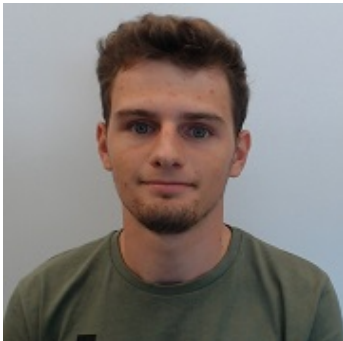


UNIVERSIDADE DO MINHO

DEPARTAMENTO DE INFORMÁTICA

PROCESSAMENTO DE LINGUAGENS

TRABALHO PRÁTICO 1
GRUPO Nº 39



Bohdan Malanka
a93300



Diogo Rebelo
a93278



Henrique Alvelos
a93316

Resumo

Este primeiro trabalho prático no âmbito da unidade curricular de processamento de linguagens consistiu na elaboração de um projeto em *python* onde por expressões regulares e filtros de texto foi desenvolvido um conversor genérico de ficheiros CSV para ficheiros JSON.

Assim sendo, ao longo do presente relatório iremos explicar como desenvolvemos os filtros de texto para extrair informação pertinente do ficheiro CSV, utilizando expressões regulares e como aplicamos métodos do *python* para converter a formatação do ficheiro CSV em JSON. Finalmente, incluímos exemplos de utilização do programa desenvolvido.

Conteúdo

1	Introdução	4
1.1	Problema e Objetivos	4
2	Análise e Especificação	4
2.1	Descrição Informal do Problema	4
2.2	Especificação dos Requisitos	4
2.3	Fichero CSV	4
2.4	Ficheiro JSON	5
3	Desenho da Resolução	5
3.1	Estruturas de Dados	5
3.2	Tratamento do Documento CSV	5
4	Codificação e Testes	5
5	Conclusão	7

1 Introdução

Os ficheiros CSV são ficheiros sem formatação muito utilizados por softwares como *Microsoft Excel*, onde cada linha representa um registo distinto. Neste formato, os valores de cada linha estão separados por delimitadores como a vírgula ou o ponto e vírgula. Os ficheiros JSON têm um formato compacto muito usado em sistemas de exportação/transferência de dados entre aplicações para assegurar a interoperabilidade como *web services*, sendo um substituto dos ficheiros XML.

Assim, com este trabalho pretendemos implementar um conversor genérico de CSV para JSON, tornando fácil e rápida a criação dos ficheiros JSON.

1.1 Problema e Objetivos

Em geral, o projeto aprofunda a aprendizagem de temáticas e conceitos já abordados nas aulas, tais como:

- aumentar a experiência de uso do ambiente Linux e de algumas ferramentas de apoio à programação;
- aumentar a capacidade de escrever Expressões Regulares (ER) para descrição de padrões em streams de texto;
- desenvolver, a partir de ER, sistemática e automaticamente, Processadores de Linguagens Regulares, que encontrem ou transformem textos com base no conceito de regras de produção Condição-Ação;
- utilizar o Python e os seus módulos *re* e *ply* para gerar os filtros de texto.

Tendo estes objetivos em mente e como já foi referido anteriormente, pretende-se através da exploração de expressões regulares e respetivas funções *python*, converter um ficheiro CSV (*Comma Separated Values*) para JSON (*JavaScript Object Notation*).

2 Análise e Especificação

2.1 Descrição Informal do Problema

Para um qualquer ficheiro gravado em formato CSV, independentemente do número de linhas e colunas, pretende-se extrair a primeira linha que funciona como cabeçalho para as restantes linhas do ficheiro. Posteriormente, vamos executar a conversão para um novo ficheiro JSON, ou seja, nada mais que um formato textual neutro e simples, baseado no conceito de um conjunto de pares {"campo": "valor"}, onde o "campo" será retirado do cabeçalho e o "valor" corresponde ao respetivo conteúdo presente na linha a fazer a correspondência para JSON.

Contudo, este conteúdo terá de ser analisado, caso o cabeçalho contenha quatro pontos('::') seguidos de uma função de agregação (sum, mult, media, mediana, maior, menor, ordemCrescente, ordemDecrescente, antiga, recente) será efetuado um *fold* sobre a lista (o conteúdo) e, no ficheiro JSON, a lista será substituída pelo valor da função agregação.

2.2 Especificação dos Requisitos

Como forma de cumprir com o objetivo do problema apresentado é necessário analisar e especificar os dados e requisitos do projeto. Para isso, é fundamental ter em consideração a estrutura e características dos ficheiros CSV e JSON usados para implementação da solução.

2.3 Ficheiro CSV

O *input* que o programa criado recebe é um ficheiro CSV. Trata-se de um ficheiro simples, em que os dados aparecem separados por um delimitador. Para o programa em questão é dado o contexto do problema, consideramos a vírgula como separador. Além disso, de forma a identificar cada campo presente em cada linha, existe um *header* na primeira linha do ficheiro. Este cabeçalho

do ficheiro CSV contém os nomes que identificam cada campo. Caso esse nome seja seguido de chavetas com um numero ou um intervalo com dois números, significa que esse campo é uma lista de comprimento *min*, *max*. Seguido de quatro pontos (':') pode haver uma função de agregação como *max*, *min*, *avg* ou *sum* entre outras referidas em cima.

2.4 Ficheiro JSON

O output que o programa produz é um ficheiro JSON. Estes ficheiros iniciam e terminam com parêntesis retos e cada registo está dividido em vários campos onde cada campo tem um valor associado cumprindo com o formato {"campo": "valor"}. No caso em que os valores sejam listas estes são representados por [] de forma a cumprir com a formatação JSON. Strings estão entre aspas e números não necessitam de aspas.

