Processamento de Linguagens
MIEI (3º Ano)
Trabalho Prático Nº 2 (GAWK)
Relatório de Desenvolvimento

Rui Costa	Rafael Silva	Ricardo Pereira
(79947)	(74264)	(77045)

28 de Abril de 2019

Resumo

Para a segunda fase do trabalho prático de Processamento de Linguagens foi-nos proposta a implementação de um processador de linguagens regulares utilizando GAWK. Para isto houve uma análise acrescida do prolema de modo a proporcionar a escolha mais acertada no que toca à escolha de expressões regulares que servissem o problema da forma mais clara e robusta. Deste modo consideramos satisfatório o resultado obtido assim como os objetivos e respostas às questões do enunciado cumpridos.

Conteúdo

Capítulo 1

Introdução

Com este trabalho prático pretendeu-se o aumento da experiencia em ambiente Linux, assim como no uso de expressões regulares e no uso do GAWK para filtragem de texto. De entre os enunciados disponibilizados, o grupo ficou com a tarefa de desenvolver exercício 2.5, - *Processador de textos preanotados com Freeling.* Não satisfeitos apenas com a resolução deste ponto, decidiu-se também desenvolver o exercício 2.4, - *Processador de CE-TEMPúblico.*

Ao nível dos enunciados dos exercícios 2.4 e 2.5, era requerido, no geral, que se procedesse à contagem de extratos, paragrafos ou frases, se extraísse listas com palavras com determinadas classes e se indicasse a frequência ed ocorrência das mesmas e se determinassem os dicinários implícitos. A grande diferença entre estes dois exercícios resume-se no formato utilizado nos extratos de texto, que implica uma variação no conteúdo das colunas ou linhas.

Capítulo 2

Processador de CETEMPúblico

2.1 Análise de extratos

Para este exercicio a anotação frásica é feita através das seguintes tags XML:

- <**ext**> - Extratos;
- <**p**> - Parágrafos;
- <**s**> - Frases.

A anotação morfossintática é extensa mas, da totalidade da informação que se pode extrair, importa especialmente a denotada por:

- **1ª Coluna** - Palavra;
- **2ª Coluna** - Secção;
- **3ª Coluna** - Semestre;
- **4ª Coluna** - Lema;
- **5ª Coluna** - pos(part of speech);
- **6ª Coluna** - tempoVerbal-modo;
- **7ª Coluna** - num-pessoa;
- **8ª Coluna** - Género;
- **9ª Coluna** - Árvore.

Na imagem em baixo podemos visualizar um exerto da infomação com que iremos trabalhar.

```
<ext n=1 sec=clt sem=92b>
<p par=ext1-clt-92b-1>
<s>
</s>
<t>
Um      clt     92b     um      DET_arti        0       S       M       >N      0       0       0
revivalismo     clt     92b     revivalismo     N       0       S       M       NPHR    0       0       0
refrescante     clt     92b     refrescante     ADJ     0       S       M       N<      0       0       0
</t>
</p>
<p par=ext1-clt-92b-2>
<s>
O       clt     92b     o       DET_artd        0       S       M       >N      0       0       0
7       clt     92b     7=e=meio        NUM_card        0       S       M       SUBJ>   0       0       0
e       clt     92b     7=e=meio        NUM_card        0       S       M       SUBJ>   0       0       0
Meio    clt     92b     7=e=meio        NUM_card        0       S       M       SUBJ>   0       0       0
é       clt     92b     ser     V       PR_IND  3S      0       FMV     0       0       0
um      clt     92b     um      DET_arti        0       S       M       >N      0       0       0
ex-libris       clt     92b     ex-libris       N       0       S       M       <SC     0       0       0
da      clt     92b     de+o    PRP+DET_artd    0       S       F       N<+>N   0       0       0
noite   clt     92b     noite   N       0       S       F       P<      0       0       0
algarvia        clt     92b     algarvio        ADJ     0       S       F       N<      0       0       0
.       clt     92b     .       PU      0       0       0       PONT    0       0       0
</s>
<s>
É       clt     92b     ser     V       PR_IND  3S      0       FMV     0       0       0
uma     clt     92b     uma     NUM_card        0       S       F       <SC     0       0       0
das     clt     92b     de+o    PRP+DET_artd    0       P       F       N<+>N   0       0       0
mais    clt     92b     mais    ADV_quant       0       0       0       >A      0       0       0
antigas clt     92b     antigo  ADJ     0       P       F       >N      0       0       0
discotecas      clt     92b     discoteca       N       0       P       F       P<      0       0       0
do      clt     92b     de+o    PRP+DET_artd    0       S       M       N<+>N   0       0       0
Algarve clt     92b     Algarve PROP    0       S       F       P<      0       0       0
,       clt     92b     ,       PU      0       0       0       PONT    0       0       0
situada clt     92b     situar  V       PCP     S       F       ICL-N<PRED      0       0       0
em      clt     92b     em      PRP     0       0       0       <ADVS   0       0       0
Albufeira       clt     92b     Albufeira       PROP    0       S       F       P<      0       0       0
,       clt     92b     ,       PU      0       0       0       PONT    0       0       0
que     clt     92b     que     SPEC_rel_clb-fs 0       S       M       SUBJ>   0       0       0
continua        clt     92b     continuar       V       PR_IND  3S      0       FS-<SUBJ        0       0       0
a       clt     92b     a       PRP     0       0       0       PRT-AUX<        0       0       0
manter  clt     92b     manter  V       INF     3       0       ICL-AUX<        0       0       0
os      clt     92b     o       DET_artd        0       P       M       >N      0       0       0
traços  clt     92b     traço   N       0       P       M       <ACC    0       0       0
decorativos     clt     92b     decorativo      ADJ     0       P       M       N<      0       0       0
```

