

Processamento de Linguagens (3º ano de MiEI)

Trabalho Prático 1

Relatório de Desenvolvimento de um Normalizador de ficheiros *BibTex*

Gustavo da Costa Gomes-Aluno
(72223)

José Carlos da Silva Brandão Gonçalves-Aluno
(71223)

Tiago João Lopes Carvalhais-Aluno
(70443)

25 de Março de 2016

Resumo

Isto é um resumo do relatório da unidade curricular Processamento de Linguagens relativamente ao Trabalho Prático 1. Este visa a produção de um Normalizador de ficheiros *BibTex* permitindo a exploração da ferramenta *Flex* acompanhada de uma pequena demonstração de quão poderosa realmente é. Também é possível encontrar numa seção, no capítulo "Anexo", que possui a resolução de um segundo enunciado, que é a conversão de ficheiros *Zim-Wiki* para slides *Beamer*.

Conteúdo

1	Introdução	2
2	Análise e Especificação	3
2.1	Descrição informal do problema	3
2.2	Especificação do Requisitos	4
2.2.1	Dados	4
3	Concepção/desenho da Resolução	5
3.1	Estruturas de Dados	5
3.1.1	Alínea a	5
3.1.2	Alínea b	6
3.1.3	Alínea c	7
4	Codificação	8
4.1	Alternativas, Decisões e Problemas de Implementação	8
4.1.1	Alínea a	8
4.1.2	Alínea b	10
4.1.3	Alínea c	11
4.2	Testes realizados e Resultados	12
4.2.1	Alínea a	12
4.2.2	Alínea b	13
4.2.3	Alínea c	14
5	Conclusão	15
A	Código do Programa	16
A.1	Alínea a	17
A.2	Alínea b	22
A.3	Alínea c	23
A.4	Extra- Zim to Beamer	24

Capítulo 1

Introdução

Este trabalho envolveu o desenvolvimento de um Normalizador de ficheiros *BibTex*, um dos enunciados disponibilizados, no qual se procede á sua análise tendo em conta os seguintes pontos,

Enquadramento Utilização de Expressões Regulares e Filtros de Texto com o objetivo de produzir novos documentos a partir de padrões existentes num outro ficheiro.

Estrutura do documento Este documento possui um anexo com todo o código produzido em cada uma das alíneas, uma conclusão final que une o exercício e as respectivas soluções elaboradas e ainda capítulos elucidativos de cada tarefa a desempenhar em cada alínea.

Resultados Os resultados serão apreciados nos respectivos capítulos correspondentes a cada uma das alíneas desenvolvidas e cujo ficheiro(ou partes dele) estará(ão) no Anexo correspondente.

Conteúdo do documento Contém a explicação do problema em si, bem como a apresentação das soluções produzidas para colmatar essa situação, auxiliada com documentação e código produzido, presente nos Anexos.

Estrutura do Relatório

Este relatório possui quatro capítulos, uma conclusão, um capítulo extra dedicado a Anexos e a respectiva Bibliografia utilizada durante a realização deste projecto. Os capítulos são Análise e Especificação do problema, Especificação dos Requisitos, Arquitectura da solução para cada um dos sub-problemas indicados no enunciado global que envolverá a explicação das estruturas de dados utilizadas e por fim o capítulo designado Codificação, que incluirá alguns aspectos relevantes de todos os testes realizados para a verificação do correcto funcionamento.

Capítulo 2

Análise e Especificação

2.1 Descrição informal do problema

BibTex é uma ferramenta de formatação de citações bibliográficas em documentos *Latex*, que separa a bibliografia consultada do restante conteúdo. Um exemplo desse ficheiro, com a extensão .bib ilustra-se de seguida,

```
1 @InProceedings{CPBFH07e,
2   author = {Daniela da Cruz and Maria Joao Varanda Pereira
3             and Mario Beron and Ruben Fonseca and
4             Pedro Rangel Henriques},
5   title = {Comparing Generators for Language-based Tools},
6   booktitle = {Proceedings of the 1.st Conference on Compiler
7 }
8   year =
9   editor =
10  month =
11  Related Technologies and Applications , CoRTAâ07
12  — Universidade da Beira Interior , Portugal},
13 {2007},
14 {},
15 {Jul},
```

Este enunciado consiste em três alíneas, nas quais são requeridas diferentes tarefas a implementar. Na alínea a é pedido que se elabore um documento *HTML* que contenha a contagem de todas as diferentes categorias presentes no documento lpbib.txt, explicado num capítulo vindouro, *Código do Programa*.

Na alínea b é pedido que se desenvolva uma ferramenta de normalização que faça *pretty-printing*, fazendo a indentação correta em cada campo, escrevendo cada autor numa linha e que coloque no início da mesma os campos autor e título, e quando um campo estiver entre aspas modifique para chavetas e se escreva o nome dos autores com o seguinte formato, *N.Apelido*.

