Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Estructura de datos P
Vacaciones de junio 2019
Catedrático: Ing. Luis Espino
Auxiliar: Javier Navarro

MiniMarket Manager

Practica de laboratorio

Contenido

1 Objetivos

1.1 Objetivo general

Aplicar los conocimientos del curso Estructuras de Datos en el desarrollo de una aplicación.

1.2 Objetivos específicos

- Demostrar los conocimientos adquiridos sobre estructuras de datos lineales poniéndolos en práctica en una aplicación de simulación.
- Utilizar el lenguaje C++ para implementar estructuras de datos lineales.
- Utilizar la herramienta Graphviz para graficar estructuras de datos lineales.

2 Descripción

La práctica consiste en el desarrollo de una aplicación que, utilizando las estructuras de datos explicadas en clase y sus algoritmos, presente una simulación de los diferentes procesos en que se dan en un supermercado.

Dicha aplicación deberá ser capaz de representar las estructuras de forma visual, mediante la utilización de librerías soportadas (Graphviz).

2.1 Simulación

Un pequeño supermercado desea realizar una simulación paso a paso del proceso de venta. Estos pasos se simularán aleatoriamente por cada estructura de datos. Solo se realizará un paso por vez.

Este supermercado tiene un número n de carretas, que se definirá dentro del sistema. Además, se podrá definir una variable m que representa el número de cajas que existen. La simulación sigue las siguientes reglas:

- Si llega un cliente y no hay ninguna carreta disponible, este cliente espera a que lo halla.
- Ningún cliente se impacienta y abandona el supermercado sin pasar por alguna de las cajas.
- Cuando un cliente finaliza su compra, se coloca en la cola de pagos y espera su turno para pasar a un cajero.
- En el momento en que un cliente paga en la caja, su carreta queda disponible.

2.2 Carretas

El número n de carretas estará definido por el usuario. Estas se organizarán en **dos pilas**, actuando cada una independiente de la otra, es decir, se puede dar el caso que una pila esté vacía y la otra completamente llena. Los datos del nodo de esta pila será solamente un número identificador de la carreta.

2.3 Cola de espera de carretas

Debido a que las carretas tienen un límite, se implementa una **cola** de espera. Cuando hay una carreta en alguna pila, el cliente que está esperando la toma y se dirige a realizar sus compras. Se toma la cantidad de carretas necesarias en cada paso.

2.4 Compras

Luego de tomar su carreta, el cliente entra a una **lista circular simplemente enlazada** de compras. La forma de salir de este proceso es generando un número aleatorio entre 0 y 100. Si el número generado supera el tamaño de la lista, entonces ningún cliente pasa a la cola. En caso de que el número generado sea un índice adecuado, ese cliente saldrá de la lista para hacer la cola de pagos.

2.5 Cola de pagos

Se realizará una única cola de pagos para pasar a las cajas. Si las cajas están disponibles, pasan directamente. Si las cajas están llenas, entonces se hace una cola en espera de que alguna caja se desocupe.

2.6 Cajas

Se explicó anteriormente que existe un número m de cajas existentes. Cada caja tendrá los siguientes datos:

- Número de caja
- Tiempo de servicio
- Estado (Libre u ocupado)
- Cliente que está siendo atendido
- Carreta que usa el cliente

Las cajas estarán organizadas en una lista doblemente enlazada ordenada por número de caja.

2.7 Salida del sistema

Al terminar de pagar, se genera un número aleatorio entre 1 y 2 que decide en qué pila se coloca la carreta que se acaba de liberar.

2.8 Modulo visible

Cada una de las operaciones antes mencionadas deben poder verse en una única gráfica que muestre lo que está sucediendo en el sistema. Además, se debe realizar una consola que informe sobre cada acción realizada. Se debe generar la gráfica una vez por turno. Se recomienda el uso de subgraph para mantener un mejor orden de las gráficas.

