Guião II

Laboratórios de Informática III

2021/2022

Atualização: 2012-12-07

- clarificação relativa às pastas onde deverão estar os vários ficheiros
- prolongamento do prazo de entrega do trabalho e relatório

Objetivos

- Consolidação de conhecimentos essenciais da linguagem C e de Engenharia de Software, nomeadamente, modularidade e encapsulamento, estruturas dinâmicas de dados, e medição de desempenho (computacional, consumo de memória, etc);
- Consolidação do uso de ferramentas essenciais ao desenvolvimento de projetos em C, nomeadamente, compilação, linkagem e depuração de erros e de gestão de repositórios colaborativos;
- Preparação do trabalho que será realizado na terceira e última fase do projeto.

Realização e avaliação do trabalho desenvolvido

O desenvolvimento deste projeto deve ser realizado colaborativamente com o auxílio de *Git* e *GitHub*. Os docentes serão membros integrantes da equipa de desenvolvimento de cada trabalho e irão acompanhar semanalmente a evolução dos projetos. A entrega do trabalho será efetuada através desta plataforma e os elementos do grupo serão avaliados individualmente de acordo com a sua contribuição no repositório. Serão fornecidas algumas regras que os grupos deverão seguir para manter e estruturar o repositório de forma a que o processo de avaliação e de execução dos trabalhos possa ser uniforme entre os grupos e que possa ser efetuada de forma automática.

- O trabalho desenvolvido por cada grupo será avaliado com base no conteúdo da pasta “guiao-2” do repositório GitHub, à data de 13 de dezembro (23:59);

- O relatório do trabalho (máximo de 10 páginas de conteúdo, ou seja, sem capas ou anexos) terá de ser disponibilizado na raiz da pasta “guiao-2” até dia 13 de dezembro (23:59), em formato PDF, e o ficheiro correspondente terá de ter o nome “relatorio-2.pdf”. O relatório deverá centrar-se na resolução dos exercícios propostos, identificando as estratégias seguidas e eventuais limitações, bem como a medição de aspetos de desempenho, tais como o custo computacional e de armazenamento das estruturas de dados;

- Esta fase do trabalho a realizar na unidade curricular de Laboratórios de Informática III terá um peso de 50% na avaliação final. A avaliação desta fase será composta por 85% do exercício + 15% do relatório;

- O trabalho terá de ser desenvolvido por todos os elementos do grupo de trabalho e todos deverão registar as suas contribuições individuais no respetivo repositório;
- O projeto terá de gerar o necessário ficheiro executável com base na preparação de um ficheiro “Makefile” (na raiz da pasta “guiao-2”) e por invocação do comando “make”. Da mesma forma, deverá limpar todos os ficheiros desnecessários ao projeto através da execução do comando “make clean”;

- O executável deverá assumir a existência e ficheiros de entrada com nomes “users-g2.csv”, “commits-g2.csv” e “repos-g2.csv” dentro de uma sub-pasta “entrada” da pasta raiz da pasta “guiao-2”;

- O trabalho realizado por cada um dos grupos será avaliado em sessões de discussão com a equipa docente, sessões essas que decorrerão entre os dias 13 e 17 de dezembro.

Arquitetura da Aplicação

No sentido de definir desde já uma orientação de base quanto à arquitetura final da aplicação a desenvolver, pretende-se que esta se estruture — como apresentado na figura seguinte — e se identifiquem de forma clara as fontes de dados, a leitura das mesmas e os módulos de dados a construir (acompanhados pela respectiva API):

- Parsing dos dados: função ou parte do código no qual é realizada a leitura (e tratamento) dos dados dos 3 ficheiros;