Por fim, na alínea c é pedido a construção de um grafo que para um determinado autor, á escolha do utilizador, mostre todos os autores que publicaram com esse autor. Para tal, recorrer-se-á á linguagem Dot do *Graph Viz2*, gerando um ficheiro com esse grafo de modo a que possa, posteriormente, desenhar o mesmo através de uma outra ferramenta que faça a leitura desses ficheiros.

2.2 Especificação do Requisitos

Neste trabalho, o objectivo é estimular a utilização de um ambiente *Linux*, da linguagem imperativa *C* e de outras ferramentas de apoio para a resolução de problemas de um modo diferente do habitual, que seria tentar resolver tudo apenas utilizando uma linguagem de programação e, para tal, visam o estudo e o desenvolvimento de Expressões Regulares, bem como, a sua manipulação por forma a atingir o resultado pretendido. Essas expressões são fundamentais para encontrar os padrões para os quais se irá tomar uma ação, que será a transformação do texto, filtrando ou removendo esses. Como auxiliar na realização de filtros de texto recorrer-se-á á utilização de um gerador designado, *Flex*.

Na realização deste problema é necessário concluir uma lista de tarefas que são, especificar os padrões de frases que se quer encontrar no texto fonte, através de Expressões Regulares, identificar as acções semânticas a realizar como reacção ao reconhecimento de cada um desses padrões, identificar as estruturas de dados globais que possa eventualmente precisar para armazenar temporariamente a informação que se vai extraindo do texto fonte ou que se vai construindo á medida que o processamento avança e por fim desenvolver um filtro de texto para fazer o reconhecimento dos padrões identificados e proceder à transformação pretendida, com recurso ao gerador *Flex*.

2.2.1 Dados

Os dados fornecidos são o ficheiro *lpbib.txt*, a ser abordado num capítulo posterior, a definição e utilização da ferramenta *Flex* e a definição de um ficheiro *BibTex*, isto é, as suas características.

É também fornecido o nome de ferramentas de apoio á resolução do problema, sendo neste problema, a ferramenta *GraphViz2*, que permitirá colocar graficamente a informação dos grafos criados, sendo que as interações entre os autores se tornam mais perceptíveis.

Capítulo 3

Concepção/desenho da Resolução

3.1 Estruturas de Dados

3.1.1 Alínea a

Nesta alínea optou-se por utilizar uma *hashtable* como estrutura de dados auxiliar que vai guardando as categorias e o respectivo contador á medida que se vai encontrando um padrão no ficheiro lpbib.txt, ou seja, a acção ao padrão, que permite encontrar as categorias todas ao longo de todo o conteúdo.

Esta estrutura segue a lógica de *Open Addressing*, isto é, através de uma função de *hash* é encontrada a posição onde se irá inserir a categoria capturada pela expressão regular e no caso de essa estar já ocupada vai tentar inserir na posição seguinte, e se chegar á última reinicia, visto que é *circular*. Apenas se implementou funções essenciais, *inserir*, *remover*, *procurar* e *imprimir* o conteúdo desta estrutura e gerar o conteúdo do ficheiro *HTML* pedido. Para verificar se uma posição já possui ou não conteúdo basta verificar a *etiqueta* associada e designada por *state*, no Anexo encontra-se o conteúdo integral da implementação desta estrutura de dados.

3.1.2 Alínea b

3.1.3 Alínea c

Capítulo 4

Codificação

4.1 Alternativas, Decisões e Problemas de Implementação

4.1.1 Alínea a

Um dos problemas de implementação passou por conseguir contabilizar as categorias de forma independente, isto é, após uma análise do documento fonte `lpbib.txt` verificou-se a existência de categorias que contem exatamente os mesmos caracteres mas escritos de diferentes formas. Um exemplo disso é a categoria *inproceedings* que pode também se encontrar como *InProceedings* e como *INPROCEEDINGS*. Então o grande desafio foi separar esta categoria em três, porque apesar de terem os mesmos caracteres, elas representam a mesma categoria mas de forma independente visto que na contagem destas se pretende ter um resultado que mostre o que realmente está no ficheiro fonte.

Na fase de implementação surgiram pequenos problemas relacionados com a expressão regular que foi definida por forma a filtrar apenas a categoria. Visto que essa estava delimitada por dois caracteres, o '@' e o '. Por vezes *yytext* não continha o conteúdo correto, o que revelava que a expressão regular ainda não estava a funcionar corretamente.

Após esse problemas estarem resolvidos conseguiu-se produzir o ficheiro *HTML* sem nenhuma dificuldade, apenas se efetuou a impressão do conteúdo da estrutura de dados que foi armazenando as categorias e atualizando os seus contadores com a indentação e os *headers* que permitem visualizar o ficheiro obtido num *browser*. Esse ficheiro *HTML* produzido chama-se `indexA.html` e pode ser visto no Anexo A.1, bem como o filtro de texto produzido, recorrendo á ferramenta *Flex*, neste documento.

