**ACTIVIDAD 10**

**Enunciado**

Realiza un programa en Java que saque por consola información relativa a la prioridad del hilo principal y de otros dos hilos.

Características de la clase principal:

* Nombre: Actividad10
* Encargada de sacar por consola todas las frases del resultado esperado, a excepción de las que empiezan por “Ejecutando…”.
* Después de haber instanciado cada uno de los dos objetos de la clase Hilo, invocar al método que corresponda para que les asigne las prioridades 3 y 7, respectivamente.
* El nombre de hilo “main” y la prioridad “5” que aparecen en la primera línea del resultado esperado se obtendrán con métodos de la clase Thread.
* Los nombres de hilo “Thread-0” y “Thread-1” y las prioridades “3” y “7” que aparecen en la segunda y tercera líneas del resultado esperado se obtendrán con métodos de la clase Thread.
* Invocar primero al método start() del hilo de prioridad 3 y luego al del hilo de prioridad 7.

Características de la clase derivada de la clase Thread:

* Nombre: Hilo
* Su método run() se encargará de sacar por consola todas las frases del resultado esperado que empiezan por “Ejecutando…”.
* Los textos “Hilo-prioridad 7” e “Hilo-prioridad 3” que aparecen en las frases que empiezan por “Ejecutando…” serán mensajes de tipo String que se pasen como argumento al programa.

No olvides asignarle al proceso asociado al IDE Eclipse un único núcleo de los que dispone tu equipo, para que el programa obtenga el resultado esperado.

**Resultado esperado:**

main tiene la prioridad 5

Thread-0 tiene la prioridad 3

Thread-1 tiene la prioridad 7

Ejecutando Hilo-prioridad 7

Final programa

Ejecutando Hilo-prioridad 3

**Solución**

**Clase main:**package paquete;

public class Actividad10 {

public static void main(String[] args) {

System.***out***.println(Thread.*currentThread*().getName() + " tiene la prioridad " + Thread.*currentThread*().getPriority());

Hilo hilo1 = new Hilo("Hilo-prioridad 3");

Hilo hilo2 = new Hilo("Hilo-prioridad 7");

hilo1.setPriority(3);

hilo2.setPriority(7);

synchronized (hilo1) {

hilo1.start();

try {

hilo1.wait(); // Esperar a que hilo1 notifique antes de continuar

} catch (InterruptedException e) {

e.printStackTrace();

}

}

synchronized (hilo2) {

hilo2.start();

try {

hilo2.wait(); // Esperar a que hilo2 notifique antes de continuar

} catch (InterruptedException e) {

e.printStackTrace();

}

}

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

}

}

**Clase Hilo:**

package paquete;

class Hilo extends Thread {

private String mensaje;

public Hilo(String mensaje) {

this.mensaje = mensaje;

}

*@Override*

public void run() {

System.***out***.println(getName() + " tiene la prioridad " + getPriority());

if (mensaje != null) {

System.***out***.println("Ejecutando " + mensaje);

}

synchronized (this) {

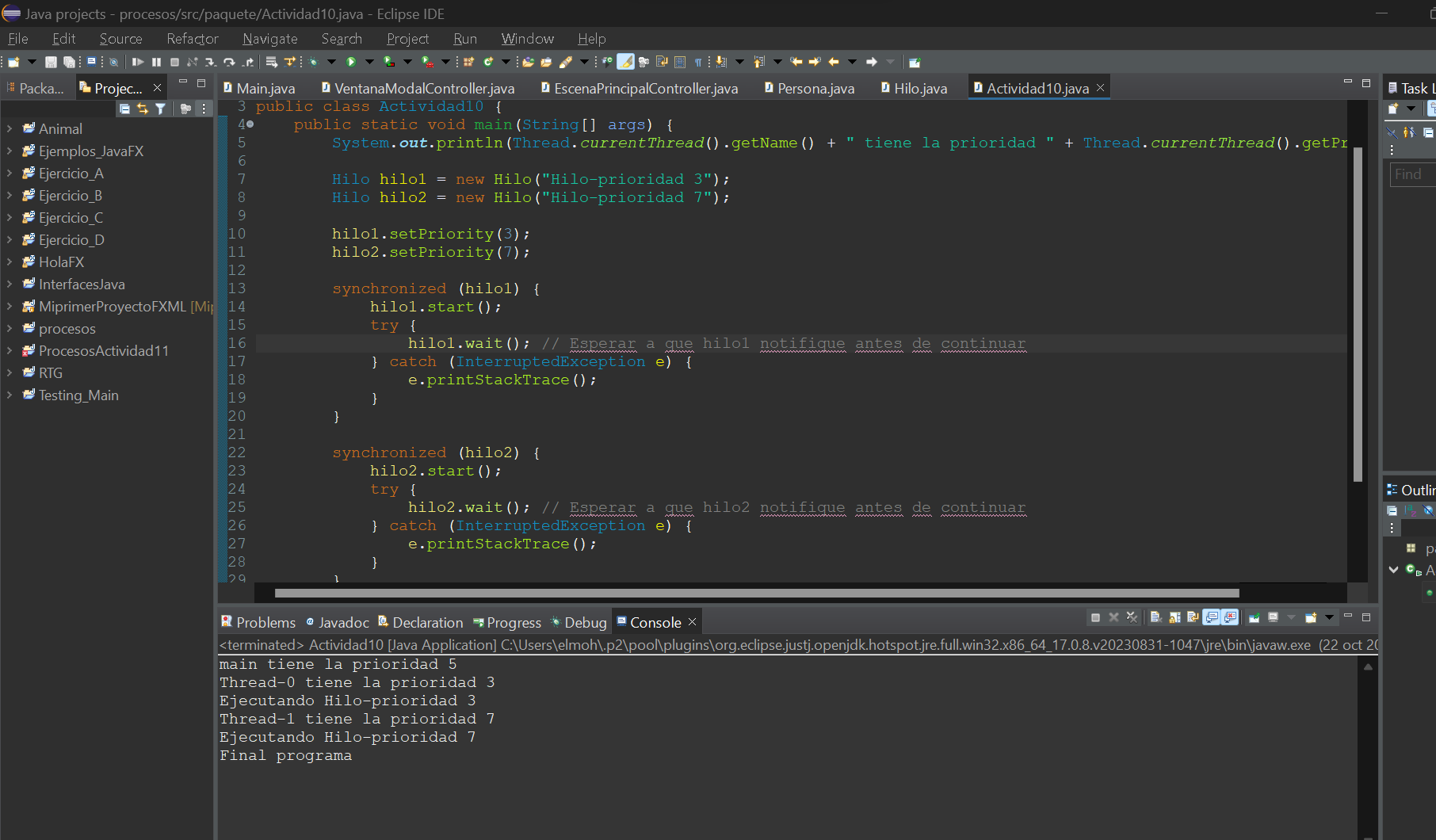
notify(); // Notificando al hilo principal de que ha terminado

}

}

}

**Resolución de problemas**



**REQUISITOS**

1. Entregar la solución de la actividad 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. El archivo 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%).