UD2. Boletín 2. Hilos

CREACIÓN DE HILOS DERIVANDO DE LA CLASE THREAD

1. Crea un hilo que en su método run() muestre un “NO” hasta un máximo de 30 veces. En el método principal (main) tras ejecutar el start() de tu hilo, mostrará “YES” hasta un máximo de 30 veces.
   Como salida obtendrás una serie alternativa de NOes YESes ya que una vez iniciada la ejecución del thread, el tiempo de la CPU se reparte entre todos los procesos y threads del sistema, con lo cual se intercalan instrucciones del método main() con instrucciones del método run().

   Posible salida:

YES YES YES YES YES YES YES YES YES YES YES YES YES
NO NO NO NO NO NO NO NO NO NO NO YES YES YES YES YES YES
YES YES YES YES YES YES YES YES YES YES YES NO NO NO NO NO
NO NO NO NO NO NO NO NO NO NO NO NO NO

(ejecútalo varias veces si no te salen intercalados).

Hilo2Ejercicio01.java

```java
package boletin2;

public class Hilo2Ejercicio01 extends Thread {

    public void run() {
        for (int i = 0; i < 30; i++) {
            System.out.print("YES");
        }
    }

}
```

MainHiloEjercicio01.java

```java
package boletin2;

public class MainHiloEjercicio01 extends Thread {

    public static void main(String[] args) {

        Hilo2Ejercicio01 hilo = new Hilo2Ejercicio01();

        hilo.start();
        System.out.println();
        for(int i = 0; i < 30; i++) {
            System.out.print("NO");
        }

    }
}
```

Problems Javadoc Declaration Console

<terminated> Ejercicio1_B [Java Application] C:\Program Files\Eclipse Adoptium\jdk-17.0.4.101-hotspot\bin\javaw.exe (8 nov 2022 8:40:35 – 8:40:35) [pid: 9496]

NONOYESYESYESYESYESYESYESYESYESYESYESYESYESYESYESYESYESYESNONONONONONONONOYESYESYESYESYESYESYESYESYESYESYESYESYESYESYESNONONONONONONONONONONONONONONONONONO

2. Vamos a hacer un programa similar al anterior pero instanciando 2 threads. En el HiloA mostrará el mensaje NO, hasta un maximo de 30 veces En el HiloB mostrará el mensaje YES, hasta un máximo de 30 veces En el programa principal mostrará un mensaje “Ejecución en HiloA” y se ejecutará el HiloA; luego mostrará un mensaje “Ejecución en HiloB” y se ejecutará el HiloB; y luego un mensaje “Ejecución en main”. Por ejemplo.
A continuación te muestro varias salidas de diferentes ejecuciones:

Salida 1

```
 Ejecución en HiloA

 Ejecución en HiloB
NO NO NO NO NO NO NO NO NO
 Ejecución en main
NO NO NO NO NO NO NO NO NO NO NO YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES Y
```

Salida 2

```
 Ejecución en HiloA

 Ejecución en HiloB

 Ejecución en main
NO NO NO NO NO NO NO NO NO NO NO YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES
```

Salida 3

```
 Ejecución en HiloA

 Ejecución en HiloB

 Ejecución en main
YES NO NO NO NO NO NO NO NO NO NO NO NO NO NO NO NO NO NO NO NO YES YES YE
```

HiloMainEjercicio02.java

```
1  package ejercicio02;
2
3  public class HiloMainEjercicio02 extends Thread{
4
5      public static void main(String [] args) {
6
7
8      //Instancia de hilos
9      HiloA hiloA = new HiloA();
10     HiloB hiloB = new HiloB();
11
12     System.out.println("\n Ejecucion HiloA");
13     hiloA.start();
14
15     System.out.println("\n Ejecucion HiloB");
16     hiloB.start();
17
18     System.out.println("\n Ejecucion en main");
19
20
21     }
22 }
23
```

HiloA.java

```
1  package ejercicio02;
2
3  public class HiloA extends Thread {
4
5      public void run() {
6          for(int i = 0; i < 30; i++) {
7              System.out.print("YES");
8          }
9      }
10 }
```

HiloB.java

```
1  package ejercicio02;
2
3  public class HiloB extends Thread{
4
5      public void run() {
6          for(int i = 0; i < 30; i++) {
7              System.out.print("NO");
8          }
9      }
10 }
11
```

```
Ejecucion HiloA

Ejecucion HiloB

Ejecucion en main
YESYESYESYESYESYESYESNONONONONONONONONONONONONONONONONONONONONONONONONONONONONONOYESYESYESYESYESYESYESYESYESYESYESYESYESYESYESYESYESYESYESYESYESYESYESYES
```