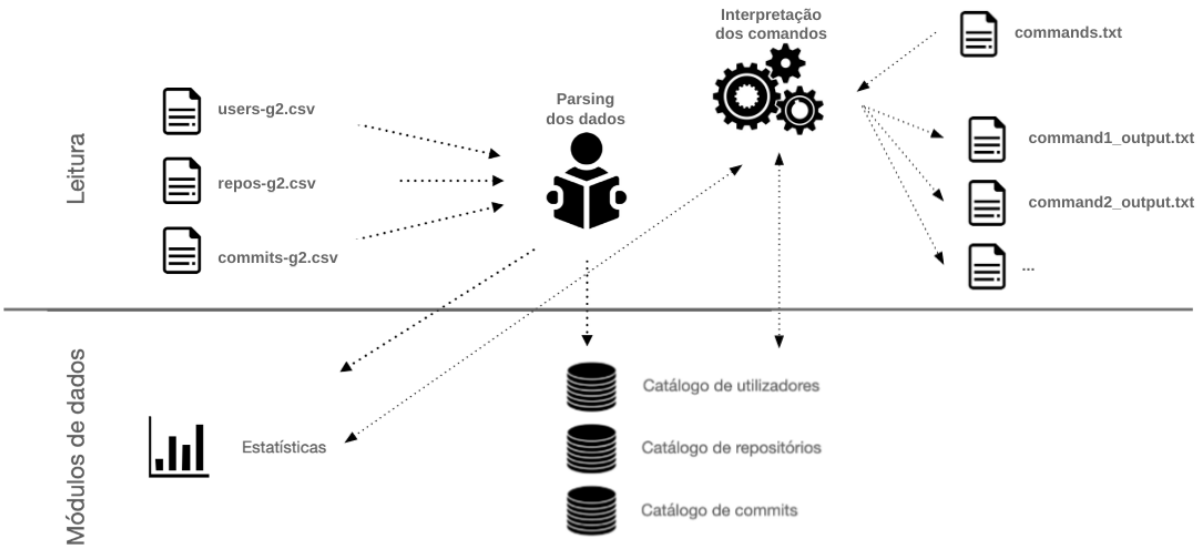
- Interpretação dos comandos: parte do código responsável por ler o ficheiro de comandos, interpretar cada um e executar a respetiva query com os argumentos indicados (se existirem);

- Catálogo de Utilizadores: módulo de dados onde deverão ser guardados todos os utilizadores provenientes de registos válidos do ficheiro users.csv, organizados por identificador;

- Catálogo de Repositórios: módulo de dados onde deverão ser guardados todos os repositórios e respectiva informação contida no ficheiro repos.csv;

- Catálogo de Commits: módulo de dados onde armazena convenientemente os commits recolhidos do ficheiro commits.csv;

Estatísticas: módulo que efetua relações entre os 3 principais modelos, proporcionando acesso mais rápido a dados e resultados pedidos nas queries do programa.



Arquitetura de referência para a aplicação a desenvolver

Para a estruturação otimizada destes módulos de dados será crucial analisar as queries que a aplicação deverá implementar. É de realçar que não devem assumir que o programa irá trabalhar sempre sobre os mesmos ficheiros de dados. Concebendo os módulos de dados anteriormente referidos, obtemos a primeira arquitetura de base para o projeto, sendo desde já de salientar que esta arquitetura (ou algo equivalente) irá ser construída por fases, módulo a módulo, testando cada fase e cada módulo até se ter a garantia de que podemos juntar todas as unidades do projeto na aplicação final.

Exercícios

De forma a convenientemente estruturar, gerir e expandir o projeto, pretende-se que na concepção do projeto se tenham em conta os princípios de Modularidade e Encapsulamento abordados nas aulas. A aplicação destes conceitos é obrigatória e será naturalmente objeto de avaliação no momento da apreciação e discussão do trabalho.

Com vista a avaliar e validar o funcionamento e a eficiência do armazenamento e gestão da informação em memória, pretende-se que o trabalho prático dê resposta a um conjunto de interrogações (queries) sobre os dados. Nesta fase, a interação do utilizador com o programa será única e exclusivamente através da sua invocação via linha de comando e da passagem do conjunto de queries a executar. As queries que o utilizador pretende executar serão especificadas como linhas de um ficheiro de texto, cujo nome é passado como argumento ao programa. O formato para a especificação de queries (dentro do ficheiro de texto) é o seguinte:

```
<query-id> [arg1 … argN]
```

O ficheiro de queries deverá então especificar **uma query por linha**. Cada uma destas linhas corresponde a um **comando**. Exemplo do conteúdo de um ficheiro contendo comandos válidos:

```
1
5 100 2010-01-01 2015-01-01
6 5 Python
7 2014-04-25
8 3 2016-10-05
9 4
6 3 Haskell
10 50
```

No exemplo acima, a primeira linha corresponde a um comando que invoca a query 1 que, à semelhança das outras queries estatísticas, não recebe nenhum parâmetro.

