Processamento de Linguagens
Trabalho Prático nº.1
Tema 3: Processador de Registos de Exames Médicos Desportivos
Grupo 8

Inês Vicente (A93269) Jorge Melo (A93308) Miguel Martins (A93280)

27 de março de 2022

1 Introdução

Para este primeiro trabalho prático da disciplina, o grupo optou por conceber uma solução para o terceiro problema apresentado no enunciado: criação de um *Processador de Registos de Exames Médicos Desportivos*. Em suma, o programa teria de receber como *imput* um *dataset* (*emd.csv*), no qual estão registadas informações sobre 300 exames médicos desportivos, e teria de processar esse ficheiro, de forma a obter os dados nele contidos. Com esses dados, teria de ser capaz de fazer a análise estatística solicitada no enunciado e colocá-la num simples *website*, através da criação de um ficheiro *index.html*, no qual a informação estatística e distribuição por categorias é apresentada ao utilizador do programa.

Nos seguintes capítulos é referida a forma como o grupo optou por dividir a criação do programa, nomeadamente na forma como:

(a) utilizou as ferramentas do módulo *re* para efetuar a leitura do *dataset*;

(b) alocou os dados do dataset;

(c) programou a solução para que cada uma das *queries* fizesse o solicitado no enunciado;

(d) gerou ficheiros *.html* para apresentar as informações de cada *query*, bem como para permitir uma navegação fluída entre páginas.

1.1 Execução do programa e dependências

Visto que utilizamos a biblioteca *dominate* para nos auxiliar na criação dos ficheiros HTML, como iremos abordar no capítulo 4, é necessário ter a seguinte dependência em conta antes de executar o programa:

```
[utilizador@Utilizador src]$ pip install dominate
```

Para executar o programa basta, na diretoria *src*, correr a *main.py* com o *path* para o ficheiro *.csv*:

```
[utilizador@Utilizador src]$ python3 main.py ../emd.csv
```

2 Leitura e armazenamento dos dados

Para armazenar as informações do *CSV* decidiu-se criar uma lista de dicionários, em que cada dicionário era a informação de cada atleta. Decidiu-se utilizar uma lista de dicionários invés de classes pois o grupo achou que seria uma estrutura que seria de mais fácil manipulação para a execução das *queries*, para além de também estar mais familiarizado com essa estrutura.
Para cada linha do *CSV*, foi aplicada a seguinte expressão regular:

```
(?P<_id>[A-Za-z0-9_À-ÿ]{24}),
(?P<index>\d+),(?P<dataEMD>\d{4}-\d{2}-\d{2}),
(?P<nome_primeiro>[A-Za-zÀ-ÿ]+),
(?P<nome_ultimo>[A-Za-zÀ-ÿ]+),
(?P<idade>\d+),(?P<genero>[Ff]|[Mm]),
(?P<morada>[A-Za-zÀ-ÿ]+),(?P<modalidade>[A-Za-zÀ-ÿ]+),
(?P<clube>[A-Za-zÀ-ÿ]+),
(?P<email>(\w+[\.]?\w+)+@(\w+\-?\w+\.)+(\w+\-?\w+)+),
(?P<federado>[Tt][Rr][Uu][Ee]|[Ff][Aa][Ll][Ss][Ee]),
(?P<resultado>[Tt][Rr][Uu][Ee]|[Ff][Aa][Ll][Ss][Ee])
```

Figura 1: Expressão regular utilizada

Na expressão regular acima foram usados grupos de captura com *tags* que correspondem aos vários atributos dos atletas presentes no ficheiro *CSV*. Essa informação será importante para podermos criar a lista de dicionários.
Para criar a lista de dicionários com as informações dos atletas, foi primeiramente criado um dicionário de listas, onde cada chave é um dos atributos capturados na expressão regular acima, e o *value* é uma lista com o conjunto desse atributo de todos os atletas. Depois disso, essa estrutura foi convertida para uma lista de dicionários, onde cada dicionário corresponde à informação de cada atleta.

3 *Queries* e Estatísticas

Depois dos dados presentes no CSV terem sido passados para um dicionário, esta parte trata de consultar esse dicionário e retornar as informações que eram pedidas na *query*. Esta parte está contida no ficheiro *queries.py*.

Na *query* **(a)**, é feito um dicionário com cada uma das datas com chave. De seguida, esse dicionário é ordenado. Depois, é feita uma lista com apenas o primeiro e último elementos desse dicionário e é retornada essa lista.

