Hilos:

1-en java, crea un programa donde un hilo haga una cuenta regresiva dado un numero, el tiempo que debe esperar el hilo debe ser aleatorio. inicia 10 hilos y revisa que hilo termino primero, asegurate de colocarle un nombre a cada uno de ellos.

```
class Decrementor extends Thread{

    private int counter=0;

    public Decrementor(String name, int numero) {
        super(name);
        this.counter=numero;
    }
    @Override
    public void run() {
        System.out.println(getName() + " comenzó.");
        for(int i = counter ; i>=0;i--) {

        System.out.println(getName() + " esta contanto hacia atras: "+i );

        try {
            Thread.sleep((long) (Math.random() * 1000));
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        }
        System.out.println(getName() + " terminó.");

    }
}

public class Main {

    public static void main(String[] args) {
        ejercicio1(10);

    }

    public static void ejercicio1(int numeroDeHilosACrear) {
        String name = "";
        Random rand = new Random();
        for (int i=0;i<numeroDeHilosACrear;i++) {
            name = "hilo " + (i+1);
            new Decrementor(name, rand.nextInt(10)+1).start();

        }
    }
```

```
hilo 4 terminó.
hilo 3 esta contanto hacia atras: 5
hilo 1 esta contanto hacia atras: 7
hilo 6 esta contanto hacia atras: 2
hilo 5 esta contanto hacia atras: 0
hilo 10 esta contanto hacia atras: 5
hilo 6 esta contanto hacia atras: 1
hilo 9 esta contanto hacia atras: 7
hilo 1 esta contanto hacia atras: 6
hilo 3 esta contanto hacia atras: 4
hilo 8 esta contanto hacia atras: 4
hilo 2 esta contanto hacia atras: 2
hilo 10 esta contanto hacia atras: 4
hilo 6 esta contanto hacia atras: 0
hilo 5 terminó.
hilo 7 terminó.
hilo 1 esta contanto hacia atras: 5
hilo 3 esta contanto hacia atras: 3
hilo 8 esta contanto hacia atras: 3
hilo 9 esta contanto hacia atras: 6
hilo 9 esta contanto hacia atras: 5
hilo 10 esta contanto hacia atras: 3
hilo 6 terminó.
hilo 2 esta contanto hacia atras: 1
hilo 3 esta contanto hacia atras: 2
hilo 8 esta contanto hacia atras: 2
hilo 10 esta contanto hacia atras: 2
hilo 8 esta contanto hacia atras: 1
hilo 1 esta contanto hacia atras: 4
hilo 10 esta contanto hacia atras: 1
hilo 3 esta contanto hacia atras: 1
hilo 9 esta contanto hacia atras: 4
hilo 3 esta contanto hacia atras: 0
hilo 1 esta contanto hacia atras: 3
hilo 2 esta contanto hacia atras: 0
hilo 10 esta contanto hacia atras: 0
hilo 1 esta contanto hacia atras: 2
hilo 3 terminó.
hilo 8 esta contanto hacia atras: 0
hilo 9 esta contanto hacia atras: 3
hilo 2 terminó.
hilo 9 esta contanto hacia atras: 2
hilo 9 esta contanto hacia atras: 1
hilo 10 terminó.
hilo 8 terminó.
hilo 9 esta contanto hacia atras: 0
hilo 1 esta contanto hacia atras: 1
hilo 1 esta contanto hacia atras: 0
hilo 9 terminó.
hilo 1 terminó.
```

