OPERACIONES CON HILOS

1) Dado el siguiente código:

```
/*
* ARCHIVO: PSP02Ejer2OperacionesHilos.java
*
*/
package psp02ejer2operacioneshilos;

import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
 * @author J
 */
public class PSP02Ejer2OperacionesHilos {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
        // TODO code application logic here
//Pongo nombre al hilo que representa el programa principal
        Thread.currentThread().setName("ProgPrin");
//creo el objeto hilo con el nombre "mihilo"
        Hilo mihilo = new Hilo("mihilo");
//compruebo si está vivo o muerto el hilo
        System.out.println("Hilo " + mihilo.getName() + " antes de iniciar: " + mihilo.isAlive());
        mihilo.start(); //arranco el hilo
//compruebo si está vivo o muerto el hilo
        System.out.println("Hilo " + mihilo.getName() + " en ejecución: " + mihilo.isAlive());
        try {
//hace dormir al hilo del programa principal
            System.out.println("HILO PRINCIPAL: " + Thread.currentThread().getName());
            Thread.currentThread().sleep(3000);
//al dormirse 3000, el hilo ya terminó porque se duerme 2000
        } catch (InterruptedException e) {
            Logger.getLogger(Hilo.class.getName()).log(Level.SEVERE, null, e);
        }
        System.out.println("Hilo " + mihilo.getName() + " después de ejecutarse: " +
mihilo.isAlive());
    }
}
```

```
/*
* ARCHIVO: Hilo.java
*
*/

package psp02ejer2operacioneshilos;

import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
 * @author WW
 */
public class Hilo extends Thread {
//le pongo nombre a mi hilo
    public Hilo(String nombre) {
        super(nombre);
    }

    @Override
    public void run() {
        try {
            sleep(2000);
        } catch (InterruptedException ex) {
            Logger.getLogger(Hilo.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
```

Responde a las siguientes cuestiones:

a) Si en el programa principal, le mando que me escriba "Thread.currentThread().getName()", ¿qué escribirá en consola?

b) Al escribir en el "void main(---)" la orden "Thread.currentThread().sleep(3000)", ¿qué hilo se va a dormir?¿y si pregunto por "Thread.currentThread().getName()" qué devolverá?

c) ¿qué vale isAlive() antes de iniciar, en ejecución y después de ejecución?

d) Cuando escribimos en el programa principal la orden "System.out.println("Hilo "+ mihilo.getName()+ " después de ejecutarse: "+ mihilo.isAlive());":

   - ¿realmente el hilo terminó ya su ejecución o no?¿por qué?
   - si realmente terminó su ejecución, ¿qué escribe esa sentencia como valor de "isAlive()"?
   - si realmente NO terminó su ejecución, ¿qué escribe esa sentencia como valor de "isAlive()"?

e) ¿Qué pasa si en el programa principal, cambio el tiempo de la orden sleep "Thread.currentThread().sleep(3000)" de 3000 a 1000? Realmente, cuando ejecute la sentencia final del programa principal "System.out.println("Hilo "+ mihilo.getName()+ " después de ejecutarse: "+ mihilo.isAlive());" ¿el hilo habrá terminado su ejecución o no? ¿por qué?. ¿Qué escribirá? Compruébalo añadiendo al hilo un "System.out.println("FIN HILO");" que se muestre cuando termine el hilo su ejecución.

f) Para comprobar si lo has entendido haz lo siguiente:

   1°)Añade la sentencia "System.out.println("FIN HILO");" como última línea del método run() en el archivo "Hilo.java"

   2°)Ejecuta ahora el programa y fíjate cuándo se muestra "FIN HILO" en la consola.

   3°)Ahora vete al programa principal "void main(---)" y cambia el tiempo de dormirse el hilo de 3000 a 1000.

   4°)Vuelve a ejecutar el programa y fíjate cuándo se muestra "FIN HILO" en la consola