

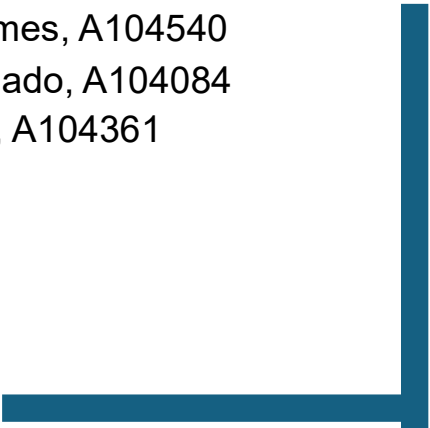


Universidade Do Minho

Relatório 2 Fase

Laboratórios de Informatica III

Grupo 49

- Pedro Miguel Araújo Gomes, A104540
 - João Nuno Pereira Machado, A104084
 - Filipe Teixeira Viana, A104361
- 

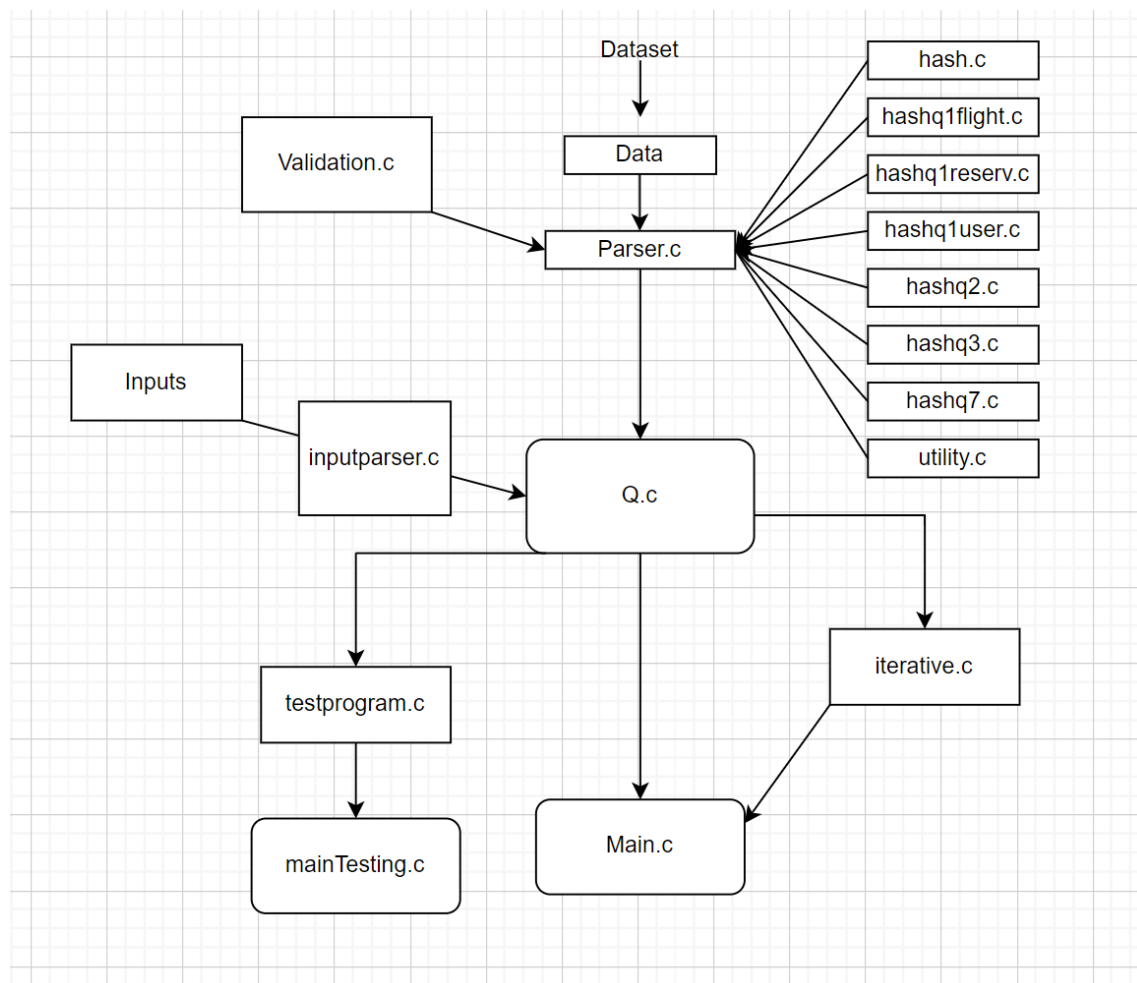
Introdução

Este documento visa abordar o projeto correspondente à UC de Laboratórios de Informática III no ano letivo 2023/2024.

Nesta segunda etapa do projeto, os objetivos do trabalho incluíram a implementação do modo interativo, melhoria e realização das restantes queries, realização de um ficheiro de testes, adaptação do projeto a um dataset de grandeza superior, gestão de memória e fugas de memória, otimização de queries, gestão do tempo de execução do programa, documentação e organização do código

Neste relatório vamos abordar um pouco sobre a realização das queries, o raciocínio por nós utilizado na realização das mesmas, algumas dificuldades na realização do projeto e aspetos que pretendemos melhorar no futuro.

Arquitetura Geral:



Query 1: Na Query1 foram usadas Tabelas Hash close addressing dedicadas a cada tipo de dados, onde as chaves foram os ids de Users, os ids de Reservas e os ids de Flights.

Query 2: Na Query2 foram usadas Tabelas Hash close addressing com listas ligadas como conteúdo, onde a chave era o “user_id” e listas ligadas com strings formato “id;date” que foram ordenadas.