2-modificar ejercicio anterior para que el tiempo de espera sea el mismo pero la prioridad sea distinta, lanzar 30 hilos.

```
class DecrementorConPrioridad extends Thread{

     private int counter=0;

     public DecrementorConPrioridad(String name, int numero,int priority) {
         super(name);
         this.counter=numero;
         this.setPriority(priority);
     }
     @Override
     public void run() {
         System.out.println(getName() + " con prioridad "+ getPriority() +" comenzó.");
         for(int i = counter ; i>=0;i--) {

         System.out.println(getName() + " esta contanto hacia atras: "+i );

        try {
            Thread.sleep(1000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
         }
         System.out.println(getName() + " terminó.");

     }
}

public class ejercicio2 {

    public static void main(String[] args) {
        ejercicio1(30);

    }

    public static void ejercicio1(int nHilos) {
        String name = "";
        Random rand = new Random();
        for (int i=0;i<nHilos;i++) {
            name = "hilo " + (i+1);
            new DecrementorConPrioridad(name, rand.nextInt(10)+1,rand.nextInt(10)+1).start();

        }
    }
}
```

```
<terminated> ejercicio2 [Java Application] C:\Program Files\Java\jdk1.8.0_202\bin\javaw.exe (22 oct. 20
hilo 4 esta contanto hacia atras: 2
hilo 5 esta contanto hacia atras: 3
hilo 29 esta contanto hacia atras: 2
hilo 27 esta contanto hacia atras: 0
hilo 21 esta contanto hacia atras: 4
hilo 26 esta contanto hacia atras: 1
hilo 24 terminó.
hilo 30 terminó.
hilo 6 terminó.
hilo 3 esta contanto hacia atras: 0
hilo 9 esta contanto hacia atras: 1
hilo 11 esta contanto hacia atras: 3
hilo 1 esta contanto hacia atras: 1
hilo 8 terminó.
hilo 23 esta contanto hacia atras: 1
hilo 19 esta contanto hacia atras: 2
hilo 20 esta contanto hacia atras: 1
hilo 4 esta contanto hacia atras: 1
hilo 27 terminó.
hilo 5 esta contanto hacia atras: 2
hilo 29 esta contanto hacia atras: 1
hilo 21 esta contanto hacia atras: 3
hilo 26 esta contanto hacia atras: 0
hilo 1 esta contanto hacia atras: 0
hilo 11 esta contanto hacia atras: 2
hilo 9 esta contanto hacia atras: 0
hilo 23 esta contanto hacia atras: 0
hilo 20 esta contanto hacia atras: 0
hilo 19 esta contanto hacia atras: 1
hilo 3 terminó.
hilo 21 esta contanto hacia atras: 2
hilo 29 esta contanto hacia atras: 0
hilo 5 esta contanto hacia atras: 1
hilo 26 terminó.
hilo 4 esta contanto hacia atras: 0
hilo 23 terminó.
hilo 9 terminó.
hilo 19 esta contanto hacia atras: 0
hilo 1 terminó.
hilo 11 esta contanto hacia atras: 1
hilo 20 terminó.
hilo 29 terminó.
hilo 5 esta contanto hacia atras: 0
hilo 21 esta contanto hacia atras: 1
hilo 4 terminó.
hilo 19 terminó.
hilo 11 esta contanto hacia atras: 0
hilo 21 esta contanto hacia atras: 0
hilo 5 terminó.
hilo 11 terminó.
hilo 21 terminó.
```

3-realizar transacciones de una cuenta bancaria usando hilos no sincronizados

```java
package JavaSeccion14Threads;

public class CuentaBancaria {

    private String name;
    private double capital=0;
    public CuentaBancaria(String name, double capital) {
        this.name = name;
        this.capital = capital;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public double getCapital() {
        return capital;
    }
    public void setCapital(double capital) {
        this.capital = capital;
    }

    public boolean validarMonto(double monto) {
        if(monto>0 && monto<=this.capital )
            return true;

            return false;

    }
    public double retirar(double montoRetirar) {
        if(validarMonto(montoRetirar)) {
            this.capital=this.capital-montoRetirar;
            return montoRetirar;
        }
        return 0;
    }

}
```

```java
package JavaSeccion14Threads;

public class Ejercicio3 {

    public static void main(String[] args) {
        CuentaBancaria cuenta = new CuentaBancaria("pepe", 6000);
        for (int i = 0; i < 5; i++) {
            new RetiroDeCliente("hilo " + (i + 1), cuenta).start();
        }
    }

}

class RetiroDeCliente extends Thread{
    CuentaBancaria cuenta=null;
    public RetiroDeCliente(String name,CuentaBancaria cuenta) {
        super(name);
        this.cuenta = cuenta;
    }

    @Override
    public void run(){
        double monto = cuenta.retirar(200);
        if (monto > 0) {
            System.out.println("Retiro exitoso de $" + monto);
            System.out.println("quedan: $" + cuenta.getCapital() + " en la cuenta");
        }
    }

}
```

```
Retiro exitoso de $200.0
Retiro exitoso de $200.0
Retiro exitoso de $200.0
Retiro exitoso de $200.0
quedan: $5000.0 en la cuenta
Retiro exitoso de $200.0
quedan: $5000.0 en la cuenta
quedan: $5000.0 en la cuenta
quedan: $5000.0 en la cuenta
quedan: $5000.0 en la cuenta
```

Ahora syncronizalos

```java
    public synchronized double retirar(double montoRetirar) {
        if(validarMonto(montoRetirar)) {
            this.capital=this.capital-montoRetirar;
            return montoRetirar;
        }
        return 0;
    }
```

```java
public void run(){
    synchronized (cuenta) {
        double monto = cuenta.retirar(200);
        if (monto > 0) {
            System.out.println("Retiro exitoso de $" + monto);
            System.out.println("Quedan: $" + cuenta.getCapital() + " en la cuenta");
        }
    }
}
```

```
<terminated> Ejercicios [Java Application] C:\Program Files
Retiro exitoso de $200.0
Quedan: $5800.0 en la cuenta
Retiro exitoso de $200.0
Quedan: $5600.0 en la cuenta
Retiro exitoso de $200.0
Quedan: $5400.0 en la cuenta
Retiro exitoso de $200.0
Quedan: $5200.0 en la cuenta
Retiro exitoso de $200.0
Quedan: $5000.0 en la cuenta
```