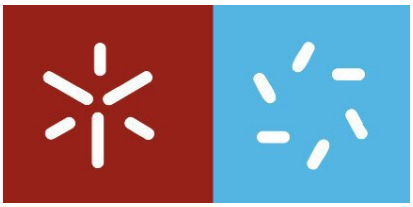


Universidade do Minho

Licenciatura em Ciências da Computação

PLC - Trabalho Prático 1

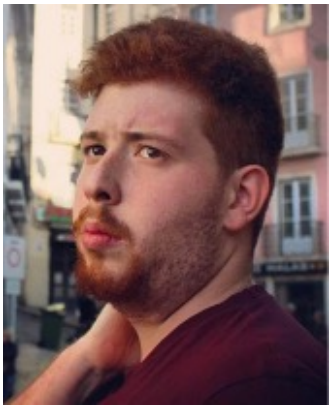
Grupo nº16

Francisco José Pereira Teófilo (A93741)

Inês Pires Presa (A90355)

Tiago dos Santos Silva Peixoto Carriço (A91695)

17 de novembro de 2021



Conteúdo

Capítulo 1

Introdução

No âmbito da disciplina de Processamento de Linguagens e Compiladores foi-nos proposto pelo docente Pedro Manuel Rangel Santos Henriques um trabalho de pesquisa cujo objetivo principal é consolidar a aprendizagem da utilização de linguagens regulares em programas em linguagem Python.

Neste sentido, ficou-nos incumbido a realização do exercício 2 que tem como intuito o cálculo de frequências de diferentes componentes num ficheiro "processos.txt", onde se pode encontrar uma lista de pessoas nos Róis de Confessados.

Neste documento apresentamos a nossa solução do problema, começando por discriminar as decisões tomadas na sua implementação e, seguidamente, apresentando alguns exemplos e os respetivos resultados da aplicação das diferentes funcionalidades realizadas.

Capítulo 2

Enunciado

2.1 Processador de Pessoas Listadas nos Róis de Confessados

Construa agora um ou vários programas *Python* para processar o texto 'processos.txt' com o intuito de calcular frequências de alguns elementos (a ideia é utilizar *arrays* associativos para o efeito) conforme solicitado a seguir

a. Calcula a frequência de processos por ano (primeiro elemento da data);

b. Calcula a frequência de nomes (considera um nome uma palavra e propaga o cálculo por todos os campos que contenham nomes);

c. Calcula a frequência dos vários tipos de relação: irmão, sobrinho, etc.;

d. Imprimir os 20 primeiros registos num novo ficheiro de output mas em formato *Json*.

Capítulo 3

Decisões Tomadas

Na resolução deste trabalho, optamos por construir um programa que permite ao utilizador escolher entre as quatro tarefas a realizar sobre o ficheiro 'processos.txt' listasdas no enunciado, desenvolvendo, para o efeito, uma função correspondente a cada alínea do problema.

a. Para a primeira alínea utilizamos um dicionário que a cada ano associa o número de processos registados. Para o efeito, iteramos sobre cada linha do ficheiro, usando a função `search()` para identificar a data através da expressão regular `[0-9]{4}(-[0-9]{2}){2}`.

```
def a(ficheiro_nome):
    ficheiro = open(ficheiro_nome, 'r')

    linha = ficheiro.readline()
    frequencia = {}

    while linha:
        x = re.search(r'[0-9]{4}(-[0-9]{2}){2}', linha)
        if x:
            ano = x.group()[:4]
            if ano not in frequencia:
                frequencia[ano] = 1
            else:
                frequencia[ano] += 1

        linha = ficheiro.readline()

    ficheiro.close()
    return frequencia
```

