Processamento de Linguagens
MIEI (3º Ano)
Trabalho Prático Nº 1 (FLEX)
Relatório de Desenvolvimento

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

31 de Março de 2019

Resumo

Este trabalho prático tem como principal objetivo o aprofundamento e aplicação dos conceitos dados nas aulas práticas e teóricas de *Processamento de Linguagens*. Para além de pretender aumentar a capacidade de escrita de *Expressões Regulares* para a descrição de padrões e, através destas, desenvolver *Processadores de Linguagens Regulares* que filtrem e transformem textos, foca-se também no uso do sistema de produção de texto em *C FLEX*.
Segundo o método de atribuição de trabalhos, foi-nos atribuido o segundo enunciado cujo objetivo é processar *Artigos de Jornal*.
Os resultados obtidos estão de acordo com os objetivos previamente definidos e com o que foi pedido no enunciado deste trabalho.

Conteúdo

Capítulo 1

Introdução

1.1 Processamento de Artigos de Jornal

Área: Processamento de Linguagens

No enunciado prático foi nos pedido a implementação de um filtro *FLEX* que transforma um documento de texto relativo a vários artigos de jornal num documento *HTML*.
Para a implementação deste trabalho optamos pelo uso da linguagem C com o FLEX visto que C disponibiliza as estruturas necessárias ao desenvolvimento deste projeto.
Através da utilização destes recursos, tivemos de por à prova não só os nossos conhecimentos de desenvolvimento de *Expressões Regulares* mas também das linguagens utilizadas.

Estrutura do Relatório

Este relatório está organizado em cinco capitulos, seguidos de anexos onde é apresentado todo o código desenvolvido.
No capítulo 2, é feita uma descrição informal do problema proposto, definindo as linhas de desenvolvimento assim como requisitos que devem ser cumpridos.
No capítulo 3 é abordado todo o processo de desenvolvimento assim como decisões tomadas ao longo deste. No capítulo 4 explicamos o porquê de todas as nossas decisões no desenvolvimento deste projeto, assim como as dificuldades encontradas na resolução do mesmo. Por fim, no capítulo 5 temos a síntese de todo o trabalho realizado, assim como uma análise crítica tanto dos resultados obtidos, como do possível trabalho futuro a desevolver.

Capítulo 2

Análise e Especificação

2.1 Descrição informal do problema

2.1.1 Desenvolvimento em FLEX de um filtro que transforme um documento TXT em HTML

Neste projeto temos dois tipos de ficheiros: TXT e HTML.
Pretende-se que sejam selecionados alguns items do ficheito txt para se efetuar experiencias e testes com estes. Pretende-se também que seja desenvolvido um processador de texto em FLEX que os transforme em ficheiros em formato HTML. Temos que gerar uma página que terá o indice de todos os artigos assim como a página de cada artigo individual. Devemos támbem efetuar uma contagem das tags usadas em todos os artigos.

2.2 Especificação do Requisitos

Os requisitos fundamentais deste enunciado são:

- Recolher todos os campos sobre um artigo de jornal presentes no ficheiro de texto;
- Criar um índice com todos os nomes de artigo;
- Criar uma página para cada artigo de jornal;
- Criar um menu de contagem de todas as tags usadas nos diferentes artigos de jornal.

Capítulo 3

Concepção/desenho da Resolução

3.1 Formato do Input

O ficheiro de input recebido consiste num ficheiro txt com milhares de notícias do Jornal Angolano "Folha 8". O formato do ficheiro segue, para cada notícia, a estrutura abaixo representada num extrato do Jornal:

```
<pub>
#TAG: tag:{Eduardo dos Santos} tag:{Petróleo} tag:{mensagem} tag:{preços}
#ID:{post-6243 post type-post status-publish format-standard has-post-thumbnail hentry
category-nacional tag-eduardo-dos-santos tag-petroleo tag-mensagem tag-precos}
Nacional

2015 será um ano difícil, diz o Presidente. Igual aos outros, acrescenta o Povo
--------------------------------------------------------------------------------

PARTILHE VIA:
#DATE: [116eb] Redacção F8 | 29 de Dezembro de 2014
2015 será um ano difícil, diz o Presidente. Igual aos outros, acrescenta o Povo - Folha 8

A baixa no preço do barril de petróleo, verificada desde Junho, está a levar o Executivo de Edua
dos Santos a traçar estratégias para contornar as dificuldades desencadeadas. Ou seja, com o pre
do petróleo em alta ou em baixa, serão sempre os mais pobres a pagar a factura.

.....
O corte nos subsídios aos combustíveis em 2015, é uma delas, prevendo o Governo angolano poupar
mais de 870 milhões de euros com essa medida. Com esta medida, que consta do Orçamento Geral do
Estado (OGE) de 2015, o Governo prevê para o próximo ano \uma redução de cerca de 109,2 biliões
kwanzas (mais de 870 milhões de euros) nos gastos com subsídios aos combustíveis", para a mesma
quantidade de consumo de 2014.

Depois de um último ajustamento ao preço dos combustíveis, em Setembro passado, com um aumento
médio de 25% ao consumidor no gasóleo e gasolina, na quarta-feira passada, registou-se a um novo
reajustamento de 20% nos preços dos mesmos tipos de combustíveis.

Etiquetas: Eduardo dos SantosPetróleomensagempreços
```

```
</pub>
```

