Indicaciones de entrega

Una vez realizada la tarea elaborarás un único fichero **.zip** donde figuren las respuestas correspondientes. El envío se realizará a través de la plataforma de la forma establecida para ello, y el archivo se nombrará siguiendo las siguientes pautas:

apellido1_apellido2_nombre_SIGxx_Tarea

Asegúrate que el nombre no contenga la letra ñ, tildes ni caracteres especiales extraños. Así por ejemplo la alumna **Begoña Sánchez Mañas para la primera unidad del MP de DAW**, debería nombrar esta tarea como...

sanchez_manas_begona_PSP_Tarea01

Cada ejercicio ha de desarrollarse en un documento distinto, siguiendo los nombres de ej1, ej2,..

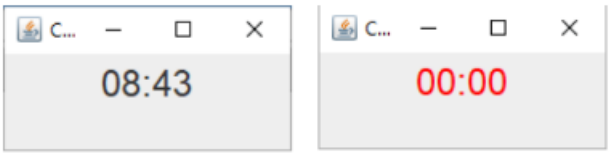
Ejercicios

1. **(0.5 pts)** Realiza una clase que cree 3 hilos (con un bucle) y visualice un mensaje donde se muestre el número de hilo que se está ejecutando y el contenido de un contador.
    a. El contador debe llegar hasta 5.
    b. A la hora de crear el hilo debe aparecer un mensaje de “Creando hilo“ y su número.
    c. Cuando el hilo termine, notificarlo.
    d. Una vez creados los 3 hilos, notificar que se han realizado estos.
    e. ¿Qué sucede? ¿Es lo que esperabas? Haz una captura de pantalla.
    f. ¿Se puede modificar el comportamiento de los hilos para que realice las acciones en el orden esperado? ¿Cómo?

2. **(0.5 pts)** Modifica el ejercicio anterior y haz que los hilos visualizen los datos que tienen por defecto (nombre, prioridad e ID).
    a. la clase del hilo, en el método run() debe mostrar sus datos por separado y además utilizando el método toString(). Luego realizará un bucle del tipo “visualizando Hilo x” 20 veces.
    b. ¿Es el resultado el mismo en todas las ejecuciones?
    c. Cambia los nombres y la prioridad (Investiga los niveles de prioridad posibles)
    d. ¿Se puede apreciar diferencia en el orden de los mensajes visualizados?

3. **(1 pts)** Crea dos clases (hilos) Java que extiendan la clase Thread.
    a. Uno de los hilos debe visualizar en pantalla en un bucle infinito la palabra TIC y el otro hilo la palabra TAC.
    b. Dentro del bucle utiliza el método *sleep()* para que nos dé tiempo a ver las palabras que se visualizan cuando lo ejecutemos, tendrás que añadir un bloque try-catch (para capturar la excepción *lnterruptedException).*
    c. Crea después la función *main()* que haga uso de los hilos anteriores. ¿Se visualizan los textos TIC y TAC de forma ordenada (es decir TICTAC TICTAC ...)?

4. **(1.25 pts)** Realiza un cronómetro con una interfaz gráfica. Programa el método run() para hacer un cronómetro que al llegar a 0 se para y se pone en rojo. Desde una clase principal hay que crear diferentes cronómetros con diferentes tiempos y ejecutarlos de forma simultánea:



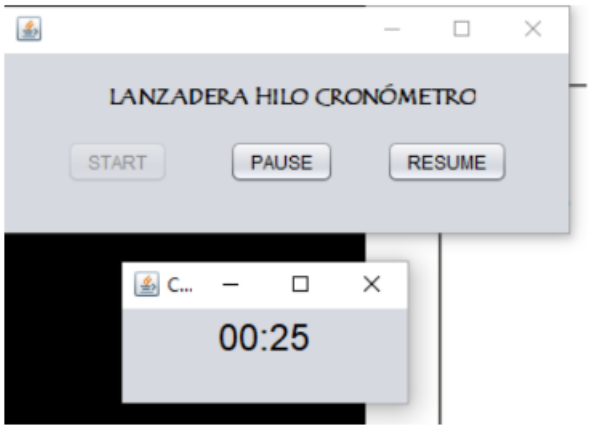
5. **(0.5 pts)** Crea una clase hilo que tenga tres métodos:
    a. run(): incrementará un contador en bucle sin retrasos y hasta que se indique mediante su correspondiente variable que debe terminar
    b. stopHilo(): activará la variable de finalización
    c. getContador(): retorna el valor del contador

