

EJERCICIO 1

a) SIN SINCRONIZACION

```
public class MainContador {

    public static void main(String[] args) {

        Contador cuenta = new Contador();

        ContadorHilo hilo1 = new ContadorHilo(cuenta, "Hilo1");
        ContadorHilo hilo2 = new ContadorHilo(cuenta, "Hilo2");
        ContadorHilo hilo3 = new ContadorHilo(cuenta, "Hilo3");
        ContadorHilo hilo4 = new ContadorHilo(cuenta, "Hilo4");

        hilo1.start();
        hilo2.start();
        hilo3.start();
        hilo4.start();

        try {
            Thread.sleep(1000);
        } catch (InterruptedException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }

        System.out.println("Fin del Main");
    }
}

public class ContadorHilo extends Thread{

    private Contador contador;

    public ContadorHilo(Contador contador, String nombre) {
        super();
        this.contador = contador;
        this.setName(nombre);
    }

    public void run() {
        for (int i = 0; i < 3; i++) {
            try {
                sleep((long) (Math.abs(new Random().nextInt()) % 1000));
            } catch (InterruptedException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
            contador.cuenta();
            System.out.println(getName() + ", valor del recurso compartido: " +
contador.getContador());
        }
        System.out.println("FIN..." + getName());
    }
}
```

```

public class Contador {
    private int contador = 0;

    public Contador() {
    }

    public void cuenta() {
        contador++;
    }

    public int getContador() {
        return contador;
    }
}

```

```

Fin del programa
Hilo1, valor del recurso compartido: 1
Hilo1, valor del recurso compartido: 2
Hilo2, valor del recurso compartido: 3
Hilo4, valor del recurso compartido: 4
Hilo3, valor del recurso compartido: 5
Hilo1, valor del recurso compartido: 6
FIN...Hilo1
Hilo2, valor del recurso compartido: 7
Hilo4, valor del recurso compartido: 8
Hilo3, valor del recurso compartido: 9
Hilo3, valor del recurso compartido: 10
FIN...Hilo3
Hilo4, valor del recurso compartido: 11
FIN...Hilo4
Hilo2, valor del recurso compartido: 12
FIN...Hilo2

```

b) CON SINCRONIZACION (Solo cambia la clase ContadorHilo)

```

public class ContadorHilo extends Thread {

    private Contador contador;

    public ContadorHilo(Contador contador, String nombre) {
        super();
        this.contador = contador;
        this.setName(nombre);
    }

    public void run() {
        synchronized (contador) {
            for (int i = 0; i < 3; i++) {
                try {
                    sleep((long) (Math.abs(new Random().nextInt()) %
1000));
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
                contador.cuenta();
                System.out.println(getName() +
", valor del recurso compartido: " +
contador.getContador());
            }
            System.out.println("FIN..." + getName());
        }
    }
}

```

```

Hilo1, valor del recurso compartido: 1
Hilo1, valor del recurso compartido: 2
Hilo1, valor del recurso compartido: 3
FIN...Hilo1
Hilo4, valor del recurso compartido: 4
Hilo4, valor del recurso compartido: 5
Hilo4, valor del recurso compartido: 6
FIN...Hilo4
Hilo3, valor del recurso compartido: 7
Hilo3, valor del recurso compartido: 8
Hilo3, valor del recurso compartido: 9
FIN...Hilo3
Hilo2, valor del recurso compartido: 10
Hilo2, valor del recurso compartido: 11
Hilo2, valor del recurso compartido: 12
FIN...Hilo2
Fin del Main

```

c) CON ORDEN ESPECIFICO

```
Hilo2, valor del recurso compartido: 1
Hilo2, valor del recurso compartido: 2
Hilo2, valor del recurso compartido: 3
FIN...Hilo2
Hilo3, valor del recurso compartido: 4
Hilo3, valor del recurso compartido: 5
Hilo3, valor del recurso compartido: 6
FIN...Hilo3
Hilo1, valor del recurso compartido: 7
Hilo1, valor del recurso compartido: 8
Hilo1, valor del recurso compartido: 9
FIN...Hilo1
Hilo4, valor del recurso compartido: 10
Hilo4, valor del recurso compartido: 11
Hilo4, valor del recurso compartido: 12
FIN...Hilo4
Fin del Main
```