3.2 Estruturas de Dados

Começamos por definir uma estrutura de dados capaz de guardar as tags do documento de modo a podermos apresentá-las assim como obter a sua contagem.

```
 typedef struct ltags {
    int valor;
    char *tag;
    char **titles;
    char **idsNews;
    int numAux;
    struct ltags *prox;
} *Tags;
```

Esta é a estrutura principal que define um artigo de jornal e todos os seus atributos relevantes para a conversão da mesma do ficheiro txt para o html.

```
struct Noticia{
    char *title;
    char **tags;
    char *idNoticia;
    char *date;
    char **text;
    int countText;
    char *category;
}noticia;
```

3.3 Algoritmos

Para a resolução deste projeto utilizamos vários algoritmos de agrupamento e filtragem de informação de modo a conseguir converter um ficheiro de texto num ficheiro de html.
Todos os algoritmos tem a sua função explicitada abaixo, sendo cada destes fundamental para a execução do programa:

- *initNoticia* - inicializa a estrutura de dados de notícia e aloca espaço para a mesma;
- *initTags* - inicializa a estrutura de dados das tags e aloca espaço para elas;
- *searchNoticia* - recebe um identificador de notícia e retorna: 1, se a notícia já foi interpretada e 0 caso contrário;
- *putListTags* - coloca a informação das tags interpretadas na respetiva estrutura, bem como o número de vezes que se repete e os títulos de notícias associadas a essa tag;

- *printIndiceTags* - coloca num ficheiro HTML todas as tags interpretadas até ao momento e os respetivos títulos associados;
- *beginTable* - função HTML que inicia a tabela do número de ocorrências das tags;
- *finishTable* - termina a tabela anteriormente iniciada para a contagem das tags;
- *htmlInit* - inicia tanto o ficheiro de índice de artigos de jornal, como o ficheiro de índice das tags;
- *htmlFinit* - fecha tanto o ficheiro de índice de artigos como o de índice de tags anteriormente inicializados;
- *printTags* - coloca na tabela das tags o número de ocorrências de cada tag no ficheiro de input;
- *beginNoticia* - coloca toda a estrutura tanto do artigo de jornal como das tags no ficheiro HTML;
- *rmSubstr* - dada uma *String* e um *Char*, se o *Char* ocorrer na string é removido;
- *trim* - dada uma *String* remove os espaços da mesma.

3.4 *Expressões Regulares*

De seguida iremos apresentar as *Expressões Regulares* utilizadas para normalizar o texto dos artigos e filtrar as informações mais importantes de cada um:

```
%x NOTICIA IGNORE
```

- Inicialmente, criamos os campos NOTICIA e IGNORE que servirão para referenciar o que diz á respeito ao que será guardado em cada notícia e o que irá ser ignorado, respetivamente;

```
<*>\<pub\> { initNoticia(); BEGIN IGNORE; BEGIN NOTICIA; }
```

- Esta expressão será encontrada no início de cada artigo pelo que, quando isso acontecer, é executado o código correspondente já acima explicado;

```
<IGNORE>{
    <*>-----------------------------------------------------------.* {;}

    <*>\<!--.* {;}

    <*>Etiquetas:.* {;}

    <*>PARTILHE.* {;}
}
```

- Neste campo, explicitamos as expressões que devem ser ignoradas;

```
<NOTICIA>\<\/pub\> { ... }
```

