**EJERCICIO 1**

**Enunciado**

Realizar en Java un programa que cree 3 hilos y que su clase principal se llame Ejercicio1. Cada uno de ellos mostrará 5 mensajes por pantalla indicando su nombre y la hora actual. Entre mensaje y mensaje de un mismo hilo se esperará 1 segundo.

El resultado esperado sería equivalente al siguiente, teniendo en cuenta que el orden de aparición de los 5 mensajes de cada hilo puede variar de una ejecución a otra del programa y que la hora actual de ejecución se corresponderá con la hora en la que se ejecute dicho programa:

Hilo 1 - 18:54:03

Hilo 3 - 18:54:03

Hilo 2 - 18:54:03

Hilo 1 - 18:54:04

Hilo 3 - 18:54:04

Hilo 2 - 18:54:04

Hilo 1 - 18:54:05

Hilo 3 - 18:54:05

Hilo 2 - 18:54:05

Hilo 3 - 18:54:06

Hilo 1 - 18:54:06

Hilo 2 - 18:54:06

Hilo 3 - 18:54:07

Hilo 1 - 18:54:07

Hilo 2 - 18:54:07

**EJERCICIO 2**

**Enunciado**

Modificar el siguiente programa en Java para conseguir el siguiente resultado:

HiloA contador vale 400

HiloB contador vale 100

La clase principal de este programa se llamará Ejercicio2 y la modificación a realizar requerirá sincronizar el objeto contador, de tal forma que el HiloA incremente 300 veces el contador, bloqueando el objeto, para que posteriormente el HiloB lo decremente.

El detalle del programa a modificar es el siguiente:

**public** **class** Ejercicio2 {

**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();

}

}

**class** Contador {

**private** **int** c=0;

Contador (**int** c) {

**this**.c = c;

}

**public** **void** incrementa() {

c=c+1;

}

**public** **void** decrementa() {

c=c-1;

}

**public** **int** getValor() {

**return** c;

}

}

**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();

**try** {

Thread.*sleep*(100);

} **catch** (InterruptedException e) {}

}

System.***out***.println(getName() + " contador vale" + contador.getValor());

}

}

**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();

**try** {

Thread.*sleep*(100);

} **catch** (InterruptedException e) {}

}

System.***out***.println(getName() + " contador vale" + contador.getValor());

}

}

**Resolución de problemas**

**REQUISITOS**

1. Entregar cada ejercicio en un archivo comprimido que contenga los ficheros \*.java empleados
2. Los archivos comprimidos que contengan los ficheros \*.java empleados seguirán la siguiente nomenclatura:

*UD2\_Ejercicion\_apellido*

donde:

*n*: valdrá 1 ó 2, representando el número del ejercicio

*apellido*: será el primer apellido del alumno

**CRITERIOS DE CORRECCIÓN**

1. Cumplimiento de los requisitos (15%).
2. Hacer lo que se indica en el enunciado (55%).
3. Claridad del código Java (15%).
4. Documentación del código Java (15%).
5. Explicación detallada de cómo se han resuelto problemas que hayan podido surgir en la realización de la actividad (hasta un 15% adicional sobre la nota obtenida). Dicha explicación se incluirá al principio de los archivos de código fuente Java mediante líneas de comentarios. La primera línea de comentarios sólo incluirá lo siguiente: // Resolución de problemas.
6. Variables no usadas (-15%).
7. Librerías no usadas (-15%).