Figura 2.1: Extrato do enunciado 2.4

2.2 Filtro de Texto - Sistema de Produção GAWK

2.2.1 Contar o número de Extratos, Parágrafos e Frases

```
# Contar número de extratos, parágrafos, e frases.

BEGIN {
  ext=0;
  p=0;
  s=0;
}

/<\/ext>/ {
  ext++;
}

/<\/p>/ {
  p++;
}

/<\/s>/ {
  s++;
}

END {
    print "FileName: " FILENAME
    print "Extratos (<ext>): " ext
    print "Paragrafos (<p>): " p
    print "Frases (<s>): " s
}
```

Figura 2.2: Código do ponto 1

Para este ponto, entre os blocos *BEGIN* e *END* foram usados três blocos, com um padrão de pesquisa em cada um, para incrementar cada um dos contadores.

- /<\/**ext**>/ - Encontra qualquer linha que contenha </ext>;
- /<\/**p**>/ - Encontra qualquer linha que contenha </p>;
- /<\/**s**>/ - Encontra qualquer linha que contenha </s>;

Sendo que os contadores de cada um destes elementos apenas incrementam quando é encontrada uma tag indicativa do final de cada elemento, conseguimos contar o numero de elementos presentes no documento.

2.2.2 Extrair a lista das multi-word-expressions e respetivo número de ocorrências

```
# Contar número de ocorrências de cada multi word expression.
# Este script é muito simples, sempre que acabamos de capturar uma multi word expression
# incrementamos array no indíce igual a essa multi word expression. Isto é facilitado
# pelo facto de que se esse indíce ainda não existir, é automaticamente inicializado a zero.
#
# Depois de terminar, reparei que estávamos a contar algumas expressões idênticas
# como diferentes por causa das maiúsculas. Por essa razão, adicionamos tolower()

BEGIN {
  capture=0;
  justfinished=0;
  mwe="";
}

/<\/mwe>/ {
  capture=0
  justfinished=1
}

capture == 1 {
  mwe = mwe " " tolower($1)
}

justfinished == 1 {
  justfinished=0
  array[mwe]++
  mwe=""
}

/<mwe.*>/ {
  capture=1
}

END {
    for (ind in array) {
      print ind "," array[ind]
    }
}
```

Figura 2.3: Código do ponto 2

No bloco *BEGIN* começamos por inicializar algumas variáveis:

- **capture** é inicializada a zero, sendo usada como flag assinalando a existencia de uma multi-word expression, guardando assim a palavra da linha atual;
- **justfinished** é inicializada a zero, sendo também uma flag que é ativada apenas ativada quando se chega ao fim de uma multi-word-expression. Esta flag indica se se deve guardar a multi-word-expression que se acumulou até ao momento;
- **mwe** é inicialixada com uma string vazia, sendo aqui acumulada cada multi-word-exrpession, à medida que é lida.

De seguida, entre os blocos *BEGIN* e *END* tem-se quatro blocos, sendo cada um deles bastante simples: O primeiro bloco deteta o fim de um bloco de uma multi-word-expression(</mwe>), e quando é ativado desativa a flag *capture* e ativa a flag *justfinished*. O segundo bloco é ativado quando a flag *capture* se encontra ativada, concatena a string mwe acumulada até ao momento com a palavra da linha atual. O uso de *tolower()* assegura que não são mantidos dois contadores para a mesma multi-word-expression com capitalização diferente. O terceiro bloco é ativado quando a flag *justfinished* está ativa, e devendo esta flag apenas ter efeito uma vez, desliga-a. De seguida incrementa o valor do índice com o valor de *mwe* em array. Esta lógica acaba por ser bem simples, uma vez que o awk inicializa automaticamente o valor do array *mwe* a zero, caso este nao tenha sido inicializado anteriormente.

No bloco *END* procede-se à impressão do array de multi-word-expressions, impromindo a expressão e o seu contador, separados por uma virgula.

2.2.3 Calcule a lista dos verbos PT: (Lema para palavras com pos = V) e respeetivo número de ocorrências

```
BEGIN {}

$5 == "V" {
  lema = tolower($4)
  array[lema]++
}

END {
    for (i in array) {
      print i "," array[i]
    }
}
```

Figura 2.4: Código do ponto 3

Para este ponto foi usado um bloco entre *BEGIN* e o *END* que é ativado quando a linha tem o valor de pos igual a *V*.

