Ciência de Dados - ALPCD

Aplicação de conhecimentos - Módulos requests/re e formatos de dados

Ano Letivo 2024/25

1 Objetivos e organização

Este trabalho prático tem como principais objetivos:

- aumentar a capacidade de escrever Expressões Regulares (ER) para descrição de padrões de frases dentro de textos;
- desenvolver, a partir de ER, sistematicamente Processadores de Linguagens Regulares, ou Filtros de Texto (FT), que filtrem ou transformem textos com base no conceito de regras de produção Condição-Ação;
- utilizar o módulo 're' — com as suas funções de search(), split(), sub()—do Python para implementar os FT pedidos.
- desenvolver conhecimentos sobre REST API e o módulo requests.

Neste TP, que se pretende que seja resolvido rapidamente, **aprecia-se a imaginação/criatividade** dos grupos ao incluir outros processamentos!

Deve colocar a sua solução final no repositório git do seu grupo até **dia 10 de Novembro**.

O programa desenvolvido será apresentado aos membros da equipa docente, totalmente pronto e a funcionar e será defendido por **todos os elementos do grupo** no dia **13 de Novembro**.

2 Enunciado

Neste exercício pretende-se trabalhar com uma API aberta que providencia informações acerca de oportunidades de emprego.

A API que fornece os dados a utilizar neste trabalho prático é acessível através do URL `https://www.itjobs.pt/api/docs`.

Devem tirar algum tempo para analisar com cuidado a documentação fornecida. A partir dessa documentação poderão conhecer as rotas, testar a API, verificar os tipos de dados de saída, verificar os parâmetros que as rotas recebem, etc ... (uso do Postman pode ser benéfico).

Pretende-se que o produto final seja uma *Command Line Interface* (CLI) que permita a um utilizador interagir e obter dados da REST API. Uma CLI recebe como argumento um texto e executa determinadas funções necessárias para produzir uma certa saída que providencie os dados solicitados. Para isso é recomendável o uso da biblioteca **typer**[1], uma vez que é uma biblioteca simples que facilita a construção de CLIs e permite melhorar o output, validar o input, etc... de maneira rápida e com menos necessidade de produzir código para atingir o objetivo final (menos manual).

Para cada alínea crie os comandos na CLI de maneira a responder ao pedido:

a) Listar os N trabalhos mais recentes publicados pela itjobs.pt. O comando deve escrever o output para o terminal em formato *json*.

[1] https://typer.tiangolo.com/

Exemplo :

```
> python jobscli.py top 10
> [{...},...]
```

b) Listar todos os trabalhos do tipo *full-time*, publicados por uma determinada empresa, numa determinada localidade. O comando deve receber o o nome da empresa, localidade e número de trabalhos a mostrar, como argumentos e escrever o output para o terminal em formato *json*. Exemplo :

```
> python jobscli.py search Braga EmpresaX 4
> [{...},...]
```

c) Extrair a informação relativa ao salário oferecido por um determinado *job id*, mesmo que a referida oferta tenha o campo *wage* a *null*; nesse caso, deve procurar em outros campos relevantes utilizando expressões regulares. O comando deve receber o *job id* como argumento (notem que o *job id* é um identificador interno da API)e escrever o output para o terminal.

Exemplo:

```
> python jobscli.py salary JOBID
> 10000
```

d) Mostrar quais os trabalhos que requerem uma determinada lista de *skills*, num determinado período de tempo. O comando deve receber a lista de *skills* como argumento e escrever o output para o terminal em formato *json*.

Exemplo:

```
> python jobscli.py skills [skill1,skill2,skill3] dataInicial dataFinal
> [{...},...]
```

e) Para cada uma das funcionalidades (a), (b) e (d) deve poder exportar para *CSV* a informação com os seguintes campos :

- titulo;
- empresa;
- descrição;
- data de publicação;
- salário;
- localização.

Para isso pode adicionar um argumento opcional a cada um dos comandos que permite indicar se pretende criar o CSV.