



Universidade do Minho
Escola de Engenharia

UNIVERSIDADE DO MINHO
DEPARTAMENTO DE INFORMÁTICA
LABORATÓRIOS DE INFORMÁTICA III

Bruno Alexandre Martins Carvalho (a89476)
Luís Alberto Barreiro Araújo (a96351)
Marisa Ferreira Soares (a92926)
7 de fevereiro de 2022

Conteúdo

1	Introdução	1
2	Solução técnica	2
2.1	Implementação do Mecanismo de interação	2
2.2	Implementação de Testes funcionais e de Desempenho	2
2.3	Implementação da Gestão de dados	2
3	Resultados	3
3.1	Mecanismo de interação	3
3.2	Testes funcionais	5
3.3	Tempos de execução	6
4	Conclusão	7

Introdução

Este relatório foi realizado no âmbito da terceira parte do projeto de grupo da unidade curricular de *Laboratórios de Informática III* e tem como principais objetivos:

- A interatividade do utilizador com o programa.
- A implementação de métodos essenciais de C, tal como, modularidade, encapsulamento, estruturas dinâmicas de dados, e medição de desempenho.
- Utilização de ferramentas de desenvolvimento de projetos em grande escala em C.
- Utilizar um formato de paginação para interatividade com o utilizador.
- Elaboração de um módulo de testes com o intuito de ser uma introdução à testagem com implementação concreta.

Solução técnica

Nesta fase inicial começamos por corrigir alguns problemas de modularidade e encapsulamento de modo a melhorar essa parte, em relação ao guião 2.

2.1 Implementação do Mecanismo de interação

Para o mecanismo de interação, sempre que não é recebido um ficheiro de comandos como argumento, construímos um menu de interatividade com o utilizador. São apresentadas dez opções, que correspondem a todas as *queries*. O utilizador escolhe um número e é lhe assim apresentado o output dessa *query*. No caso da *query* necessitar de vários argumentos (*queries* parametrizáveis), estes também são solicitados ao utilizador antes da execução. Nomeadamente a partir da *query* 5, cujo output é uma lista extensa, existe uma paginação, para que o utilizador possa navegar por páginas. Este pode avançar uma página ("p"), regressar à página anterior ("a") e também pode avançar para uma página específica se indicar o número que pretende ("s n" em que "n" é a página para a qual o utilizador pretende saltar).

2.2 Implementação de Testes funcionais e de Desempenho

Em relação ao testes funcionais, executamos o programa normalmente, onde são criados ficheiros de saída para cada *query*, para que sejam comparados com ficheiros pré-definidos ("expected_fileX.txt") criados anteriormente. Estes são resultado do output obtido da execução de cada *query* previamente assim como referido na *FAQ*. Sendo assim, para cada *query* é comparado o conteúdo de cada ficheiro, caractere a caractere, de output e expected, assim como é verificado se o tempo de execução é inferior a um limite mínimo para elaboração das *queries*. Por fim é apresentada uma frase para cada *query*. Por exemplo, se o tempo de execução é inferior a 5 segundos e os ficheiros são iguais, aparece a frase "*Teste válido*". Se o tempo de execução for válido, mas se os ficheiros forem diferentes, a frase apresentada é "*Tempo válido, mas testes incorretos*". Caso os ficheiros sejam iguais e o tempo exceda o esperado, "*Testes corretos, mas excedeu o tempo*". No caso de exceder o tempo e os ficheiros serem diferentes, a mensagem apresentada é "*Teste inválido*".

2.3 Implementação da Gestão de dados

No contexto de gestão de dados, ficamos apenas por criar um modulo de gestão *sg.c* que inicializa os catálogos e tem uma estrutura onde guarda os catálogos em si, este método é feito de modo a não ter sempre todas as listas a passar de um lado para o outro pois ficaria muito mais custoso ao programa, no entanto no que remete à utilização dos métodos referidos pelos professores de criação de ficheiros intermédios para a gestão dos dados não foi elaborado nesse sentido o que acaba consequentemente por tornar mais custoso na memória do utilizada. No caso do nosso trabalho, de modo a melhorar o esforço do programa para ficheiros tão grandes guarda-mos, no caso do ficheiro de *commits*, apenas o tamanho das mensagens, uma vez que o seu conteúdo não é necessário a nenhuma *query* existindo assim menos informação em memória.

Resultados

3.1 Mecanismo de interação

Na imagem seguinte, está apresentado o menu de interação desenvolvido.

