Informe proyecto Nº1 sistemas operativos: el barbero dormilón

Profesor: Ricardo Pérez

Ayudante: Álvaro Elgueta

Autor: Matías Erenchun

Curicó 21 de abril 2019

Descripción de la solución

Entorno

La solución al problema plateado se desarrolló en el lenguaje de programación Java en la versión 1.8 del JDK, para esto se utilizó el entorno de desarrollo de IntelliJ IDEA en su versión Ultimate 2019.1.

Propuesta de solución

Para empezar, debemos decir que como el proyecto solicitaba que la implantación se hiciera mediante hilos y dado mi mayor manejo de java sobre c además de ver que java provee una seria de clases e interfaces que soportan la implementación hilos, como son Thread y Runnable, Ademas dado que personalmente estoy más familiarizado con el uso de Thread decidí desarrollar la solución en base a esta clase mediante la aplicación de la herencia de la clase Thread en las clases que requirieran ser un hilo independiente.

Por otro lado, personal mentes me familiarizo con el uso semáforos para controlar la interacción de distintos hilos sobre variables compartidas, aun a sabiendas de esto último me decidí a utilizar monitores ya que desconocía el cómo su implementación y luego de leer e investigar sus ventajas y falencias me decanté por probar el uso de monitores.

Para la solución del problema se planteó una solución utilizando Clases los cuales se describen a continuación:

Main: Clase principal donde se Crean las instancias de las demás clases y se hacen funcionar los distintos hilos mediante el método ***start***.

GeneradorCliente: Clase que extiende ***Thread***, se encarga de generar los clientes e iniciando el funcionamiento de su respectivo hilo mediante el método ***start***.

Barbero: Clase que extiende ***Thread***, mediante el método ***atender*** define el tiempo que se demora en atender a cada uno de los clientes.

Barberia: Clase que mediante métodos que implementan ***synchronized*** coordina la comunicación de los distintos hilos.

Cliente: Clase que extiende ***Thread***, en el método ***run*** el cual está sobre escrito se ejecuta el proceso de atención el cual se define en el enunciado.

Como podemos ver de lo anterior la clase encargada de sincronizar el acceso a los recursos compartidos por los hilos es la clase barbería la que funcionaría como el monitor que controla a los hilos clientes.

Para finalizar cabe mencionar que el programa entregado entregara por consola un estado de los hilos que están esperando el cliente que está siendo atendido y cuando este deja de utilizar los recursos.

Se imprime en pantalla los siguientes mensajes según el estado de un cliente:

"se fue indignado": el barbero estaba ocupado y no quedaban sillas libres en la barbería.

"se sentó a esperar": el cliente entro en la barbería, pero el barbero estaba ocupado.

"se sentó en la silla del barbero": el cliente está siendo atendido, si estaba esperando se levanta de su silla y deja espacio para que otro cliente puede sentarse a esperar.

"se termino de atender" "se fue feliz": estos mensajes indican que un cliente termino de ser atendido y se fue feliz.

Conclusión

Ajuicio personal creo que, si bien mi solución e implementación no son ni las más limpias ni la mejores, cumplen con el objetivo del proyecto aun cuando queda mucho por mejora, pulir y entender del uso de las distintas técnicas de control sobre el uso de hilos.

PD: al final no logre detectar porque un hilo nunca se finalizaba al correr el programa.