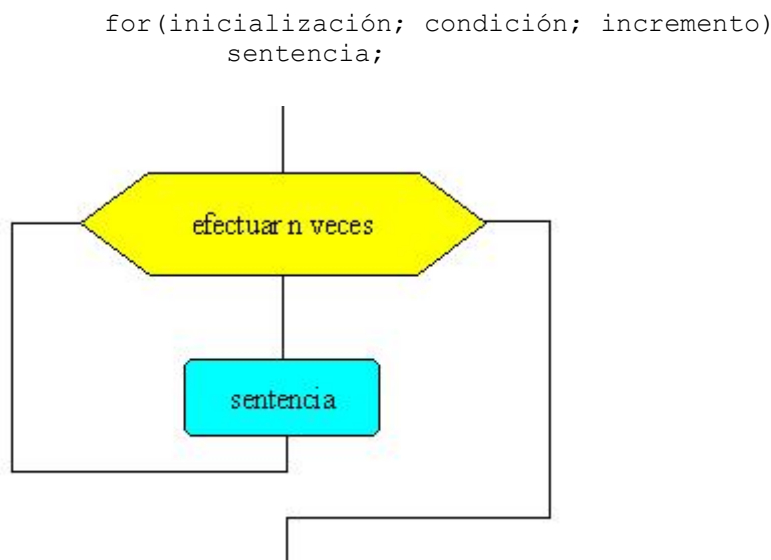


El flujo de un programa (sentencias iterativas)

Tan importantes como las sentencias condiciones son las sentencias iterativas o repetitivas. Normalmente, las sentencias de un programa son ejecutadas en el orden en el que aparecen. Cada sentencia es ejecutada una y solamente una vez. El lenguaje Java, como la mayoría de los lenguajes, proporciona sentencias que permiten realizar una tarea una y otra vez hasta que se cumpla una determinada condición, dicha tarea viene definida por un conjunto de sentencias agrupadas en un bloque. Las sentencias iterativas son **for**, **while** y **do...while**

La sentencia *for*

Esta sentencia se encuentra en la mayoría de los lenguajes de programación. El bucle **for** se empleará cuando conocemos el número de veces que se ejecutará una sentencia o un bloque de sentencias, tal como se indica en la figura. La forma general que adopta la sentencia **for** es



El primer término *inicialización*, se usa para inicializar una variable índice, que controla el número de veces que se ejecutará el bucle. La *condición* representa la condición que ha de ser satisfecha para que el bucle continúe su ejecución. El *incremento* representa la cantidad que se incrementa la variable índice en cada repetición.

Ejemplo: Escribir un programa que imprima los primeros 10 primeros números enteros empezando por el cero

```
for (int i = 0; i < 10; i++) {
    System.out.println(i);
}
```

El resultado será: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

La variable índice *i* se declara y se inicializa en el término *inicialización*, la *condición* se expresa de modo que *i* se debe mantener estrictamente menor que 10; la variable *i* se incrementa una unidad en cada repetición del bucle. La variable *i* es local al bucle, por lo que deja de existir una vez que se sale del bucle.

Ejemplo: Escribir un programa que imprima los números pares positivos menores o iguales que 20

```
for (int i=2; i <=20; i += 2) {  
    System.out.println(i);  
}
```

Ejemplo: Escribir un programa que imprima los números pares positivos menores o iguales que 20 en orden decreciente

```
for (int i=20; i >= 2; i -= 2) {  
    System.out.println(i);  
}
```

Ejemplo: Escribir un programa que calcule el factorial de un número empleando la sentencia iterativa **for**. Guardar el resultado en un número entero de tipo **long**.

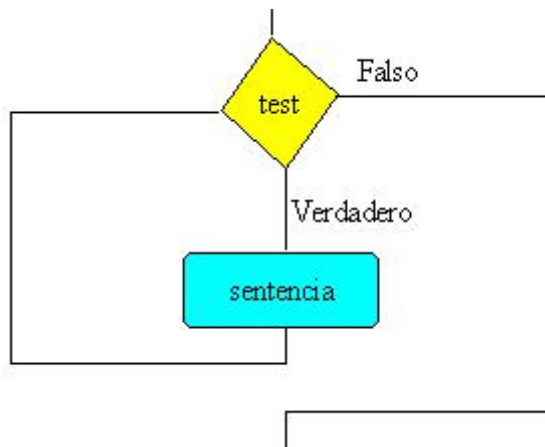
Definición: el factorial de un número *n* es el resultado del producto $1 \cdot 2 \cdot 3 \cdot \dots \cdot (n-1) \cdot n$.

```
public class FactorialApp {  
    public static void main(String[] args) {  
        int numero=4;  
        long resultado=1;  
        for(int i=1; i<=numero; i++){  
            resultado*=i;  
        }  
        System.out.println("El factorial es "+resultado);  
    }  
}
```