Para terminar falta proceder á análise das expressões regulares utilizadas e a respectiva acção a tomar quando estas forem encontradas.

- i. `@[a?zA?Z]+\{`
- ii. `.\|\\n`

A expressão regular ii. é para filtrar todo o texto, mas é absorvente e por isso é preciso muito cuidado com a sua utilização e ,associado a esta, a acção de fazer *print* que é a acção por defeito, quando se fornece

`}`

A expressão regular i. é a expressão que foi desenvolvida para a resolução do problema pedido, contagem das categorias, que exige inicialmente a captação das categorias e apenas das categorias presentes no ficheiro `lpbib.txt`, que contem muita mais informação. O objectivo desta é encontrar todos os padrões que estejam contidos entre os caracteres '@' e '.', visto que essa é a definição de categoria num ficheiro *BibTex*. E como as categorias apenas podem conter letras temos de restringir os padrões encontrados entre esses dois caracteres ao facto de que só podem ter letras, quer minúsculas quer maiúsculas, daí `'[a?zA?Z]+'`. O símbolo '+' refere-se á possibilidade de encontrar uma

ou mais ocorrências, isto é, entre esses dois caracteres encontra-se apenas uma ou mais letras. Visto que não existe nenhuma restrição quanto á forma da palavra categoria, isto é, por exemplo a exigência de começar por letra maiúscula e seguida apenas de letras minúsculas não é necessário efetuar mais nenhuma restrição ao padrão que permitirá filtrar as categorias.

Associado a esta, última expressão regular, está a acção de copiar o conteúdo de *yytext+1*, que apenas considera o texto após o '@' até *yytext-2*, onde está o '' para um *char** local que será então inserido na estrutura de dados através da instrução *insertTable(ht, str, (int ?) count)* ; sendo que *count* é um contador inicial que apenas serve para iniciar o contador na estrutura de dados. Esta instrução está codificada por forma a verificar logo se a categoria já existe ou não, e caso exista apenas incrementa a ocorrência dessa categoria e ignora o parâmetro *count* recebido. E caso não exista procede então ao início do contador com valor de *count* recebido e insere na estrutura de dados.

Por fim é necessário criar o ficheiro *HTML* pretendido com o conteúdo da estrutura de dados que foi sendo atualizada até se chegar ao fim do ficheiro *lpbib.txt* e para tal na função *main* do ficheiro *tp1A.l* recorreu-se á chamada da função *printHashTable (ht)* ; que foi codificada no ficheiro *hashtable.c* por forma a criar o ficheiro *HTML* com a formatação necessária, produzindo, desse modo, o resultado final desta alínea.

4.1.2 Alínea b

4.1.3 Alínea c

4.2 Testes realizados e Resultados

Mostram-se a seguir alguns testes feitos (valores introduzidos) e os respectivos resultados obtidos: lpbib.txt nas diferentes alíneas deste problema.

4.2.1 Alínea a

Para este problema fizeram-se testes sucessivos até encontrar a expressão regular que permitisse armazenar apenas a categoria na estrutura de dados evitando desse modo que a manipulação de *strings* fosse feita através de funções definidas na linguagem *C*, tal como, *strtok* ou outra semelhante, uma vez que o objectivo é a utilização de expressões regulares que façam todo o trabalho de manipulação de texto.

Os testes realizados passaram todos por correr a seguinte *makefile*

```
1 all: 1 2
2
3 1:
4         flex tp1.l
5
6 2:
7         gcc lex.yy.c hashtable.c -ll -o tp1
```

E de seguida efetuar

```
./tp1 < lpbib.txt
```

E verificando se o resultado obtido era realmente o resultado pretendido. Antes de se chegar ao resultado correcto teve-se de corrigir a situação de contagem de categorias com os mesmos caracteres mas que representavam categorias distintas porque o que acontecia era, para cada uma dessas categorias o contador estava a incluir as outras, mas só mostrava as ocorrências de uma dessas categorias. Por exemplo, para a categoria *INPROCEEDINGS* verifica-se que no ficheiro fonte lpbib.txt apenas ocorre cinco vezes, mas o que acontecia era que se se imprimisse o conteúdo da estrutura de dados ela apresentava apenas essas cinco ocorrências mas o valor do contador não correspondia á contagem real porque estava a incluir os contadores das categorias *InProceedings* e *inproceedings*.

Como resultado final obteve-se o ficheiro *HTML* com o formato pretendido, que é categoria x e o valor do contador para essa categoria x. Pode ser verificada abrindo o ficheiro lpbib.txt e procurando uma categoria qualquer e verificar que o número de ocorrências coincidem e para visualizar o aspecto da solução apenas é necessário abrir o indexA.html com um *browser* qualquer.