Query 3: Na Query3 foi usada 1 Tabela Hash close addressing com listas ligadas como conteúdo onde a chave era o “hotel_id” que nos levava aos “ids” referentes a reservas que eram chaves na tabelas hash dedicada as reservas. Os ratings e o número de ratings coletados eram acumulados para poder realizar o quociente correspondente ao rating médio.

Query 4: Na Query 4 foram usadas tabelas hash closing addressing com listas ligadas como conteúdo onde a chave era o “hotel_id” que nos levava aos “ids” referentes a reservas que eram chaves na tabelas hash dedicada as reservas. Essas reservas eram guardadas em arrays e posteriormente reordenadas segundo os critérios do enunciado prático.

Query 5: Na Query 5 era o “hotel_id” que nos levava aos “flight_ids” referentes a voos que eram chaves na tabelas hash dedicada aos voos. Se a data prevista para a partida do voo estiver contida no intervalo de tempo dado pelo input então esses voos são guardados num array e posteriormente segundo os critérios do enunciado prático.

Query 6: Na Query6 foram usadas Tabelas Hash com listas ligadas onde as strings eram do formato “airport;numberofpassengers” onde “airport” será um aeroporto correspondente a “origin”, “destination” ou ambos e o “numberofpassengers” foi calculado pela soma do número de passageiros que passaram por cada aeroporto.

Query 7: Na Query7 foram usadas Tabelas Hash close addressing com listas ligadas como conteúdo onde a chave era a “origin” ou seja, o aeroporto de partida. Ao percorrer esta tabela calculamos a mediana de atrasos de cada aeroporto e guardamos cada uma dessas medianas num array que vamos depois reordenar de forma decrescente, como pede o enunciado prático.

Query 8: Na Query 8 foram usadas tabelas hash close addressing com listas ligadas como conteúdo, onde utilizamos o “hotel_id” para selecionar o nodo da tabela cujo vamos percorrer a lista ligada, lista ligada essa que contém os ids das reservas onde

iremos verificar quantas noites em comum existem em relação ao intervalo de datas dado pelo input. Multiplicamos as noites em comum pelo preço por noite da reserva de forma a obter a receita total do hotel para o intervalo de datas anteriormente referido.

Query 9: Na Query 9 foi usada a tabela hash close addressing dedicada aos users. Ao percorrer esta tabela, depois de ter removido possíveis aspas dos inputs utilizados, verificamos se o input é prefixo do nome de utilizador que estamos a percorrer. Se sim, guardamos o id e o nome do utilizador num array que iremos posteriormente reordenar alfabeticamente como manda o enunciado prático.

Modo Interativo:

```
joaonpm@Ubuntu:~/grupo-49/trabalho-pratico$ ./programa-principal
Please, present the data/path you'd like to access: dataset/data_clean

      WELCOME TO OUR APP
    Queries: 1 2 3 4 5 6 7 8 9
CHOOSE WHICH QUERY YOU'D LIKE TO CONSULT: 7
CHOOSE "N" FOR NORMAL DISPLAY AND "F" FOR BATCH DISPLAY: F
Please, choose the "N" to check the TopN medians of delay of the existent airports: 5
--- 1 ---
name: IST
median: 900

--- 2 ---
name: LIS
median: 900

--- 3 ---
name: VIE
median: 900

--- 4 ---
name: WAW
median: 450

--- 5 ---
name: AMS
median: 300
```

Programa de Testes:

```
0 input nº3 é igual.
0 input nº4 é igual.
0 input nº5 é igual.
0 input nº6 é igual.
0 input nº7 é igual.
0 input nº8 é igual.
0 input nº9 é igual.
0 input nº10 é igual.
0 input nº11 é igual.
0 input nº12 é igual.
0 input nº13 é igual.
0 input nº14 é igual.
0 input nº15 é igual.
0 input nº16 é igual.
0 input nº17 é igual.
0 input nº18 é igual.
0 input nº19 é igual.
0 input nº20 é igual.
0 input nº21 é igual.
0 input nº23 é igual.

Memory usage of the program: 120872 KB

Elapsed time of the whole program: 0.584 seconds
Executed: 9 queries.

CPU time used threw the whole program: 0.556 seconds

CPU time used parsing all the information into the data structures : 0.3198 seconds
CPU time used executing Query 1 multiple times : 0.0071 seconds. Called 36 times.
CPU time used executing Query 2 multiple times : 0.0695 seconds. Called 12 times.
CPU time used executing Query 3 multiple times : 0.0149 seconds. Called 6 times.
CPU time used executing Query 4 multiple times : 0.0172 seconds. Called 6 times.
CPU time used executing Query 5 multiple times : 0.0014 seconds. Called 6 times.
CPU time used executing Query 6 multiple times : 0.0401 seconds. Called 6 times.
CPU time used executing Query 7 multiple times : 0.0011 seconds. Called 4 times.
CPU time used executing Query 8 multiple times : 0.0040 seconds. Called 10 times.
CPU time used executing Query 9 multiple times : 0.0164 seconds. Called 4 times.
joaonpm@Ubuntu: ~/grupo-49/trabalho-pratico$
```

Dificuldades: Os memory leaks e a gestão da memória foram grandes problemas no nosso projeto. Conseguimos reduzir os leaks porém não conseguimos atingir o nosso objetivo. Trabalhar com o dataset grande também foi um desafio que nos fez ver que as nossas estruturas não eram as melhores e tivemos de fazer várias alterações e testes.

Podemos Melhorar?

Podemos sempre melhorar, especialmente no critério de modularidade e encapsulamento, melhorar as nossas estruturas e a nossa gestão de memória.

