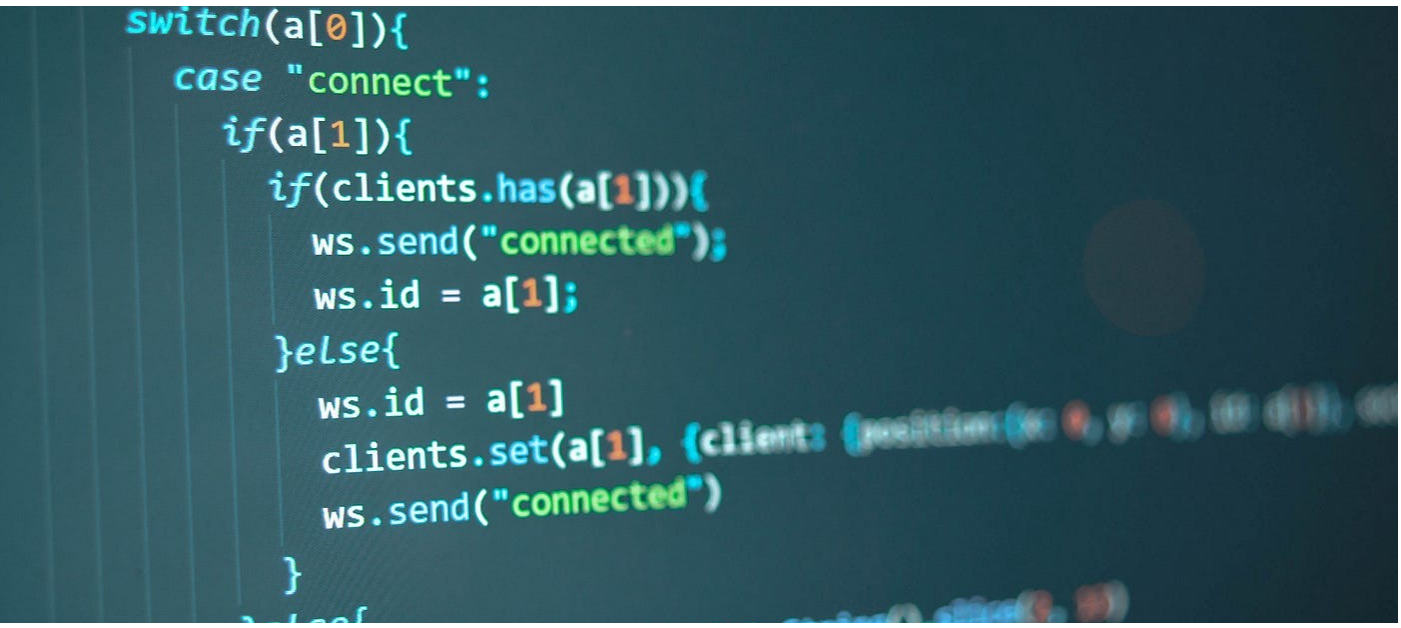


2. Estructuras de repetición

Date	Empty
Status	Not started
Type	Empty
Units	3. Estructuras de control

Estructuras de repetición

- for
- while // do while

Estructuras de repetición

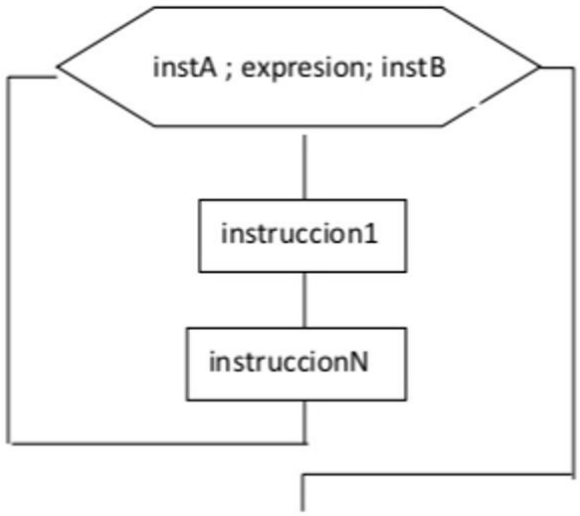
Son aquellas que mediante la evaluación de una sentencia, ejecuta n veces una serie de líneas de código. Los principales son: for (foreach), while, do while

for

Una sentencia for es utilizada cuando se necesita recorrer un conjunto de datos determinado, como por ejemplo un array de datos. Para ello

lo que se necesita indicar el inicio, el final y el incremente utilizado cada vez que se realice un recorrido.