EJERCICIO 2

Sin la sincronizacion:

```
public class Cajero {

    private int localidades;
    private String nombre;

    public Cajero(String nombre) {
        super();
        this.nombre = nombre;
    }

    public void sumarLocalidades() {
        localidades++;
    }

    public int getLocalidades() {
        return localidades;
    }

    public String getNombre() {
        return nombre;
    }
}
```

```

public class Terminal extends Thread{

    private Cajero cajero;
    private String nombre;
    private int localidad;

    public Terminal(Cajero cajero, String nombre) {
        super();
        this.cajero = cajero;
        this.nombre = nombre;
    }

    public void run() {
        for(int i = 0; i < 20000; i++) {
            cajero.sumarLocalidades();//suma en el total
            localidad++;
        }
        System.out.println(nombre + " vendio " + localidad);
    }
}

```

```

public class MainCajero {

    public static void main(String[] args) {
        Cajero cajero = new Cajero("Santander");

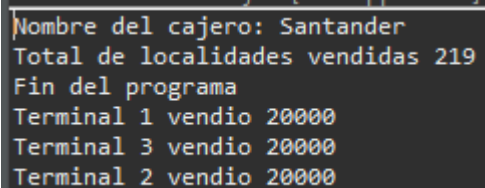
        Terminal terminal1 = new Terminal(cajero, "Terminal 1");
        Terminal terminal2 = new Terminal(cajero, "Terminal 2");
        Terminal terminal3 = new Terminal(cajero, "Terminal 3");

        System.out.println("Nombre del cajero: " + cajero.getNombre());

        terminal1.start();
        terminal2.start();
        terminal3.start();

        System.out.println("Total de localidades vendidas " +
cajero.getLocalidades());
        System.out.println("Fin del programa");
    }
}

```



```

Nombre del cajero: Santander
Total de localidades vendidas 219
Fin del programa
Terminal 1 vendio 20000
Terminal 3 vendio 20000
Terminal 2 vendio 20000

```

CON SINCRONIZACION (Modificamos la clase MainCajero y el Terminal)

```
public class MainCajero {

    public static void main(String[] args) {
        Cajero cajero = new Cajero("Santander");

        Terminal terminal1 = new Terminal(cajero, "Terminal 1");
        Terminal terminal2 = new Terminal(cajero, "Terminal 2");
        Terminal terminal3 = new Terminal(cajero, "Terminal 3");

        System.out.println("Nombre del cajero: " + cajero.getNombre());

        terminal1.start();
        terminal2.start();
        terminal3.start();

        try {
            Thread.sleep(1000);
        } catch (InterruptedException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }

        System.out.println("Total de localidades vendidas " +
cajero.getLocalidades());
        System.out.println("Fin del programa");

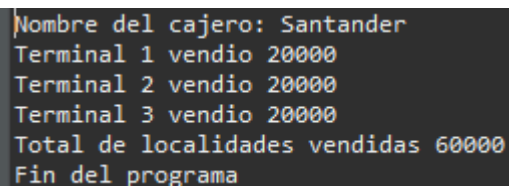
    }
}

public class Terminal extends Thread {

    private Cajero cajero;
    private String nombre;
    private int localidad;

    public Terminal(Cajero cajero, String nombre) {
        super();
        this.cajero = cajero;
        this.nombre = nombre;
    }

    public void run() {
        synchronized (cajero) {
            for (int i = 0; i < 20000; i++) {
                cajero.sumarLocalidades();// suma en el total
                localidad++;
            }
            System.out.println(nombre + " vendio " + localidad);
        }
    }
}
```

A screenshot of a terminal window showing the output of the Java program. The text is as follows:
Nombre del cajero: Santander
Terminal 1 vendio 20000
Terminal 2 vendio 20000
Terminal 3 vendio 20000
Total de localidades vendidas 60000
Fin del programa