Escuela Técnica Superior de Ingeniería

Prácticas
de
Programación Concurrente y Distribuida

3º Curso de Grado en Ingeniería Informática

Curso 2016-17

EXAMEN

Enero de 2017

CONSIDERACIONES PREVIAS:

- **No se permite el uso de ningún tipo de documentación.**
- **El acceso a Internet está desactivado conscientemente.**
- **Apague el teléfono móvil.**

ANTES DE COMENZAR EL EXAMEN:

- Cree una carpeta con su nombre y primer apellido en el **Escritorio** separados por un guión bajo (ejemplo: **`Pedro_Abad`**).
- En dicha carpeta aparecerá un proyecto por cada una de las preguntas del examen. Dichos proyectos se denominarán **`Proyecto1`**, **`Proyecto2`**,…, **`Proyecto4`**.

ENUNCIADO:

Un parking de vehículos de una zona turística tiene capacidad para 5 coches y 2 autobuses. Los autobuses sólo pueden aparcar en los puestos que tienen designados, pero los coches podrán aparcar, además de en la zona de coches, en la zona de autobuses, si ningún autobús está esperando para hacerlo.

PROYECTO 1.

Tiempo estimado: 40 minutos.
Puntos: 4

Será el proyecto base para solucionar el enunciado. Contendrá las siguientes clases:

- **`Parking`**. La clase `Parking` mantendrá el estado de ocupación del parking e implementará los siguientes métodos:
  - **`EntraBus`**. Que deberá ser invocado por los autobuses cuando quieren acceder al parking .
  - **`SaleBus`**. Que deberá ser invocado por los autobuses al salir del parking.
  - **`EntraCoche`**. Que deberá ser invocado por los coches cuando quieren entrar en el parking.
  - **`SaleCoche`**. Que deberá ser invocado por los coches al salir del parking.
- **`Autobus`**. Representará cada uno de los autobuses mediante un hilo. El hilo se creará heredando de la clase `Thread`. El hilo pondrá un mensaje de inicio indicando su identificador, intentará acceder al parking usando la clase `Parking`, permanecerá en el parking una cantidad de tiempo aleatoria de entre
- 3 y 5 segundos y saldrá del parking.
- **`Coche`**. Representará cada uno de los coches mediante un hilo. El hilo se creará implementando el *interface* `Runnable`. El hilo pondrá un mensaje de inicio indicando su identificador, intentará acceder al parking usando la clase `Parking`, permanecerá en el parking una cantidad de tiempo aleatoria de entre 2 y 5 segundos y saldrá del parking.

- **`Generador`**. Contendrá el método `main` y será quién comience la ejecución. Debe lanzar, de forma aleatoria, coches o autobuses a intervalos de tiempo de entre 1 a 3 segundos. La frecuencia de llegada de coches será del 70% y la de autobuses del 30%. Deberá esperar a que finalicen todos los hilos para finalizar.

El control de la concurrencia y la sincronización se realizará en la clase `parking`, mediante las primitivas de Java `wait(), notify()` y/o `notifyAll()`.

PROYECTO 2.

Tiempo estimado: 20 minutos.
Puntos: 3

Se modificará el *Proyecto2* para que la clase `Parking` controle la concurrencia mediante `ReentrantLocks` y `Conditions`.

No podrá usarse el método `signalAll()` de las `Conditions`.

PROYECTO 3.

Tiempo estimado: 15 minutos.
Puntos: 2

Tomará como base el *Proyecto1* y se modificará la clase **`generador`** para que use un *ThreadPool* con un tamaño fijo de 3 hilos para lanzar los coches. Al finalizar, cada coche deberá devolver el tiempo que ha estado estacionado, y generador pondrá un mensaje final indicando el tiempo total de ocupación del parking por los coches.

PROYECTO 4.

Tiempo estimado: Depende de la implementación que se pretenda
Puntos: 1

Se creará un *Applet* que visualice de forma gráfica, mediante un *Canvas*, la situación del parking y las colas de espera del *Proyecto 1*.