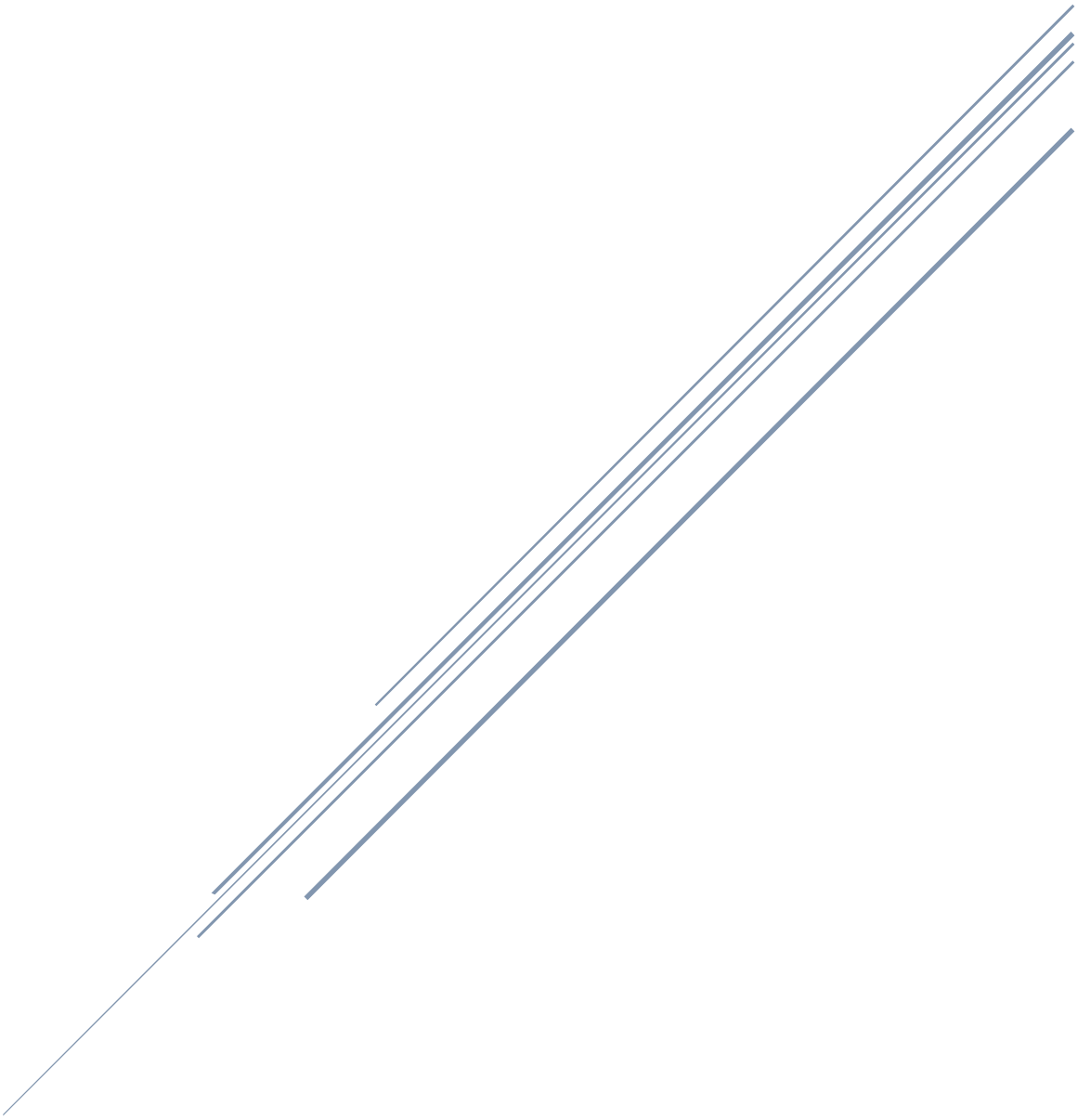
LABORATÓRIOS DE INFORMÁTICA 3

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.

Inicialmente tentamos fazê-lo com *arrays* estáticos, mas percebemos que não iria resultar porque o tamanho dos *arrays* não iria ser sempre o mesmo e íamos acabar por utilizar mais memória do que aquela que realmente precisaríamos para organizar os dados.

Posteriormente, surgiu a ideia de utilizar *arrays* dinâmicos onde tínhamos de alocar o espaço estritamente necessário para cada *array*, e recorremos também a estruturas de dados (*Data Structures*) para os dados fornecidos serem organizados de uma maneira simples e de fácil compreensão.

No final, chegamos a uma estrutura de organização satisfatória, pois separamos a execução das *queries* em respetivos ficheiros .c, tal como o *parsing* dos dados (utilizadores, voos, passageiros e reservas) - o que facilitou imenso a compreensão e organização do código, bem como a facilidade de corrigir bugs ou erros de execução que fossem aparecendo.

Recorremos ao uso do *Valgrind* e do *GDB*, de modo a conseguirmos eliminar as *memory leaks* (Valgrind), reduzir o tempo de execução do programa (*Valgrind*) e resolver *segmentation faults* que fossem aparecendo (*GDB*).

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

Conclusão

De uma forma geral, o nosso projeto está bem construído, foi feito de uma forma eficiente (sendo, porém, possível melhorar o tempo de execução, algo que gostaríamos de explorar na segunda fase) e está bem organizado, sendo que é fácil fazer todo o tipo de alterações no código sem alterar a estrutura do projeto.

Conseguimos atingir um dos objetivos principais do projeto que foi a execução do **programa-principal** sem *memory leaks* - erro bastante comum se alguma vez alguém se esquecer de libertar memória alocada dinamicamente a alguma variável.