1. **Ejercicios UD 2. PROGRAMACIÓN DE HILOS**
2. En el siguiente ejemplo dos hilos comparten un objeto de la clase Contador. Esta clase define un atributo contador v tres métodos, (uno de ellos incrementa una unidad su valor, el otro lo decrementa y el tercero devuelve su valor; el constructor asigna un valor inicial al contador:

class Contador {

private int c = 0; //atributo contador

Contador(int c) {this.c = c;}

public void incrementa () {

c = c + 1;

}

public void decrementa () {

c = c - 1;

}

public int getValorO {

return c;

}

}// CONTADOR

Para probar el objeto compartido se definen dos clases que extienden Thread. En la clase Hilo A se usa el método del objeto contador que incrementa en uno su valor. En la clase HiloB se usa el método que decrementa su valor. Se añade un sleep() intencionadamente para probar que un hilo se duerma y mientras el otro haga otra operación con el contador, así la CPU no realiza de una sola vez todo un hilo y después otro, y podemos observar mejor el efecto:

class HiloA extends Thread {

private Contador contador;

public HiloA(String n, Contador c) {

setName(n);

contador = c;

}

public void run() {

for (int j = 0; j < 300; j++) {

contador.incrementa(); //incrementa el contador

try {

sleep (100);

} catch (InterruptedException e) {}

}

System.out.println(getName() + " contador vale " +

contador.getValor());

}

}// FIN HILOA

class HiloB extends Thread {

private Contador contador;

public HiloB(String n, Contador c) {

setName(n);

contador = c;

}

public void run() {

for (int j = 0; j < 300; j++) {

contador.decrementa(); //decrementa el contador

try {

sleep(100) ;

} catch (InterruptedException e) {}

}

System.out.println(getName() + " contador vale " +

contador.getValor());

}

}// FIN HILOB

A continuación, se crea el método main(), donde primero se define un objeto de la clase Contador y se le asigna el valor inicial de 100. Después, se crean los dos hilos pasándoles dos parámetros: un nombre y el objeto Contador. Seguidamente se inicia la ejecución de los hilos:

public class Compartirlnf1 {

public static void main(String[] args) {

Contador cont = new Contador(100);

HiloA a = new HiloA("HiloA", cont);

HiloB b = new HiloB("HiloB”, cont);

a .start();

b .start();

}

}

Nos puede dar la impresión de que, al ejecutar los hilos, el valor del contador en el hilo A debería ser 400, ya que empieza en 100 y le suma 300; y en B, 100 ya que se resta 300; pero no es así. Al ejecutar el programa los valores de salida pueden no ser los esperados y variará de una ejecución a otra.

Plantea una solución de forma que se evite que esto suceda, es decir, que las operaciones de incremento y decremento del objeto contador se hagan de forma atómica.

1. Escribe una clase llamada Check que cree dos threads que accedan simultáneamente a un buffer de 10.000 enteros. Uno de ellos lee en el buffer y el otro escribe en el mismo. El thread escritor debe escribir el mismo valor en todos los elementos del buffer incrementando en uno el valor en cada pasada. El thread lector debe ir comprobando que todos los números del buffer son iguales, mostrando un mensaje de error en caso contrario o un mensaje de “correcto” si la condición se cumple. El código a realizar utilizará un monitor para acceder al buffer si se indica un parámetro al ejecutar el programa. En caso contrario, los threads accederán al buffer sin hacer uso del monitor.
2. Crea un proyecto java llamado Banco.

Se define la clase Cuenta, con atributos saldo y numero\_reintegros. Tiene tres métodos:

* Devolver el valor del saldo.
* Restar al saldo una cantidad.
* Comprobar que el saldo sea >= que la cantidad que se quiere retirar.

Se define la clase Cliente, con atributos cuenta, nombre, cantidad y total, que retire cantidades de dinero periódicas cada 50 ms. Dicha cantidad se indicará al crear al cliente.

El cliente seguirá retirando dinero mientras sea posible (mientras tenga saldo disponible), guardando el total de dinero retirado.

Desarrollar la clase Banco que en su método main() cree una cuenta con un saldo de 1000 € y dos clientes que actúen sobre la misma cuenta. Uno de ellos obtiene reintegros de 55 € y otro de 25 €. Al final se deberá mostrar el dinero que ha sacado cada cliente y el saldo restante en la cuenta. De esta forma se comprueba si las actualizaciones de la cuenta se han realizado de manera adecuada.

Realiza pruebas con distintos reintegros de los clientes, y distintos saldos iniciales de la cuenta.