- Esta expressão é encontrada no final de cada notícia.

```
<NOTICIA>{
```

- Inicia o campo NOTICIA, onde irão ser encontradas as tags, os id's, a data, o título, a categoria e o texto do artigo;

```
<*>#TAG:.*        { ... }
```

- Expressão usada para capturar o texto que contém *TAG:*, que nos será útil para obtermos as tags que correspondem à notícia em questão;

```
<*>#ID:.*         { ... }
```

- Expressão usada para capturar o texto que contém *ID:*, que nos será útil para obtermos o ID da notícia em questão;

```
<*>#DATE:.*       { ... }
```

- Expressão usada para capturar o texto que contém *DATE:*, que nos será útil para obtermos a data da notícia em questão;

```
<*>[^.\n].*       { ... }
```

- Expressão utilizada para nos capturar o que resta da notícia, isto é, a categoria, o título e o corpo da notícia, esta expressão usa vários carácteres para obter esse efeito;

```
<*>\n\n\n\n\n\n.*     { ... }
```

- Esta expressão é útil para sabermos que a notícia vai terminar e executarmos certas operações necessárias.

Capítulo 4

Codificação e Testes

4.1 Alternativas, Decisões e Problemas de Implementação

Para resolução do problema imposto optamos por usar a linguagem de programação C e FLEX de modo a podermos converter um ficheiro txt em HTML.
A maior dificuldade foi a decisão de escolha da estrutura de dados a utilizar devido à elevada quantidade de dados a ser interpretada. Optamos por criar uma estrutura "Noticia"que interpreta e cria um ficheiro HTML para cada artigo no ficheiro de texto.

4.2 Testes realizados e Resultados

De modo a conseguirmos testar o projeto de forma económica decidimos fazer uma seleção reduzida dos artigos a filtrar.

De seguida executamos o programa para os artigos selecionados.

Como a seleção de notícias é muito curta o tempo de execução é quase nulo:

```
[MBP-de-Rafael:TP1 rafaelsilva$ make clean                                      ]
 rm -f Indice-Jornal.html lex.yy.c Jornal Noticias/*.html Indice-Tags.html Tags.h
 tml
[MBP-de-Rafael:TP1 rafaelsilva$ make                                            ]
 flex Jornal.fl
 cc -o Jornal lex.yy.c
[MBP-de-Rafael:TP1 rafaelsilva$ ./Jornal teste.txt                              ]
```

Com este teste podemos também obter um índice de tags reduzido:

Indice Tags

Number of News: 5

Tag: Apreensão

- 100 milhões de dólares em droga

Tag: Droga

- 100 milhões de dólares em droga

Tag: EUA

- 100 milhões de dólares em droga

Tag: São Tomé e Príncipe

- 100 milhões de dólares em droga

Tag: narcotráfico

- 100 milhões de dólares em droga

Tag: Sonangol

- 140 milhões de dólares para dois petroleiros

Tag: coreia do sul

- 140 milhões de dólares para dois petroleiros

Tag: petroleiros

- 140 milhões de dólares para dois petroleiros

Tag: Angola

- 1.500 desalojados pelas chuvas no Zaire

Tag: Desalojados

- 1.500 desalojados pelas chuvas no Zaire

Tag: chuvas

- 1.500 desalojados pelas chuvas no Zaire

Tag: zaire

- 1.500 desalojados pelas chuvas no Zaire

Tag: 2014

- Ano da vitória da baderna

Tag: Aqui falo eu

- Ano da vitória da baderna

Tag: Corrupção

- Ano da vitória da baderna

Tag: MPLA

- Ano da vitória da baderna

Tag: baderna

- Ano da vitória da baderna

Tag: balanço

- Ano da vitória da baderna

Tag: Petróleo

- 2015? Como sempre, ano mau para a maioria dos angolanos

Tag: Receitas

- 2015? Como sempre, ano mau para a maioria dos angolanos

Tag: crise

- 2015? Como sempre, ano mau para a maioria dos angolanos

Obtemos também uma lista de tags mais reduzida:

List of Tags

List contain number of occurrences of a tag

Total tags: 21

Tag Name	Number of Occurrences
Apreensão	**1**
Droga	**1**
EUA	**1**
São Tomé e Príncipe	**1**
narcotráfico	**1**
Sonangol	**1**
coreia do sul	**1**
petroleiros	**1**
Angola	**1**
Desalojados	**1**
chuvas	**1**
zaire	**1**
2014	**1**
Aqui falo eu	**1**
Corrupção	**1**
MPLA	**1**
baderna	**1**
balanço	**1**
Petróleo	**1**
Receitas	**1**
crise	**1**