b. Para calcular a frequência de cada nome decidimos, mais uma vez, usar um dicionário que atribui a cada nome a sua frequência. Neste caso, utilizamos a função `split()` para guardar numa lista os diferentes campos de cada linha, que se encontram divididos por "::". De seguida, acedemos aos campos que continham nomes e fomos adicionando cada um ao dicionário, usando a expressão regular `[a-zA-Z]+` para identificar cada palavra. Para além disso, uma vez que consideramos relevante analisar a frequencia de cada nome por ano, criamos também um dicionário com o intuito de fazer esse registo.

```
def b(ficheiro_nome):
    ficheiro = open(ficheiro_nome, 'r')

    linha = ficheiro.readline()
    frequencia = {}
    anos = {}

    while linha:
        linha_dividida = re.split('::', linha)

        if len(linha_dividida) >= 2:

            ano = linha_dividida[1][:4]
            if ano not in anos:
                anos[ano] = {}

            for campo in linha_dividida[2:5]:
                listaPalavras = re.findall(r'[a-zA-Z]+', campo)
                for palavra in listaPalavras:
                    if palavra not in frequencia:
                        frequencia[palavra] = 1
                    else:
                        frequencia[palavra] += 1

                    if palavra not in anos[ano]:
                        anos[ano][palavra] = 1
                    else:
                        anos[ano][palavra] += 1

        linha = ficheiro.readline()

    ficheiro.close()
    nomes_mais_frequentes = {}

    for ano in anos:
        nomes_mais_frequentes[ano] = max(anos[ano], key=lambda x: anos[ano][x])

    return frequencia, sorted([(x, nomes_mais_frequentes[x]) for x in nomes_mais_frequentes])
```

c. No cálculo dos vários tipos de relação voltamos a utilizar um dicionário para guardar cada relação e o número de vezes que é mencionada no ficheiro. Ao analisar o ficheiro constatamos que cada grau de parentesco aparece após uma vírgula e antes de um ponto, portanto utilizamos a função `split()` para dividir as linhas por cada vírgula. De seguida, iteramos sobre cada elemnto, identificando cada relação através da expressão regular:

`[a-zA-Z]*(Irmao|Tio|Sobrinh[ao]|Pai|[nN]et[ao]|Meio|[Aa]vo)[a-zA-Z ]*\.`

```
def c(ficheiro_nome):
    ficheiro = open(ficheiro_nome, 'r')

    linha = ficheiro.readline()
    frequencia = {}

    while linha:
        lista = re.split(r',', linha)

        for elemento in lista[1:]:
            x = re.match(r'[a-zA-Z]*(Irmao|Tio|Sobrinh[ao]|Pai|[nN]et[ao]|Meio|[Aa]vo)[a-zA-Z ]*\.',
            elemento)
```

```
            if x:
                relacao = x.group()[:-1]
                if relacao not in frequencia:
                    frequencia[relacao] = 1
                else:
                    frequencia[relacao] += 1

        linha = ficheiro.readline()

    ficheiro.close()
    return frequencia
```

d. Na organização dos primeiros vinte registos em formato *Json*, decidimos estabelecer uma lista com um dicionário para cada linha, cujas chaves são `['numero', 'data', 'nome', 'pai', 'mae', 'informacao adicional']` de modo a identificar e guardar cada campo separado por "::". Por fim, usamos a função `validateJSON()` para verificar se a estrutura do ficheiro *Json* é válida.

```
def validateJSON(jsonData):
    try:
        json.load(jsonData)
    except ValueError as err:
        return False
    return True

def d(ficheiro_nome):
    ler = open(ficheiro_nome, 'r')
    escrever = open("output.json",'w')
    escrever.write("[\n")
    lista = []

    for y in range(20):
        linha = ler.readline()
        escrever.write("{")
        campos = re.split(r'::', linha)
        chaves = ['numero', 'data', 'nome', 'pai', 'mae', 'informacao adicional']

        for x in range(len(campos)-1):
            if campos[x] != "":
                escrever.write(f"\n\"{chaves[x]}\": \"{campos[x]}\",")

        escrever.seek(escrever.tell()-1,0)

        escrever.write("\n}")
        if y == 19:
            escrever.write("\n")
        else:
            escrever.write(",\n")

    escrever.write("]")

    ler.close()
    escrever.close()

    ficheiro = open("output.json", 'r')
    res = validateJSON(ficheiro)
    ficheiro.close()
    return res
```