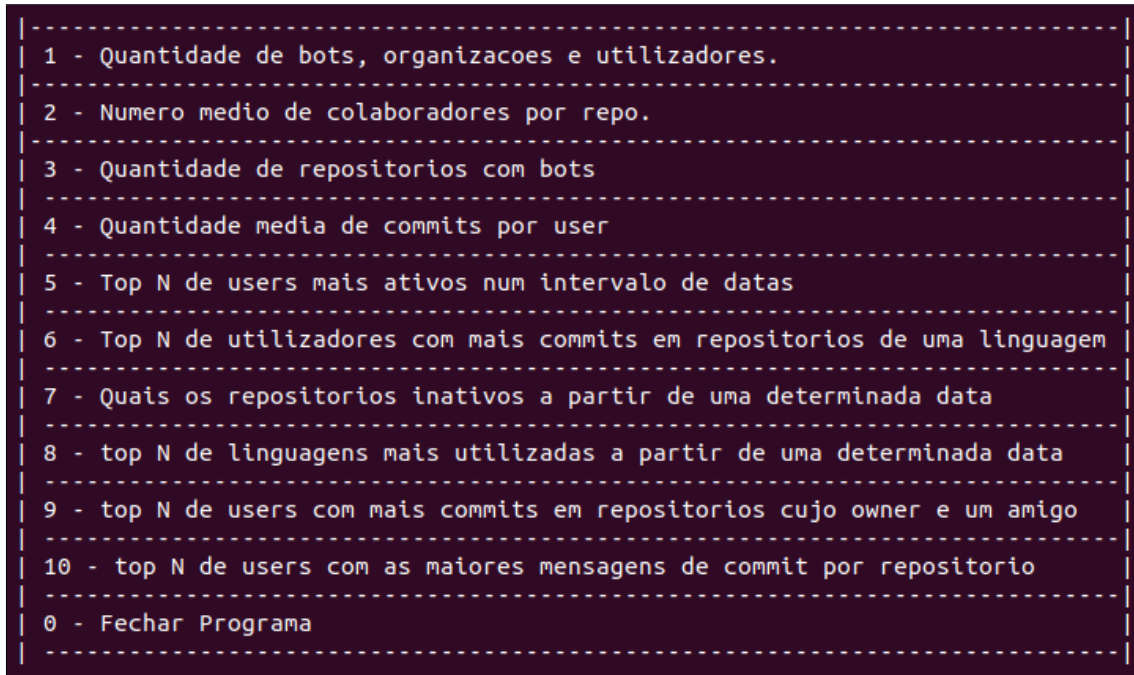


Figura 3.1: Menu de interação com o utilizador

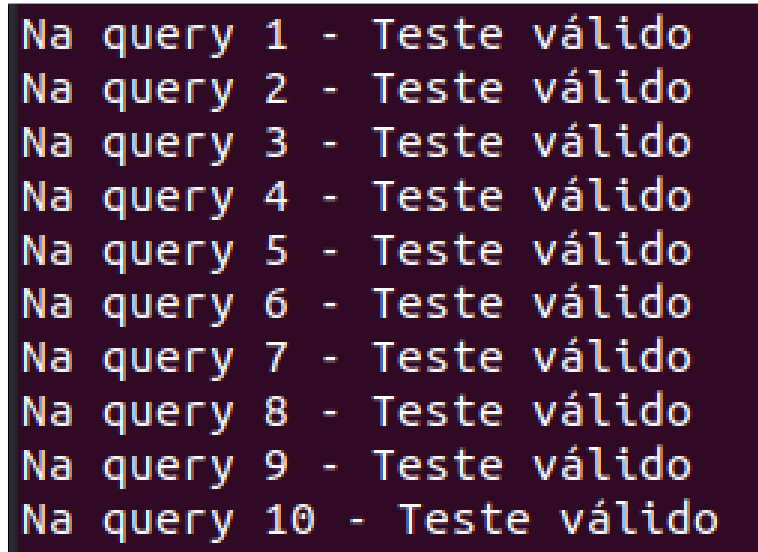
Na figura apresentada, podemos ver um exemplo de navegação por páginas. Em cada página são apresentadas 24 linhas, logo neste caso há 5 páginas para serem apresentadas. É importante referir que a disposição da informação não está na melhor formatação possível, sendo que não era um dos principais pontos deste projeto.

```
Escolha uma opcao: 5
Qual o total desejado de users?
100
Qual a data inicial (YYYY-MM-DD)?
2016-11-11
Qual a data final (YYYY-MM-DD)?
2017-11-11
| ID      |Login      |Commit_qtty1
26547456 |grishma-atanur |30
23735251 |mred5683 |30
10245674 |garkavenkov |30
23018401 |jekafortuna |30
20396460 |swielgus |30
30262420 |Axiuf |30
24638031 |rngchan |30
32073369 |ChloeBoudreault |30
29403806 |weihanchia |30
22873405 |VaDikOo8 |30
23111373 |White-CHN |30
15342275 |arpitp1912 |30
30784651 |retro0101 |30
23157616 |rkrsolutions |30
26254595 |DoHaiBinh |30
4186841 |pirate7radio |30
25015145 |patseev |30
25443248 |Martyn-Lewis |30
26253488 |pikuechan |30
23727061 |svscom |30
20914290 |igorko1235 |30
26490645 |Soporteases |30
27967350 |samdaw13 |30
31827273 |gtwark904 |30
      Pagina 1 de 5
p -> proxima pagina
a -> pagina anterior
s -> saltar para pagina
```

