Iterativas

Programación I

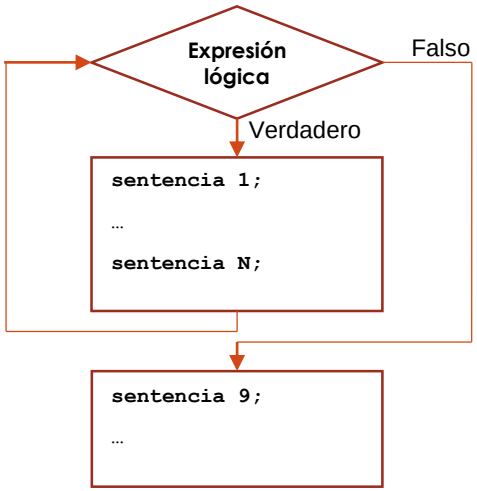
Objetivos del tema

- Comprender el funcionamiento de las sentencias iterativas (while, do-while y for)
- Codificar programas utilizando la combinación de sentencias iterativas y condicionales

Sentencias iterativas

- Una sentencia iterativa se utiliza cuando se requiere realizar una o muchas tareas varias veces (varias veces puede ser ninguna, una o más).
- En la sentencia iterativa **while (expresión logica){…}** hay una expresión lógica que:
  - si es cierta o verdadera, ejecuta las sentencias entre llaves, y luego vuelve al inicio de la sentencia iterativa para evaluar nuevamente la expresión lógica.
  - si es falsa, sale de la iteración.

```
while (expresion logica){
  sentencia 1;
  …
  sentencia N;
}
sentencia 9;
…
```



Sentencias iterativas

- Mientras se verdadera la expresión lógica de **while (expresión logica){…}** va a ejecutarse lo que esta dentro de la {…} en el orden en que aparecen.
- Eso significa que algún valor dentro de la expresión lógica deberá cambiar en algún ciclo de la repetición, sino se producirá un ciclo infinito (la expresión lógica siempre valdrá lo mismo y el programa no podrá salir del ciclo).
- La expresión lógica debe contener una o más variables que se modifiquen dentro de las sentencias en las {…}, asegurándose que la expresión lógica con ese cambio de valor en al algún momento sea falsa (evitar el ciclo infinito).

Ejemplo 1

```
/*Dado un numero entero con valor inicial 1, hacer una iteración que haga
  incrementar numero de a uno hasta un valor MAX = 4 (constante). Mientras
  itera deberá imprimir numero
*/
public class Clase_5_Ejemplo_1 {
 public static void main (String [] args) {
        final int MAX = 4;
        int numero = 1;
        while (numero <= MAX){
                System.out.println("El numero es: " + numero);
                //al cambiar de valor el numero significa que el valor de la
                //expresion logica va a cambiar
                numero++;
        }
  }
}
```

Ejemplo 1

¿Qué hace el programa Clase_5_Ejemplo_1 cuando se ejecuta?

Ejecuta cada sentencia en el orden en que aparecen

Define una constante MAX = 4;
Define una variable numero = 1;

Pregunta mientras numero<=MAX, en valores sería si (1<=4) es verdadero, entonces
Imprime El numero es: 1
Luego ejecuta numero++, que es igual a numero=numero+1, por lo tanto numero=2
Por estar dentro de la sentencia while(...){...} se vuelve al comienzo de dicha sentencia

Pregunta mientras numero<=MAX, en valores sería si (2<=4) es verdadero, entonces
Imprime El numero es: 2
Luego ejecuta numero++, que es igual a numero=numero+1, por lo tanto numero=3
Por estar dentro de la sentencia while(...){...} se vuelve al comienzo de dicha sentencia

Pregunta mientras numero<=MAX, en valores sería si (3<=4) es verdadero, entonces
Imprime El numero es: 3
Luego ejecuta numero++, que es igual a numero=numero+1, por lo tanto numero=4
Por estar dentro de la sentencia while(...){...} se vuelve al comienzo de dicha sentencia

Pregunta mientras numero<=MAX, en valores sería si (4<=4) es verdadero, entonces
Imprime El numero es: 4
Luego ejecuta numero++, que es igual a numero=numero+1, por lo tanto numero=5
Por estar dentro de la sentencia while(...){...} se vuelve al comienzo de dicha sentencia

Pregunta mientras numero<=MAX, en valores sería si (5<=4) es falso,
entonces pasa a la próxima sentencia después de las llaves {...} del while
Como no hay más sentencias para ejecutar el programa termina.

```
public class Clase_5_Ejemplo_1 {
 public static void main (String [] args){
  final int MAX = 4;
  int numero = 1;
  while (numero <= MAX){
   System.out.println("El numero es:
   "+numero); numero++;
  }
  }
}
```

Sentencias iterativas y declaraciones de variables

- La utilización de sentencias iterativas implica reubicar sentencias declarativas como la de **`BufferedReader`**, para no iterar con una declaración dentro de las llaves {…} del ciclo (evitar que en cada ciclo se declare la misma variable).