La sentencia *while*

A la palabra reservada **while** le sigue una condición encerrada entre paréntesis. El bloque de sentencias que le siguen se ejecuta siempre que la condición sea verdadera tal como se ve en la figura. La forma general que adopta la sentencia **while** es:

```
while (condición)  
    sentencia;
```



Ejemplo: Escribir un programa que imprima los primeros 10 números enteros empezando por el cero, empleando la sentencia iterativa *while*.

```

int i=0;

while (i<10) {
    System.out.println(i);
    i++;
}

```

El valor inicial de i es cero, se comprueba la condición ($i < 10$), la cual resulta verdadera. Dentro del bucle, se imprime i , y se incrementa la variable contador i , en una unidad. Cuando i vale 10, la condición ($i < 10$) resulta falsa y el bucle ya no se ejecuta. Si el valor inicial de i fuese 10, no se ejecutaría el bucle. Por tanto, el bucle **while** no se ejecuta si la condición es falsa.

Ejemplo: Escribir un programa que calcule el factorial de un número empleando la sentencia iterativa **while**

```

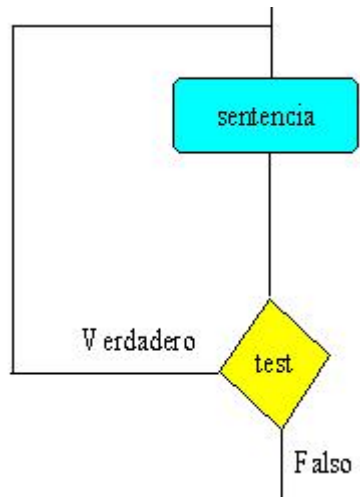
public class FactorialApp1 {
    public static void main(String[] args) {
        int numero=4;
        long resultado=1;
        while(numero>0){
            resultado*=numero;
            numero--;
        }
        System.out.println("El factorial es "+resultado);
    }
}

```

La sentencia *do...while* (opcional)

Como hemos podido apreciar las sentencias **for** y **while** la condición está al principio del bucle, sin embargo, **do...while** la condición está al final del bucle, por lo que el bucle se ejecuta por lo menos una vez tal como se ve en la figura. **do** marca el comienzo del bucle y **while** el final del mismo. La forma general es:

```
do{
    sentencia;
}while(condición);
```



Ejemplo: Escribir un programa que imprima los primeros 10 primeros números enteros empezando por el cero, empleando la sentencia iterativa *do..while*.

```
int i=0;

do{
    System.out.println(i);
    i++;
}while(i < 10);
```

El bucle **do...while**, se usa menos que el bucle **while**, ya que habitualmente evaluamos la expresión que controla el bucle al comienzo, no al final.

La sentencia *break*

A veces es necesario interrumpir la ejecución de un bucle **for**, **while**, o **do...while**.

```
for(int i = 0; i < 10; i++){
    if (i == 8) break;
    System.out.println(i);
}
```

Consideremos de nuevo el ejemplo del bucle **for**, que imprime los 10 primeros números enteros, se interrumpe la ejecución del bucle cuando se cumple la condición de que la variable contador *i* valga 8. El código se leerá: "salir del bucle cuando la variable contador *i*, sea igual a 8".

Como podemos apreciar, la ejecución del bucle finaliza prematuramente. Quizás el lector pueda pensar que esto no es de gran utilidad pues, el código anterior es equivalente a

```
for(int i = 0; i <=8; i++)
```

```
System.out.println(i);
```

Sin embargo, podemos salir fuera del bucle prematuramente si se cumple alguna condición de finalización.

```
while(true){  
    if (condicionFinal)    break;  
    //...otras sentencias  
}
```

Como podemos apreciar en esta porción de código, la expresión en el bucle **while** es siempre verdadera, por tanto, tiene que haber algún mecanismo que nos permita salir del bucle. Si la condición de finalización es verdadera, es decir la variable *condicionFinal* del tipo **boolean** toma el valor **true**, se sale del bucle, en caso contrario se continua el procesamiento de los datos.

La sentencia *continue*

La sentencia **continue**, fuerza al bucle a comenzar la siguiente iteración desde el principio. En la siguiente porción de código, se imprimen todos los números del 0 al 9 excepto el número 8.

```
for(int i = 0; i < 10; i++){  
    if (i == 8)    continue;  
    System.out.println(i);  
}
```