4.2.2 Alínea b

4.2.3 Alínea c

Capítulo 5

Conclusão

Síntese do Documento.

Estado final do projecto; Análise crítica dos resultados.

Trabalho futuro.

Apêndice A

Código do Programa

Lista-se a seguir um excerto do ficheiro *BibTex* que foi utilizado para demonstrar o funcionamento, correto, do código desenvolvido para a resolução do problema. Ficheiro esse que foi disponibilizado em <http://di.uminho.pt/~prh/lp.bib>

```
1 @string{ eth = "Institut fur Informatik , ETH Zurich" }
2
3 @techreport{BW83a,
4   author = "Manfred Broy and Martin Wirsing",
5   title = "Generalized Heterogeneous Algebras and Partial Interpretations",
6   year = 1983,
7   month = Feb,
8   institution = "Institut fur Informatik , TUM",
9   note = "(draft version)",
10  annote = "espec algebrica"
11 }
12
13 @inbook{Val90a ,
14   author = "Jos\'e M. Valen\c{c}a",
15   title = "Processos , {O}bjectos e {C}omunica\c{c}\~ao
16           ({O}p\c{c}\~ao I - {MCC})",
17   chapter = 2,
18   year = 1990,
19   month = Oct,
20   publisher = gdcc ,
21   address = um,
22   annote = "programacao oobjectos , proc comunicantes , espec formal"
23 }
```

A.1 Alínea a

O código do programa desenvolvido em *Flex*, tal como está no ficheiro fonte *tp1A.l* encontra-se de seguida.

```
1 %{
2     #include <stdlib.h>
3     #include <stdio.h>
4     #include <string.h>
5     #include "hashtable.h"
6     HashTable ht;
7     int count=1;
8     int position=0;
9 %}
10
11
12
13 %%
14
15 @[a-zA-Z]+\{
16         {
17             char* str = (char *) malloc(sizeof(char)*1000);
18             strcpy(str,yytext+1);
19             str[yytext-2] ='\0';
20
21             insertTable(
22                 ht, str,
23                 (int *)
24                 count);
25
26         }
27
28 .|\n
29         {}
30
31
32
33
34
35
36 %%
37 int main(){
38     initializeTable(ht);
39     int s;
40     s=yylex();
41     while(s){printf("%d",s);s=yylex();}
42     printHashTable(ht);
43     printf("\n");
44     return 0;
45 }
```

Apesar de existirem bibliotecas disponíveis com estruturas de dados já implementadas, tomou-se a liberdade de reutilizar uma biblioteca já produzida numa unidade curricular anterior em vez de utilizar *hsearch.h* disponível em *glib2.h*.

O código da estrutura de dados utilizada na resolução da alínea *a* deste problema foi desenvolvido na linguagem *C* e apresenta-se de seguida o código na íntegra, que pode ser encontrado no ficheiro fonte *hashtable.h*. Relembra-se que as operações de inserir, remover e procura nesta estrutura de dados encontra-se em *hashtable.c*.

Ficheiro *.h*,

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 #define HASHSIZE 31
6 #define KEYTYPE_SIZE 30
7 #define EMPTY 0
8 #define DELETED -1
9 #define FULL 1
10
11 //Open Addressing – Linear Probing (insertTable) and Quadratic Probing (insertTable2)
12 //After hash_function calculated the position if it is not available it try insert in the
   next position with
13 //label "empty" or "deleted". "deleted" is required when retrieving data only stops when
   finds "empty"
14
15 typedef char Keytype[KEYTYPE_SIZE];
16 typedef void *Info;
17 typedef struct entry{
18     int state;
19     Keytype key;
20     Info info;
21 }Entry;
22
23 typedef Entry* HashTable[HASHSIZE];
24
25 int hash_function(Keytype key);
26 void initializeTable(HashTable ht);
27 void clearTable(HashTable ht);
28 void insertTable(HashTable ht, Keytype k, Info i);
29 int retrieveTable(HashTable ht, Keytype k);
30 void deleteTable(HashTable ht, Keytype k);
31 void printHashTable(HashTable ht);
```

Ficheiro .c,