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
public class Clase_5_Ejemplo_2 {
  public static void main(String[] args) {
  //ubicar el buffer entrada cerca de la región de declaración de variables
  BufferedReader entrada = new BufferedReader(new InputStreamReader(System.in));
  try{
        while (…){
        }
  }
  catch (Exception exc){
        System.out.println(exc);
  }
  }
}
```

Ejemplo 2

```
/*Hacer un programa que mientras que el usuario cargue un numero entero distinto de 0 lo imprima
*/
import java.io.BufferedReader;
import java.io.InputStreamReader;
public class Clase_5_Ejemplo_2 {
  public static void main(String[] args) {
   int numero;
   //la declaración del buffer entrada la ubico al principio junto con las otras declaraciones
   BufferedReader entrada = new BufferedReader(new InputStreamReader(System.in));
   try{
           //el usuario carga un valor la primera vez
           System.out.println("Ingrese un numero entero (0 para salir): ");
           numero = Integer.valueOf(entrada.readLine());
           while (numero != 0){
                   //si numero es distinto de 0 lo imprime, vuelve a pedir su carga, y regresa al while
                   System.out.println("El valor es: " + numero);
                   System.out.println("Ingrese un numero entero (0 para salir): ");
                   numero = Integer.valueOf(entrada.readLine());
           }
   }
   catch (Exception exc){
           System.out.println(exc);
   }
  }
}
```

Video de explicación

https://drive.google.com/file/d/1tNSIIhiHz2eXc9HH1tBkO5xrgyljMXBN/view?usp=sharing

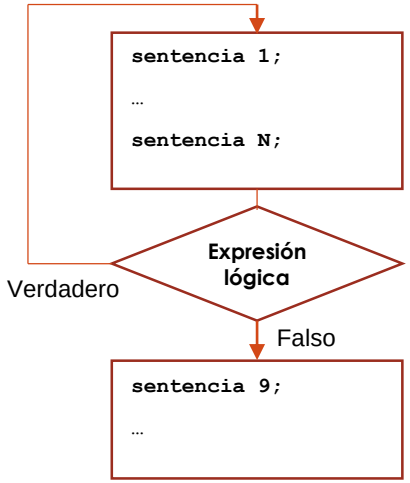
Ejemplo 3

```
/*Hacer un programa que mientras que el usuario cargue un numero entero distinto de 0 lo imprima si es par
*/
import java.io.BufferedReader;
import java.io.InputStreamReader;
public class Clase_5_Ejemplo_3 {
  public static void main(String[] args) {
   int numero;
   BufferedReader entrada = new BufferedReader(new InputStreamReader(System.in));
   try{
          System.out.println("Ingrese un numero entero (0 para salir):");
          numero = Integer.valueOf(entrada.readLine());
          while (numero != 0){
                    if ((numero % 2) == 0) {
                              System.out.println("El valor es :" + numero);
                    }
                    System.out.println("Ingrese un numero entero (0 para salir):");
                    numero = Integer.valueOf(entrada.readLine());
          }
   }
   catch (Exception exc){
          System.out.println(exc);
   }
  }
}
```

Sentencias iterativas

- Hay sentencias iterativas donde la expresión lógica se evalúa luego de ejecutar al menos una vez las sentencias entre llaves {}. Por ejemplo **do{…}while(expresión logica)**

```
do {
  sentencia 1;
  …
  sentencia N;
} while(expresion logica);
sentencia 9;
…
```




```
//se ejecutan una vez las sentencias entre llaves y
//luego mientras la expresión lógica sea verdadera
//se van a ejecutar las sentencias entre llaves
//produciéndose un ciclo
```

Ejemplo 4