Quando o bloco é ativado, obtém-se o lema na quarta coluna usando *tolower()* de modo a evitas contadores para o mesmmo verbo com capitalização diferente, e o valor de *array[lema]* é incrementado.

Por fim, no bloco *END*, imprimimem-se os verbos e seu respetivo contador.

2.2.4 Determinar o dicionário implícitp no córpora - calcule a lista das palavras associando-lhes os possíveis (lema,pos)

Neste ponto são usados dois arrays:

- **array[palavra]** é uma string de vários (lema, pos) que foram associados à palavra do índice, separados por vígulas.
- **arrayaux[palavra lema pos]** é um array de palavras que tomam o valor 1 quando a combinação de palavra lema/pos já foi encontrada anteriormente, de maneira a evitar repetições no dicionário.

O único bloco entre *BEGIN* e *END* é ativado para todas as linhas que tenham pelo menos 5 campos, sendo que o último se refere ao point of speech(pos).

Para evitar repetições, a palavra e o lema são capturados com *tolower()*. O campo pos encontra-se sempre em maiúsculas e por isso nao precisamos de aplocar o *tolower()*.

De seguida, se a combinaçã de palavra/lema/pos ainda nao foi encontrada, a flag arrayaux ainda nao se encontra ativa, ativa-se essa mesma flag e adiciona-se essa mesma combinação ao dicionário. Inicializa-se, depois, a entrada em array dessa palavra como string vazia e, depois, junta-se essa string à combinação.

Por fim, no bloco *END*, é imprimido um array, com a palavra e a string das várias combinações, separadas por vírgulas.

```
BEGIN {}

# pos is $5
NF >= 5 {

  palavra = tolower($1)
  lema = tolower($4)
  pos = $5

  # Evitar entradas repetidas
  if (!((palavra lema pos) in arrayAux)) {

    # Marcar esta combinacao para evitar repetidos
    arrayAux[palavra lema pos] = 1

    # Inciar a entrada no dicionario
    if (!(palavra in array)) {
      array[palavra] = ""
    } else {
      # Adicionar virgula
      array[palavra] = array[palavra] ","
    }

    # Esta lista de parts of speech por extenso não é exaustiva
    if (pos == "V") {
      pos = "Verbo"
    } else if (pos == "N") {
      pos = "Nome"
    } else if (pos == "ADJ") {
      pos = "Adjetivo"
    } else if (pos == "ADV") {
      pos = "Advérbio"
    } else if (pos == "DET_artd") {
      pos = "Determinante artigo definido"
    } else if (pos == "DET_arti") {
      pos = "Determinante artigo indefinido"
    } else if (pos == "NUM_year_date_card") {
      pos = "Número cardinal ano/data"
    } else if (pos == "NUM_card") {
      pos = "Número cardinal"
    }

    array[palavra] = array[palavra] "(" lema "," pos ")"
  }
}

END {
    for (ind in array) {
      print ind "  :  " array[ind]
    }
}
```

Figura 2.5: Código do ponto 4

2.3 Análise de resultados

```
FileName: Files/Cetempublico01.txt
Extratos (<ext>): 5454
Paragrafos (<p>): 12843
Frases (<s>): 28863
```

Figura 2.6: Resultado do ponto 1, enunciado 2.4

```
uma vez                         47
de resto                        48
dezenas de                      48
tudo o que                      48
desta vez                       51
em conta                        52
dentro de                       53
por causa                       54
pela primeira vez               55
como se                         57
em casa                         57
relativamente                   57
de novo                         58
ao mesmo tempo                  59
ainda que                       61
por parte                       63
cada um                         64
um pouco                        64
fora                            66
cada vez mais                   68
a partir de                     71
desde que                       71
bem como                        75
```

Figura 2.7: Resultado do ponto 2, enunciado 2.4

```
dominar                 51
fornecer                51
indicar                 51
adquirir                52
comentar                52
desejar                 52
optar                   52
solicitar               52
transmitir              52
visar                   52
julgar                  53
suceder                 53
fugir                   54
merecer                 54
prender                 54
relacionar              54
alcançar                55
corresponder            55
repetir                 55
sublinhar               55
beneficiar              56
compor                  56
convidar                56
derrotar                56
informar                56
prosseguir              56
valer                   56
crescer                 57
interpretar             57
respeitar               57
```