Capítulo 4

Exemplos de utilização e análise dos resultados

4.1 Exemplo 1

Nesta secção vamos apresentar os resultados das diferentes funcionalidades do nosso programa aplicadas ao seguinte conjunto de linhas:

```
260::1712-08-11::Luis Silva::Francisco Silva Rego::Ana Vale::Constantino Silva
Rego,Irmao. Proc.32346.::

1336::1786-02-21::Francisco Goncalves Azevedo::Gervasio Pires Alvares::Maria
Goncalves Azevedo::Antiga Freguesia de PARADA OUTEIRO GERES,Sao Tome. Manuel
Alvares Pires Carvalho,Tio Paterno. Proc.16797. Andre Goncalves Azevedo,Tio
Materno. Proc.7722. Francisco Venancio Goncalves Azevedo,Sobrinho Paterno.
Proc.14975.::

674::1705-08-25::Joao Freitas::Pedro Freitas::Helena Oliveira::Antonio Freitas
Sampaio,Irmao. Proc.7035.::

620::1847-04-20::Paulo Manuel Dias::Custodio Dias::Teresa Alves::Joao Dias,Tio
Paterno. Proc.24810. Joao Manuel Alvares,Tio Materno.Proc.25325.::

1114::1807-04-03::Joao Francisco Nogueira::Manuel Francisco::Ana Luisa
Nogueira::::
```

4.1.1 Frequência de Processos por Ano

```
Introduza um ano (Sair - 0): 1807
1807: 1
Introduza um ano (Sair - 0): 1904
1904: 0
```

Como esperado, o programa encontrou um processo no documento no ano 1807 e nenhum no ano 1904.

4.1.2 Frequência de Nomes

```
Introduza um nome (Sair - 0): Joao
Joao: 2
Introduza um nome (Sair - 0): Francisco
Francisco: 4
Introduza um nome (Sair - 0): Tiago
Tiago: 0
```

De acordo com o previsto, o programa encontrou duas pessoas com o nome Joao, quatro chamadas Francisco e nenhum Tiago.

4.1.3 Frequência dos Vários Tipos de Relação

```
c
{'Irmao': 2, 'Tio Paterno': 2, 'Tio Materno': 2, 'Sobrinho Paterno': 1}
```

Conforme esperável, o programa encontrou dois irmãos, dois tios paternos, dois tios paternos e apenas um sobrinho paterno.

4.2 Exemplo 2

Nesta secção vamos apresentar os resultados das diferentes funcionalidades do nosso programa aplicadas ao ficheiro "processos.txt".

4.2.1 Frequência de Processos por Ano

```
Introduza um ano (Sair - 0): 1807
1807: 602
Introduza um ano (Sair - 0): 1904
1904: 52
```

4.2.2 Frequência de Nomes

```
Introduza um nome (Sair - 0): Joao
Joao: 10467
Introduza um nome (Sair - 0): Francisco
Francisco: 8316
Introduza um nome (Sair - 0): Tiago
Tiago: 13
```

4.2.3 Nomes Mais Frequentes por Ano

```
Introduza um ano (Sair - 0): 1904
1904: Jose
Introduza um ano (Sair - 0): 1786
1786: Jose
Introduza um ano (Sair - 0): 1712
1712: Maria
```