3 Desenho da Resolução

3.1 Estruturas de Dados

As estruturas de dados usadas são nativas do Python: Dicionários e Listas. Usa-se os dicionários para armazenar dados sobre cada linha do ficheiro .csv e as listas para guardar esses dicionários e para guardar o resultado do match das expressões regulares.

3.2 Tratamento do Documento CSV

Existem duas expressões regulares (uma para o cabeçalho, outra para as restantes linhas):

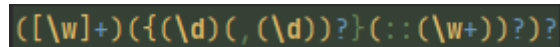


Figura 1: Expressão regular para o cabeçalho

Esta expressão significa que cada elemento do cabeçalho precisa de ter um nome (conjunto de caracteres) e, a seguir, poderá conter um/dois limites e uma função de agregação, daí que tenha a possibilidade de ter um/dois dígitos, separados por uma vírgula e que estejam entre chavetas e, além disso, quatro pontos e um nome para a função de agregação.

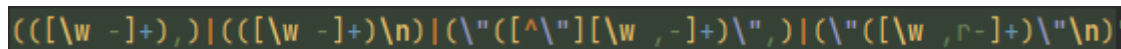


Figura 2: Expressão regular para as restantes linhas

Para esta última adicionamos o caso de que um valor contenha uma vírgula. Para obter uma vírgula, este necessita de estar entre aspas. Assim, para o valor ser válido, tem de:

- Estar entre aspas, tiver uma vírgula, pelo menos e acabar com uma vírgula;
- Estar entre aspas, tiver uma vírgula, pelo menos e acabar com um *end of line*;
- Conter caracteres e acabar com uma vírgula;
- Conter caracteres e acabar com um *end of line*;

Com isto, chegamos à expressão acima.

4 Codificação e Testes

Como forma de testar o programa implementado utilizou-se o ficheiro alunos.csv, disponível na blackboard. Em baixo encontra-se um pedaço desse ficheiro CSV e a sua respetiva conversão para JSON.

```

PL_TP > resources > alunos.csv
1 id_aluno,nome,curso,tpc1{3,4}::sum,,,
2 "a1","Aysha Melanie Gilberto","LEI",12,8,19,,
3 "a2","Igor André Cantanhede","ENGFIS",12,,18,20
4 "a3","Laurénio Narciso","ENGFIS",8,14,15,14
5 "a4","Jasnoor Casegas","LCC",14,20,17,
6 "a5","Tawseef Rebouças","ENGBIOM",13,14,13,17
7 "a6","Eryk Clementino","LEI",,19,11,14
8 "a7","Ianna Noivo","ENGBIOM",15,15,17,16
9 "a8","Ayla Thaissa Reina","ENGBIOM",12,8,8,18
10 "a9","Cássia Viviane Coitã","ENGFIS",19,9,14,10

```

Figura 3: alunos.csv

```

PL_TP > out > alunos.json > ...
1 [
2   {
3     "id_aluno": "a1",
4     "nome": "Aysha Melanie Gilberto",
5     "curso": "LEI",
6     "sum": 39.0
7   },
8   {
9     "id_aluno": "a2",
10    "nome": "Igor André Cantanhede",
11    "curso": "ENGFIS",
12    "sum": 50.0
13  },
14  {
15    "id_aluno": "a3",
16    "nome": "Laurénio Narciso",
17    "curso": "ENGFIS",
18    "sum": 51.0
19  },
20  {
21    "id_aluno": "a4",
22    "nome": "Jasnoor Casegas",
23    "curso": "LCC",
24    "sum": 51.0
25  },
26  {
27    "id_aluno": "a5",
28    "nome": "Tawseef Rebouças",
29    "curso": "ENGBIOM",
30    "sum": 57.0
31  },
32  {
33    "id_aluno": "a6",
34    "nome": "Eryk Clementino",
35    "curso": "LEI",
36    "sum": 44.0
37  },
38  {
39    "id_aluno": "a7",
40    "nome": "Ianna Noivo",
41    "curso": "ENGBIOM",
42    "sum": 63.0
43  },
44  ]

```

Figura 4: alunos.json

5 Conclusão

Dada por concluída a realização do trabalho prático, consideramos boa prática fazer uma apreciação crítica realçando não só os aspetos positivos como também as dificuldades que surgiram e o modo como estas foram colmatadas.

No que diz respeito aos pontos fortes, destacamos a universalidade do trabalho, uma vez que funciona para qualquer tipo de ficheiro CSV cujo delimitador seja ",", i.e, funciona independentemente do número de colunas ou linhas e podemos usar várias funções de agregação. Destacamos também a eficiência que trouxe a estrutura de dados implementada, que se refletiu na leitura e análise do cabeçalho apenas uma vez, e permitiu diminuir significativamente o número de linhas lidas.

Durante a realização deste trabalho surgiram algumas dificuldades, tais como: lidar com os casos de erro no formato do ficheiro CSV, as possíveis vírgulas dentro do valor e que condicionavam a divisão de cada linha e decidir qual a melhor opção de implementação. Apesar disso, consideramos que os problemas foram ultrapassados com sucesso.

Desta forma, pretendemos explicitar que a realização deste projeto foi um aspeto essencial para aprimorar e cimentar os conhecimentos sobre Expressões Regulares e Filtros de Texto. Para além disso, uma vez que todas as dificuldades foram ultrapassadas de forma eficaz concluímos que o balanço do resultado final foi positivo.