    Desde la clase principal, crea 3 contadores y asígnales diferentes prioridades: 1, 5 y 10, o lo que es lo mismo: los atributos MIN_PRIORITY, NORM_PRIORITY Y MAX_PRIORITY de la clase Thread

    Espera 2 segundos y visualiza el contenido de los 3 contadores.

6. **(0.5 pts)** Crea un hilo que saque de forma continua un mensaje por pantalla mientras no reciba una interrupción. El programa principal le enviará la interrupción a los 5 segundos de iniciarse.

7. **(0.75 pts)** Recupera el ejemplo anterior de los cronómetros de forma que se lance solamente un hilo cronómetro, pero el hilo principal tenga dos botones para parar y reanudar el Cronómetro.



8. **(1 pts)** Haz un Juego que sea la carrera de los hilos: Al iniciar 3 barras de progreso competirán por llegar al final, debe haber una clase hilo que incremente un valor aleatorio el objeto ProgressBar que se le manda como parámetro y otra clase hilo que monitorice las progressBar y Etiquetas para ver quien gana.

9. **(1,25 ptos.) Tratamiento de ficheros concurrente**. Vamos a realizar un proceso con ficheros similar al de prácticas anteriores. Consistirá en contar el número total de líneas y lo haremos concurrentemente. En este ejercicio hay que utilizar threads básicos. No se pueden usar pools de threads, ni objetos Callable, ni Future.
    a. Elige tres ficheros de texto cualquiera de tu disco (si no los encuentras créalos con el Bloc de notas). Asegúrate de que tienen muchas líneas. Para apreciar la diferencia de tiempo de ejecución entre las versiones concurrente y secuencial del programa es deseable que los ficheros sean grandes (ej 10 Mb). Si no son suficientemente grandes, copia y pega repetidas veces el contenido de cada fichero en cada fichero.
    b. Crea una clase que pueda ser ejecutada concurrentemente y que cuente el número de líneas que contenga un fichero.
    c. Crea una aplicación que instancie tres “objetos buscadores”, uno para cada uno de los ficheros elegidos en el punto (a), y los arranque.
    d. El programa principal (el que lance los threads) pedirá el resultado a cada objeto buscador con el fin de contar el número total de líneas de los ficheros (la suma). **NO DEBEN APARECER RESULTADOS PARCIALES, SOLO EL RESULTADO GLOBAL.**
    e. Mide el tiempo que tarda el proceso desde que comienza (justo antes de lanzar los hilos) hasta que la estadística es agregada completamente.
    f. Haz lo mismo, pero de manera secuencial. Mide el tiempo que tarda en este caso y compara con el caso concurrente.

10. (1,5 ptos.) Comprobador de actualización de directorio.

a. Hacer un programa que liste por pantalla el contenido de un directorio del disco.
b. Tras el primer listado, el programa debe comprobar si hay cambios, por lo que debes programar una tarea periódica (que se ejecute cada 10 sg.) que compruebe si hay actualizaciones en el directorio, y, en tal caso, lo liste de nuevo.
c. Para simplificar, supón que el contenido del directorio ha cambiado si cambia su fecha

(**CUIDADO**: en sistemas de ficheros FAT32, como muchos USB, la fecha de modificación de un directorio no cambia aunque cambie su contenido).

11. (1,25 pts) Modelo productor-consumidor

¿Cómo podrías resolver el ejercicio de TIC TAC usando el modelo productor consumidor y los métodos sincronizados?

PISTA: tendrás que modificar el Productor para reciba una cadena de texto que será el mensaje que el consumidor ponga en la Cola, y tras coger dicho mensaje, mostrarlo por la consola. El main tendrá dos hilos Productor, uno que escriba TIC y otro que escriba TAC