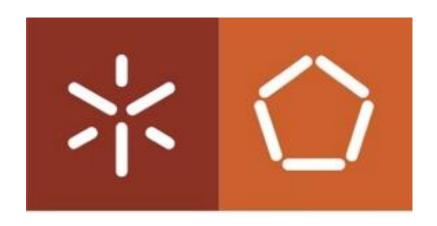
## Laboratórios de Informática III

### Guião 1



#### Universidade do Minho

João Gonçalo de Faria Melo, nº 95085 Cláudio Alexandre Freitas Bessa, nº 97063 Eduardo Miguel Pacheco Silva, nº 95345

Novembro de 2020

# Índice

1. Introdução	3
2. Desenvolvimento do trabalho	3
2.1. Exercício "users100.csv"	3
2.2. Exercício 1	4
2.3. Exercício 2	5
3. Conclusão	6

#### 1. Introdução

Ao longo deste relatório é abordado o desenvolvimento do trabalho, centrando a atenção na resolução dos exercícios propostos pelos docentes, no Guião 1, assim como nas estratégias e métodos utilizados. No Guião 1 está presente em que consiste o trabalho e os respetivos objetivos.

#### 2. Desenvolvimento do trabalho

#### 2.1. Exercício "users100.csv"

Começado o trabalho, o ponto de partida foi perceber o enunciado e aquilo que nos estava a ser pedido para fazer. Após ter esta compreensão, discutimos em grupo qual a abordagem que tomaríamos para a realização dos exercícios 1 e 2 propostos no Guião.

O primeiro passo foi, consolidando alguns conhecimentos relativos a operações com ficheiros, elaborar a função main de modo a abrir o ficheiro "users100.csv" disponibilizado pelos docentes e conseguir ler esse ficheiro linha a linha. Para fazer esta análise das várias linhas do ficheiro utilizamos funções como a fopen, fclose, fgets e strsep.

Era necessário agora analisar o conteúdo dos vários campos(colunas) presentes em cada linha do ficheiro e verificar se este era válido e/ou o formato correspondia àquele exigido. Para isto voltamos a usar a função strsep, separando a string que correspondia à linha a ser analisada pelos ";" que iam surgindo.

Após conseguirmos ler cada coluna individualmente, criamos as funções que nos permitem testar a validade dos campos. Se estes campos não estivessem corretos, a linha inteira teria que ser removida e apenas a linhas corretas seriam escritas num ficheiro de saída.

Começamos a trabalhar, então, numa função que faria estas verificações a partir de um ciclo que termina quando o ficheiro original(a ser analisado) não tiver mais linhas, e que daria indicação à função main para escrever as linhas corretas num ficheiro de saída.

Assim, conseguimos concluir a primeira parte dos nossos objetivos que era remover todas as linhas inválidas do ficheiro "users100.csv" e criar um ficheiro novo com apenas as linhas válidas, também em formato csv.

#### 2.2. Exercício 1

Na resolução do exercício 1 utilizamos as funções de validação criadas anteriormente para os utilizadores e acrescentamos algumas funções que avaliavam as restrições necessárias para lidar com os novos campos que apareciam em ficheiros relativos a commits e a repositórios.

Estas funções estão aplicadas numa função Parse1 que a main executa quando é indicada para realizar o exercício 1.

Durante a realização deste exercício percebemos que a ordem dos campos pode diferir de ficheiro para ficheiro, e, por isso, decidimos criar uma forma de generalizar o código de modo a aceitar qualquer ordem. Para isso, assumimos uma ordem fixa "standard" dos campos e fizemos uma função que permitia analisar as linhas quaisquer que fosse a ordem das colunas de acordo com o "standard" criado, não afetando a ordem dos campos no ficheiro de saída(esta continua a ser a mesma do original)

Questionámos-nos se alguns formatos em que as datas eram apresentadas, como "/", "h" e "m" em vez de ":" seriam também válidas e realizamos uma função permitia trocar entre estes carateres e corrigir o formato da data para aquele pedido no enunciado(no entanto, confirmamos mais tarde que apenas o formato exato dado pelos docentes deveria ser considerado);

Nesta fase, fizemos também umas alterações ao código para a imprimir no terminal, aquando da execução do exercício 1 na main, quantas linhas estavam inválidas em cada um dos ficheiros, de modo a podermos testar o funcionamento do nosso código e verificar se estava tudo a correr bem. Após deparamo-nos com a falta de algumas restrições e umas que tínhamos a mais.

Outro ponto a notar foi que dividimos o projeto nos seguintes ficheiros:

- main.c : contém a função main e as funções de parse;
- eval.c: contém as funções responsáveis pela avaliação do conteúdo das colunas e a sua validade;
  - eval.h: contém os headers das funções do eval.c;
  - define.h: contém todos os "defines" (ficheiros, buffer, etc).

#### 2.3. Exercício 2

Neste exercício era necessário comparar os ids que apareciam nos ficheiros dos commits e dos repositórios com os dos utilizadores e verificar se estes utilizadores existiam. Era também necessário remover commits cujos repositórios não existissem e remover os repositórios que não tivessem qualquer commit. Criámos, portanto, funções para fazer estas remoções.

Tal como foi feito no exercício 1, realizamos uma função Parse2 que a main executa quando é indicada para fazer o exerício 2.

No entanto, a procura dos ids mostrou-se ser, ao início, um problema, visto que o tamanho dos ficheiros era muito grande(até 10000000 de linhas), e a quantidade de ids que era preciso percorrer até encontrar aquele que se estava à procura podia ser muito elevada ou até ser necessário ter que se percorrer o ficheiro inteiro caso o id que se estava à procura não existisse.

Até este ponto, estávamos a utilizar apenas strings e portanto esta procura de ids fazia com que a execução do exercício 2 demorasse cerca de 15-20 minutos até terminar.

Decidimos tentar implementar uma hashtable para resolver este problema mas a sua implementação mostrou-se difícil e não conseguimos o seu correto funcionamento.

A nossa decisão final, então, para esta parte do trabalho foi implementar uma procura binária que permitiu que a execução fosse muito mais rápida, cerca de x segundos.

#### 3. Conclusão

No geral, consideramos que o grupo esteve bem organizado e trabalhou bem, e estamos contentes com o resultado final do trabalho.