Escuela Técnica Superior de Ingeniería

Prácticas
de
Programación Concurrente y Distribuida

3º Curso de Grado en Ingeniería Informática

Curso 2019-20

EXAMEN

Enero de 2020

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 delegación del Registro de la Propiedad trabajan tres oficiales y el registrador. Al registro llegan dos tipos de clientes, los que quieren obtener *notas simples informativas* y los que van a realizar registro de propiedades. Los clientes que desean notas simples pueden ser atendidos, indistintamente por cualquiera de los oficiales o por el registrador. Los clientes que van a realizar registros, necesitan ser atendidos por un oficial y el registrador al mismo tiempo.

PROYECTO 1.

Tiempo estimado: 40 minutos.
Puntos: 4

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

- **`Registro`**. La clase `Registro` mantendrá el estado de ocupación del Registro e implementará los siguientes métodos:
  - **`EntraSimple`**. Que deberá ser invocado por los clientes que quieren notas simples al llegar al Registro.
  - **`SaleSimple`**. Que deberá ser invocado por los clientes que quieren notas simples al salir.
  - **`EntraPropiedad`**. Que deberá ser invocado por los clientes que quieren registrar propiedades al llegar a la Registro.
  - **`SalePropiedad`**. Que deberá ser invocado por los clientes que quieren registrar propiedades al salir.
- **`Nsimple`**. Representará cada uno de los clientes que quieren notas simples mediante un hilo. El hilo se creará heredando de la clase `Thread`. El hilo pondrá un mensaje de inicio indicando su identificador, intentará ser atendido usando la clase `Registro`, será atendido durante un intervalo aleatorio entre 2 y 3 segundos y saldrá.
- **`Propiedad`**. Representará cada uno de los clientes que quieren registrar propiedades mediante un hilo. El hilo se creará implementando el *interface* `Runnable`. El hilo pondrá un mensaje de inicio indicando su identificador,

intentará ser atendido usando la clase `Registro`, será atendido durante un intervalo aleatorio entre 3 y 5 segundos y saldrá.

- **`Generador`**. Contendrá el método `main` y será quién comience la ejecución. Debe lanzar, de forma aleatoria, clientes de ambos tipos a intervalos de tiempo entre 1 y 3 segundos. La frecuencia de llegada de los clientes para obtener notas simples será del 80% y la de clientes para realizar registros del 20%. Deberá esperar a que acaben todos los hilos para finalizar.

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

PROYECTO 2.

Tiempo estimado: 20 minutos.
Puntos: 3

Se modificará el *Proyecto1* para que la clase `Registro` controle la concurrencia mediante `ReentrantLocks` y `Conditions`.

En este caso, los clientes que van a realizar registros tienen prioridad con respecto a los de la nota simple.

PROYECTO 3.

Tiempo estimado: 15 minutos.
Puntos: 2

Tomará como base el *Proyecto 1* y se modificará en la forma necesaria, de manera que **generador** permita el acceso de los clientes al Registro de forma remota, haciendo uso de R.M.I. Es decir, generador hará de servidor del objeto de la clase `Registro` para los clientes remotos.

PROYECTO 4.

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

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