Universidad de los Andes

Ingeniería de Sistemas y Computación
ISIS1206 – Estructura de Datos
Taller 5

Taller 5: Colas de Prioridad

Objetivos

- Analizar y comprender las operaciones críticas de una cola de prioridad implementada mediante un Heap, así como su complejidad temporal.
- Analizar un algoritmo de Heapsort.

Contexto

La Pizzería de los Andes cuenta con una promoción que consiste en que si un pedido se demora más de 30 minutos en llegar a su lugar de destino, el pedido es gratis.

Este fin de semana la Pizzería tuvo un inconveniente técnico y cuenta con un sólo horno para cocinar sus pizzas. La idea es ganar la mayor cantidad de dinero y para esto se ha decidido tomar los pedidos de acuerdo al precio de cada uno (Pedidos Recibidos), de tal forma que el pedido con mayor prioridad sea el de mayor precio y luego, para despachar estos pedidos, se ha decidido hacerlo del más cercano al más lejano (Cola de Despachos).

Tenga en cuenta que un pedido se ingresa de la siguiente forma:

<Primer Nombre> <Precio> <Cercanía (1-5)>

En la plantilla existen tres formas diferentes para recibir, atender y despachar un pedido, las cuales son: 1) Interfaz de usuario, 2) Ingreso manual y 3) Archivo de pruebas.

Instrucciones de desarrollo

1. Descargue la plantilla de laboratorio y abra el proyecto en Eclipse.
2. Lea la documentación de la interfaz **IHeap** que se encuentra en el paquete **taller.estructuras**.
3. Revise las clases del mundo **Pedido** y **Pizzeria** y comprenda sus métodos.

Nota: *NO modifique las signaturas de los métodos que ya existen. Tampoco modifique la clase Main.*

Lo que usted debe hacer

Parte 1 – Preparación (en casa)

La pizzería de los Andes desea *atender* los pedidos del más costoso al más económico y *despachar* los pedidos que ya tienen listos en orden del más cercano al más lejano del restaurante. Para esto, cada pedido tiene un precio y una cercanía de 1 a 5 al restaurante (donde 1 es la zona más cercana y 5 es la más lejana).

- Implemente los TODOs que se encuentran en las clases Pizzería y Pedido, de tal manera que se puedan agregar pedidos a la cola de prioridad y sacarlos de ella.
- Utilizar un max-heap que ordene de mayor a menor los pedidos recibidos por su precio. Para esto debe crear la clase Heap, la cual debe usar objetos T que sean Comparables.

Parte 2 – Trabajo en clase

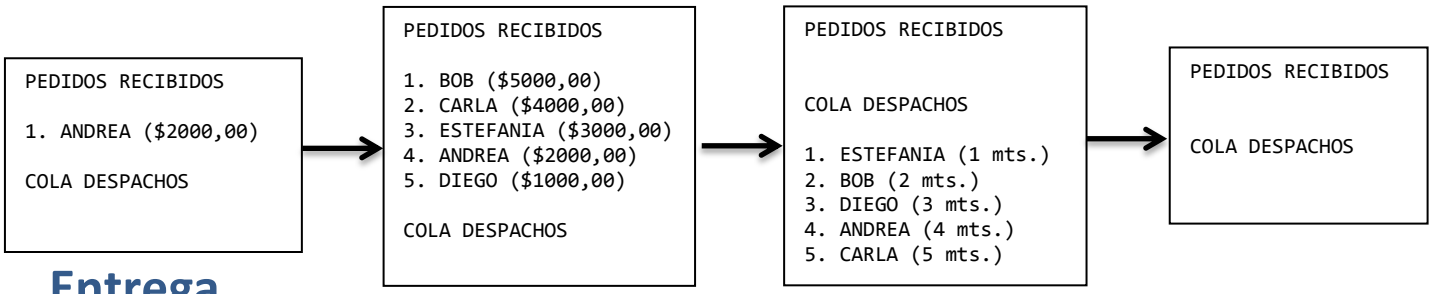
- Utilizar un min-heap que ordene de menor a mayor los pedidos ya atendidos por su cercanía. Para esto debe adicionar el min-heap a la clase Heap, la cual también debe usar objetos T que sean Comparables.
- Modifique las clases Pizzería y Pedido para que soporten este nuevo requerimiento, de tal forma que una vez sea recibido un pedido utilice la estructura max-heap ya creada y luego cuando este pedido sea atendido utilice la estructura min-heap previamente creada.
- Haga la estimación de complejidad y justifíquela para las operaciones de agregar elemento, retornar el elemento máximo/mínimo, retirar el elemento máximo/mínimo, mover el último elemento arriba en el árbol y mover la raíz abajo en el árbol del heap.
- Además explique el funcionamiento y analice la complejidad de los algoritmos para ordenar los pedidos por precio y por cercanía que implementó con el heap.

Verificación de funcionamiento

Una vez terminado el trabajo en casa y el trabajo en clase se espera que pueda ejecutar la opción tres (3) “Archivo de pruebas” del menú, la cual realizará las siguientes acciones:

1. Se leerá un archivo de texto (tests.txt).
2. Recibirá los pedidos que ahí se encuentran.
3. Atenderá los pedidos del más costoso al más económico.
4. Despachará los pedidos del más cercano al más lejano.

Ejemplo visual de consola, recuerde que existe una vista por cada ingreso de pedido, atención de pedido y despacho de pedido; es decir se recibe uno (1) por uno (1) hasta llegar al límite, se atienden uno (1) por uno (1) hasta llegar al límite y se despacha uno (1) por uno (1) hasta llegar al límite:



Entrega

1. Verifique que su proyecto cumple con los requisitos de la entrega de talleres.
2. Cree un archivo *estimation.txt* con la estimación de complejidad de las operaciones e inclúyalo en la carpeta ***data*** del proyecto.
3. Entregue su taller por medio de **BitBucket**. Recuerde, si su repositorio no tiene el taller o está vacío, su taller no será calificado por más de que lo haya desarrollado.