Mini Proyecto 2 – Simulación de un cine

Cuando una persona se encuentra en la cola de un cine para comprar entradas, en general, es posible pensar que, cuantos más cajeros, más rápida se moverá la fila. Cuando sólo hay una persona para vender entradas, debe atender todas las compras, generándose una cola, y en cada momento solamente una persona puede ser atendida. Con dos personas atendiendo, la situación es diferente. Teniendo esto en cuenta, el propietario de un cine desea que sus clientes estén satisfechos con el mecanismo de compra. Sin embargo, no quiere contratar más personas que las que realmente necesite. Actualmente, sólo tiene una persona contratada. Si el propietario pudiera simular el número de cajeros necesarios (en hora punta), de tal manera que el tiempo medio de atención de cada cliente (tiempo de llegada más el tiempo de espera y compra) sea de 2 minutos, que es el tiempo únicamente de compra sin esperar, podría contratar más personas, pero sólo las necesarias.

Ejercicio: diseño e implementación de un simulador de un cine

Consideraciones iniciales:

- Únicamente hay una fila para comprar entradas. La primera persona que entra en la fila es la primera en ser atendida.
- Los clientes, en hora punta, llegan cada 15 segundos.
- Si hay un cajero disponible, el cliente es atendido inmediatamente en el momento que llega. En caso contrario, debe esperar.
- El tiempo considerado para atender cada cliente, desde que llega al cajero hasta que compra la entrada, es de 2 minutos (120 segundos).
- El número máximo de cajeros es 10.
- La simulación se hará con 100 clientes.

Interfaz Queue<T>

El primer paso será pensar la estructura de datos adecuada para modelar el problema. Todo apunta hacia algún tipo de estructura de orden secuencial ya que los clientes llegan en un orden concreto y éste se ha de mantener. Una pila no parece adecuada, ¿no? Debe ser una cola … *First In First Out*. Por lo tanto, el siguiente paso consiste en implementar una **cola** del siguiente modo:

- Primero, definid una **interfaz** Queue<T> con tres métodos:
    - public boolean isEmpty();
        - devuelve True si la cola está vacía, False en caso contrario
    - public void enqueue(T x);
        - añade x como el último elemento de la cola
    - public T dequeue();
        - devuelve el primer elemento de la cosa
        - lanza NoSuchElementException si la cola está vacía

Clase LinkedQueue<T>

- Diseñaremos la implementación de la cola como una estructura de nodos simplemente enlazados (por tanto, más simple que la clase LinkedList presentada en clase). Cada nodo tendrá un apuntador al siguiente nodo (ya que la cola solamente se mueve en un sentido) y la cola, tendrá referencias tanto al primer nodo como al último, ya que necesitamos acceder a ambos extremos (uno para sacar elementos y otro para meterlos).
- Con la interfaz y la clase del nodo, lo siguiente es implementar la clase que representa la cola del cine, que se podría llamar **LinkedQueue<T>**. A continuación, se proporciona el esqueleto de esta clase:

Clase Client

El siguiente paso, antes de la simulación, es modelar los clientes del cine. Estos serán los elementos con los que debe trabajar la cola previamente definida e implementada.

Para cada cliente, en base a la descripción realizada, interesa su **tiempo de llegada** y **salida**. Además, interesa calcular el **tiempo total** desde que llega al cine hasta que finaliza la compra.

Clase Simulation

Ahora sí, el siguiente y último paso, es la simulación y ésta clase representará el programa principal (tendrá definido el método main).

- El programa realizará las simulaciones del tratamiento de los 100 clientes considerando que el número de cajeros varía desde 1 hasta 10.
- Para cada número de cajeros:
    - Creará la cola con los 100 clientes, cada uno con su tiempo de llegada.
    - Creará un array en el que, para cada cajero, anotaremos el instante de tiempo en el que quedará libre (inicialmente el cajero no está atendiendo a nadie, por lo que se inicializará con ceros).
    - La clave de la solución es averiguar cómo se modifican los valores en el array de tiempos de los cajeros y los tiempos de finalización de cada cliente.
    - Cuando se acaba de tratar a un cliente, su siempo de finalización se actualiza.
    - Para cada cliente hemos de acumular el tiempo de tratamiento (para poder calcular el promedio en cada simulación).

Pistas que simplifican (enormemente) la solución

- Ésta ya la hemos comentado: tener la cola formada por los 100 clientes desde el principio. No es como sucede en la realidad, pero no afecta para nada al resultado.

Universitat de Lleida
Departament d'Informàtica
i Enginyeria Industrial

- Como los clientes llegan separados en el tiempo y los cajeros tardan el mismo tiempo en atenderles, los cajeros siguen un orden cíclico a la hora de ir tratando clientes.
- Considerad, por ejemplo, tres cajeros:
    - Llega el primer cliente a t=0 y le atiende el cajero 0 (que está libre desde t=0). Este cajero volverá a estar libre a t=120
    - Llega el segundo cliente a t=15 y le atiende el cajero 1 (que está libre desde t=0). Este cajero volverá a estar libre a t=135
    - Llega el tercer cliente a t=30 y le atiende el cajero 2 (que está libre desde t=0). Este cajero volverá a estar libre a t=150
    - Cuando llega el cuarto cliente a t=45
        - ¿Puede ser atendido? No, ningún cajero está libre a t=45.
        - ¿Qué cajero será el primero que lo podrá atender? Si miramos los tiempos, son 120, 135 y 150, es decir, el cajero 0 será el primero que quedará libre
    - Es por ello que la asignación de cajeros a clientes sigue el ciclo
        - 0, 1, 2, 0, 1, 2, 0, 1, 2, 0, ...

Resultados de la simulación

¿Cuánto tiempo tardamos en promedio en atender a un cliente en la situación inicial de 1 cajero? Nuestra simulación indica que ¡casi hora y media!

¿Cuántos cajeros son necesarios para atender 100 clientes sin que esperen (que su tiempo medio sea 120s)? Nuestra simulación nos indica que con 8 es suficiente.

¿Cuántos cajeros son necesarios si el propietario indica que desea un tiempo medio de compra de entrada inferior a 7 minutos (420 segundos)? En este caso, nuestra simulación nos dice que con 6 cajeros es suficiente.

Entrega

La entrega de este ejercicio consiste en un proyecto Netbeans con los ficheros que contienen vuestra implementación.

El resultado obtenido debe ser entregado por medio de un único fichero comprimido, **uno por cada grupo,** con el nombre "MP2_NombreIntegrantes".

Consideraciones de Evaluación

A la hora de evaluar el laboratorio se tendrán en cuenta los siguientes aspectos:

- Desgraciadamente hemos de indicar que una solución que se detecte como copiada tendrá **una nota igual a 0 y no será recuperable**.
- Corrección de la solución (una solución incorrecta tendrá una nota inferior a 4).
- Calidad y limpieza del código (nombres de variables y métodos, descomposición funcional, etc. etc).
- Diseño orientado a objetos.
- Realización de tareas opcionales.
    - Podéis considerar variaciones que hagan la simulación más realista:
        - Los clientes no llegan de 15 en 15 segundos sino que estos varían aleatoriamente en un cierto intervalo (p.e. entre 5 y 20 segundos)
        - Los cajeros no tardan siempre 120 segundos en atender a un cliente sino que el tiempo varía en un cierto intervalo (p.e. entre 60 y 120 segundos)
    - En estos casos, al haber factores aleatorios, para tener mayor seguridad en las conclusiones, deberemos, para cada número de cajeros, ejecutar varias simulaciones y promediar los resultados (también se suelen indicar variancia, mínimo y máximo)