```
1 #include "hashtable.h"
2
3
4 int hash_function(Keytype key){
5     int i=0,sum=0;
6     for (;i<KEYTYPE_SIZE-1;i++){ //ends with '\0' so we must subtract one iteration
7         sum+=key[i];
8     }
9     return (sum%HASHSIZE);
10 }
11
12 void initializeTable(HashTable ht){
13     int i=0;
14     for (;i<HASHSIZE;i++){
15         Entry* new = (Entry*) malloc(sizeof(struct entry));
16         new->state=EMPTY;
17         strncpy(new->key,"Empty",KEYTYPE_SIZE);
18         new->info=NULL;
19         ht[i]=new;
20     }
21 }
22
23 void clearTable(HashTable ht){
24     int i=0;
25     for (;i<HASHSIZE;i++) free(ht[i]);
26 }
27
28 void insertTable(HashTable ht, Keytype k, Info i){
29     int position=hash_function(k);
30     int found=retrieveTable(ht,k);
31     if(found!=0){ // k já existe
32         ht[found]->info=ht[found]->info + 1;
33     }
34     else{ //k não existe
35         if(ht[position]->state!=FULL){
36             strncpy(ht[position]->key,k,KEYTYPE_SIZE);
37             ht[position]->info=i;
38             ht[position]->state=FULL;
39         }
40         else{
41             //try to insert in the next and so on — Linear Probing
42             while(ht[position]->state==FULL) position=(position+1)%HASHSIZE;
43             //it founded an EMPTY or DELETED
44             strncpy(ht[position]->key,k,KEYTYPE_SIZE);
45             ht[position]->info=i;
46             ht[position]->state=FULL;
47         }
48     }
49 }
50
51 //retrieving with linear probing from the initial position
52 int retrieveTable(HashTable ht,Keytype k){
53     int position=hash_function(k);
```

```

54  Entry* aux;
55  int res=0;
56  for (; ht[position]->state!=EMPTY && position<HASHSIZE; position=(position+1)%HASHSIZE){
57      if (strncmp(ht[position]->key,k,KEYTYPE.SIZE)==0){
58          aux=ht[position]; res=position;
59      }
60  }
61  return res;
62 }
63
64 void deleteTable(HashTable ht, Keytype k){
65     int position=hash_function(k), i=0;
66     for (; ht[position]->state!=EMPTY && i<HASHSIZE; i++){
67         if (strncmp(ht[position]->key,k,KEYTYPE.SIZE)==0){
68             ht[position]->state=DELETED;
69             strncpy(ht[position]->key," Deleted",KEYTYPE.SIZE);
70             ht[position]->info=NULL;
71         }
72     }
73 }
74
75 void printHashTable(HashTable ht){
76     int i=0;
77     Entry* aux;
78     FILE * fp = fopen("index.html","wr");
79     fprintf(fp,"<HTML>\n<BODY>\n");
80     fprintf(fp,"<h1> BibTex File </h1>\n");
81     for (; i<HASHSIZE; i++){
82         aux=ht[i];
83         if (aux->state!=EMPTY && aux->state!=DELETED){
84             fprintf(fp,"<t<li>Categoria: %s, Contagem: %d\n",aux->key,(int) aux->info);
85         }
86     }
87     fprintf(fp,"<t<BODY/>\n<HTML/>");
88 }

```

O ficheiro com a resolução da alínea *a* é apresentado em baixo, e o seu conteúdo está no ficheiro *indexA.html*.

```
1 <HTML>
2     <BODY>
3         <li>Categoria: MISC, Contagem: 1
4         <li>Categoria: TechReport, Contagem: 2
5         <li>Categoria: mastersthesis, Contagem: 2
6         <li>Categoria: techreport, Contagem: 138
7         <li>Categoria: proceeding, Contagem: 1
8         <li>Categoria: ARTICLE, Contagem: 1
9         <li>Categoria: MISC, Contagem: 1
10        <li>Categoria: unpublished, Contagem: 15
11        <li>Categoria: INPROCEEDINGS, Contagem: 5
12        <li>Categoria: phdthesis, Contagem: 21
13        <li>Categoria: string, Contagem: 31
14        <li>Categoria: incollection, Contagem: 6
15        <li>Categoria: manual, Contagem: 13
16        <li>Categoria: BOOK, Contagem: 2
17        <li>Categoria: InProceedings, Contagem: 20
18        <li>Categoria: inbook, Contagem: 3
19        <li>Categoria: inproceedings, Contagem: 184
20        <li>Categoria: book, Contagem: 44
21        <li>Categoria: misc, Contagem: 39
22        <li>Categoria: proceedings, Contagem: 4
23        <li>Categoria: article, Contagem: 131
24        <li>Categoria: Article, Contagem: 10
25        <li>Categoria: Misc, Contagem: 20
26        <li>Categoria: Book, Contagem: 1
27    <BODY/>
28 <HTML/>
```

A.2 Alínea b

A.3 Alínea c

A.4 Extra- Zim to Beamer

Decidiu-se também resolver o último enunciado que fiz respeito á conversão de ficheiros *Zim-wiki* para slides *Beamer*, que é uma variação de *LaTeX*, cujos slides são criados consoante o tipo de *Header* lido (apenas os tipos 3,4,5 permitem criar novo slide).

Ficheiro *Zim-Wiki* escrito em .txt,