4.2.4 Frequência dos Vários Tipos de Relação

```
{'Tio Paterno': 2245, 'Tio Materno': 2463, 'Irmao': 13168,
 'Sobrinho Materno': 1698, 'Pai': 525, 'Sobrinho Paterno':
 1642, 'Irmaos': 686, 'Sobrinhos Maternos': 98, 'Irmao Pat
erno': 497, 'Neto Materno': 41, 'Sobrinhos Paternos': 57,
'Sobrinho Neto Paterno': 97, 'Tio Avo Paterno': 154, 'Tio
Avo Materno': 230, 'Irmao Materno': 55, 'Sobrinho Bisneto
Paterno': 3, 'Tios Maternos': 20, 'Irmaos Paternos': 21, '
Sobrinho Neto Materno': 145, 'Avo Materno': 48, 'Bisavo Ma
terno': 2, 'Avo Paterno': 11, 'Neto Paterno': 8, 'Tios Pat
ernos': 12, 'Tio Bisavo Materno': 5, 'Tio Bisavo Paterno':
 6, 'Irmaos Maternos': 4, 'Sobrinhos Netos Paternos': 2, '
Sobrinho Neto': 2, 'Tio Avo': 3, 'Tio': 5, 'Sobrinhos Neto
s Maternos': 5, 'Meio Irmao': 3, 'Sobrinho Bisneto Materno
': 3}
```

4.2.5 Imprimir os 20 Primeiros Registos em formato *Json*


```
[
    {
    "numero": "575",
    "data": "1894-11-08",
    "nome": "Aarao Pereira Silva",
    "pai": "Antonio Pereira Silva",
    "mae": "Francisca Campos Silva"
    },
    {
    "numero": "582",
    "data": "1909-05-12",
    "nome": "Abel Almeida",
    "pai": "Antonio Manuel Almeida",
    "mae": "Teresa Maria Sousa"
    },
    {
    "numero": "569",
    "data": "1867-05-23",
    "nome": "Abel Alves Barroso",
    "pai": "Antonio Alves Barroso",
    "mae": "Maria Jose Alvares Barroso",
    "informacao adicional": "Bento Alvares Barroso,Tio Paterno. Proc.32057.
    Domingos Jose Alvares Barroso,Tio Materno. Proc.32235."
    },
    {
    "numero": "576",
    "data": "1896-11-28",
    "nome": "Abel Augusto Oliveira",
    "pai": "Francisco Jose Oliveira",
    "mae": "Antonia Rosa Rebelo",
    "informacao adicional": "Jose Antonio Oliveira,Irmao. Proc.5020."
    },
    {
    "numero": "579",
    "data": "1904-05-27",
    "nome": "Abel Gomes Abreu Reis",
    "pai": "Antonio Gomes Abreu",
    "mae": "Ana Sequeira Reis"
    },
    {
    "numero": "579",
    "data": "1904-05-21",
    "nome": "Abel Marques Reis",
    "pai": "Jose Joaquim Marques Reis",
    "mae": "Bernardina Dantas"
    },
    {
    "numero": "578",
    "data": "1901-11-12",
    "nome": "Abel Martins Pereira",
    "pai": "Serafim Martins Pereira",
```

```
"mae": "Emilia Goncalves"
},
{
"numero": "572",
"data": "1883-02-01",
"nome": "Abel Pedro Pereira Freitas",
"pai": "Joao Freitas Oliveira",
"mae": "Cecilia Rosa Pereira"
},
{
"numero": "578",
"data": "1900-08-30",
"nome": "Abel Silva Carvalho",
"pai": "Constantino Silva Rego",
"mae": "Margarida Rosa Carvalho"
},
{
"numero": "575",
"data": "1894-04-30",
"nome": "Abelardo Jose Cerqueira Araujo",
"pai": "Jose Maria Araujo",
"mae": "Leopoldina Cerqueira Ribeiro Araujo"
},
{
"numero": "575",
"data": "1894-04-30",
"nome": "Abelardo Jose Cerqueira Araujo",
"pai": "Jose Maria Araujo",
"mae": "Leopoldina Cerqueira Ribeiro Araujo"
},
{
"numero": "572",
"data": "1883-11-24",
"nome": "Abilio Acacio Conceicao Guerreiro",
"pai": "Jose Antonio Pereira Dantas Guerreiro",
"mae": "Maria Rita Pereira Monteiro"
},
{
"numero": "579",
"data": "1902-10-23",
"nome": "Abilio Aires Sousa Pereira Guimaraes",
"pai": "Joaquim Aires Sousa Pereira Guimaraes",
"mae": "Josefa Rosa Gomes"
},
{
"numero": "571",
"data": "1880-01-24",
"nome": "Abilio Antonio Alves",
"pai": "Joao Francisco Alves",
"mae": "Maria Jesus Rocha"
},
{
"numero": "573",
"data": "1889-12-03",
```

```
"nome": "Abilio Augusto Arantes",
"pai": "Sebastiao Arantes",
"mae": "Maria Sousa"
},
{
"numero": "581",
"data": "1908-08-11",
"nome": "Abilio Augusto Galvao",
"pai": "Antonio Augusto Galvao",
"mae": "Perpetua Duarte Galvao"
},
{
"numero": "581",
"data": "1908-05-20",
"nome": "Abilio Augusto Magalhaes",
"mae": "Maria Jesus Magalhaes"
},
{
"numero": "581",
"data": "1908-05-20",
"nome": "Abilio Augusto Magalhaes",
"mae": "Maria Jesus Magalhaes"
},
{
"numero": "581",
"data": "1908-05-20",
"nome": "Abilio Augusto Magalhaes",
"mae": "Maria Jesus Magalhaes"
},
{
"numero": "569",
"data": "1869-12-02",
"nome": "Abilio Augusto Santos",
"pai": "Jose Joaquim Santos",
"mae": "Teresa Jesus",
"informacao adicional": "Antonio Jose Adao,Tio Materno. Proc.12530.
Albino Antonio Ribeiro,Primo Paterno. Proc.12721."
}
]
```