Figura 2.8: Resultado do ponto 3, enunciado 2.4

```
colocar-se  :  (colocar+se,V_v-pron_mv_vH+PERS_refl),(colocar+se,V+PERS)
colocaram  :  (colocar,Verbo)
colocarem  :  (colocar,Verbo)
colocaria  :  (colocar,Verbo)
colocará  :  (colocar,Verbo)
colocasse  :  (colocar,Verbo)
colocava  :  (colocar,Verbo)
colocação  :  (colocação,Nome)
colocou  :  (colocar,Verbo)
colocou-se  :  (colocar+se,V_v-pron_mv_fmc_vH+PERS_refl),(colocar+se,V+PERS)
colocá-la  :  (colocar+ela,V+PERS)
colocá-las  :  (colocar+elas,V+PERS)
colocá-lo  :  (colocar+ele,V+PERS)
colombiana  :  (colombiano,Adjetivo)
colombianas  :  (forças=armadas=revolucionárias=colombianas,PROP)
```

Figura 2.9: Resultado do ponto 4, enunciado 2.4

Capítulo 3

Processador de textos preanotados com Freeling

Este tema visa a análise de ficheiros sobre os *corpora*, que agrupam grandes quantidades de texto, e aos quais se adiciona informação de anotação frásica norfossintática. Para este exercício foram fornecidos cinco *datasets* em formato *Freeling*, fl0, fl1, fl2, harrypotter1 e por fim harrypotter2.

3.1 Análise dos extratos

Para uma melhor compreesão do problema proposto, a análise aprofundada dos extratos foi fundamental, sendo esta baseada na observação dos elementos das diferentes colunas do ficheiro. Tendo isto em conta apresentamos de seguida uma parcela do texto fonte com todas os campos relativos ao problema.

```
4  de                    de                    SP      SP  pos=adposition|type=preposition                                    - - (sp-de:4
- - - -
5  Mário_Centeno         mário_centeno         NP00000 NP  pos=noun|type=proper                                               - - (sn:5(grup-nom-ms:5(w-ms:5))))
- - - -
6  para                  para                  SP      SP  pos=adposition|type=preposition                                    - - (grup-sp:6(prep:6)
- - - -
7  a                     o                     DA0FS0  DA  pos=determiner|type=article|gen=feminine|num=singular             - - (sn:9(espec-fs:7(j-fs:7))
- - - -
8  chamada               chamar                VMP00SF VMP pos=verb|type=main|mood=pastparticiple|num=singular|gen=feminine  - - (grup-nom-fs:9(s-a-fs:8(parti-fs:8))
- - - -
9  presidência           presidência           NCFS000 NC  pos=noun|type=common|gen=feminine|num=singular                     - - (grup-nom-fs:9(n-fs:9)))))
- - - -
10 de                    de                    SP      SP  pos=adposition|type=preposition                                    - - (sp-de:10
- - - -
11 o                     o                     DA0MS0  DA  pos=determiner|type=article|gen=masculine|num=singular           - - (sn:13(espec-ms:11(j-ms:11))
- - - -
12 chamado               chamar                VMP00SM VMP pos=verb|type=main|mood=pastparticiple|num=singular|gen=masculine  - - (grup-nom-ms:13(s-a-ms:12(parti-ms:
12)) - - - -
13 Eurogrupo             eurogrupo             NP00000 NP  pos=noun|type=proper                                               - - (grup-nom-ms:13(w-ms:13)))))
- - - -
14 revela                revelar               VMIP3S0 VMI pos=verb|type=main|mood=indicative|tense=present|person=3|num=singular - - (grup-verb:14(verb:14))
- - - -
15 a                     o                     DA0FS0  DA  pos=determiner|type=article|gen=feminine|num=singular           - - (sn:16(espec-fs:15(j-fs:15))
- - - -
16 actualidade           actualidade           NCFS000 NC  pos=noun|type=common|gen=feminine|num=singular                     - - (grup-nom-fs:16(n-fs:16)))
- - - -
17 de                    de                    SP      SP  pos=adposition|type=preposition                                    - - (sp-de:17
- - - -
```

Figura 3.1: Extrato do código fonte

Com isto verificamos a existência de colunas separadas nos diferentes atributos, separados por espaços, comuns em quase todos os ficheiros fornecidos.

3.2 Filtro de texto - Sistema de Produção GAWK

3.2.1 Contagem de Extratos

Neste exercício, foi pedido no enunciado a contagem dos números de extratos nos vários ficheiros. Para isto, inicialmente no campo *BEGIN* definiu-se a variável count a zero que serve de contador de extratos. Para o corpo do programa definiu-se a clausula de filtragem, em que, quando se encontra *$0~/^$/* incrementa o contador. Por fim, na secção *END* é imprimido o valor finala da variável conut e o nome do ficheiro interpretado.

```
BEGIN {
    count=0;
}

$0~/^$/{
    count++
}

END {
    print "Numero de extratos: " count
    print "Nome do ficheiro: " FILENAME
}
```

Figura 3.2: Código do exercício 1

3.2.2 Calcular a lista dos personagens do HarryPotter (nome próprio) e respetivo número de ocorrências

Neste ponto foi pedida a contagem do número de personagens do Harry Potter, indicando o seu nome próprio e respetivo número de ocorrências. Para isto precisamos de percorrer as linhas e contar aquelas que contêm *noun* e *proper*, assim como guarda-los numa lista.

```
BEGIN{}

/noun.*proper/{
        nomes[$3]++;
}

END{
        for(ind in nomes){
                print nomes[ind] "," ind
        }
}
```

Figura 3.3: Código do exercício 2

Num instante inicial, no bloco *BEGIN* não se encontra nada declarado devido ao facto de nao ser necessária nenhuma variavél inicial.