```
/*Dado un numero entero con valor inicial 1, hacer una iteración que haga
  incrementar numero de a uno hasta un valor MAX = 5 (constante). Mientras
  itera deberá imprimir si numero es par
*/
public class Clase_5_Ejemplo_4 {
 public static void main (String [] args) {
        final int MAX = 5;
        int numero = 1;
        do{
                if (numero % 2 == 0) {
                        System.out.println(numero + " es par");
                }
                //al cambiar de valor el numero significa que el valor de la
                //expresion logica va a cambiar
                numero++;
        }while (numero <= MAX);
  }
}
```

Video de explicación

https://drive.google.com/file/d/1wX0TVrIXY4ROaQzIHYQvUGLICWugZk6a/view?usp=sharing

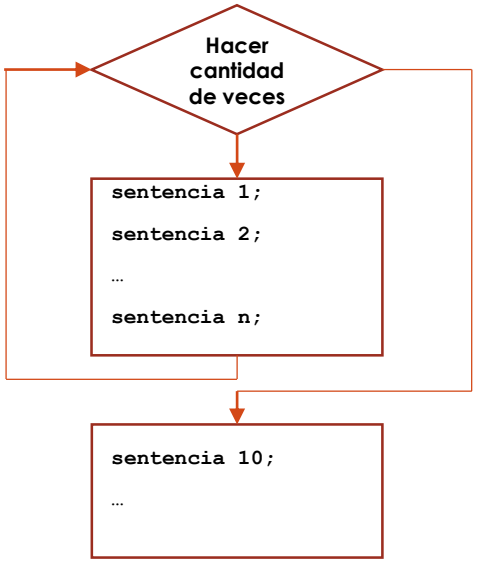
Ejemplo 5

```
/*Hacer un programa que mientras que el usuario cargue un numero entero distinto de 0 lo
  imprima
*/
import java.io.BufferedReader;
import java.io.InputStreamReader;
public class Clase_5_Ejemplo_5 {
  public static void main(String[] args) {
  int numero;
  BufferedReader entrada = new BufferedReader(new InputStreamReader(System.in));
  try{
        do{
                System.out.println("Ingrese un numero entero (0 para salir):");
                numero = Integer.valueOf(entrada.readLine());
                //la primera vez imprime el valor aunque sea igual a 0
                System.out.println("El valor es :" + numero);
        }while (numero != 0);
  }
  catch (Exception exc){
        System.out.println(exc);
  }
  }
}
```

Sentencia iterativa for

- La sentencia iterativa **for(condición inicial de variable de control; expresión lógica sobre variable de control; actualización de la variable de control){…}** es una sentencia que **SOLO** se utiliza cuando se conoce exactamente la cantidad de iteraciones.

```
for (condición inicial de variable de control; expresión lógica
  sobre variable de control; actualización de la variable de
  control){
  sentencia 1;
  sentencia 2;
  …
  sentencia n;
}
sentencia 10;
…
```



Ejemplo 6

```
/*Dado un numero entero con valor inicial 1, hacer una iteración que haga
  incrementar numero de a uno hasta un valor MAX = 4 (constante). Mientras
  itera deberá imprimir numero
*/
public class Clase_5_Ejemplo_6 {
 public static void main (String [] args) {
        final int MAX = 4;
        //declara numero de tipo entero como variable de control con valor
        //inicial 1
        //expresión lógica (que incluye variable de control) cuyo valor indicará
        //si ejecuta o no las sentencias dentro de las {…}
        //incremento de la variable de control luego de que se ejecuta la ultima
        //sentencia dentro de las {…}
        for (int numero = 1; numero <= MAX; numero++){
                System.out.println("El numero es: " + numero);
        }
  }
}
```

Ejemplo 6

¿Qué hace el programa Clase_5_Ejemplo_6 cuando se ejecuta?

