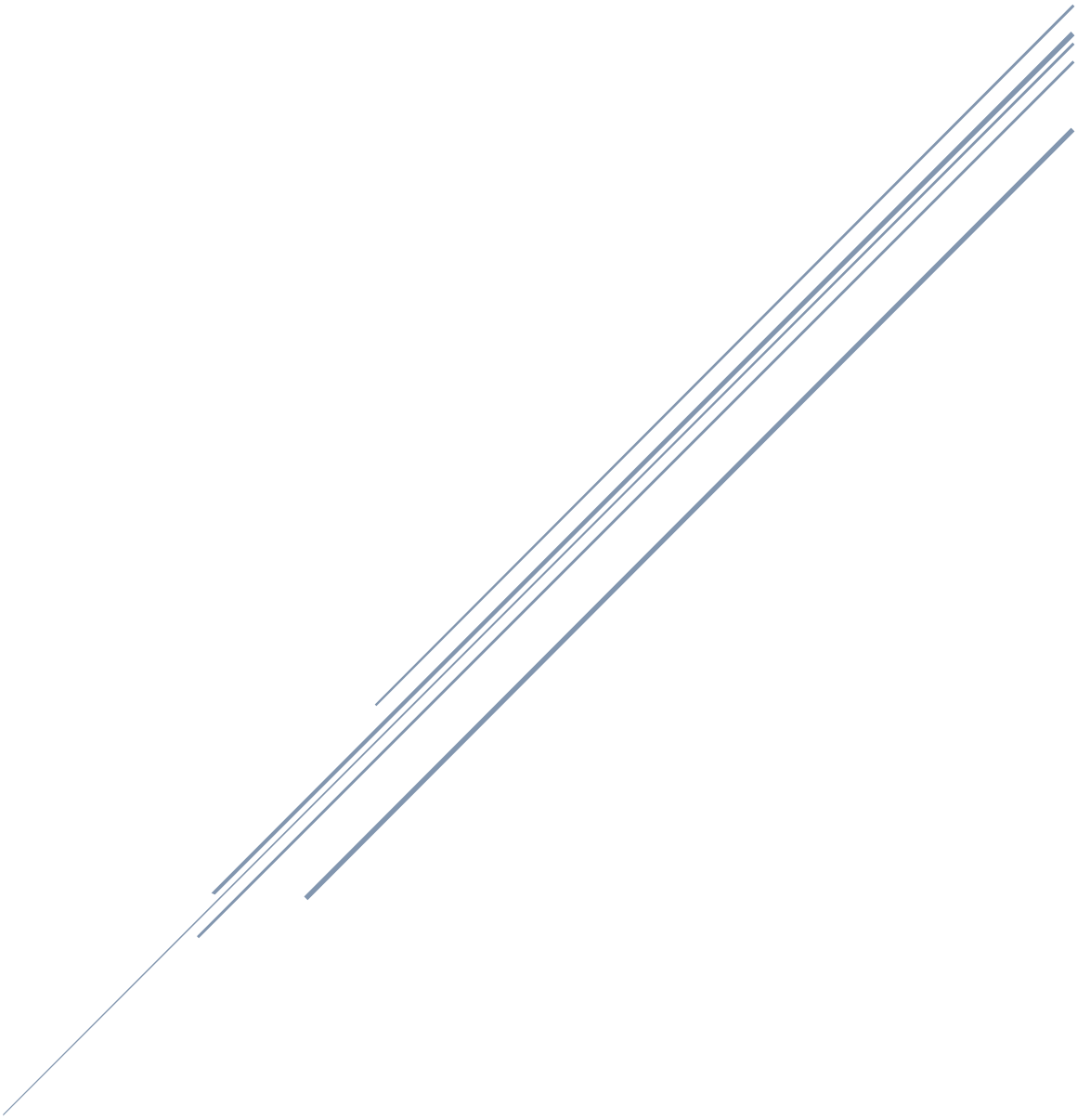
LABORATÓRIOS DE INFORMÁTICA 3

Grupo 2

Nuno Ribeiro (a104177)	Rodrigo Lopes (a104440)	João Costa (a104258)



Universidade do Minho
Licenciatura em Engenharia Informática

Conteúdo

Introdução

Este projeto tem como objetivo organizar dados de 4 ficheiros que nos foram fornecidos (utilizadores, voos, passageiros e reservas). Depois disso, foram nos atribuídas um conjunto de *queries*, onde tivemos de apresentar e organizar os dados da maneira correta que cada *query* pedia.

O projeto foi feito na linguagem C, na qual tivemos de encontrar o modo mais eficiente de lidarmos com a grande quantidade de informação que nos foi atribuída nos ficheiros .CSV.

Na segunda fase do projeto, como o dataset é bastante superior ao da primeira fase, tivemos de fazer algumas alterações a nível da sua estrutura. Primeiro, adicionámos catálogos para os 4 tipos (utilizadores, voos, passageiros e reservas) e os respetivos ficheiros de cada um dos tipos. Para além disso, em vez de fazermos extern a partir do header file, criámos função de get e set para cada um dos parâmetros que precisávamos de modo a manter o encapsulamento e a opacidade do código.