Figura 3.2: Exemplo de paginação

3.2 Testes funcionais

Na imagem seguinte, está apresentado o output resultante da execução dos testes.



```
Na query 1 - Teste válido
Na query 2 - Teste válido
Na query 3 - Teste válido
Na query 4 - Teste válido
Na query 5 - Teste válido
Na query 6 - Teste válido
Na query 7 - Teste válido
Na query 8 - Teste válido
Na query 9 - Teste válido
Na query 10 - Teste válido
```

Figura 3.3: Output resultante da execução dos testes

3.3 Tempos de execução

O tempo de execução de ler os três ficheiros *.csv* e armazenar os dados nos respetivos módulos de dados foi de 3.020746 segundos com os ficheiros do guião 2, com os ficheiros *.csv* do dataset 3 é de 14.328750 segundos. No guião 2, este tempo era de 2.67967 segundos. Como neste guião há a validação das linhas de cada ficheiro, assim como o tamanho extremamente maior dos ficheiros do guião 3 isto justifica o aumento de tempo.

Nas tabelas seguintes estão apresentados os tempos de execução de cada query, nos guiões respetivos, para que se possa ver a comparação entre ambos.

O hardware utilizado para os testes foi: processador *i5-8300H*, 16 *GiB* de memória *RAM* e *SSD*.

Query	Tempo (s)
Query1	0.164284
Query 2	0.061801
Query 3	0.132436
Query 4	0.000006
Query 5	0.203460
Query 6	0.219348
Query 7	0.219634
Query 8	0.096965
Query 9	0.217670
Query 10	0.161197

Tabela 3.1: Tempo de execução das queries no Guião 2

Query	Tempo (s)
Query1	2.154364
Query 2	0.305166
Query 3	0.337529
Query 4	0.000003
Query 5	5.860250
Query 6	2.741809
Query 7	0.372926
Query 8	27.188631
Query 9	1.104103
Query 10	0.596863

Tabela 3.2: Tempo de execução das queries no Guião 3

Houve uma ligeira diferença em relação ao tempo de execução de algumas queries, devido às alterações que fizemos, alguns tempos subiram e outros desceram.

Conclusão

Concluindo, fazemos uma ilação positiva sobre o desenvolvimento deste projeto, apesar de uma menos boa gestão dos dados, tornando assim o programa mais pesado, principalmente na utilização dos novos ficheiros que são bastante maiores. Uma das possíveis soluções para este problema seria por exemplo armazenar tanto as *hashtables*, ou mais simplesmente as descrições dos repositórios, ou as maiores strings do programa e depois reaver essa informação apenas quando as *queries* necessitarem. É também necessário notar que existe particularmente, em alguns momentos no código onde poderá ocorrer perdas de memória(memory leaks). No entanto conseguimos implementar todos os testes, assim como a interação com o utilizador e a paginação, que eram grande parte desta fase do projeto. No final queríamos apenas referir que no que toca aos *warnings*, estes poderiam estar corrigidos mas tivemos alguns problemas com informação a apontar para sítios errados, apesar de sabermos como deveria estar realizada a estrutura das funções que têm *warnings* a nossa implementação de soluções para este problema nao estava na melhor das condições por isso deixamos estar assim para nao alterar resultados.