```
public class Clase_5_Ejemplo_6 {
 public static void main (String [] args){
  final int MAX = 5;
  for (int numero = 1; numero <= MAX; numero++){
   System.out.println("El numero es: " + numero);
  }
  }
}
```

Ejecuta cada sentencia en el orden en que aparecen

Define una constante MAX = 4;

Comienza la sentencia for (){} definiendo una variable numero = 1;

Pregunta numero<=MAX, en valores sería si (1<=4) es verdadero, entonces
Imprime El numero es: 1
Luego de que termina de ejecutar las sentencias dentro {} del for ejecuta numero++,
por lo tanto numero=2, y regresa a la expresión lógica de dicha sentencia

Pregunta numero<=MAX, en valores sería si (2<=4) es verdadero, entonces
Imprime El numero es: 2
Luego de que termina de ejecutar las sentencias dentro {} del for ejecuta numero++,
por lo tanto numero=3, y regresa a la expresión lógica de dicha sentencia

Pregunta numero<=MAX, en valores sería si (3<=4) es verdadero, entonces
Imprime El numero es: 3
Luego de que termina de ejecutar las sentencias dentro {} del for ejecuta numero++,
por lo tanto numero=4, y regresa a la expresión lógica de dicha sentencia

Pregunta numero<=MAX, en valores sería si (4<=4) es verdadero, entonces
Imprime El numero es: 4
Luego de que termina de ejecutar las sentencias dentro {} del for ejecuta numero++,
por lo tanto numero=5, y regresa a la expresión lógica de dicha sentencia

Pregunta numero<=MAX, en valores sería si (5<=4) es falso,
entonces pasa a la próxima sentencia después de las llaves {...} del for
Como no hay más sentencias para ejecutar el programa termina.

Ejemplo 7

```
/*Dado un numero entero con valor inicial 1, hacer una iteración que haga
  incrementar numero de a uno hasta un valor MAX = 5 (constante). Mientras
  itera deberá imprimir si numero es par y el valor del numero
  independientemente de si es par o no
*/
public class Clase_5_Ejemplo_7 {
 public static void main (String [] args) {
        final int MAX = 5;
        //incremento de la variable de control luego de que se ejecuta la ultima
        //sentencia dentro de las {…})
        for (int numero = 1; numero <= MAX; numero++){
                if (numero % 2 == 0) {
                        System.out.println(numero + " es par");
                }
                System.out.println("El numero es: " + numero);
        }
  }
}
```

Video de explicación

https://drive.google.com/file/d/1VdMVfuwc0AqAN7VigavERUJT3IkirNk1/view?usp=sharing

Ejemplo 8

```
/*Dado un numero entero con valor inicial 3, imprimir la tabla de multiplicar de numero
*/
public class Clase_5_Ejemplo_8 {
 public static void main (String [] args) {
        //declaro una constante MAX=10 ya que en la tabla de multiplicar 10
        //siempre es el maximo multiplicador y no cambia
        final int MAX = 10;
        int numero = 3;
        System.out.println("Tabla de multiplicar de: " + numero);
        //multiplicador es la variable de control que hace de multiplicador
        for (int multiplicador=1; multiplicador <=MAX; multiplicador++) {
                System.out.println(multiplicador +" * "+numero+" = "+(multiplicador *numero));
        }
  }
}
```

Recomendaciones para while, do while y for

- Las sentencias iterativas **while(expresión lógica){…}** y **do{…}while(expresión lógica)** se utilizan cuando se conoce exactamente o no la cantidad de iteraciones.
- La sentencia iterativa **for(; ;){…}** se utiliza cuando se conoce exactamente la cantidad de iteraciones.
- Por ejemplo cuando la expresión lógica tiene una variable que se carga dentro de la **{…}** o su valor es resultado de una cálculo (no de un incremento o decremento) usar **while(expresión lógica){…}** y **do{…}while(expresión lógica)** ya que se desconoce cuantas iteraciones hará la sentencia.

Sentencias iterativas anidadas

- Dentro de los bloques **{…}** de las sentencias iterativas hay sentencias que pueden ser o no otras sentencias iterativas.

```
public class Clase_5_Ejemplo_9 {
  public static void main(String[] args) {
        …
        while (…){
                …
                do{
                        …
                }while(…);
                …
                for( ; ; ){
                        …
                }
        }
        …
  }
}
```

Ejemplo 9

```
/*Hacer un programa que mientras que el usuario cargue un numero entero entre 1 y 3, imprima el numero y la tabla de multiplicar del numero ingresado
*/
import java.io.BufferedReader;
import java.io.InputStreamReader;
public class Clase_5_Ejemplo_9 {
    public static void main(String[] args) {
                final int MAXMULTIPLICADOR = 10;
                final int MAX = 3;
                final int MIN = 1;
                int numero;
                BufferedReader entrada = new BufferedReader(new InputStreamReader(System.in));
                try{
                                System.out.println("Ingrese un numero entero entre "+MIN+" y "+MAX+", otro numero para salir:");
                                numero = Integer.valueOf(entrada.readLine());
                                //1 y 3 deberían ser constantes si se usaran mas de una vez, ej en otra sentencia
                                while ((MIN <= numero)&&(numero <= MAX)){
                                                System.out.println("Tabla de multiplicar de: " + numero);
                                                for (int multiplicador = 1; multiplicador <= MAXMULTIPLICADOR; multiplicador++) {
                                                                System.out.println(multiplicador +" * "+numero+" = "+(multiplicador*numero));
                                                }
                                                System.out.println("Ingrese un numero entero entre "+MIN+" y "+MAX+", otro numero para salir:");
                                                numero = Integer.valueOf(entrada.readLine());
                                }
                }
                catch (Exception exc){
                                System.out.println(exc);
                }
     }
}
```

Ejemplo 10