No corpo do programa apenas foi necessário definir o padrão de procura por nome e, uma vez encontrados são guardados na estrutura *nomes*.

Por fim, no bloco *END* foi implementado um ciclo que precorre todo a estrutura *nomes* e imprime todos os elementos desta lista, formada pelos resultados pretendidos.

3.2.3 Calcular a lista dos verbos, substantivos, adjetivos e advérbios PT e criar um ficheiro HTML com cada uma destas listas

Para este ponto, era exigido o cálculo de quatro listas, de verbos, substantivos, adjetivos e advérbios de um determinado excerto de texto. Após esta filtragem, para cada uma das listas, seria criado um ficheiro HTML com o seu conteúdo. Para isto, consultamos cada linha de texto e, para os verbos verificou-se o campo *verb*, para os substantivos o *noun*, para os adjetivos o *adjective* e para os advérbios o campo *adverb*.

```
BEGIN {
        countVerbos=0; countSubs=0; countAdj=0; countAdv=0;
        printf "<html>\n<head>\n\t<meta charset='utf-8'/>\n\t</head>\n\t<body>\n\t<h2>Index Alinea 3 - %s</h2>\n\t<table border='1'>\n\t\n", ARGV[1] > "indexAlinea3.html";
        printf "<html>\n<head>\n\t<meta charset='utf-8'/>\n\t</head>\n\t<body>\n\t<h3>Lista de Verbos</h3>\n\t<table border='1'>\n\t<tr><th>Verbo</th></tr>\n" > "Alinea3/verbos.html";
        printf "<html>\n<head>\n\t<meta charset='utf-8'/>\n\t</head>\n\t<body>\n\t<h3>Lista de Substantivos</h3>\n\t<table border='1'>\n\t<tr><th>Substantivo</th></tr>\n" > "Alinea3/substantivos.html";
        printf "<html>\n<head>\n\t<meta charset='utf-8'/>\n\t</head>\n\t<body>\n\t<h3>Lista de Adjetivos</h3>\n\t<table border='1'>\n\t<tr><th>Adjetivo</th></tr>\n" > "Alinea3/adjetivos.html";
        printf "<html>\n<head>\n\t<meta charset='utf-8'/>\n\t</head>\n\t<body>\n\t<h3>Lista de Adverbios</h3>\n\t<table border='1'>\n\t<tr><th>Adverbio</th></tr>\n" > "Alinea3/adverbios.html";
}

/\=verb/{
         verbos[$3]++;
}

/\=noun/{
        nomes[$3]++;
}

/\=adjective/{
        adjectivos[$3]++;
}

/\=adverb/{
        adverbios[$3]++;
}

END {
        for(ind in verbos){
                printf "\t\t<tr><td>%s</td></tr>\n", ind > "Alinea3/verbos.html";
                countVerbos++;
        }
        for(ind in nomes){
                printf "\t\t<tr><td>%s</td></tr>\n", ind > "Alinea3/substantivos.html";
                countSubs++;
        }
        for(ind in adjectivos){
                printf "\t\t<tr><td>%s</td></tr>\n", ind > "Alinea3/adjetivos.html";
                countAdj++;
        }
        for(ind in adverbios){
                printf "\t\t<tr><td>%s</td></tr>\n", ind > "Alinea3/adverbios.html";
                countAdv++;
        }
        printf "\t\t<tr><th><a href=\"Alinea3/verbos.html\">Verbos</a></th><th>%s</th></tr>\n", countVerbos > "indexAlinea3.html";
        printf "\t\t<tr><th><a href=\"Alinea3/substantivos.html\">Substantivos</a></th><th>%s</th></tr>\n",countSubs > "indexAlinea3.html";
        printf "\t\t<tr><th><a href=\"Alinea3/adjetivos.html\">Adjetivos</a></th><th>%s</th></tr>\n", countAdj > "indexAlinea3.html";
        printf "\t\t<tr><th><a href=\"Alinea3/adverbios.html\">Adverbios</a></th><th>%s</th></tr>\n", countAdv > "indexAlinea3.html";
        printf "\t</table>\n\t</body>\n</html>" > "Alinea3/verbos.html";
        printf "\t</table>\n\t</body>\n</html>" > "Alinea3/substantivos.html";
        printf "\t</table>\n\t</body>\n</html>" > "Alinea3/adjetivos.html";
        printf "\t</table>\n\t</body>\n</html>" > "Alinea3/adverbios.html";
        printf "\t</table>\n\t</body>\n</html>" > "indexAlinea3.html";
}
```

Figura 3.4: Código do exercício 3

Analisando o código desenvolvido para este exercício, no bloco *BEGIN* temos a inicialização dos contadores de todos os atributos que queremos considerar a zero, assim como a vriação de vada página *HTML*.