```
1 ===== Home =====
2 Created Tuesday 22 March 2016!
3 This is a test!
4
5
6 ===== Header 2 =====
7 Should be a new slide!!
8 Next items should produce a List in Bold with three different bullets!
9
10 * item 1
11 * item 2
12 * item 3
13     1. item 3
14         a. item 3a
15
16
17 ===== Header 3 =====
18 Should be a new slide!!
19 Now it should be appearing //check boxes in italic//
20 ===== head 4 =====
21 Should not be a new slide underlined!!
22
23 [ ] empty
24 [*] checked wright
25     [*] sub checked wright
26     [ ] sub empty
27 [x] checked wrong
28
29
30 ===== Header 3 =====
31 Should be a new slide!!
32
33 ===== head 5 =====
34 Should not be a new slide!!
35
36 Now an example of ~~strike through~~
37
38 To finish this presentation an image must be shown!
39 An import of an image!!!
40
41 {{./atom.jpg}}
42
43
44 ===== End =====
45 Now an example of verbatim working ''int main() {return(0);} in verbatim ''!!!
```

Ficheiro *Ficheiro.l* com as expressões regulares que permitem resolver o problema,

```

1  %{
2      #include <stdio.h>
3      int firstSlide=0;
4      FILE *fp;
5  %}
6  %%
7  ^{=}{6}[ ]{1}[a-zA-Z0-9 ]+[ ]{1}{=}{6}          {
8
9
10
11
12
13
14
15
16
17
18
19
20 ^{=}{5}[ ]{1}[a-zA-Z0-9 ]+[ ]{1}{=}{5}          {
21
22

```

```

if (firstSlide==0){
fp=fopen("zim2beamer.tex","w
");
fprintf(fp,"%cdocumentclass{
beamer}\n %chypersetup{
pdfstartview={Fit}}\n %
cusetheme{Madrid}\n
",92,92,92);
fprintf(fp,"%cusecolortheme{
beaver}\n %cusepackage{
pifont}\n %cusepackage{
cancel}\n %cbegin{
document}\n
",92,92,92,92);
fprintf(fp,"%c title [Teste
Beamer]{The Real Deal}\n
%c subtitle {Example of
Beamer}\n",92,92);
fprintf(fp,"%c author [Ze
Carlos]\n",92);
fprintf(fp,"%c F.~ Author %
cinst{1} }\n %cinstitute
[University of Minho]{\n
%cinst{1}\n",92,92,92);
fprintf(fp,"Department of
Computer Science%c%c\n
",92,92);
fprintf(fp,"University of
Minho\n}\n %cdate [PT
2016]{Try number 3,
2016}\n %csubject{
Computer Science}\n
",92,92);
}
yytext[yytext-7]='\0';
fprintf(fp,"\\n%cbegin{
frame}\n\\t%cframetitle{%
s}\n",92,92,yytext+7);
firstSlide++;
}

if (firstSlide==0){
fprintf(fp,"%cdocumentclass{beamer}\n
%cchypersetup{pdfstartview={Fit
}}\n %cusetheme{Madrid}\n
",92,92,92);

```

```

23 fprintf(fp,"%cusecolortheme{beaver}\n
    %cusepackage{pifont}\n %
    cusepackage{cancel}\n %cbegin{
    document}\n",92,92,92,92);
24 fprintf(fp,"%ctitle[Teste Beamer]{
    The Real Deal}\n %csubtitle{
    Example of Beamer}\n",92,92);
25 fprintf(fp,"%cauthor[Ze Carlos]\n
    ",92);
26 fprintf(fp,"%{F.~ Author %cinst{1} }\n
    %cinstitute[University of Minho
    ]{\n %cinst{1}\n",92,92,92);
27 fprintf(fp,"Department of Computer
    Science%c%c\n",92,92);
28 fprintf(fp,"University of Minho\n\n
    %cdate[PT 2016]{Try number 3,
    2016}\n %csubject{Computer
    Science}\n",92,92);
29 }
30 yytext[yytext-6]='\0';fprintf(fp,"\n
    %cbegin{frame}\n\t%cframetitle{%
    s}\n",92,92,yytext+6);firstSlide
    ++;
31
32 ^[=]{4}[ ]{1}[a-zA-Z0-9 ]+[ ]{1}[=]{4} {
33
34 if(firstSlide==0){
    fprintf(fp,"%cdocumentclass{beamer}\n %
    chypersetup{pdfstartview={Fit}}\n %
    cusetheme{Madrid}\n",92,92,92);
35 fprintf(fp,"%cusecolortheme{beaver}\n
    %cusepackage{pifont}\n %
    cusepackage{cancel}\n %cbegin{
    document}\n",92,92,92,92);
36 fprintf(fp,"%ctitle[Teste Beamer]{
    The Real Deal}\n %csubtitle{
    Example of Beamer}\n",92,92);
37 fprintf(fp,"%cauthor[Ze Carlos]\n
    ",92);
38 fprintf(fp,"%{F.~ Author %cinst{1} }\n
    %cinstitute[University of Minho
    ]{\n %cinst{1}\n",92,92,92);
39 fprintf(fp,"Department of Computer
    Science%c%c\n",92,92);
40 fprintf(fp,"University of Minho\n\n
    %cdate[PT 2016]{Try number 3,
    2016}\n %csubject{Computer
    Science}\n",92,92);
41 }
42 yytext[yytext-5]='\0';fprintf(fp,"\n
    %cbegin{frame}\n\t%cframetitle{%
    s}\n",92,92,yytext+5);firstSlide
    ++;
43
44 \n\n\n {fprintf(fp,"%cend{frame}\n",92);}
45 ^[=]{3}[ ]{1}[a-zA-Z0-9 ]+[ ]{1}[=]{3} {yytext[yytext-4]='\0';fprintf(fp,"\n%s\n",
    yytext+4);}

```

