TAD Cola de Prioridad

Práctica 16
Aplicación de colas de prioridad

Objetivo

El objetivo de esta práctica es que el alumno ejercite el uso de colas de prioridad, desarrollando una aplicación que simula la venta y compra de acciones de una empresa.

Descripción general

Esta práctica consiste en desarrollar una simulación de un mercado de acciones en donde los usuarios puedan vender o comprar acciones de una empresa. El usuario hace una orden de compra o venta indicando cuántas acciones quiere comprar o vender y el precio por acción al que está dispuesto a comprar o vender.

El programa leerá de un archivo de texto las órdenes de compra y venta de acciones, y las irá procesando para tratar de cerrar o completar las órdenes. Si se completa una orden entonces se le llama transacción. Para ello se requiere comparar la orden de compra de mayor precio contra la orden de venta de menor precio, y así poder cerrar la o las transacciones. En caso de que el precio de la orden de compra sea mayor que el de la orden de venta se realizará la transacción tomando el precio de la orden de compra. Si en una transacción se tiene que el número de acciones de la compra es mayor que el de la venta, se deberá de tratar de completar la orden de compra para el resto de las acciones. Toda transacción será guardada para poder ser mostrada en pantalla al terminar de la lectura del archivo.

El formato de una orden de compra o venta es la siguiente:
NombreUsuario,TipoOrden,NumeroDeAcciones,Precio. Cada dato estará separado por una coma. El dato *TipoOrden* debe ser una cadena con el valor “comprar” o “vender”. El dato *Precio* usará unidades enteras de la moneda peso.

Ayuda: hacer uso de dos colas de prioridad, una almacenará las órdenes de venta en una cola de prioridad mínima, mientras que para las órdenes de compra se usará una cola de prioridad máxima.

Material

El material de esta práctica consta de los siguientes archivos:

- **InterfazColaConPrioridad.class** interfaz para trabajar con colas de prioridad.
- **Heap.class** clase que implementa el TAD colas de prioridad haciendo uso de heaps. **Nota:** usar las clases desarrolladas en la clase anterior.
- **Orden.java** clase que implementará la estructura de una orden de compra o venta de acciones.
- **ComparaOrdenesVenta.java** clase que creará un comparador de órdenes de venta de acciones.
- **ComparaOrdenesCompra.java** clase que creará un comparador de órdenes de compra de acciones.
- **MercadoAcciones.java** clase que implementará la simulación del mercado de acciones.
- **PruebaMercadoAcciones.class** programa para probar la clase `MercadoAcciones`.
- **ordenes.txt** archivo usado por la clase `PruebaMercadoAcciones`.
- Documentación:
    - InterfazColaConPrioridad.html documentación de la interfaz `InterfazColaConPrioridad`.
    - Heap.html documentación de la clase `Heap`.

Desarrollo

1. Descargar los archivos `InterfazColaConPrioridad.class, Orden.java, MercadoAcciones.java, PruebaMercadoAcciones.class` y `ordenes.txt` en el directorio donde se va a desarrollar la práctica. **No olvidar** incluir los archivos de la clase `Heap` desarrollados en la práctica anterior.

2. Leer la documentación y revisar el material de la práctica. Recordar que la firma de los métodos proporcionados y de los requeridos en el desarrollo de la práctica no podrán ser modificados.

3. En la clase `Orden` se implementarán los siguientes métodos:

    a. **Constructor** que reciba todos los atributos.
    b. Métodos **modificadores**.
    c. Métodos de **acceso**.
    d. Método **toString** para convertir una orden a una cadena de caracteres.

4. Escribir la clase `ComparaOrdenesVenta` que implemente la interfaz Comparator de Java de acuerdo con su precio de venta. **Ayuda:** considerar que se requiere hacer uso de una cola de prioridad mínima para almacenar este tipo de orden.

5. Escribir la clase `ComparaOrdenesCompra` que implemente la interfaz Comparator de Java de acuerdo con su precio de compra. **Ayuda:** considerar que se requiere hacer uso de una cola de prioridad máxima para almacenar este tipo de orden.

6. En la clase `MercadoAcciones` se implementarán los siguientes métodos:

    a. **Constructor por omisión** que inicializa las dos colas de prioridad.
    b. Método **simular**(**String**) que simula la compra y venta de acciones. Recibe como parámetro el nombre del archivo con las órdenes de compra y venta. Procesa cada orden para tratar de completar transacciones.
    Al terminar de leer el archivo, muestra en pantalla lo siguiente:

        - Las transacciones realizadas, con el siguiente formato:
        NombreUsuario1 compra a NombreUsuario2 acciones:# precio: $

        Donde *NombreUsuario1* corresponde al nombre de usuario de la orden de compra, y *NombreUsuario2* corresponde al nombre de usuario de la orden de venta que se completó. # es el número de acciones compradas, y *$* el precio por acción.

        - Las órdenes de compra y venta que quedaron pendientes en orden ascendente.

    Nota: Es posible utilizar todos los métodos privados necesarios para la implementación de la clase.

7. Ejecutar el programa `PruebaMercadoAcciones` para probar la implementación realizada. Si el programa funciona adecuadamente se verán los siguientes mensajes:

```
Yazmin compra a Liliana acciones:5 precio: $50
Rosario compra a Ernesto acciones:4 precio: $80
Rosario compra a Ariadna acciones:1 precio: $80
Rod compra a Juan acciones:3 precio: $100
Lara compra a Yazmin acciones:1 precio: $96
Lara compra a Manuel acciones:2 precio: $96
Lara compra a Juan acciones:1 precio: $96

----Ordenes de Venta pendientes:
Juan acciones:1 precio: $95

----Ordenes de Compra pendientes:
Nami acciones:4 precio: $81
Angie acciones:1 precio: $80
```