Já no corpo do programa, foram definidos vários padrões de procura para as seguintes situações:

- \=**verb**/**verbos[$3]**++ - filtra todos os verbos e coloca-os no respetivo array;
- \=**noun**/**nomes[$3]**++ - filtra todos os substantivos e coloca-os no respetivo array;
- \=**adjective**/**adjetivos[$3]**++ - filtra todos os adjetivos e coloca-os no respetivo array;
- \=**adverb**/**adverbios[$3]**++ - filtra todos os advérbios e coloca-os no respetivo array.

Por fim no bloco *END* colocamos todos os elementos presentes diferentes arrays no ficheiro HTML.

3.2.4 Determinar o dicionáriono córpora - lista contendo os lema, pos e palavras dele derivadas

Este último exercício pede a criação de um dicionário, ou seja, uma lista com a palavra, lema e sua pos. Tomamos assim a decisão de ordenar o dicionário com a palavra em primeiro lugar com o formato: **palavra : (lema, pos)**.

Como podemos ver na imagem do código desenvolvido para este problema, o bloco *BEGIN* encontra-se vazio pois não temos variáveis iniciais.

No corpo do programa começa-se por abter a palavra, o lema e a pos usando o *tolower()* , para evitar contadores diferentes para a mesma palavra. Depois usamos dois arrays:

- **array[word]** - string de vários (lema,pos) que foram associados à palavra do índice, separados por vírgulas.
- **array[word lema pos]** - allay de flags que tomam o valor 1 caso a combinação palavra/lema/pos já tiver sido encontrada, de forma a evitar repetições no dicionário.

```
BEGIN{}

NF>0{
    word = tolower($2)
    lema = tolower($3)
    pos = $5

    # Evitar entradas repetida
    if(!((word lema pos) in arrayAux)){

        # Marcar esta combinação encontrada para evitar repetições
        arrayAux[word lema pos] = 1

        # Iniciar entrada no dicionario
        if(!word in array){
            array[word] = ""
        } else {
            # Adiciona virgula
            array[word] = array[word] ","
        }

        array[word] = array[word] "(" lema "," pos ")"
    }
}

END{
    for(ind in array){
        print ind, "::", array[ind]
    }
}
```

Figura 3.5: Código do exercício 4

3.3 Análise de resultados

```
Numero de extratos: 5569
Nome do ficheiro: Files/harrypotter1
```

Figura 3.6: Resultado do ponto 1, enunciado 2.5, Ficheiro HarryPotter1

```
1143  harry
400   ron
337   hagrid
241   hermione
150   snape
143   dumbledore
122   dudley
109   malfoy
108   neville
106   vernon
100   quirrell
96    mc_gonagall
92    gryffindor
67    hogwarts
57    petúnia
56    slytherin
54    quidditch
52    dursley
51    potter
49    wood
48    filch
47    dursleys
39    voldemort
38    harry_potter
35    percy
31    peeves
29    muggles
28    fred
25    weasley
25    norbert
24    gringotts
24    fluffy
22    ollivander
22    hermione_granger
```

Figura 3.7: Excerto do Resultado do ponto 2, enunciado 2.5, Ficheiro HarryPotter1

```
1429  harry
665   ron
296   hermione
191   lockhart
147   hagrid
144   dobby
137   dumbledore
135   malfoy
111   riddle
103   slytherin
102   ginny
97    fred
92    snape
89    gryffindor
86    mc_gonagall
79    harry_potter
79    george
75    hogwarts
63    muggles
62    percy
60    mrs._weasley
51    mr._weasley
48    filch
46    câmara_dos_segredos
44    nick
41    potter
41    murta
41    mr._malfoy
39    vernon
38    quidditch
37    goyle
```

Figura 3.8: Excerto do Resultado do ponto 2, enunciado 2.5, Ficheiro HarryPotter2

Index Alinea 3 - Files/harrypotter1

Verbos	1264
Substantivos	2736
Adjetivos	698
Adverbios	329

Figura 3.9: Resultado do ponto 3, enunciado 2.5, Ficheiro HarryPotter1

Lista de Verbos

Verbo
assinar
poder
comedir
recusar
enlouquecer
depender
erguer
recair
prometer

Figura 3.10: Excerto do Resultado do ponto 3, enunciado 2.5, Ficheiro HarryPotter1

Lista de Substantivos

Substantivo
jim
poder
dentada
podes
elefante
direita
lenha
ás
inocente
subida
espinho
hannah_abbott
cimento
meteorologista

Figura 3.11: Excerto do Resultado do ponto 3, enunciado 2.5, Ficheiro HarryPotter1

Lista de Adjetivos

Adjetivo
eficaz
ave
and
escanzelado
sinistro
doente
inultrapassável
escolar
doloroso

Figura 3.12: Excerto do Resultado do ponto 3, enunciado 2.5, Ficheiro HarryPotter1

