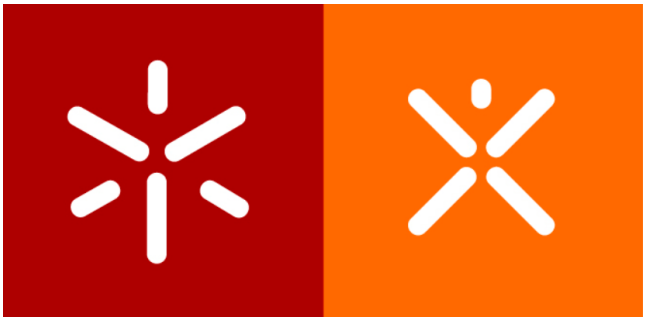

Universidade do Minho

Departamento de Informática

Licenciatura em Engenharia Informática

Laboratórios de Informática 3

Grupo 55 - Fase 1

Elementos do Grupo:

Guilherme Rio a100898

Diogo Cunha a100481

Rui Cerqueira a100537

Índice

Introdução

Este trabalho tem como objetivo a criação de código em C capaz de organizar e armazenar estruturas de dados complexas, mais especificamente, de planeamento de viagens, sendo necessária ainda a implementação de ferramentas capazes de realizar operações predefinidas sobre esses dados (queries).

Módulos e estruturas de dados

- **Users.c –** Este módulo é responsável pela formatação do utilizador e por inserir o mesmo na sua respetiva estrutura de dados, optamos por utilizar uma hashtable para armazenar os utilizadores devido à necessidade frequente de acesso aos dados baseados no nome de utilizador. O nome de utilizador serve como chave única para cada utilizador na hashtable, facilitando um acesso rápido e eficiente às informações de cada utilizador.

- **User_stats.c –** Este módulo é responsável por guardar as "stats" necessárias para a resposta posterior das queries, para isso criamos uma hashtable com o nome do utilizador como chave para rápido acesso as stats de um utilizador, dentro desta temos guardamos o seu número de reservas, o seu número de voos e o seu total gasto. Decidimos também guardar a sua lista de voos e lista de reservas, para estas decidimos usar Listas Ligadas em que cada nodo iriamos guardar os voos/reservas do utilizador.

- **Flights.c –** Este módulo é responsável pela formatação dos voos e por inserir os mesmos na sua estrutura de dados, utilizados uma hashtable para armazenar os dados por frequentemente procuramos voos a partir do seu id, por isso utilizamos o seu id como chave

- **Airport_stats.c –** Este módulo é responsável por guardar as "stats" necessárias para os aeroportos, para isto utilizamos uma hashtable com o id do aeroporto sendo a sua localização (ex: MAN), guardamos também o seu número de passageiros por ano, um array contendo todos os atrasos, o número de voos, e uma Lista Ligada que contem a lista de todos os voos.

- **Passengers.c –** Este módulo é responsável pela formatação dos passageiros e por inserir o mesmo na sua estrutura de dados, para esta estrutura optamos por um array.

- **Reservations.c –** Este módulo é responsável pela formatação das reservas e por inserir as mesmas na sua estrutura de dados, para esta estrutura decidimos usar uma hashtable pois procuramos pelas reservas pelo seu id constantemente durante o projeto.

- **Hotel_stats.c –** Este módulo é responsável por guardar as "stats" acerca dos hotéis, para este usamos uma hastable com o id do hotel como chave, o número de reservas, a lista de reservas numa Lista Ligada, e guardamos também, o seu averageScore, e a sua soma dos ratings(usados para calcular o avgScore).

- **Stats.c –** Este módulo é responsável pela criação das estruturas de dados para as stats.

- **Validation.c –** Este módulo é responsável pela validação de utilizadores, voos e reservas.

- **Handle.c –** Este módulo é responsável por ler as linhas do input e executar as respetivas queries.

- **Utils.c –** Contem funções auxiliares necessárias ao longo do programa

Queries

- Querie 1 – Nesta querie temos 3 tipos de input
    - Para o input em que recebemos o id de uma reserva, começamos por procurar o id na hashtable correspondente e verificar se o mesmo existe, e após isso imprimimos os seus dados através de getters definidos nos respetivos módulos.
    - Para o input em que recebemos o id de um utilizador, começamos por procurar o id na hashtable correspondente e verificar se o mesmo existe, depois verificamos se a sua conta é ativa e caso seja, ainda temos de verificar se foram criadas as stats para o utilizador pois elas podem não ser criadas caso o utilizador não tenha nem voos nem reservas, e depois fazemos os respetivos prints.
    - Para o input em que recebemos o id de um voo, verificamos se o mesmo existe na sua hashtable, e imprimimos os seus respetivos dados.

- Querie 3 – Nesta querie apenas nos pedem o average rating de um hotel, o qual já temos guardados nas stats do hotel, ou seja, só temos de verificar se existem stats para esse hotel, e efetuar os prints respetivos.

- Querie 4 – Para esta querie começamos por verificar se existem stats para o hotel, e após isso efetuamos o sort da lista de reservas presentes nas stats, e efetuamos os prints dos dados necessários.

- Querie 5 – Para esta querie começamos por retirar as aspas presentes no input para conseguirmos efetuar a comparação entre datas, depois criamos uma Lista Ligada que será freed no final da função onde guardamos os voos que cumprem os requisitos a querie, e efetuamos os respetivos prints no final.

- Querie 6 – Para esta querie começamos por organizar a lista dos stats que criamos para os aeroportos pelo número de passageiros, e depois usamos um ciclo while para efetuar os prints enquanto uma variável i que começa com valor 1 é >= ao valor de N (top N).

- Querie 7 – Para esta querie começamos por organizar a lista dos stats dos aeroportos pela sua mediana e depois efetuamos o ciclo while como na querie 6 para efetuarmos prints apenas do top N medianas.

Conclusão

Ao longo da realização desta primeira fase fomos confrontados várias vezes com obstáculos inesperados, apesar disto, consideramos ter realizado uma primeira fase satisfatória, que cumpre os principais objetivos desta primeira fase, cujas complicações nos ajudarão a melhorar o trabalho antes da sua apresentação final.

Nesta primeira fase o nosso trabalho falha na parte de encapsulamento e modularidade, no entanto é algo que pretendemos resolver completamente na próxima fase do trabalho.