```
/*Hacer un programa que mientras que el usuario cargue un numero entero entre 0 y 1000 (no incluidos),
guarde el menor de los numeros ingresados. Al finalizar el ciclo imprima el menor por pantalla
*/
import java.io.BufferedReader;
import java.io.InputStreamReader;
public class Clase_5_Ejemplo_10 {
    public static void main(String[] args) {
                final int MIN = 0;
                final int MAX = 1000;
                int numero;
                int menor = MAX;
                BufferedReader entrada = new BufferedReader(new InputStreamReader(System.in));
                try{
                                System.out.println("Ingrese un numero entero entre "+MIN+" y "+MAX+", con otro valor sale del ciclo");
                                numero = Integer.valueOf(entrada.readLine());
                                while ((MIN < numero)&&(numero < MAX)){
                                                if (numero<menor){
                                                                menor=numero;
                                                }
                                                System.out.println("Ingrese un numero entero entre "+MIN+" y "+MAX+", con otro valor sale del ciclo");
                                                numero = Integer.valueOf(entrada.readLine());
                                }
                }
                catch (Exception exc){
                                System.out.println(exc);
                }
                if (menor != MAX){
                                System.out.println("El menor ingresado es: " + menor);
                }
    }
}
```

Video de explicación

https://drive.google.com/file/d/16GFHtLPG57S6hfodGYoo_ub35H9YVRyH/view?usp=sharing

Práctico

1. Escribir un programa que mientras el usuario ingrese un número entero menor que 10 y mayor a 1, muestre por pantalla si el número es múltiplo de 2 y múltiplo de 3 simultáneamente. (*¿Los valores mencionados deberían ser constantes?. De a poco habría que definirlos como constantes*).
2. Escribir un programa que solicite al usuario el ingreso de un número entero positivo, y muestre por pantalla los valores entre el numero ingresado y 0 de manera decreciente.
3. Escribir un programa que mientras el usuario ingresa un caracter distinto del caracter '*', muestre por pantalla si es caracter digito, o si es caracter vocal minúscula.
4. Escribir un programa que mientras que el usuario ingrese un número entero distinto de 0, pida ingresar otro número entero y lo imprima.
5. Escribir un programa que mientras que el usuario ingrese un caracter dígito o caracter letra minúscula, imprima si es caracter dígito o caracter letra minúscula, y si es letra minúscula imprimir si es vocal o consonante.
6. Escribir un programa que mientras que el usuario ingrese un número entero entre 1 y 10 inclusive, lleve la suma de los números ingresados. Finalmente, cuando sale del ciclo muestre por pantalla el resultado de la suma.

Práctico

7. Escribir un programa que mientras que el usuario ingrese un número entero entre 1 y 10 inclusive, lleve la suma de los números ingresados y la cantidad de sumas que realizó. Finalmente, cuando sale del ciclo muestre por pantalla el resultado del promedio de todo lo ingresado.

8. Escribir un programa que mientras el usuario ingrese un caracter letra minúscula, acumule la cantidad de vocales que ingreso. Finalmente muestre por pantalla dicha cantidad.

9. Escribir un programa que mientras el usuario ingrese un caracter letra minúscula, se quede con la menor y la mayor letra ingresada. Finalmente muestre por pantalla dichas letras.

10. Escribir un programa que mientras el usuario ingresa un numero de mes (entero) entre 1 y 12 inclusive, muestre por pantalla la cantidad de días del mes ingresado (suponer febrero de 28 días).

11. Escribir un programa que mientras que el usuario ingrese un caracter letra minúscula, pida ingresar un numero entero. Si el número ingresado está entre 1 y 5 inclusive deberá imprimir la tabla de multiplicar de dicho numero.