3. Vamos a ver como una variable “contador” no funciona correctamente debido a que se accede concurrentemente a una misma zona de memoria sin que se produzca exclusión mutua (esto intentaremos solucionarlo más adelante cuando se trate el sincronismo de los hilos).
   Creas una subclase Thread la cual tendrá un atributo de clase[1] que es una variable entera (contador) que se inicializa a 0 y un atributo de objeto[2] que es una cadena; como métodos:
   - Se redefine su constructor pasándole un string como argumento, se llama al constructor de la superclase, se le asigna al atributo de objeto como valor el del string pasado como argumento.
   - El método run() mostrará hasta un máximo de 10 veces el valor de contador incrementado y luego el valor del atributo de objeto (el string).

   En el programa principal (main) -> creas 2 objetos de tu subclase Thread (2 hilos) cada uno con un valor, por ejemplo, el hilo1 con SI y el hilo2 con NO. Ejecutas ambos hilos y comprueba que el contador no funcionó todo lo bien que debía ya que no se ha incrementado secuencialmente (como sí tenía que hacerlo).

[1] *Los atributos de clase son los declarados como static*
[2] *Este atributo de objeto lo definirás como private*

Por ejemplo:

Salida 1

```
Comienzo de la ejecución concurrente
1:NO 2:NO 1:SI 4:SI 5:SI 3:NO 6:SI 8:SI 9:SI 10:SI 11:SI 12:SI 13:SI 7:NO 14:NO 15:NO 16:NO 17:NO 18:NO 19:NO
```

Salida 2

```
Comienzo de la ejecución concurrente
1:SI 2:SI 3:SI 4:SI 5:SI 6:SI 7:NO 8:NO 9:NO 10:NO 11:NO 13:NO 12:SI 14:NO 16:NO 17:NO 18:NO 15:SI 19:SI 20:SI
```

Salida 3

```
Comienzo de la ejecución concurrente
1:SI 3:SI 2:NO 5:NO 6:NO 4:SI 8:SI 9:SI 10:SI 7:NO 11:SI 12:NO 13:SI 14:NO 15:SI 16:NO 17:SI 18:NO 19:NO 20:NO
```

HiloMainEjercico03.java

```java
package ejercicio03;

public class HiloMainEjercico03 {

    public static void main(String [] args) {

        HiloA hilo1 = new HiloA("YES");
        HiloA hilo2 = new HiloA("NO");

        System.out.println("Comienzo de la ejecucion concurrente:");
        hilo1.start();
        hilo2.start();
    }
}
```

HiloA.java

```java
package ejercicio03;

public class HiloA extends Thread {

    static Integer contador = 0;
    private String cadena;

    public HiloA(String cadena) {
        super();
        this.cadena = cadena;
    }

    public void run() {
        for(int i = 0; i < 10; i++) {
            System.out.print(++contador + ":" + this.cadena + " ");
        }
    }
}
```

```
<terminated> HiloMainEjercico03 [Java Application] C:\Program Files\Eclipse Adoptium\jdk-17.0.4.101-hotspot\bin\javaw.exe  (8 nov 2022 9:21:55 – 9:21:55) [pid: 97
Comienzo de la ejecucion concurrente:
1:YES 3:YES 4:YES 2:NO 5:YES 6:NO 7:YES 8:NO 9:YES 10:NO 11:YES 13:YES 12:NO 14:YES 15:NO 16:YES 17:NO 18:NO 19:NO 20:NO
```

4. Diseña un programa que escriba dos sucesiones ascendentes de números de forma simultánea. Es decir, un programa con dos hilos en ejecución concurrente.

   El hilo tendrá:
   - 2 atributos de tipo entero que marcarán el comienzo y fin de la secuencia que queremos que se muestre (por ejemplo, desde el 18 hasta el 45), una cadena para el nombre del hilo.
   - El constructor, al que se le pasarán el comienzo y fin de la secuencia y el nombre del hilo. Es decir, cuando se instancia un objeto hilo se le pasarán el intervalo de valores a mostrar y como se llamará.
   - El método run(), mostrará el nombre del hilo indicando que comienza, luego la secuencia de valores y cuando haya finalizado indicará que terminó.

   En el main:
   - creas dos objetos de tu hilo, pasándoles el inicio, fin y nombre del hilo. Por ejemplo: Un ThreadA que comenzará en 1 y terminará en 10 Un ThreadB que empezará en 20 y terminará en 30
   - Un mensaje indicando que "Vamos a iniciar los dos threads"
   - Un mensaje indicando que los hilos se han inicializado
   - Lanzas los dos hilos
   - Y un mensaje de que el programa principal ha terminado.

