Departamento de Programación
Facultad de Informática
Universidad Nacional del Comahue

Estructuras de Datos
Trabajo Práctico Final 2020

EDAT Viajes.com

Se desea desarrollar un sistema para consulta y venta de viajes aéreos. Deberá guardar información de los aeropuertos, de los cuales almacenará el nombre aeronáutico formado por tres letras, que es único (Ej: NQN para Neuquén, AEP para Aeroparque de la Ciudad de Buenos Aires, etc), ciudad y número de teléfono. Además, se guardará información de los vuelos entre aeropuertos, indicando para cada caso la duración estimada en minutos.

En otra estructura se almacenará información de los vuelos, cuya clave es un código conformado por dos letras que identifican a la compañía y un número de 4 dígitos que identifica al vuelo (Ej: AA2279 = es el vuelo 2279 de Aerolíneas Argentinas), aeropuerto de origen y destino, hora de salida y hora de llegada, y una lista de los vuelos de cada día, que guarda fecha, cantidad de asientos totales y cantidad de asientos vendidos.

Por otro lado, se guardará información de los clientes, incluyendo tipo y número de DNI (que conformarán la clave única), nombre y apellido, fecha de nacimiento, domicilio y número de teléfono.

Además, en un mapeo uno a muchos se guardará para cada cliente, los pasajes aéreos comprados, indicando el vuelo, fecha, número de asiento y estado (pendiente, cancelado, volado).

Para la implementación se deberá utilizar un grafo etiquetado para el mapa de aeropuertos, con etiqueta de tipo numérico. Los clientes y vuelos se almacenarán en Tablas de Búsqueda implementadas con árbol AVL. Para el mapeo a muchos se podrá utilizar una implementación propia de hash abierto o HashMap de Java.

Desarrollar una clase EDATViajes con un menú de opciones para realizar las siguientes tareas:

1. ABM (Altas-Bajas-Modificaciones) de aeropuertos
2. ABM (Altas-Bajas-Modificaciones) de clientes
3. ABM (Altas-Bajas-Modificaciones) de vuelos
4. ABM (Altas-Bajas-Modificaciones) de pasajes
5. Consultas sobre clientes:
    - Dado tipo y Nro de DNI, verificar y mostrar información de contacto de un cliente, incluyendo un listado de los pasajes comprados pendientes de volar.
    - Dado un cliente, mostrar qué ciudades ha visitado, según su historial de vuelos.
5. Consultas sobre vuelos:
    - Dado un código de vuelo y una fecha, mostrar toda la información del mismo
    - Dados dos códigos de vuelo, mostrar todos los códigos existentes que están en el rango entre el menor y el mayor de ambos códigos (incluyendo ambos extremos)
6. Consultas sobre tiempos de viaje:

    - Dados dos aeropuertos A y B, mostrar si es posible que el cliente que parte del aeropuerto A llegue al B en como máximo X vuelos (recorrer lo mínimo indispensable)
    - Dados dos aeropuertos A y B, obtener el camino que llegue de A a B de menor tiempo de vuelo (recorrer lo mínimo indispensable)
    - (*) Dados dos aeropuertos A y B, obtener el camino que llegue de A a B pasando por la mínima cantidad de aeropuertos
    - (*) Dados tres aeropuertos A, B, y C, obtener el camino que llegue más rápido de A a B y que pase por el aeropuerto C
7. A efectos de realizar una promoción a clientes fieles, una vez por mes la agencia quiere un listado de los clientes que han comprado más pasajes, para ofrecerles un descuento especial. La tabla debe mostrarlos ordenados de mayor a menor según cantidad de pasajes comprados (puede utilizar alguna estructura de datos auxiliar que considere apropiada para asegurar la eficiencia del cálculo)
8. Mostrar sistema: es una operación de debugging que permite ver todas las estructuras utilizadas con su contenido (grafo, Diccionarios y Mapeo) para verificar, en cualquier momento de la ejecución del sistema, que se encuentren cargadas correctamente.

Requisitos importantes:

- El programa debe permitir la ejecución por separado de cada una de las operaciones especificadas.
- El programa debe ser eficiente: Debe recorrer las estructuras sólo lo necesario y haciendo buen uso de la memoria.
- Las estructuras deben estar implementadas de forma genérica para elementos de tipo Object o Comparable de Java, según el propósito de la estructura.
- La **carga inicial** del sistema debe hacerse a partir de un archivo de texto con el formato indicado. Ejemplo, la información se puede ingresar de a una por línea con una identificación inicial para reconocer el tipo (A: Aeropuerto, V: Vuelo, C: Cliente, P: pasajes), o en archivos diferentes
    - A: NQN, Neuquén, Neuquén,0299-4490800, …, …, …
    - V: AA2279, NQN, AEP, 8:00, 9:30, …
    - C: DNI, 34.456.678, José, García, ...
    - P: AA2279, 20200520, 24, pendiente, ...
- Guardar **un archivo de LOG** (archivo de texto) para guardar la siguiente información:
    - Estado del sistema al momento de terminar la carga inicial (todas las estructuras)
    - Listado de operaciones de ABM que se realizan a lo largo de la ejecución (Ej: “Se crea el aeropuerto X”, “Se eliminó el cliente Y”, “El cliente XX compró un pasaje XXX”, etc)
    - Estado del sistema al momento de terminar de ejecutarse (todas las estructuras incluyendo la tabla de posiciones)

Condiciones y fechas de entrega:

- El programa debe realizarse de manera individual y debe presentarse personalmente a alguno de los docentes que indicarán si está aprobado. Luego deberá subirlo a la plataforma PEDCO para su archivo.
- Al momento de la defensa, deberá presentar un dibujo en papel del mapa de aeropuertos (grafo) y las tablas de clientes y vuelos (AVL) subido en la carga inicial desde archivo de texto.
- Los estudiantes que promocionan la materia tendrán tiempo para entregarlo hasta el **viernes 21 de agosto** pero NO DEBEN realizar los módulos marcados con (*)
- Los estudiantes que no promocionan podrán entregarlo y aprobarlo en cualquier momento, como mínimo 2 semanas antes de presentarse a rendir el final regular.