Lista de Adverbios

Adverbio
instintivo
precipitado
de_acordo
todavia
à_medida_que
infeliz
igual
tenh'aqui
agressivo
desde_pequeno

Figura 3.13: Excerto do Resultado do ponto 3, enunciado 2.5, Ficheiro HarryPotter1

```
abafada :: ,(abafar,VMP)
abafado :: ,(abafar,VMP)
abafador :: ,(abafador,NC)
abafando :: ,(abafar,VMG)
abafar :: ,(abafar,VMN)
abaixo :: ,(abaixo,RG)
abalada :: ,(abalar,VMP)
abalado :: ,(abalar,VMP)
abanando :: ,(abanar,VMG)
abanava :: ,(abanar,VMI)
abandonar :: ,(abandonar,VMN)
abandonaram :: ,(abandonar,VMI)
abandonavam :: ,(abandonar,VMI)
abandoná :: ,(abandonar,VMN)
abanou :: ,(abanar,VMI)
abateu :: ,(abater,VMI)
abatido :: ,(abater,VMP)
abelhudo :: ,(abelhudo,AQ)
abençoado :: ,(abençoar,VMP)
aberta :: ,(aberto,AQ),(aberta,NC)
```

Figura 3.14: Excerto do Resultado do ponto 4, enunciado 2.5, Ficheiro HarryPotter1

```
aborrecê :: ,(aborrecer,VMN)
abotinados :: ,(abotinar,VMP)
abram :: ,(abrir,VMS)
abrandar :: ,(abrandar,VMN)
abrandou :: ,(abrandar,VMI)
abras :: ,(abrir,VMS)
abraçou :: ,(abraçar,VMI)
abraçá :: ,(abraçar,VMN)
abre :: ,(abrir,VMI)
abri :: ,(abrir,VMN),(abrir,VMI)
abria :: ,(abrir,VMI)
abriam :: ,(abrir,VMI)
abrigada :: ,(abrigar,VMP)
abrigava :: ,(abrigar,VMI)
abrindo :: ,(abrir,VMG)
abrir :: ,(abrir,VMN)
abrira :: ,(abrir,VMI)
```

Figura 3.15: Excerto do Resultado do ponto 4, enunciado 2.5, Ficheiro HarryPotter2

Capítulo 4

Exercicio Extra - Previsão de palavras

Na resolução deste exercício extra foram usados os *datasets* do exercício **2.5**. Os mesmo ficheiro na foram explicados na sua respetiva secção.

Criamos este exercicio extra com o objetivo de simular o que na actualidade está presente nos teclados dos smartPhones, que é a denotada previsão da próxima palavra a ser introduzida tendo em conta o que foi escrito. Neste casos os nossas *datasets* serão usados como a nossa base de conhecimento.

4.1 Secção BEGIN

```
BEGIN {

  exists=0
  word
  saveProx=0
  saved=0
  numSugestion=0
  sugestedWord
  putComma=0

  pritn "<!DOCTYPE html>"
  print "<html lang='pt-PT'>"
    print "<body>"

    print "<h1>" word "</h1>"

    print "<div id='piechart'></div>"

    print "<script type='text/javascript' src='https://www.gstatic.com/charts/loader.js'></script>"

    print "<script type='text/javascript'>"
    print "// Load google charts"
    print "google.charts.load('current', {'packages':['corechart']});"
    print "google.charts.setOnLoadCallback(drawChart);"

    print "// Draw the chart and set the chart values"
    print "function drawChart() {"
    print "var data = google.visualization.arrayToDataTable(["
}
```

Figura 4.1: Secção BEGIN do algoritmo

Neste Bloco *Begin* usaram-se as seguintes variáveis:

- **exists** guarda o numero de vezes que a palavra á procurar foi encontrada. No caso seria a palavra acabada de escrever no teclado.
- **word** palavra que o utilizador pretende pesquisar.

- **savedProx** trata-se de uma flag, que tem como função de, após ser introduzida a palavra a procurar, fica a 1, permitindo registar a palavra seguinte para entrar na contagem de probabilidades.
- **saved** número de palavras guaradas, para entrar em contas de probabilidades.
- **numSugestion** permite saber no fim qual é o número de vezes que a palavra mais frequente apareceu, tem efeitos de debug.
- **sugestedWord** é a palavra seguinte que foi sugerida.
- **putComma** é inicializada a 0, pois assim nao é possivel introduzir uma ","na variável **data** no HTML quando o algoritmo é corrido pela primeira vez.

As restantes das linhas são para a inicialização do HTML vem como do seu script e funções.

4.2 Secção de Processamento

```
$2~/^[a-zA-Z]+[^']?[a-zA-Z]*$/ {

    # word after the one which we are looking for
    if(saveProx==1) {
        previsao[$2]++;
        saveProx=0;
        saved++;
    }

    # check if its the word we are looking for
    if(word==$2) {
        exists++;
        saveProx=1;
    }

}
```

Figura 4.2: Secção de processamento do algoritmo

Foi criada uma ER para ser usada neste algoritmo. A mesma tem como função filtar palvras, nao captando a sua pontuação, filtrando também o carácter "' "(pelica), visto que os mesmo fazem estragos nas páginas HTML.