Capítulo 5

Conclusão

Com a realização deste trabalho alcançamos os objetivos esperados, sendo que nos sentimos mais experientes na manipulação de expressões regulares e na sua aplicação a casos concretos. Para o efeito, aplicamos os conhecimentos adquiridos durante as aulas, concretamente a utilização de funções como *split*, *search* e *findall*.

A elaboração deste projeto levou-nos também a conhecer melhor uma forma de representação e armazenamento de dados, o *Json*.

Finalmente, consideramos que este trabalho fundamentou as nossas bases para que consigamos tirar o melhor proveito do que será lecionado posteriormente nesta unidade curricular.

Apêndice A

Código do Programa

```
import re
import json

def a(ficheiro_nome):
    ficheiro = open(ficheiro_nome, 'r')

    linha = ficheiro.readline()
    frequencia = {}

    while linha:
        x = re.search(r'[0-9]{4}(-[0-9]{2}){2}', linha)
        if x:
            ano = x.group()[:4]
            if ano not in frequencia:
                frequencia[ano] = 1
            else:
                frequencia[ano] += 1

        linha = ficheiro.readline()

    ficheiro.close()
    return frequencia

def b(ficheiro_nome):
    ficheiro = open(ficheiro_nome, 'r')

    linha = ficheiro.readline()
    frequencia = {}
    anos = {}

    while linha:
        linha_dividida = re.split('::', linha)

        if len(linha_dividida) >= 2:

            ano = linha_dividida[1][:4]
            if ano not in anos:
                anos[ano] = {}
```

```
            for campo in linha_dividida[2:5]:
                listaPalavras = re.findall(r'[a-zA-Z]+', campo)
                for palavra in listaPalavras:
                    if palavra not in frequencia:
                        frequencia[palavra] = 1
                    else:
                        frequencia[palavra] += 1

                    if palavra not in anos[ano]:
                        anos[ano][palavra] = 1
                    else:
                        anos[ano][palavra] += 1

        linha = ficheiro.readline()

    ficheiro.close()

    nomes_mais_frequentes = {}

    for ano in anos:
        nomes_mais_frequentes[ano] = max(anos[ano], key=lambda x: anos[ano][x])

    return frequencia, sorted([(x, nomes_mais_frequentes[x]) for x in nomes_mais_frequentes])

def c(ficheiro_nome):
    ficheiro = open(ficheiro_nome, 'r')

    linha = ficheiro.readline()
    frequencia = {}

    while linha:
        lista = re.split(r',', linha)

        for elemento in lista[1:]:
            x = re.match(r'[a-zA-Z]*(Irmao|Tio|Sobrinh[ao]|Pai|[nN]et[ao]|Meio|[Aa]vo)[a-zA-Z ]*\.',
            elemento)
            if x:
                relacao = x.group()[:-1]
                if relacao not in frequencia:
                    frequencia[relacao] = 1
                else:
                    frequencia[relacao] += 1

        linha = ficheiro.readline()

    ficheiro.close()
    return frequencia

def validateJSON(jsonData):
    try:
        json.load(jsonData)
    except ValueError as err:
        return False
    return True

def d(ficheiro_nome):
    ler = open(ficheiro_nome, 'r')
    escrever = open("output.json",'w')
```

```
    escrever.write("[\n")
