestiones interesantes sobre este bucle.

```
public class RepetitivaFor {
    /*
    En este ejemplo se utiliza la estructura repetitiva 'for'
    para representar en pantalla la tabla de multiplicar del siete.
    */
    public static void main(String[] args) {
        // Declaración e inicialización de variables
        int numero = 7;
        int resultado;

        // Salida de información
        System.out.println("Tabla de multiplicar del " + numero);
        System.out.println(".....");

        // Utilizamos ahora el bucle 'for'
        for (int contador = 1; contador <= 10; contador++) {
            /*
            La cabecera del bucle incorpora la inicialización de la variable
            de control, la condición de multiplicación hasta el 10 y el
            incremento de dicha variable de uno en uno en cada iteración del
            bucle. En este caso, 'contador++' incrementará en una unidad el valor de
            dicha variable.
            */
            resultado = contador * numero;
            System.out.println(numero + " x " + contador + " = " + resultado);
        }
        /*
        A través del operador '+' aplicado a cadenas de caracteres,
        concatenamos los valores de las variables con las cadenas de
        texto.
        */
    }
}
```

```

        caracteres que necesitamos para representar correctamente la
        salida de cada multiplicación.
    */
}
}
}

```

Autoevaluación

Cuando construimos la cabecera de un bucle for, podemos prescindir de alguno de los tres elementos que la forman e incluso, podemos utilizar más de una variable contadora separando éstas por comas. Pero, ¿qué conseguiremos si construimos un bucle de la siguiente forma?

```
for (;;) { //instrucciones }
```

Un bucle infinito.

Nada, dará un error.

Un bucle que se ejecutaría una única vez.

Solución:

Será un bucle infinito, al no tener ninguna condición o instrucción de ruptura.

1.2. Estructura for/in.

Junto a la estructura `for`, `for/in` también se considera un bucle controlado por contador. Este bucle es una mejora incorporada en la versión 5.0. de Java.

Este tipo de bucles permite realizar recorridos sobre `arrays` y colecciones de objetos. Los `arrays` son colecciones de variables que tienen el mismo tipo y se referencian por un nombre común. Así mismo, las colecciones de objetos son objetos que se dice son iterables, o que se puede iterar sobre ellos.

Este bucle es nombrado también como bucle `for` mejorado, o bucle `foreach`. En otros lenguajes de programación existen bucles muy parecidos a este.

La sintaxis es la siguiente:

```

for (declaración: expresión) {
    sentencial;
    ...
    sentenciaN;
}

```

- Donde `expresión` es un array o una colección de objetos.
- Donde `declaración` es la declaración de una variable cuyo tipo *sea compatible* con expresión. *Normalmente, será el tipo y el nombre de la variable a declarar.*

El funcionamiento consiste en que para cada elemento de la expresión, guarda el elemento en la variable declarada y realiza las instrucciones contenidas en el bucle.

Después, en cada una de las iteraciones del bucle tendremos en la variable declarada el elemento actual de la expresión.

Por tanto, para el caso de los `arrays` y de las colecciones de objetos, se recorrerá desde el primer elemento que los forma hasta el último.

Observa el contenido del código, puedes apreciar cómo se construye un bucle de este tipo y su utilización sobre un `array`.

```

public class RepetitivaForIn {

    public static void main(String[] args) {
        // Declaración e inicialización de variables
        String[] semana = {"Lunes", "Martes", "Miércoles", "Jueves", "Viernes", "Sábado", "Domingo"};

        // Salida de información
        // Utilizamos ahora el bucle for/in
        for (String día : semana) {
            /*
             * La cabecera del bucle incorpora la declaración de la variable día
             * a modo de contenedor temporal de cada uno de los elementos que forman
             * el array semana.
             * En cada una de las iteraciones del bucle, se irá cargando en la variable
             * día el valor de cada uno de los elementos que forman el array semana,
             * desde el primero al último.
             */
            System.out.println(día);
        }
    }
}

```

Los bucles `for/in` permitirán al programador despreocuparse del número de veces que se ha de iterar, pero no sabremos en qué iteración nos encontramos salvo que se añada artificialmente alguna variable contadora que nos pueda ofrecer esta información.

1.3. Estructura while.

El bucle `while` es la primera de las estructuras de repetición controladas por sucesos que vamos a estudiar.

La utilización de este bucle responde al planteamiento de la siguiente pregunta:

¿Qué podemos hacer si lo único que sabemos es que se han de repetir un conjunto de instrucciones mientras se cumpla una determinada condición?

La característica fundamental de este tipo de estructura repetitiva estriba en ser útil en aquellos casos en los que las instrucciones que forman el cuerpo del bucle podría ser necesario ejecutarlas o no.

Es decir, en el bucle `while` siempre se evaluará la condición que lo controla, y si dicha condición es cierta, el cuerpo del bucle se ejecutará una vez, y se seguirá

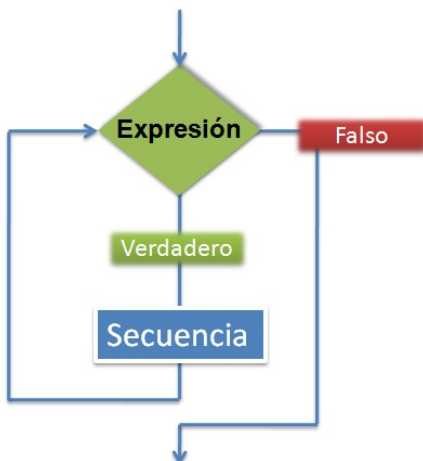
ejecutando mientras la condición sea cierta.