```

46  ^[=]{2}[ ]{1}[a-zA-Z0-9 ]+[ ]{1}[=]{2}           {yytext[yylen-3]='\0';fprintf(fp,"\\n%s\\n",
      yytext+3);}
47  \\{\\.\\*\\}
      {yytext[yylen-2]='\0';fprintf(fp,"\\n%cbegin{
      figure}[!ht]\\n\\t %ccentering\\n\\t %cincludegraphics[width=0.25 %ctextwidth]{%s} \\n\\t %
      ccaption{Atom}\\n %cend{figure}\\n",92,92,92,92,yytext+4,92,92);}
48  \\*+\\*
      {yytext[yylen-2]='\0';fprintf(
      fp,"%ctextbf{%s}\\n",92,yytext+2);}
49  \\-+\\-
      {yytext[yylen-2]='\0';fprintf(
      fp,"\\n%cbegin{\\n",92,yytext+2);}
50  [/]{2}[a-zA-Z0-9 ]+[/]{2}
      {yytext[yylen-2]='\0';fprintf(fp,"\\n%ctextit{%s}\\n",92,yytext+2);}
51  [~]{2}[a-zA-Z0-9 ]+[~]{2}
      {yytext[yylen-2]='\0';fprintf(fp,"\\n%ccancel{%s}\\n",92,yytext+2);}
52  [']{2}.+[']{2}
      {yytext[yylen-2]='\0';fprintf(fp,"\\n%cbegin{frame}\\n%
      cbegin{frame}[fragile]\\n%cbegin{verbatim}\\n%s\\n%cbegin{verbatim}\\n%cbegin{frame}\\n
      ",92,92,92,yytext+2,92,92);}
53  [0-9]\\. [A-Za-z0-9]+
      {fprintf(fp,"\\n%cbegin{itemize}\\n\\n%cbegin{itemize}\\n%cbegin{itemize}\\n%cbegin{itemize}\\n%cbegin{itemize}\\n",92,92,92,yytext,92,92);}
54  [a-z]\\. [ ]{1}[A-Za-z0-9]+
      {fprintf(fp,"\\n%cbegin{itemize}\\n\\n%cbegin{itemize}\\n%cbegin{itemize}\\n%cbegin{itemize}\\n%cbegin{itemize}\\n",92,92,92,yytext,92,92);}
55  ^[A-Z][A-Za-z0-9 \\!]+
      {fprintf(fp,"%s\\n",yytext);}
56  \\[[ ]\\}\\*
      {fprintf(fp,"\\n%cding{112} %s %cpar\\n",92,yytext+4,92);}
57  .\\[[ ]\\}\\*
      {fprintf(fp,"%chspace{5ex}%cding{112} %s %cpar\\n",92,92,
      yytext+4,92);}
58  .\\[\\*\\]\\*
      {fprintf(fp,"%chspace{5ex}%cverb|%ccheckmark%s %cpar\\n
      ",92,92,92,yytext+4,92);}
59  \\[\\*\\]\\*
      {fprintf(fp,"%cverb|%ccheckmark %s %cpar\\n",92,92,yytext
      +4,92);}
60  \\[[X|x]\\}\\*
      {fprintf(fp,"%cverb|%cxmark X %s %cpar\\n",92,92,yytext+4,92);}
61  \\n\\*[ ]{1}[a-zA-Z0-9 ]+
      {fprintf(fp,"\\n%cbegin{itemize}\\n%cbegin{itemize}\\n%cbegin{itemize}\\n",92,92,yytext+3,92)
      ;}
62  .|\\n
      {}
63  %%
64  int main(){
65      int s;
66      s=yylex();
67      while(s){fprintf(fp,"%d",s);s=yylex();}
68      fprintf(fp,"%cend{document}\\n",92);
69      fclose(fp);
70      return 0;
71  }

```

Ficheiro *Ficheiro .tex* com os slides criados a partir de *zim2beamer.txt*, que representa o conteúdo de um ficheiro *Zim-Wiki*.

Para criar os respetivos slides apenas bastará compilar o ficheiro .tex para se obter o ficheiro PDF.

