Modelado y Programación
Proyecto Final

Profesor: Alejandro Hernández Mora *
Ayudante: Pablo Camacho González †
Ayudante Lab: Luis Manuel Martínez Dámaso ‡

1 Supermercado

Un supermercado te contrata para que hagas un simulador de ventas, con la finalidad de ver el número ideal de cajas rápidas (para los clientes que lleven 20 artículos o menos) que les conviene poner. Tu deber es hacer un programa que simule el comportamiento del supermercado y que haga suficientes simulaciones variando el número de cajas rápidas para poder calcular el número ideal de cajas rápidas que hay que abrir. Afortunadamente el curso de modelado y programación nos dio las herramientas necesarias para hacer el programa.

2 Requerimientos

Escribir un programa en `Java` o algún otro lenguaje orientado a objetos que simule el comportamiento del supermercado. El programa debe cumplir con los siguientes requisitos:

1. Debe ser resistente a fallos (por ningún motivo debe lanzar ningún tipo de excepción).

2. Se guardará un banco de datos (*Almacén*), que tenga almacenados los datos de los productos en venta que tiene el supermercado. Cada producto deberá contar con un `id` único, nombre, precio y cantidad de existencias en el supermercado. Por ejemplo:

   ```
   12345, Coca-cola 300 ml, $13, 200
   ```

   Donde el primer número es el `id`, el segundo el nombre, el tercero el precio y el cuarto nos indica que quedan 200 de estos productos en el almacen.

3. El programa deberá ser persistente, es decir que simulemos el comportamiento del supermercado por un día (Algún tiempo arbitrario que decidas en milisegundos, por ejemplo 100000) y al final guarde el estado del almacén después de las ventas del día. Además de descontar a cada producto la cantidad de elementos vendidos, se deberá llevar la cuenta del monto vendido en total.

4. El supermercado cuenta con 15 cajas, el dueño desea saber cuántas de éstas deberían convertirse en cajas rápidas para un funcionamiento óptimo. Puedes considerar una unifila, es decir una sola cola para `x` número de cajas rápidas, o simplemente formar a los clientes en distintas cajas rápidas pero cuidando que se distribuyan uniformemente.

*`alejandrohmora@ciencias.unam.mx`
†`pablopcg1@ciencias.unam.mx`
‡`luismanuel@ciencias.unam.mx`

5. Deberás simular el ingreso de un nuevo cliente al supermercado cada cierto tiempo de manera aleatoria. Así como debes simular su ingreso, debes simular cuántos y cuáles productos comprará.

6. Para cada compra se generará un ticket con un `id de venta`. El ticket deberá contener: id del producto comprado, número de veces que se compró ese artículo en esta compra, nombre del producto, el precio unitario del producto, la suma del costo de cada producto el número de veces que fue comprado y por último el monto e iva total de la compra completa. Ejemplo:

```
TICKET DE COMPRA
--------------------------------------------------
#Producto   Cantidad    Nombre       Precio    Total
123456789   1           Galletas     $15       $15
234567890   5           Pluma azul   $5        $25
--------------------------------------------------


                                  Subtotal: $33.60
                                  IVA:      $6.4
                                  Total:    $40.0
--------------------------------------------------
¡GRACIAS POR TU COMPRA, VUELVE PRONTO!
--------------------------------------------------
```

   Al final del día se deberán guardar todos los tickets de compra del día en un Archivo que lleve como nombre *Ventas Fecha*, donde Fecha será la fecha (puede ser con hora, para que no tengas que estar borrando el archivo y genere distintos archivos). Al final de todos los tickets debe de venir la suma del total de todas las compras y deberás mostrar la caja (no rápida) con el máximo número de clientes formados en todo el día. Considera a tu criterio la forma de justificar el número de cajas rápidas que debe abrir el supermercado.

3 Modelo y manejo de datos

Se les pedirán algunos objetos que ustedes deben modelar de manera óptima. Se tomarán en cuenta las estructuras de datos qe utilicen para modelar sus objetos y también los patrones de diseño que puedan o no utilizar para el proyecto. A manera de consejo, hay tres patrones de diseño muy obvios que pueden utilizar en el modelo de su programa.

1. Deberás crear un objeto `Producto`, que tenga los atributos necesarios para cubrir con los requerimientos. Si deseas, puedes agregar otros atributos y funcionalidades.

2. Deberás modelar el almacén de artículos con una estructura de datos, ten cuidado al elegirla, pues si utilizas una estructura que aumente la complejidad del programa considerablemente, será penalizada la decisión. Considera la estructura ordenable, donde el índice de comparación de los artículos será el `id`. Recuerda que el supermercado puede decidir agregar nuevos artículos, así que hay que generar un `id` único cada que esto sucede (HINT: contador estático de la clase). También podemos resurtir de productos, así que si un producto se encuentra agotado, no debemos borrarlo del almacén, y deberá ser posible aumentar el número de existencias al producto.