   Ejecuta varias veces tu aplicación para ver que obtienes diferentes salidas dependiendo de los tiempos que asigne la CPU a cada hilo.

   Ejemplos de salidas son:

```
Vamos a iniciar los dos threads
Hilos inicializados
Programa principal terminado
ThreadA empieza...
ThreadA dice: 1.
ThreadA dice: 2.
ThreadA dice: 3.
ThreadA dice: 4.
ThreadA dice: 5.
ThreadA dice: 6.
ThreadA dice: 7.
ThreadA dice: 8.
ThreadA dice: 9.
ThreadA dice: 10.
ThreadA acaba.
ThreadB empieza...
ThreadB dice: 20.
ThreadB dice: 21.
ThreadB dice: 22.
ThreadB dice: 23.
ThreadB dice: 24.
ThreadB dice: 25.
ThreadB dice: 26.
ThreadB dice: 27.
ThreadB dice: 28.
ThreadB dice: 29.
ThreadB dice: 30.
ThreadB acaba.
```

```
Vamos a iniciar los dos threads
Hilos inicializados
Programa principal terminado
ThreadB empieza...
ThreadB dice: 20.
ThreadB dice: 21.
ThreadB dice: 22.
ThreadB dice: 23.
ThreadB dice: 24.
ThreadB dice: 25.
ThreadB dice: 26.
ThreadB dice: 27.
ThreadB dice: 28.
ThreadB dice: 29.
ThreadB dice: 30.
ThreadB acaba.
ThreadA empieza...
ThreadA dice: 1.
ThreadA dice: 2.
ThreadA dice: 3.
ThreadA dice: 4.
ThreadA dice: 5.
ThreadA dice: 6.
ThreadA dice: 7.
ThreadA dice: 8.
ThreadA dice: 9.
ThreadA dice: 10.
ThreadA acaba.
```

```
Vamos a iniciar los dos threads
Hilos inicializados
Programa principal terminado
ThreadA empieza...
ThreadA dice: 1.
ThreadA dice: 2.
ThreadA dice: 3.
ThreadA dice: 4.
ThreadA dice: 5.
ThreadA dice: 6.
ThreadB empieza...
ThreadB dice: 20.
ThreadA dice: 7.
ThreadA dice: 8.
ThreadB dice: 21.
ThreadB dice: 22.
ThreadB dice: 23.
ThreadB dice: 24.
ThreadB dice: 25.
ThreadB dice: 26.
ThreadB dice: 27.
ThreadB dice: 28.
ThreadA dice: 9.
ThreadB dice: 29.
ThreadB dice: 30.
ThreadB acaba.
ThreadA dice: 10.
ThreadA acaba.
```

HiloMainEjercicio04.java | HiloEjercio04.java

```java
package ejercicio04;

public class HiloMainEjercicio04 extends Thread{

    public static void main(String [] args) {

        HiloEjercio04 hilo1 = new HiloEjercio04(10, 20, "ThreadA");
        HiloEjercio04 hilo2 = new HiloEjercio04(29, 34, "ThreadB");

        System.out.println("Vamos a iniciar los dos Threads");
        System.out.println("Hilos iniciados");
        hilo1.start();
        hilo2.start();
        System.out.println("Programa principal terminado");
    }
}
```

HiloMainEjercicio04.java | HiloEjercio04.java

```java
package ejercicio04;

public class HiloEjercio04 extends Thread{

    private Integer principio, fin;
    private String nombre;

    public HiloEjercio04(Integer principio, Integer fin, String nombre) {
        super();
        this.principio = principio;
        this.fin = fin;
        this.nombre = nombre;
    }

    public void run() {
        System.out.println(nombre + " empieza...");
        while (principio < fin) {
            System.out.println(nombre + " dice: " + principio + ".");
            principio++;
        }
        System.out.println(nombre + " acaba.");
    }
}
```

<terminated> HiloMainEjercicio04 [Java App

```
Vamos a iniciar los dos Threads
Hilos iniciados
Programa principal terminado
ThreadA empieza...
ThreadB empieza...
ThreadB dice: 29.
ThreadA dice: 10.
ThreadB dice: 30.
ThreadB dice: 31.
ThreadB dice: 32.
ThreadB dice: 33.
ThreadB acaba.
ThreadA dice: 11.
ThreadA dice: 12.
ThreadA dice: 13.
ThreadA dice: 14.
ThreadA dice: 15.
ThreadA dice: 16.
ThreadA dice: 17.
ThreadA dice: 18.
ThreadA dice: 19.
ThreadA acaba.
```