El diagrama de flujo es el siguiente:



La sintaxis de esta estructura es:

```
for (inicio; final: incremento){ ejecuciones }
```

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

foreach

Existe una construcción especial del bucle for que recorre de forma completa una colección de datos sin necesidad de indicarle cual es el principio y le final de la misma. Este tipo de estructura recibe el nombre de foreach donde se indica el tipo de elementos que tiene la colección y la colección a recorrer. Automáticamente iterará sobre la colección hasta recorrerla de forma completa

```
for(tipo constante: coleccion){ }
```

Un ejemplo sería

```
int[] numeros = {1,2,3,4,5,6,7,8,9,10}; for (int index: numer
```

```
os) { System.out.println( Ejecucion numero:  + index); }
```

Ejemplos a realizar:

1. Pedir un número por teclado. En el caso que el número introducir sea mayor que 10 o menor que 0 el programa parará la ejecución con el mensaje "dato erróneo". En caso contrario mostrará la tabla de multiplicar del número introducido con la siguiente estructura: "5 por 1 = 5" "5 por 2 = 10" ...
2. Realizar un programa que lea por teclado 10 números y los sume. Al final del proceso mostrará la suma total
3. Generar 100 números aleatorios entre 1 y 1000 (Math.random()*1001), ambos incluisve, mostrar cada número generado y contabilizar cuántos de ellos son pares.

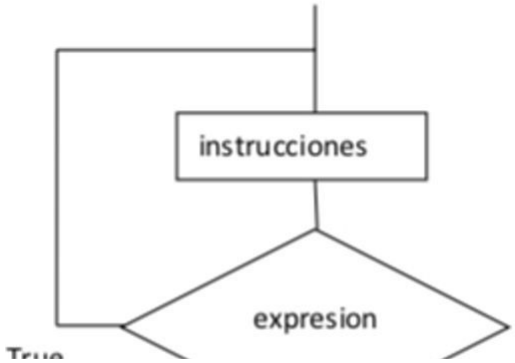
while // do while

Una sentencia do o do while ejecuta un conjunto de líneas de código siempre y cuando la condición a evaluar se cumpla y devuelva un valor verdadero. Este tipo de estructura se utiliza para evaluar los valores de la colección de datos o una señal que puede actuar como semáforo. Hay que tener cuidado ya que para no convertir este tipo de estructuras en un bucle infinito se debe modificar el valor de la variable que está incluida en la condición para que así en algún momento se pueda obtener un resultado diferente (o bien utilizar la sentencia break / continue). Dependiendo del orden de ejecución – evaluación se utilizará una sentencia y otra:

do while

Esta estructura ejecuta una serie de líneas de código para después evaluar si una condición es verdadera. En este tipo de ejecuciones las líneas de código se ejecutan al menos una vez.

El diagrama de flujo es el siguiente:



La sintaxis de esta estructura es:

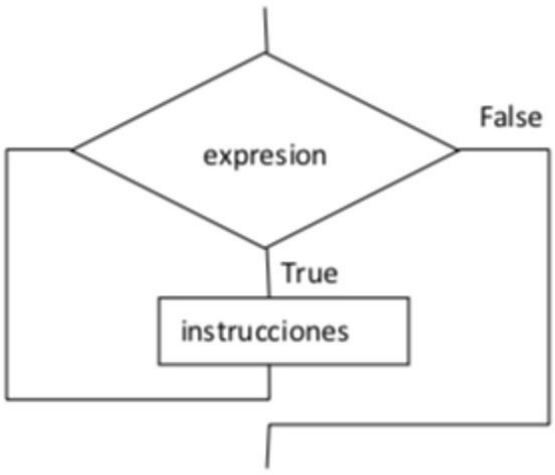
```
do{ //líneas de código que se quieren ejecutar //importante e
s modificar el valor de la condición //que está dentro del wh
ile }while(condición) //siempre que la condición se cumpla co
mo verdadera las lineas de código ubicadas dentro del while s
e volverán a ejecuar
```

```
do{ System.out.println("Ejecución número: "+numero); numero
--; }while(numero>0);
```

while

Una estructura while es exactamente lo mismo que la anterior con la diferencia de que antes de ejecutar las líneas de código se evalúa la condición de entrada, donde en el caso de ser verdadera ejecuta las líneas de código.

El diagrama de flujo es el siguiente:



La sintaxis a utilizar es:

```
while(condición){ //líneas de código a ejecutar //es importan
te modificar el valor de la variable que se evalúa en la cond
ición }
```

```
int numero = 10; while (numero>0){ System.out.println("Ejecuc
ión número: "+numero); numero--; }
```

Ejemplos a realizar:

1. Realiza un programa que lee enteros pares hasta introducir un impar. El programa cuenta el número de positivos de los números leído y lo mostrará cuando se introduzca un impar
2. Generar números aleatorios entre 0 y 1000 hasta obtener un cero (momento en el cual el programa terminará), y contabilizar cuántos de ellos son pares. Al final del programa se mostrará cuantos pares se han obtenido
3. Realizar un menú con 4 posibilidades, donde cada vez que se pulse una opción aparezca el mensaje "Pulsada la opción 1". Tras mostrar el mensaje se volverá a mostrar el menú con su correspondiente ejecución. La 4 posibilidad parará la ejecución con el mensaje "saliendo ...". En el caso de introducir un número que no esté entre 1 y 4 aparecerá el mensaje "número no contemplado"