Nas *queries* **(b)** até **(g)**, a construção do novo dicionário é idêntica para todas. São construídos, em paralelo, dois dicionários de dicionários: ***dist*** e ***distStats***, cujas chaves são aquilo que a *query* pede para filtrar e os valores (de cada um dos dicionários dentro do dicionário principal) são uma lista das informações dos atletas correspondentes a essa chave (em ***dist***) e o número de atletas nessa lista (em ***distStats***). Depois, o ***dist*** é ordenado e os valores do ***distStats*** são mudados, de forma a serem percentuais. No ***distStats***, é também acrescentado um ou dois valores de distribuições totais, para algumas *queries*, como pedido no enunciado.

No fim, cada *query* retorna um *tuplo* com os dois dicionários criados.

4 Produção dos ficheiros *.html*

Esta secção do projeto, responsável por gerar não só o ficheiro HTML principal do projeto (*index.html*), mas também todos os ficheiros HTML das restantes páginas do *website*, tem a sua lógica inserida no ficheiro *htmlGenerator.py*, sendo os ficheiros gerados alocados na diretoria *out*.

Este, para além de possuir uma função capaz de criar a página principal do *website* com redirecionamento para cada uma das páginas das *queries*, também possui funções que, recebendo o *output* de cada *query*, são capazes de gerar um ficheiro *.html* principal para cada uma delas na respetiva diretoria, bem como gerar diretorias e outros ficheiros *.html* que sejam necessários para alocar a informação da *query*. De forma a clarificar o que foi referido anteriormente, podemos usar o processo da produção do HTML referente à *query* (b), a título de exemplo.

O objetivo da *query* é apresentar a distribuição de atletas por género em cada ano e no total. Assim sendo, a *query* usa o seguinte procedimento:

1. Criação na diretoria *out/queryB* de uma diretoria para cada ano. Isto é efetuado, recorrendo à função *makedirs* do módulo *os*.

2. Criação do ficheiro para cada género em cada ano. Ou seja, dentro da diretoria do ano 2019, p.e, temos um HTML para o sexo feminino e outro para o masculino. Estes ficheiros possuem listas de atletas ordenadas alfabeticamente por primeiro nome.

3. Ainda na diretoria de cada ano, será criado um ficheiro principal para cada um dos anos, capaz de redirecionar para cada um dos ficheiros referidos em (2), e com a percentagem referente a cada um deles.

4. Criação da página principal da *query*, capaz de redirecionar para cada uma das páginas dos anos, referidas em (3), com a percentagem de atletas em cada ano, bem como um conjunto de estatísticas gerais, que indica a percentagem de atletas de cada sexo no geral.

Esta secção do projeto, encarrega-se também de, na diretoria *athletes*, criar um ficheiro HTML para cada um dos atletas do *dataset*, com as suas informações mais importantes.

É de notar que, apesar de ser perfeitamente exequível a criação de funções que nos auxiliassem na escrita de HTML, o grupo optou por tirar proveito das várias bibliotecas e código *open-source* a que um programador tem acesso em *Python*. Assim sendo, escolhemos utilizar a ferramenta dominate concebida por Tom Flanagan.

5 Arquitetura da solução

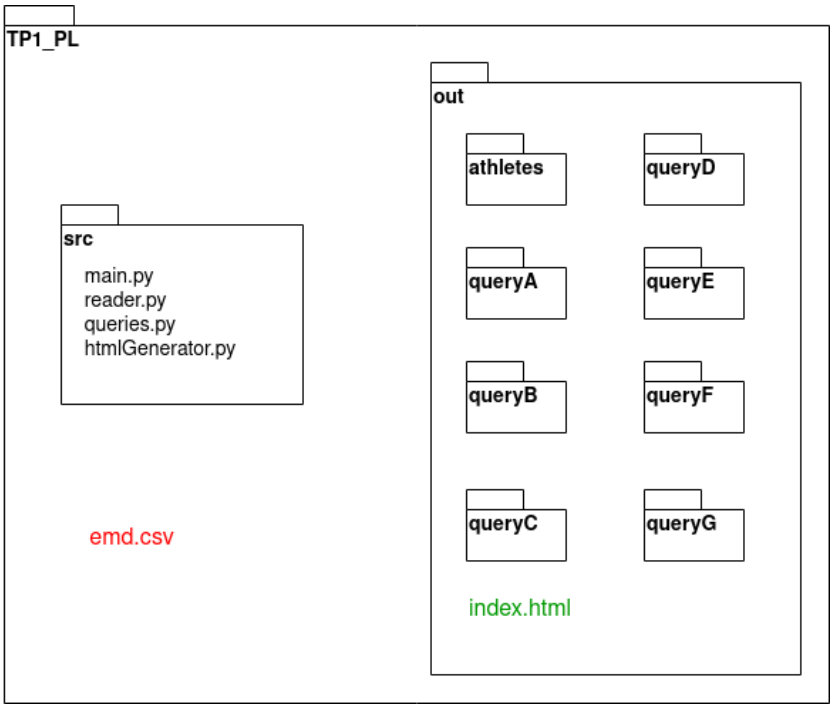

Figura 2: Arquitetura da solução

Tendo em conta os aspetos referidos nos capítulos anterior, optámos por representar visualmente a arquitetura geral do nosso projeto.

