**ACTIVIDAD 15**

**Enunciado**

Realizar en Java un programa que cree 3 hilos y que su clase principal se llame Actividad15. 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

**Solución**

**Actividad15.java:**

import java.text.SimpleDateFormat;

import java.util.Date;

public class Actividad15 {

public static void main(String[] args) {

// Crear los hilos

Thread hilo1 = new MiHilo("Hilo 1");

Thread hilo2 = new MiHilo("Hilo 2");

Thread hilo3 = new MiHilo("Hilo 3");

// Iniciar los hilos

hilo1.start();

hilo2.start();

hilo3.start();

}

static class MiHilo extends Thread {

String nombre;

public MiHilo(String nombre) {

this.nombre = nombre;

}

@Override

public void run() {

for (int i = 0; i < 5; i++) {

// Obtener la hora actual

String hora = new SimpleDateFormat("HH:mm:ss").format(new Date());

// Mostrar el mensaje

System.out.println(nombre + " - " + hora);

// Esperar 1 segundo

try {

Thread.sleep(1000);

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

}

}

**Resolución de problemas**

No me han surgido problemas a la hora de realizar el ejercicio.

**ACTIVIDAD 16**

**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 Actividad16 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** Actividad16 {

**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());

}

}

**Solución**

**Actividad16.java:**

package ejer13\_14;

public class Actividad16 {

public static void main (String[] args) {

// Se crea un objeto Contador con el valor inicial 100.

Contador cont = new Contador (100);

// Se crean dos hilos, uno para incrementar el contador y otro para decrementarlo.

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

HiloB b = new HiloB("HiloB", cont);

// Se inician los dos hilos.

a.start();

b.start();

}

}

class Contador {

private int c = 0;

private boolean incrementado = false;

// Constructor que inicializa el contador con el valor pasado como parámetro.

Contador (int c) {

this.c = c;

}

// Método sincronizado para incrementar el contador.

public synchronized void incrementa() {

c = c + 1;

}

// Método sincronizado para decrementar el contador.

public synchronized void decrementa() {

c = c - 1;

}

public synchronized void setIncrementado() {

incrementado = true;

notifyAll();

}

public synchronized void waitForIncrement() {

while (!incrementado) {

try {

wait();

} catch (InterruptedException e) {}

}

}

// Método para obtener el valor actual del contador.

public int getValor() {

return c;

}

}

class HiloA extends Thread {

private Contador contador;

// Constructor que inicializa el nombre del hilo y el objeto Contador.

public HiloA (String n, Contador c) {

setName(n);

contador = c;

}

public void run () {

// Se incrementa el contador 300 veces.

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

contador.incrementa();

try {

Thread.sleep(100);

} catch (InterruptedException e) {}

}

// Se imprime el valor actual del contador.

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

contador.setIncrementado();

}

}

class HiloB extends Thread {

private Contador contador;

// Constructor que inicializa el nombre del hilo y el objeto Contador.

public HiloB (String n, Contador c) {

setName(n);

contador = c;

}

public void run () {

contador.waitForIncrement();

// Se decrementa el contador 300 veces.

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

contador.decrementa();

try {

Thread.sleep(100);

} catch (InterruptedException e) {}

}

// Se imprime el valor actual del contador.

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

}

}

**Resolución de problemas**

No me han surgido problemas a la hora de realizar el ejercicio.

**REQUISITOS**

1. Entregar las soluciones de las actividades en un único archivo comprimido cuyo nombre es: el nombre original de la actividad, seguido de un guión bajo y del primer apellido del alumno. Donde tendremos:
   1. Un archivo de extensión .odt o .docx que contendrá
      1. Una copia del código del programa
      2. Una captura de pantalla del resultado de la ejecución del programa
   2. Los archivos con el código fuente de extensión .java

**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%).