EXAMEN - UNIDAD 2

CONSIDERACIONES A TENER EN CUENTA PARA REALIZAR LOS EJERCICIOS

1. Se debe hacer una gestión correcta de excepciones, no agrupándolas y no mostrando únicamente el printStacktrace de la excepción. Se deben mostrar mensajes indicando por qué se ha podido producir la excepción.
2. El código debe estar comentado para explicar qué se hace en cada caso.
3. Realiza ambos ejercicios en el mismo proyecto pero en distintos paquetes. Entrega el proyecto comprimido.
4. Puntuaciones de los ejercicios:
    - Ejercicio 1: 3,5 ptos + 0,25 ptos de comentarios
    - Ejercicio 2: 4,75 ptos + 0,5 ptos de comentarios

1. En una peluquería hay barberos y sillas para los clientes. Un cliente puede llegar a la barbería y encontrar alguna silla libre, en cuyo caso, el cliente se sienta y esperará a que algún barbero le afeite. Puede ocurrir que el cliente llegue y no haya sillas libres, en cuyo caso se marcha. Simular el comportamiento de la barbería mediante un programa Java teniendo en cuenta que:

    a. Se generan 10 clientes, algunos encuentran silla y otros no. Los que no consigan silla desaparecen (terminan su ejecución).

    b. Puede haber más sillas que barberos y al revés (poner constantes para poder cambiar fácilmente entre ejecuciones). Realiza el caso en que tengamos 2 barberos y 4 sillas.

    c. Imprimir por consola todos los estados por los que pasan los hilos.

    d. Pon nombre a los hilos para saber qué hilo se está ejecutando en cada momento.

    e. Imprimir si un cliente está esperando una silla y si la consigue. Si no la consigue y se marcha hay que indicarlo también.

    f. Si un cliente consigue una silla, imprimir el momento en el que se queda esperando a que lo atienda un barbero. Imprimir también el momento en el que está siendo atendido y cuándo sale de la barbería, liberando por tanto el barbero y la silla.

    g. Realiza una depuración del programa y añade capturas a la entrega donde se puedan ver varios hilos en ejecución y en qué punto del código se encuentra cada hilo. (0, 25 ptos)

2. Se tienen 4 hilos que realizan una colecta. En un tiempo aleatorio de entre 10 y 200 ms, cada hilo consigue una cantidad de entre 4 y 25. Una vez llegan a recolectar entre todos una cantidad de 2000, no pueden seguir recolectando dinero.

   Además, hay hilos consumidores que toman una cantidad de dinero de la ya recolectada por los recolectores, por ejemplo, entre 10 y 40, a intervalos de tiempo aleatorios entre 20 y 300 ms.

   Los hilos recolectores y los consumidores se ejecutan indefinidamente. Los hilos recolectores no pueden recoger dinero cuando se ha llegado a la cantidad máxima (2000), deben esperar a que los hilos consumidores retiren dinero, de manera que puedan aportar la cantidad que han recogido.

   Si los hilos consumidores no pueden retirar la cantidad de dinero que necesitan, porque no hay suficiente recolectado, deben esperar a que los hilos recolectores aporten suficiente dinero.

   En ambos casos, la espera debe ser no activa. Deben probarse ambas situaciones: los hilos recolectores esperan porque se ha pasado del máximo que se puede recolectar, y los hilos consumidores esperan porque no se ha recolectado suficiente dinero. Para ello se puede jugar con el número de hilos consumidores. Incluir en comentarios una breve explicación acerca del número de hilos consumidores con los que se puede probar cada una de estas situaciones.

   Ten en cuenta las siguientes consideraciones:

   - Comienza inicialmente por 4 hilos recolectores y 4 consumidores, y ve variando el número de consumidores para que se puedan dar los diferentes casos que se comentan en el último párrafo. Defínelos como constantes.
   - Ponle nombres a los hilos. En todo momento registra qué hilo recolector aporta dinero y cuánto, y qué hilo consumidor coge dinero y cuánto.
   - Establece las siguientes **prioridades a los hilos recolectores**: 1, 4, 7 y 10. ¿Se percibe que hay hilos que recolectan más dinero que otros? Añade comentarios en el código indicando si se percibe alguna tendencia.