3. Deberás crear un objeto `Cliente`, que lleve un carrito de compras. Para modelar el carrito, en el cuál agregarás n artículos, donde n se generó de manera aleatoria.

4. Deberás crear un objeto Caja, que simulará atender a los clientes que están formados, los cuáles serán atendidos como van llegando.

   Además considerarás las cajas rápidas, que sólo reciben a clientes con carritos con una cantidad de artículos menores o iguales a 20.

   Una parte importante de la simulación es que cada caja se tarde cierto tiempo en atender a un cliente, ese tiempo debe ser directamente proporcional al número de artículos que lleva. Para hacer la simulación más real, considera un intervalo de tiempo que el cajero puede tardar en cada artículo cobrado, genera el tiempo de manera aleatoria para cada artículo; cada cajero tiene una velocidad distinta, así que el intervalo puede cambiar entre cajas (aunque no mucho).

5. Los clientes son inteligentes, así que se formarán en la caja con menos clientes. Si las cajas rápidas cuentan con unifila, los clientes con pocos artículos se formarán ahí; si no tiene unifila, de igual manera deben formarse en la caja rápida con menos clientes formados.

6. Como parte de la simulación, las cajas tendrán un recurso compartido más. El gerente es la única persona que puede cancelar la venta de un producto que se ha pasado por el lector de código de barras en la caja. Las cajas podrán llamar al gerente para cancelar un producto cuando sea necesario. Para el correcto funcionamiento de esta simulación, deberás hacer que este procedimiento se utilice cuando mucho con el 3% de los productos vendidos.

   El gerente no necesariamente tiene que ser un hilo aparte, pero es claro que es una variable compartida y acceder a sus funciones son parte de la sección crítica. En la compra, en el ticket debe aparecer cada que se descuenta un articulo cancelado. Recuerda que ele gerente es único en la tienda, no puede haber más de una instanca de este objeto.

7. Al final del día, crearás un archivo con los datos de los productos agotados (o con pocas existencias), así el supermercado podrá saber qué artículos resurtir. También generarás un reporte sobre cuál fue la caja que atendió más clientes, la caja que vendió más productos y la caja que cobró más dinero.

8. Tu programa deberá tener un menú donde puedas elegir entre hacer labores administrativas del supermercado (resurtir la tienda), ejecutar una simulación en tiempo real (que muestre los mensajes de las cajas atendiendo a sus clientes), y una opción que haga la simulación con distintos números de cajas rápidas.

   La última opción debe mostrar el análisis de la ejecución iterando el número de cajas rápidas, en el formato de su preferencia y dependiendo de el criterio que elijan, su efectividad.

9. El programa deberá considerar hilos, pues la idea es que sea una simulación en tiempo real. El hilo principal deberá ser el supermercado, simular que un cliente llega es crear e inmediatamente formar a un cliente nuevo en una caja (OJO, el almacén es una variable compartida y puede ser accedida por varios elementos, deberás impedir que dos elementos la tomen al mismo tiempo). Las cajas también deberán ser hilos, pues atienen a los clientes cada que uno se forma.

10. **Punto extra:** Puedes hacer que una caja esté inhabilitada por un tiempo aleatorio y que abra de nuevo.

11. **Punto extra:** Haz que tu programa genere un archivo txt con los datos de la simulación iterando el número de cajas rápidas, de manera que se pueda hacer una gráfica con este archivo. Entrega

además un script de **gnuplot** que lea este archivo y genere las gráficas. Incluyelas en tu reporte y hazlo más completo.

Recuerda que la presentación de la información es la que vende los proyectos, así que si te queda muy bien esta parte podrás ganar hasta dos puntos extra.

4 Reporte

Ademas de los que lleva el reporte se espera que se mencionen los siguientes puntos:

- Diagrama UML del proyecto
- Patrones de diseño que se usaron y la razon del por que se usaron
- Reporte del speedup que se obtuvo de los diferentes experimentos
- Una conclusion del cual es la mejor configuracion de cajas rapidas y por que, este hecho debe estar basado en el calculo del speedup

5 Entrega

La entrega será conforme a los requisitos de entrega de las prácticas, únicamente a través de correo electrónico en la dirección `pablopcg1@ciencias.unam.mx`. Para quienes entregan esto como proyecto final, el asunto deberá ser **[MYP-Proyecto]**, para quien haga final el asunto será **[MYP-FINAL]**. El proyecto deberá ser entregado antes del día **viernes 7 de junio de 2018 a las 23:59**.

Dado que es el proyecto, **NO SE RECIBEN TAREAS DESPUÉS DE ESTA FECHA**.

¡Éxito en el final, ocupa las sesiones de dudas también!