Práctica #5 – Bucles

RESULTADOS DE APRENDIZAJE:

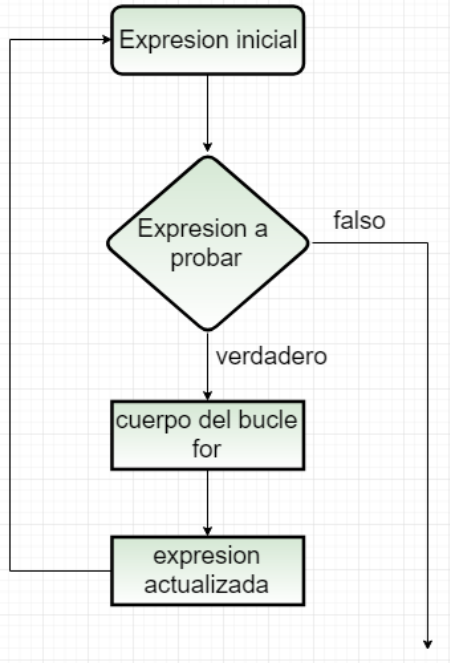
Al finalizar esta sesión de práctica el estudiante deberá:

- ✓ Crear código que permita la captura de datos.
- ✓ Realizar operaciones de asignación de datos.
- ✓ Comprender el uso de la clase MATH de Java.
- ✓ Realizar operaciones matemáticas en Java.

En Java hay 3 tipos de sentencias de control (for, while y do while) para ejecutar un código un número determinado de veces y aunque sabiendo usar un solo tipo te las puedes arreglar para hacer cualquier cosa siempre hay un tipo de bucle más apropiado para cada situación, si no solo habría uno ¿no?

BUCLE FOR EN JAVA

El bucle for sirve para ejecutar un código un número conocido de veces, por ejemplo recorrer un array o cualquier otro tipo de colección o simplemente ejecutar el código un número concreto de veces.



En java hay dos posibles tipos de bucle for:

El más habitual es en el que se establece el numero desde el que va a empezar el bucle, hasta cuando se va a seguir ejecutando el bucle y finalmente el modo en el que se actualiza el contador del bucle, que aunque habitualmente lo que se hace es incrementar en una unidad el contador esto no tiene que ser obligatoriamente así.

```
for (inicializacion; condicion; actualizacion) {
    ...
}
```

El otro formato de bucle for es el ideal para recorrer colecciones de objetos sean del tipo que sean (arrays, ArrayList, HashMap, ...) y en este caso hay que definir un iterador que nos devolverá un elemento de la colección en cada iteración y la colección que se quiere recorrer.

```
for (iterador; coleccion) {
    ...
}
```

ACTIVIDAD 1

Este es un ejemplo muy sencillo de bucle for en el que lo único que hacemos es imprimir por pantalla los números entre 1 el numero que nosotros queramos y como puedes ver en este caso al contrario que como ocurre en los bucles while y do while sabemos con certeza el numero de veces que se va a ejecutar el bucle.

```
package bucles1;

public class Bucles1 {
    public static void main(String[] args) {
        int numero = 10;
        System.out.println("Cuenta hasta: " + numero);
        for (int i = 1; i <= numero; i++) {
            System.out.println(i);
                  }
      }
}
```

Modifica el código anterior para que acepte el número que el usuario introduzca

ACTIVIDAD 2

Visto lo sencillo del ejemplo anterior vamos a ver lo sencillo que es recorrer una colección de objetos si usamos el otro modelo de bucle for.

```
package bucles2;

public class Bucles2 {
    public static void main(String[] args) {
        String[] dias = {"Lunes", "Martes", "Miercoles", "Jueves", "Viernes",
"Sabado", "Domingo"};
        System.out.println("Los dias de la semana son:");
        for (String d: dias) {
            System.out.println(d);
            }
      }
}
```

ACTIVIDAD 3

En este caso lo que estamos recorriendo es un array o colección que también se podría recorrer de forma sencilla con la otra forma de bucle for aunque como podrás ver nos obliga a escribir algo más de código, por lo que para recorrer cualquier tipo de colección normalmente usaremos la opción anterior.

```
package bucles3;
public class Bucles3 {
    public static void main(String[] args) {
        String[] dias = {"Lunes", "Martes", "Miercoles", "Jueves", "Viernes",
"Sabado", "Domingo"};

        System.out.println("Los dias de la semana son:");
        for (int i = 0; i < dias.length; i++) {
            System.out.println(dias[i]);
        }
    }
}
```

BUCLE WHILE EN JAVA

El bucle while es tan sencillo como decir mientras se cumpla la condición se ejecuta el código que haya dentro del bucle, y en el momento que ya no se cumpla esa condición se sale del bucle.

```
while (condicion) {
    ...
}
```

Por lo tanto, este tipo de bucle es el idóneo cuando necesitamos que un fragmento de código se repita un número de veces variable.

ACTIVIDAD 4

El siguiente programa imprime una línea 10 veces

```
package bucles4;

public class Bucles4 {
    public static void main(String[] args) {
      int i = 1;

      while (i <= 10) {
         System.out.println("Linea " + i);
         ++i;
      }
    }
}
```

Modifica el programa anterior para que la linea se repita las veces que el usuario solicite.

ACTIVIDAD 5

Otro ejemplo claro de un bucle que no sabemos cuántas veces se va a ejecutar es cuando le pedimos al usuario que introduzca algo por teclado, porque se puede equivocar o introducirlo mal porque le apetezca. El siguiente programa pedirá un numero positivo, si el usuario no lo introduce seguirá pidiéndolo.

```
package bucles5;
import java.util.Scanner;

public class Bucles5 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int numero = -1;
        while (numero <= 0) {
            System.out.println("Introduce un numero positivo: ");
            numero = sc.nextInt();
        }
        sc.close();
    }
}
```

En este ejemplo no se hace control de excepciones por lo que si el usuario mete algo que no sea un número el programa fallaría.

ACTIVIDAD 6

Este programa imprime la suma de los números naturales

```
package bucles6;

public class Bucles6 {
    public static void main(String[] args) {
        int sum = 0, i = 100;
        while (i != 0) {
            sum += i;      // sum = sum + i;
            --i;
        }
        System.out.println("Suma = " + sum);
    }
}
```

Modifica el programa para que el usuario introduzca hasta que numero sumar

BUCLE DO WHILE EN JAVA

El bucle do while es prácticamente igual al while, pero con la diferencia de que el código del bucle se ejecutara al menos una vez ya que la comprobación se hace después de cada iteración y no antes como en el caso del while.

```
do {
    ...
} while (condicion)
```

Este tipo de bucle es el idóneo cuando necesitamos que un fragmento de código se ejecute al menos una vez y dependiendo de las circunstancias puede ser que se vuelva a repetir un número indeterminado de veces o ninguna.

ACTIVIDAD 7

Un ejemplo claro de bucle do while puede ser el siguiente, en el que necesitamos obtener un número par aleatorio menor que el que nosotros le digamos.

Está claro que al menos se tiene que ejecutar una vez para tener un número aleatorio y si el número aleatorio no es par pues tendrá que repetirse hasta que sí que lo sea.

```
package bucles7;

public class Bucles7 {
    public static void main(String[] args) {
       int numero = 65;
       System.out.println("Numero par menor que: " + numero);
       int n;
       do {
           n = (int) (Math.random() * numero);
           System.out.println(n);
       } while (n % 2 != 0);
       System.out.println("Y el numero par elegido es: " + n);
    }
}
```

Modifica el código para que el usuario pueda introducir el valor para la variable numero.