Na primeira fase, utilizámos arrays para guardar os dados de cada um dos tipos, mas nesta fase optámos por estruturas mais eficientes para uma maior quantidade de dados, como por exemplo, Hash Tables e Àrvores binárias.

Arquitetura do Projeto

```
trabalho-pratico
|- include
|  |- ...
|- src
|  |- ...
|- Resultados
   |- ...
|- relatorio-fase1.pdf
|- relatorio-fase2.pdf
|- programa-principal (depois do make)
|- programa-testes (depois do make)
```

Esta foi a estrutura do projeto, que foi mantida ao longo das suas alterações:

- Na pasta ***include*** estão guardados todos os ficheiros .h (ficheiros *header*).
- Na pasta ***src*** é onde estão localizados todos os ficheiros .c.
- Na pasta ***resultados*** foram guardados os ficheiros .csv com os erros (utilizadores, voos, passageiros e reservas) e com os outputs respetivos a cada *query*.

Para além destas pastas temos o ficheiro executável **programa-principal** onde vai ser executado todo o código (validação, *parsing* e *queries*).

Ficheiros Fornecidos

Utilizadores

Para armazenar os dados relativos aos utilizadores recorremos ao uso de arrays dinâmicos. Criámos uma estrutura de dados (um *array* de *structs*) onde foram armazenados individualmente todos os dados relativos aos utilizadores.

Primeiro fizemos uma função que validava os dados fornecidos, criando um ficheiro somente com os erros e outro sem nenhum. Por fim, criámos um módulo para percorrer linha a linha o ficheiro dos utilizadores (sem erros), guardando cada informação respetiva aos utilizadores, nas respetivas variáveis das estruturas de dados.

Essa informação foi posteriormente usada para a execução das *queries*.

❖ **Users**

- ID
- Name
- Email
- Phone Number
- Birth Date
- Sex
- Passport
- Country Code
- Address
- Account Creation
- Payment Method
- Account Status
- Age

Voos

Para armazenar os dados relativos aos utilizadores recorremos ao uso de arrays dinâmicos. Criámos uma estrutura de dados (um *array* de *structs*) onde foram armazenados, individualmente, todos os dados relativos aos utilizadores.

Primeiro fizemos uma função que validava os dados fornecidos, operando da mesma forma que a utilizada para os utilizadores. Por fim, utilizamos o mesmo método dos utilizadores para armazenar cada elemento na respetiva variável localizada na estrutura de dados.

Essa informação foi posteriormente usada para a execução das *queries*.

- **Flights**
  - ID
  - Airline
  - Plane Model
  - Total Seats
  - Origin
  - Destination
  - Scheduled Departure Date
  - Scheduled Arrival Date
  - Real Departure Date
  - Real Arrival Date
  - Pilot
  - Copilot
  - Notes

Passageiros

Para armazenar os dados relativos aos utilizadores recorremos ao uso de arrays dinâmicos. Criámos uma estrutura de dados (um *array* de *structs*) onde foram armazenados individualmente todos os dados relativos aos utilizadores.

Começando, igualmente aos anteriores, pela mesma separação de informação. Por fim, outra vez na semelhança, usamos o mesmo método, já que as bases das estruturas de dados são idênticas.

Essa informação foi posteriormente usada para a execução das queries.

- **Passengers**
  - Flight ID
  - User ID

Reservas

Como anteriormente, usamos a mesma forma que usamos para os utilizadores de armazenar os dados fornecidos, apenas a execução é separada para ser utilizado aquilo estritamente necessário para cada função.

- **Reservations**
  - ID
  - User ID
  - Hotel ID
  - Hotel Name
  - Hotel Stars
  - City Tax
  - Address
  - Begin Date
  - End Date
  - Price Per Night
  - Includes Breakfast
  - Room Details
  - Rating
  - Comment

Lógica das queries

Nas queries, optámos por criar uma data structure para cada query, de modo a ser o mais eficiente possível e não precisarmos de guardar dados desnecessários (nos catálogos) para cada query.

Escolhemos usar Hash Tables e Árvores binárias, porque foi o que nos pareceu mais eficiente para os problemas. Usámos Hash Tables, quando as queries pediam para aceder a um certo campo de um tipo (p.e. Nas reservas, aceder ao campo “rating”), pois essa busca é feita em tempo constante O(1), logo é a maneira mais eficiente de o fazer. Para as outras queries que pediam para ordenar certos tipos de dados, escolhemos utilizar Árvores binárias, porque em comparação às Hash Tables que demoram O(n) para ordenar, as árvores demoram O(log(n)), que para um dataset maior, diminui bastante o tempo de execução do projeto gasto para essa query.

Conclusão

De uma forma geral, o nosso projeto está bem construído, foi feito de uma forma eficiente e está bem organizado, sendo que é fácil fazer todo o tipo de alterações no código sem alterar a estrutura do projeto.

Conseguimos atingir os objetivos principais do projeto para esta fase que foi a modularidade e o encapsulamento das estruturas utilizadas e a utilização de estruturas mais eficientes (para o dataset grande).

O único problema que tivemos foi na lógica de guardar os passageiros para serem utilizados nas queries.

Concluindo, percebemos a intenção deste projeto que foi aprender a organizar os dados em estruturas e saber fazê-lo da forma mais eficiente possível, utilizando por exemplo Hash Tables e Árvores binárias.