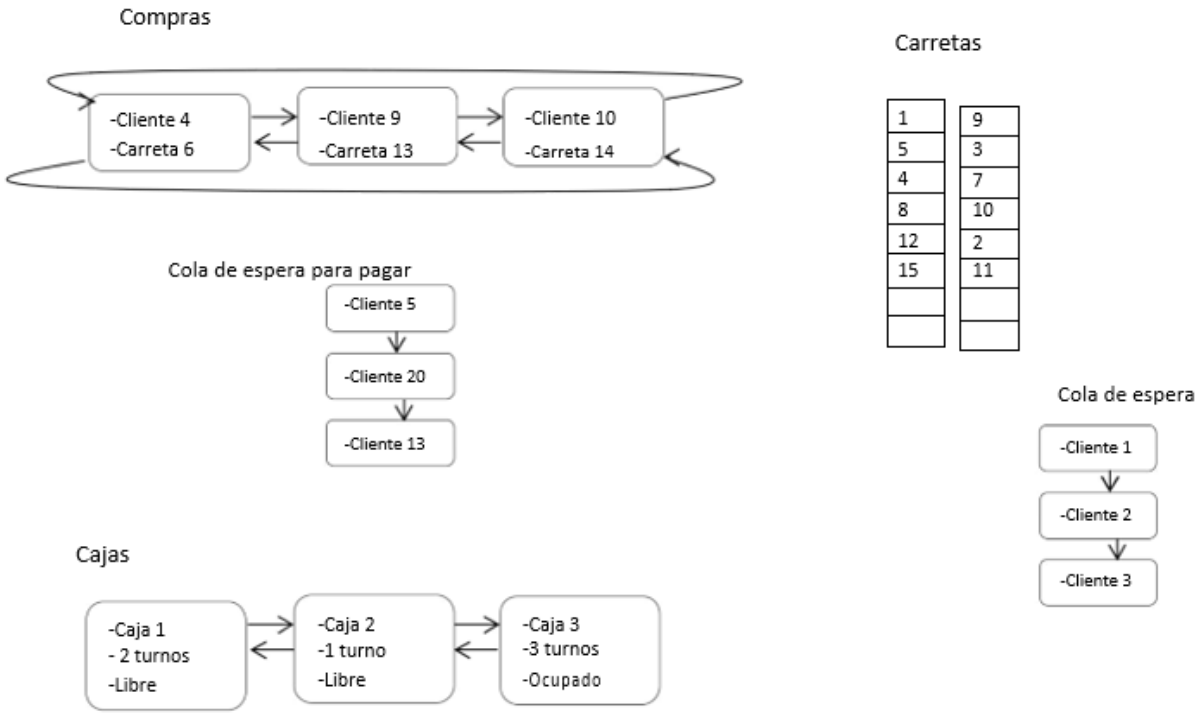
2.9 Representación grafica del sistema

Se puede ver un diagrama sugerido de cómo se debe representar la aplicación, mostrando los datos requeridos. Cada paso se puede dar haciendo una llamada al método por medio de un botón o una instrucción de consola (queda a discreción del estudiante). Antes de iniciar la simulación, se debe definir todos los parámetros necesarios para el inicio de esta:

- Cantidad clientes en cola de espera de carretas
- Cantidad de carretas por pila
- Cantidad de clientes comprando
- Cantidad de clientes en cola de pagos
- Cantidad de cajas

Con cada llamada, se debe preguntar cuántos clientes nuevos entran al sistema.

Ejemplo:



La salida en consola seria la siguiente:

```
CONSOLA
**********Paso 54**********
Cliente 15 está siendo atendido por caja 3.
Cliente 14 está siendo atendido por caja 2
**********Paso 55**********
Llega cliente 3 y se agrega a cola de espera
Cliente 15 está siendo atendido por caja 3
Cliente 14 sale del sistema. Libera carreta 11 y la caja 2.
Cliente 5 entra a cola de pagos.
```

3 Observaciones

- Lenguaje de programación a utilizar: C/C++
- Sistema Operativo: Libre
- IDE: Libre.
- La gráfica debe mostrarse dentro de la aplicación, no buscarse en carpetas ajenas. La aplicación que no genere gráfica no podrá ser calificada, se debe utilizar Graphviz.
- Todas las estructuras deben de ser realizadas por el estudiante, sin el uso de librerías específicas de ningún IDE o Framework.
- Durante la calificación se harán preguntas para validar que el estudiante realizo la práctica, de no responder correctamente anulara la nota obtenida en la o las secciones en la que aplique tal concepto
- La aplicación será compilada y ejecutada al momento de la calificación.

4 Fecha y modo de entrega:

- Jueves 13 de junio antes de las 8:00 horas vía Classroom
- Si no se cumple con la hora de entrega, se tendrá una penalización de 20 %.
- Si no se entrega en Classroom, no se calificará.
- Copias tendrán nota de 0 puntos y serán reportadas al catedrático y a la Escuela de Sistemas.

REFERENCIA: MYNOR MARCOS, ENUNCIADO DE VACACIONES 2012 DEL CUAL SE EXTRAJO LA IDEA DEL PRESENTE.