6 Aspeto da solução obtida

Para tornar elucidativa a forma como o utilizador irá ver o *website* gerado, segue uma imagem que ilustra como é feita a navegação ao longo das páginas para a *query* (c).

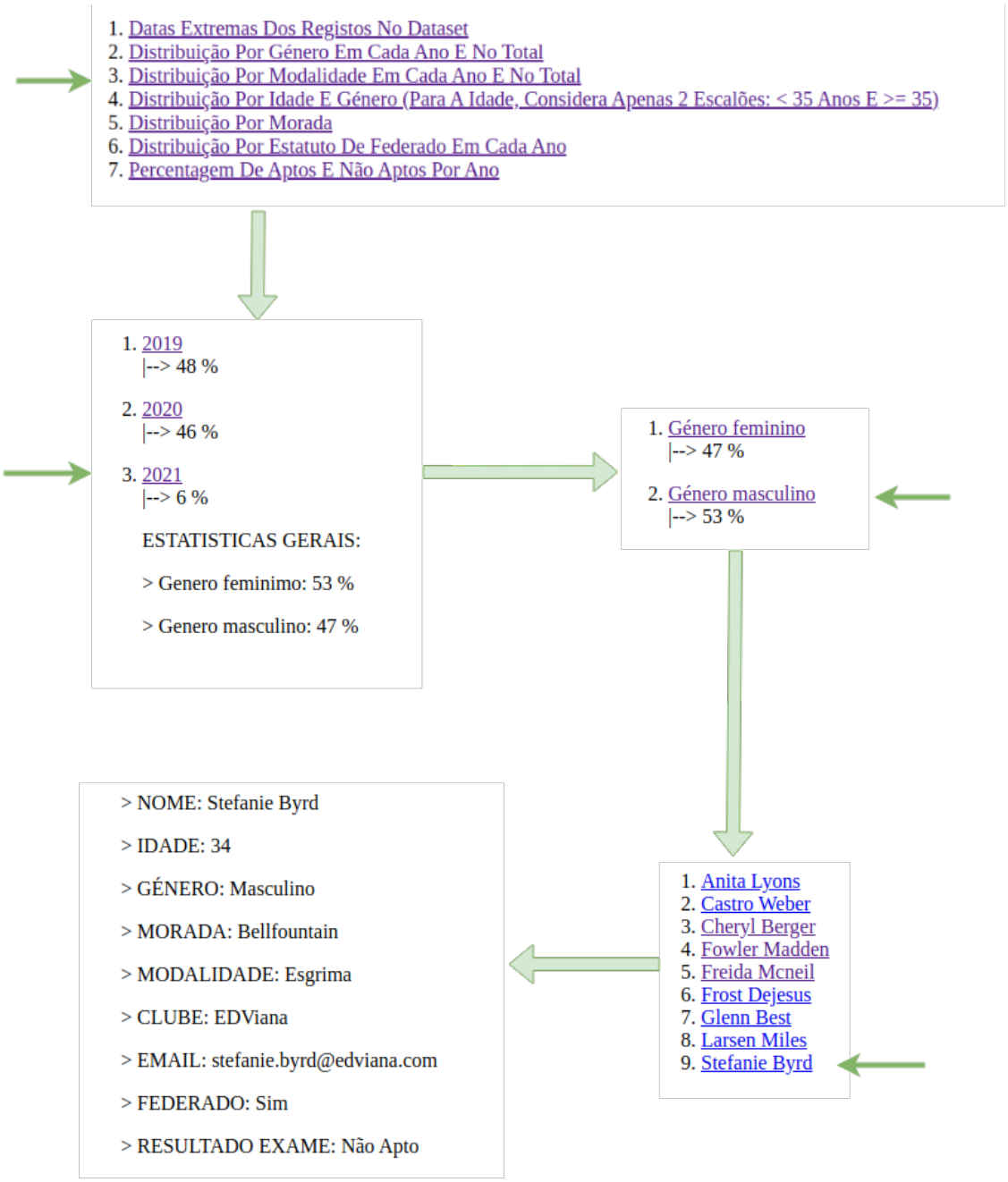

Figura 3: *Query* (c) ao longo das páginas do *website*

7 Conclusão

Consideramos que o trabalho final corresponde decentemente àquilo que era pedido no enunciado. Conseguimos apenas apontar um ponto fraco: há muita repetição de código. No entanto, estamos, de um modo geral, satisfeitos com o resultado final, especialmente porque percebemos, na prática, o quão simples os programas podem ficar ao usar as funcionalidades do ***regex***.