Pero si en la evaluación inicial de la condición ésta no es verdadera, el cuerpo del bucle no se ejecutará.

Es imprescindible que en el interior del bucle `while` se realice alguna acción que modifique la condición que controla la ejecución del mismo, en caso contrario estaríamos ante un bucle infinito.

Estructura repetitiva while	
Sintaxis:	Sintaxis:
<code>while (condición)</code>	<code>while (condición) {</code>
<code> sentencia;</code>	<code> sentencia1;</code>
	<code> ...</code>
	<code> sentenciaN;</code>
	<code>}</code>
Funcionamiento:	
Mientras la condición sea <i>cierta</i> , el bucle <i>se repetirá</i> , ejecutando la/s instrucción/es de su interior.	
En el momento en el que la condición no se cumpla, el control del flujo del programa pasará a la siguiente instrucción que exista justo detrás del bucle while.	
La condición se evaluará siempre al principio, y podrá darse el caso de que las instrucciones contenidas en él no lleguen a ejecutarse nunca si no se satisface la condición de partida.	

En la siguiente imagen puedes ver un diagrama de flujo que representa el funcionamiento de este tipo de estructura repetitiva.



Debes conocer

Accede al siguiente código java y podrás analizar un ejemplo de utilización del bucle `while` para la impresión por pantalla de la tabla de multiplicar del siete. Lee atentamente los comentarios incluidos en el código, pues aclaran algunas cuestiones interesantes sobre este bucle. Como podrás comprobar, el resultado de este bucle es totalmente equivalente al obtenido utilizando el bucle `for`.

```
public class repetitiva_while {
    public static void main(String[] args) {

        // Declaración e inicialización de variables
        int numero = 7;
        int contador;
        int resultado=0;

        //Salida de información
        System.out.println ("Tabla de multiplicar del " + numero);
        System.out.println ("..... ");

        //Utilizamos ahora el bucle while
        contador = 1; //inicializamos la variable contadora
        while (contador <= 10) //Establecemos la condición del bucle
        {
            resultado = contador * numero;
            System.out.println(numero + " x " + contador + " = " + resultado);
            //Modificamos el valor de la variable contadora, para hacer que el
            //bucle pueda seguir iterando hasta llegar a finalizar
            contador++;
        }
    }
}
```

```

    }
}
}

```

1.4. Estructura do-while.

La segunda de las estructuras repetitivas controladas por sucesos es **do-while**. En este caso, la pregunta que nos planteamos es la siguiente: ¿Qué podemos hacer si lo único que sabemos es que se han de ejecutar, *al menos una vez*, un conjunto de instrucciones y seguir repitiéndose hasta que se cumpla una determinada condición?

La característica fundamental de este tipo de estructura repetitiva estriba en ser útil en aquellos casos en los que las instrucciones que forman el cuerpo del bucle necesitan ser ejecutadas, *al menos, una vez* y repetir su ejecución mientras que la condición sea verdadera. Por tanto, en esta estructura repetitiva **siempre se ejecuta el cuerpo del bucle una primera vez**.

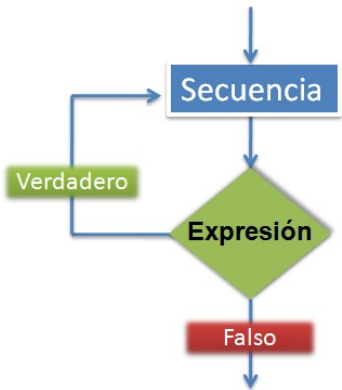
Es imprescindible que en el interior del bucle se realice alguna acción que modifique la condición que controla la ejecución del mismo, en caso contrario estaríamos ante un bucle infinito.

Estructura repetitiva **do-while**

Sintaxis: <div>do</div> <div>sentencia;</div> <div>while (condición);</div>	Sintaxis: <div>do</div> <div>{</div> <div>sentencia1;</div> <div>...</div> <div>sentenciaN;</div> <div>}</div> <div>while (condición);</div>
---	--

Funcionamiento:
 El cuerpo del bucle se ejecuta la primera vez, a continuación se evaluará la condición y, si ésta es cierta, el cuerpo del bucle volverá a repetirse. El bucle finalizará cuando la evaluación de la condición sea falsa. En ese momento el control del flujo del programa pasará a la siguiente instrucción que exista justo detrás del bucle do-while. La condición se evaluará siempre después de una primera ejecución del cuerpo del bucle.

En la siguiente imagen puedes ver un diagrama de flujo que representa el funcionamiento de este tipo de estructura repetitiva.



Debes conocer

Estudia el siguiente programa java y podrás analizar un ejemplo de utilización del bucle `do-while` para la impresión por pantalla de la tabla de multiplicar del siete. Lee atentamente los comentarios incluidos en el código, pues aclaran algunas cuestiones interesantes sobre este bucle. Como podrás comprobar, el resultado de este bucle es totalmente equivalente al obtenido utilizando el bucle `for` y el bucle `while`.

```

public class RepetitivaDoWhile {
    public static void main(String[] args) {
        // Declaración e inicialización de variables
        int numero = 7;
        int contador;
        int resultado = 0;

        // Salida de información
    }
}

```

```
System.out.println("Tabla de multiplicar del " + numero);
System.out.println(".....");

// Utilizamos ahora el bucle do-while
contador = 1; // Inicializamos la variable contadora
do {
    resultado = contador * numero;
    System.out.println(numero + " x " + contador + " = " + resultado);
    // Modificamos el valor de la variable contadora, para hacer que el
    // bucle pueda seguir iterando hasta llegar a finalizar
    contador++;
} while (contador <= 10); // Establecemos la condición del bucle
}
```