Cuaderno de ejercicios

Tema 3 - Programación multihilo

1. Crea una clase llamada “MostrarASCII” que implemente Runnable y que tenga como atributo un número entero denominado “tipo”. Si el tipo es 1, la clase mostrará los caracteres ASCII impares y si es 2, los pares. La aplicación (método main) deberá crear dos hilos, uno para mostrar los impares y otro para los pares.

2. Implementa una clase denominada “Contador” que implemente Runnable con tres atributos: un atributo de tipo String denominado “nombreHilo”, otro atributo de tipo entero denominado “inicioContador” y otro de tipo entero denominado “limiteContador”, que indicará el fin de la cuenta. El programa principal deberá crear 5 hilos contadores con límites distintos y que se muestre por pantalla la cuenta de cada uno.

3. Crea una clase denominada “Caracol” que implemente Runnable y que tenga dos atributos: velocidad (tipo double) y nombre (tipo String). La aplicación deberá simular una carrera de 1 m entre 5 caracoles (con nombres y velocidades distintas), mostrando por pantalla el progreso (en porcentaje) de cada caracol hasta que recorra la distancia completa.

4. Crea una variación del programa anterior que asigne a todos los caracoles la misma velocidad, pero prioridades distintas a sus hilos correspondientes.

5. El delegado de clase ha comprado un bote extra grande de alitas de pollo en el KFC (100 alitas) y los va a repartir entre los 30 compañeros de clase. Para ello crea una clase que implementa Runnable y que tiene un método “consumirAlita” que mientras queden alitas permite consumir el número que pida cada compañero (entre 1 y 10 alitas, de forma aleatoria), un método “run” (obligado al implementar Runnable) que llama a “consumirAlita” y un “main” que crea un hilo por cada compañero. El “main” deberá mostrar al final de la ejecución cuántas alitas se han comido entre todos. Realiza dos ejecuciones: sin y con sincronización del método “consumirAlita” (problema similar al de las entradas).

6. Basándote en el problema del Productor-Consumidor (comunicación entre hilos), desarrolla un programa que tenga una clase “ControlSemaforos” con dos métodos que simulan dos semáforos distintos (“encenderSemaforo1” y “encenderSemaforo2”, que hace alternar los semáforos entre verde y rojo cada 3 segundos, no pudiendo estar los dos en el mismo estado a la vez) y un método “main” que crea un objeto “ControlSemaforos” y dos hilos, uno por cada semáforo.

   Puedes utilizar una variable de tipo int para controlar qué semáforo está en verde, de forma que cuando 1 esté en verde, el semáforo 2 espere en rojo. Cuando pasen 5 segundos, el semáforo 1 que está en verde cambiará a rojo y lo notificará al semáforo 2, que dejará de esperar y pasará a verde durante otros 5 segundos.

7. Crea una ampliación del programa de los caracoles (del 3 o del 4, es indiferente) para que se controle el momento en que el primer caracol termina la carrera (pista: utilizar un fichero que escribirá en disco el primero que acabe). Cuando esto suceda, el programa deberá realizar dos acciones:

- Hacer que el resto de participantes se detengan (en realidad, cada participante deberá “controlar por su cuenta” si alguien ha acabado ya (comprobando un fichero) y, si es el caso, finalizará automáticamente la carrera, mostrando por pantalla un mensaje indicando que abandona y en qué punto de la carrera -en porcentaje- se encontraba al abandonar).
- Mostrar un mensaje final indicando quién es el ganador.

8. Resuelve el problema de los NEOs de la NASA (Actividad Entregable 2) utilizando hilos (multithreading) en vez de procesos (multiprocessing). Compara los tiempos de ejecución entre resolverlo de una forma y de otra.