    lista = []

    for y in range(20):
        linha = ler.readline()
        escrever.write("{")
        campos = re.split(r'::', linha)
        chaves = ['numero', 'data', 'nome', 'pai', 'mae', 'informacao adicional']

        for x in range(len(campos)-1):
            if campos[x] != "":
                escrever.write(f"\n\"{chaves[x]}\": \"{campos[x]}\",")

        escrever.seek(escrever.tell()-1,0)

        escrever.write("\n}")
        if y == 19:
            escrever.write("\n")
        else:
            escrever.write(",\n")

    escrever.write("]")

    ler.close()
    escrever.close()

    ficheiro = open("output.json", 'r')
    res = validateJSON(ficheiro)
    ficheiro.close()
    return res

inpt = -1
ficheiro = 'processos.txt'
while inpt != '0':
    inpt = input(f"Ficheiro: {ficheiro}\nEscolha o que pretende fazer: " +
        "\n0. Sair" +
        "\na. Calcular a frequência de processos por ano" +
        "\nb. Calcular a frequência de nomes" +
        "\nc. Calcular a frequência dos vários tipos de relação" +
        "\nd. Imprimir os 20 primeiros registos num ficheiro em formato Json" +
        "\ne. Alterar ficheiro de input\n")

    if inpt == "a":
        res = a(ficheiro)

        ipt = input("Introduza um ano (Sair - 0): ")
        while ipt != '0':
            if ipt not in res:
                print(f'{ipt}: 0')
            else:
                print(f'{ipt}: {res[ipt]}')

            ipt = input("Introduza um ano (Sair - 0): ")

    elif inpt == 'b':
        frequencia, nomes_mais_frequentes = b(ficheiro)

        ipt = input("Escolha o que pretende fazer: " +
```

```
            "\n0. Sair" +
            "\n1. Calcular a frequência de um nome" +
            "\n2. Calcular o nome mais frequente de um ano\n")
        while ipt != '0':
            if ipt == '1':
                nome = input("Introduza um nome (Sair - 0): ")
                while nome != '0':
                    if nome not in frequencia:
                        print(f'{nome}: 0')
                    else:
                        print(f'{nome}: {frequencia[nome]}')

                    nome = input("Introduza um nome (Sair - 0): ")
            elif ipt == '2':
                ano = input("Introduza um ano (Sair - 0): ")
                while ano != '0':
                    listaX = [x for x,y in nomes_mais_frequentes if x == ano]
                    listaY = [y for x,y in nomes_mais_frequentes if x == ano]
                    if ano not in listaX:
                        print(f'{ano}: 0')
                    else:
                        print(f'{ano}: {listaY[0]}')

                    ano = input("Introduza um ano (Sair - 0): ")
            else:
                print("Opção inválida.")

            ipt = input("Escolha o que pretende fazer: " +
                "\n0. Sair" +
                "\n1. Calcular a frequência de um nome" +
                "\n2. Calcular o nome mais frequente de um ano\n")
    elif inpt == 'c':
        print(c(ficheiro))
    elif inpt == 'd':
        print(d(ficheiro))
    elif inpt == 'e':
        ficheiro = input("Escolha o novo ficheiro: ")
    elif inpt != '0':
        print("Opção inválida.")
```