Os comandos 2 a 6 correspondem à invocação das queries 5 a 9, respetivamente. Visto serem queries parametrizáveis, têm valores a seguir ao primeiro número da linha (ID da query). Estes valores são os argumentos que devem ser considerados para a execução da query.

De notar que o comando 7 volta a invocar a query 6 mas, desta vez, com parâmetros diferentes. Ou seja, é de realçar que a mesma query pode ser invocada mais do que uma vez num ficheiro de comandos.

A última linha corresponde a um comando que invoca a query 10.

Queries estatísticas:

- Quantidade de *bots*, organizações e utilizadores.
    - ID: 1
    - Descrição: Quantidade de cada tipo de utilizador
    - Output: (X, Y e Z representam as quantidades calculadas)

```
Bot: X
Organization: Y
User: Z
```

- Número médio de colaboradores por repositório:
    - ID: 2
    - Descrição: (Total colaboradores) / (Total repositórios)
    - Output: Média calculada arredondada a duas casas decimais (Avg)

```
Avg
```

- Quantidade de repositórios com *bots*;

    - ID: 3
    - Descrição: Número de repositórios contendo *bots* como colaboradores;
    - Output: Total de repositórios

```
Total
```

- Qual a quantidade média de *commits* por utilizador?
    - ID: 4
    - Descrição: (Total de commits) / (Total de utilizadores)
    - Output: Média calculada arredondada a duas casas decimais (Avg)

```
Avg
```

Queries parametrizáveis:

- Qual o top N de utilizadores mais ativos num determinado intervalo de datas?
    - ID: 5
    - Descrição: Top N (indicado por parâmetro) de utilizadores com mais commits num determinado intervalo de datas.
    - Input:
        - `N (nr desejado de users)`
        - `data_inicio (YYYY-MM-DD)`
        - `data_fim (YYYY-MM-DD)`
    - Output: N linhas em que cada uma indica o *id*, *login* e nr. de *commits* de um utilizador.

```
ID1;Login1;Commit_qtty1
ID2;Login2;Commit_qtty2
...
```

- Qual o top N de utilizadores com mais commits em repositórios de uma determinada linguagem?
    - ID: 6
    - Descrição: Top N (indicado por parâmetro) de utilizadores presentes em commits de uma determinada linguagem (indicada por parâmetro)
    - Input:
        - `N (nr desejado de users)`
        - `linguagem (case insensitive)`
    - Output: N linhas em que cada uma indica o *id*, *login* e nr. de *commits* de um utilizador.

```
ID1;Login1;Commit_qtty1
ID2;Login2;Commit_qtty2
...
```

- Quais os repositórios inativos a partir de uma determinada data?

    - ID: 7
    - Descrição: Lista de repositórios sem commits a partir de uma determinada data (indicada por parâmetro).
    - Input: `data (YYYY-MM-DD)`
    - Output:

```
Repo1;Descricao1
Repo2;Descricao2
...
```

- Qual o top N de linguagens mais utilizadas a partir de uma determinada data?
    - ID: 8
    - Descrição: Top N de linguagens presentes com mais frequência em repositórios a partir de uma determinada data.
    - Input:
        - `N (nr desejado de linguagens)`
        - `data (YYYY-MM-DD)`
    - Output: N linhas em que cada uma apresenta o nome de uma linguagem.

```
Lang1
Lang2
...
```

- Qual o top N de utilizadores com mais *commits* em repositórios cujo *owner* é um amigo seu?
    - ID: 9
    - Descrição: Top N (indicado por parâmetro) de utilizadores com mais *commits* em repositórios cujo owner está contido na sua lista de *followers* e *following.*
    - Input: `N (nr desejado de users)`
    - Output:

```
ID1;Login1
ID2;Login2
...
```

- Qual o top N de utilizadores com as maiores mensagens de *commit* por repositório?
    - ID: 10
    - Descrição: Top N de utilizadores, ordenados pelo tamanho da maior mensagem de *commit*, em cada repositório.
    - Input: `N (nr desejado de users)`
    - Output:

```
ID1;Login1;Commit_msg_size1
ID2;Login2;Commit_msg_size2
...
```