```
1 \documentclass{beamer}
2 \hypersetup{pdfstartview={Fit}}
3 \usetheme{Madrid}
4 \usecolortheme{beaver}
5 \usepackage{pifont}
6 \usepackage{cancel}
7 \begin{document}
8 \title[Teste Beamer]{The Real Deal}
9 \subtitle{Example of Beamer}
10 \author[Ze Carlos]
11 {F.~Author \inst{1} }
12 \institute[University of Minho]{
13 \inst{1}
14 Department of Computer Science\\
15 University of Minho
16 }
17 \date[PT 2016]{Try number 3, 2016}
18 \subject{Computer Science}
19
20 \begin{frame}
21     \frametitle{Home}
22     Created Tuesday 22 March 2016!
23     This is a test!
24 \end{frame}
25
26 \begin{frame}
27     \frametitle{Header 2}
28     Should be a new slide!!
29     Next items should produce a
30     \textbf{List in Bold}
31
32 \begin{itemize}
33 \item item 1
34 \end{itemize}
35
36 \begin{itemize}
37 \item item 2
38 \end{itemize}
39
40 \begin{itemize}
41 \item item 3
42 \end{itemize}
43
44 \begin{itemize}
45
46 \begin{itemize}
47 \item 1. item 3
48 \end{itemize}
49 \end{itemize}
50
51 \begin{itemize}
52
```

```

53 \begin{itemize}
54 \item a. item
55 \end{itemize}
56 \end{itemize}
57 \end{frame}
58
59 \begin{frame}
60     \frametitle{Header 3}
61     Should be a new slide!!
62     Now it should be appearing
63
64     \textit{check boxes in italic}
65
66     head 4
67     Should not be a new
68
69     \underline{slide underlined}
70
71     \ding{112} empty \par
72     \verb|\checkmark checked wright \par
73     \hspace{5ex}\verb|\checkmark sub checked wright \par
74     \hspace{5ex}\ding{112} sub empty \par
75     \verb|\xmark X checked wrong \par
76 \end{frame}
77
78 \begin{frame}
79     \frametitle{Header 3}
80     Should be a new slide!!
81
82     head 5
83     Should not be a new slide!!
84     Now an example of
85
86     \cancel{strike through}
87     To finish this presentation an image must be shown!
88     An import of an image!!!
89
90     \begin{figure}[!ht]
91         \centering
92         \includegraphics[width=0.25 \textwidth]{atom.jpg}
93         \caption{Atom}
94     \end{figure}
95 \end{frame}
96
97 \begin{frame}
98     \frametitle{End}
99     Now an example of verbatim working
100
101 \end{frame}
102 \begin{frame}[fragile]
103 \begin{verbatim}
104 int main() {return(0);} in verbatim
105 \end{verbatim}
106 \end{frame}
107 \end{document}

```

Alguns *prints* do ficheiro PDF final, visto que não fazia sentido incluir um PDF dentro de outro. Pretende-se apenas mostrar três slides que contém a maior parte das características de uma apresentação.

Header 2

Should be a new slide!! Next items should produce a **List in Bold**

- item 1
- item 2
- item 3
 - 1. item 3
 - a. item

Navigation icons

Ze Carlos (University of Minho)

Teste Beamer

PT 2016 2 / 6

Header 3

Should be a new slide!!

head 5 Should not be a new slide!! Now an example of ~~strike-through~~

To finish this presentation an image must be shown! An import of an image!!!



Figure: Atom

Navigation icons

Ze Carlos (University of Minho)

Teste Beamer

PT 2016 4 / 6

Header 3

Should be a new slide!! Now it should be appearing

check boxes in italic

head 4 Should not be a new

slide underlined

☐ empty

✓checked wright

 ✓sub checked wright

☐ sub empty

X checked wrong

Durante a elaboração deste enunciado os testes realizados passaram pela geração do seus ficheiros PDF e ir comparando com o ficheiro .txt que possui o conteúdo de um ficheiro *Zim-Wiki* e observando se os slides estavam de acordo com a sua especificação.

O ficheiro .l contém todo o conjunto de expressões regulares que permitem capturar os padrões que dizem respeito às regras de escrita de um ficheiro *zim* e as ações que toma, escrever o respectivo conteúdo no ficheiro *LaTeX Beamer*.

Os problemas que ocorreram relacionam-se com o o facto de por vezes não estar alinhado mas nada de especial. O maior problema foi apenas o facto de ao fazer "fprintf" para o ficheiro .tex não colocava 'é como tal recorreu-se á utilização do seu valor *ASCII*, que é o número 92 e verificava-se quando surge o símbolo percentagem seguido de 'c'.

Bibliografia

- [1] *V. Aho, Alfred* , *S. Lam, Monica* , *Sethi, Ravi* and *D. Ullman, Jeffrey* Second Edition, *Compilers Principles Techniques and Tools*.
- [2] ShareLatex examples and tutorials, <https://www.sharelatex.com>