Dentro temos dois **if**, o segundo **if** irá verificar se a palavra em questão é aquela que o utilizador introduziu, no primeiro **if** consiste em verifcar se aquele palavra é para guardar para a possível sugestão.

```
END {
    for (i in previsao) {

        if (previsao[i] > numSugestion) {
            numSugestion=previsao[i];
            sugestedWord=i;
        }

        # Print , after the first argument
        if (putComma) print ",\n"
        else print "['Prevision', 'Word possibilities'],"

        # Print the arguments for html graph
        print "['" i "', " previsao[i] "]"

        putComma=1
    }

  print "]);"

  print "// Optional; add a title and set the width and height of the chart"
  print "var options = {'title':'Palavra seguinte: " sugestedWord "', 'width':880, 'height':640};"

  print "// Display the chart inside the <div> element with id='piechart'"
  print "var chart = new google.visualization.PieChart(document.getElementById('piechart'));"
  print "chart.draw(data, options);"
  print "}"
  print "</script>"

  print "</body>"
  print "</html>"
}
```

Figura 4.3: Secção de END do algoritmo

Nesta parte do algoritmo, dentro do ciclo **for**, o primeiro **if** serve para obter a palavra mais frequente dentro das palavras próximas possíveis.

Na página HTML será apresentada essa palavra em formato de destaque.

No segundo **if** irá verificar a variável putComma que foi explicada em cima. O seguinte print serve para colocar a infomação do array no HTML.

4.3 Análise de resultados

mas

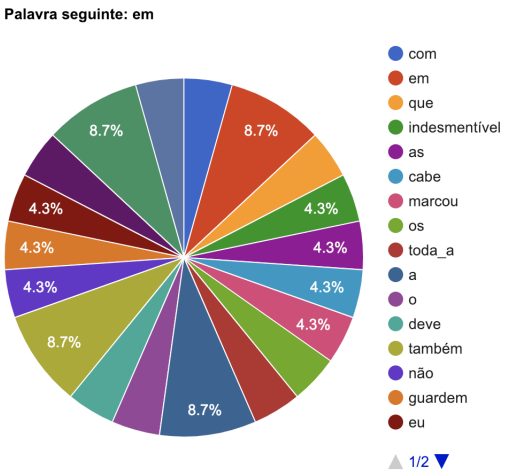

Figura 4.4: Resultado do exercicio extra, usando o fl0, para a palavra 'mas'

Harry

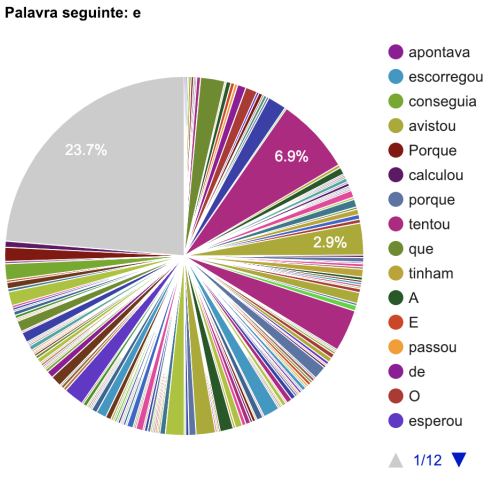

Figura 4.5: Resultado do exercicio extra, usando o ficheiro harrypotter1, para a palavra 'Harry'

Capítulo 5

Conclusão

A resolução deste segundo trabalho prático foi bastante importante e enriquecedora, pois permitiu aos membros do grupo explorar ainda mais o conceito de ER e a utilizão da ferramenta GAWK, sendo assim possível utilizar os conhecimentos adquiridos nas aulas práticas e teóricas da Unidade Curricular de Processamento de Linguagens.

Deste modo, foram adquiridos conhecimemtos de programação da linguagem *awk* e da sua ferramenta que foi mencionada em cima.

Tendo o grupo notado que a utilização desta duas ferramentas é muito mais eficaz para este tipo de problemas, do que as linguagens que estamos habituados a utilizar, estando satisfeitos com o trabalho desenvolvido.

Na resolução de ambos os exercicios, onde o grupo encontrou mais dificuldade foi na análise dos *datasets*.

Na maioria dos exercicios resolvidos a sua resolução foi simples, na execção de um ou dois em que os algortimos dos mesmos requeriam mais trabalho por parte do grupo.

No exercicio extra, foi-nos mais dificil e implementação do HTML, visto que o grupo nao tinham qualquer conhecimento de como construir gráficos em HTML.

Em suma, é feita uma apreciação positiva ao trabalaho desenvolvido, visto que as funcinalidades pedidas foram realizada, bem como uma exercicio extra, que neste caso é o **2.4** e outra alinea extra. O grupo tirou conhecimentos deste trabalho e sendo capaz que os utilizar num futuro.