Escuela Técnica Superior de Ingeniería

Prácticas
de
Programación Concurrente y Distribuida

3º Curso de Grado en Ingeniería Informática

EXAMEN
Enero

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:

En una revisión de exámenes hay tres profesores que revisan simultáneamente. Todos los profesores pueden revisar tanto la teoría como las prácticas, pero sólo dos profesores puede revisar prácticas al mismo tiempo. A la revisión llegarán estudiantes que desearán revisar sólo la teoría o sólo las prácticas. **Los estudiantes que quieren revisar prácticas tendrán prioridad, si pueden entrar.**

PROYECTO 1.

Tiempo estimado: 40 minutos.
Puntos: 4

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

- **`Teoría`**. Representará a cada uno de los estudiantes que revisarán la teoría mediante un hilo. El hilo se creará heredando de la clase `Thread`. El estudiante pondrá un mensaje de inicio, revisará un tiempo aleatorio entre 2 y 5 segundos y pondrá un mensaje de finalización al acabar.

- **`Prácticas`**. Representará a cada uno de los estudiantes que revisarán las prácticas mediante un hilo. El hilo se creará implementando el *interface* `Runnable`. El estudiante pondrá un mensaje de inicio, revisará un tiempo aleatorio entre 2 y 5 segundos y pondrá un mensaje de finalización al acabar.

- **`Generador`**. Contendrá el método `main` y será quién comience la ejecución. Debe lanzar 10 estudiantes a intervalos de tiempo de entre 1 a 3 segundos. El tipo de estudiante generado será aleatorio, con un probabilidad del 60% para teoría y del 40% para prácticas. La ejecución finalizará cuando todos los hilos hayan finalizado.

- **`Revision`**. La clase `Revision` se usará para controlar la entrada de los estudiantes a la revisión e implementará los siguientes métodos:

  - **`EntraTeoria`**. Que deberá ser invocado por los estudiantes que revisarán la teoría cuando quieren ser atendidos.

  - **`SaleTeoria`**. Que deberá ser invocado por los por los estudiantes que revisarán la teoría al terminar la revisión.

  - **`EntraPracticas`**. Que deberá ser invocado por los estudiantes que revisarán las prácticas cuando quieren ser atendidos.

  - **`SalePracticas`**. Que deberá ser invocado por los por los estudiantes que revisarán las prácticas al terminar la revisión.

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

PROYECTO 2.

Tiempo estimado: 35 minutos.
Puntos: 3

Se modificará el *Proyecto1* para que la clase `Revision` controle el acceso al taller mediante `ReentrantLocks` y `Conditions`.

En esta solución 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 **`Revision`** para que use un *ThreadPool* con un tamaño fijo de 5 hilos para lanzar los estudiantes de prácticas. Al finalizar, cada estudiante de practicas deberá devolver el tiempo que ha estado revisando, y generador pondrá un mensaje cuando final indicando el tiempo total que se ha empleado en revisar las prácticas de todos los alumnos.

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 taller y las colas de espera del *Proyecto 1*.

El nivel de representación gráfica es libre, pero al menos se deberán identificar los dos tipos de estudiantes, la ocupación de los profesores y las colas de espera.