**ACTIVIDAD 4**

**Enunciado**

Realiza el siguiente ejercicio utilizando una única clase Java (Posicion) que sustituya a las clases Primero y Segundo.

El nombre de la clase principal será Actividad4 y el número de veces que se sacará por la consola las cadenas de caracteres “Primero” y “Segundo” será 15.

// Hilos2.java

**public** **class** Hilos2 {

**public** **static** **void** main(String arg[]) {

Thread p = **new** Thread(**new** Primero());

Thread s = **new** Thread(**new** Segundo());

p.start();

s.start();

System.***out***.println( "Fin programa ");

}

}

// Primero.java

**public** **class** Primero **implements** Runnable {

**public** **void** run() {

**for** (**int** i=1;i<100;i++)

System.***out***.println( "Primero " + i );

}

}

// Segundo.java

**public** **class** Segundo **implements** Runnable {

**public** **void** run() {

**for** (**int** i=1;i<100;i++)

System.***out***.println( "Segundo " + i );

}

}

**Solución**

**clase Posicion.java:**

package ejer4;

//Clase Posicion que implementa Runnable para poder ser utilizada en un hilo

class Posicion implements Runnable {

private final String nombre; // Nombre del hilo, puede ser "Primero" o "Segundo"

private final int veces; // Número de veces que se imprimirá el mensaje

// Constructor que inicializa el nombre y el número de veces

public Posicion(String nombre, int veces) {

this.nombre = nombre;

this.veces = veces;

}

// Método que se ejecuta cuando se inicia el hilo

@Override

public void run() {

for (int i = 1; i <= veces; i++) {

System.out.println(nombre + " " + i); // Imprimir el mensaje

}

}

}

**clase Actividad4.java:**

package ejer4;

// Clase principal que crea y gestiona los hilos

public class Actividad4 {

public static void main(String[] args) {

// Crear dos objetos de la clase Posicion, uno para "Primero" y otro para "Segundo"

Thread p = new Thread(new Posicion("Primero", 15));

Thread s = new Thread(new Posicion("Segundo", 15));

// Iniciar los hilos

p.start();

s.start();

try {

// Esperar a que ambos hilos terminen

p.join();

s.join();

} catch (InterruptedException e) {

e.printStackTrace();

}

// Imprimir el mensaje final

System.out.println("Fin programa");

}

}

**Resolución de problemas**

**ACTIVIDAD 5**

**Enunciado**

Utiliza el método sleep de la clase Thread en las clases Primero y Segundo del siguiente programa, de acuerdo a los siguientes requerimientos:

1. En la clase Primero, modifica el bucle para que durante 15 repeticiones detenga la ejecución del hilo durante 100 milisegundos.
2. En la clase Segundo, modifica el bucle para que durante 15 repeticiones detenga la ejecución del hilo durante 0,2 sappendegundos.

El nombre de la clase principal será Actividad5.

//Actividadx.java

**public** **class** Actividadx {

**public** **static** **void** main(String arg[]) {

Primero p = **new** Primero();

Segundo s = **new** Segundo();

p.start();

s.start();

System.***out***.println("Fin programa");

}

}

//Primero. java

**class** Primero **extends** Thread {

**public** **void** run() {

**for** (**int** i=1;i<10;i++)

{

System.***out***.println( "Primero " + i );

}

}

}

//Segundo. java

**class** Segundo **extends** Thread {

**public** **void** run() {

**for** (**int** i=1;i<=10;i++)

{

System.***out***.println( "Segundo " + i );

}

}

}

**Solución**

**Primero.java:**

// Clase Primero que extiende la clase Thread

class Primero extends Thread {

public void run() {

for (int i = 1; i <= 15; i++) {

System.out.println("Primero " + i);

try {

// Dormir durante 100 milisegundos

Thread.sleep(100);

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

}

**Segundo.java:**

// Clase Segundo que extiende la clase Thread

class Segundo extends Thread {

public void run() {

for (int i = 1; i <= 15; i++) {

System.out.println("Segundo " + i);

try {

// Dormir durante 200 milisegundos (equivalente a 0.2 segundos)

Thread.sleep(200);

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

}

**Actividad5.java:**

// Clase principal que crea y gestiona los hilos

public class Actividad5 {

public static void main(String[] args) {

Primero p = new Primero();

Segundo s = new Segundo();

p.start();

s.start();

try {

// Esperar a que ambos hilos terminen

p.join();

s.join();

} catch (InterruptedException e) {

e.printStackTrace();

}

// Imprimir el mensaje final

System.out.println("Fin programa");

}

}

**Resolución de problemas**

**CRITERIOS DE CORRECCIÓN**

1. Nivel de ajuste a la nomenclatura de objetos definida en el enunciado de la actividad y nivel de ajuste al almacenamiento de la información requerida en las subcarpetas definidas para la actividad y unidad didáctica correspondientes (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%).