Podemos também observar um índice do jornal em questão:

Indice Jornal

- post-1643
- post-5702
- post-4816
- post-6225
- post-5959

Por fim temos a página de uma notícia:

Title: 100 milhões de dólares em droga

Date: Redacção F8 — 30 de Setembro de 2014

Tags: Apreensão || Droga || EUA || São Tomé e Príncipe || narcotráfico

Category: Lusofonia

Um navio de carga com bandeira são-tomense que transportava mais de cem milhões de dólares em droga foi apresado há cerca de três semanas pela marinha dos EUA, revelou hoje uma fonte governamental são-tomense. Segundo a mesma fonte, o navio de nome Borocho, capturado nas águas da América Central, continha muita droga, cujo valor ronda mais de 100 milhões de dólares", além de outras mercadorias ilegais. A embarcação estava matriculada sob o número 003651 nas águas da América Central. Num comunicado do Conselho de Ministro distribuído esta semana indica-se que "o Governo foi informado que no dia 07 deste mês as competentes autoridades da Guarda Costeira em parceria com as cooperações de países amigos deram início a diligências para interceptar no navio de carga de nome Borocho, supostamente registado com bandeira são-tomense". O governo não dá mais pormenores, adiantando apenas que o referido navio é desconhecido do Instituto Marítimo e Portuário de São Tomé e Príncipe (IMAP-STP), a única instituição vocacionada para registar navios e atribuir bandeira são-tomense. "Neste momento, o referido navio com registo ilegal foi apresado e encontra-se sob custódia com estatuto de embarcação sem nacionalidade", refere-se no comunicado do governo são-tomense. Sublinha-se ainda que "perante a gravidade deste caso", que afecta "a imagem do país, o conselho de ministros insta as autoridades competentes do sector no sentido de prosseguirem com as investigações, identificar cumplicidades e prevaricadores para efeitos de procedimento disciplinar e criminal". A nível interno em São Tomé, as autoridades não revelam pormenores deste caso, mas sabe-se que o navio transportava 700 quilos de cocaína e que foram detidos 13 tripulantes. A embarcação é de origem do Panamá e a documentação que as autoridades do arquipélago dizem ser falsas foram emitidos em 23 de Julho último e é válida até a mesma data de 2018.

Após este teste reduzido testamos para o ficheiro completo, para o qual obtemos os seguintes resultados de tempo de execução:

```
[MBP-de-Rafael:TP1 rafaelsilva$ make clean
rm -f Indice-Jornal.html lex.yy.c Jornal Noticias/*.html Indice-Tags.html Tags.h
tml
[MBP-de-Rafael:TP1 rafaelsilva$ make
flex Jornal.fl
cc -o Jornal lex.yy.c
[MBP-de-Rafael:TP1 rafaelsilva$ time ./Jornal folha8.OUT.txt

real    0m3.141s
user    0m1.964s
sys     0m1.022s
```

Obtemos também um contador de tags e notícias:

List of Tags

List contain number of occurrences of a tag

Total tags: 4909

> # Indice Tags
>
> ## Number of News: 4912

Capítulo 5

Conclusão

Em suma, este projeto permitiu-nos aprofundar os nossos conhecimentos sobre *Processamento de Linguagens* em relação aos conteúdos lecionados nas aulas teóricas e práticas, tendo como principal foco a ferramenta de processamento de texto FLEX.
Como sugestões de trabalho futuro, poderia ter sido incluído um método de procura por título ou tag de modo a simplificar a consulta de artigos do jornal ao utilizador. Contudo achamos que o nosso trabalho cumpre todos os requisitos pedidos no enunciado de modo correto, simples e eficiente.

Apêndice A

Código do Programa

Lista-se a seguir o CÓDIGO C e FLEX do programa:

```
      %option noyywrap
%{
#include <string.h>
    #include <stdio.h>
    #include <stdlib.h>
    #include <ctype.h>

    struct Noticia{
        char *title;
        char **tags;
        char *idNoticia;
        char *date;
        char **text;
        int countText;
        char *category;
    }noticia;

    typedef struct ltags {
     int valor;
     char *tag;
        char **titles;
        char **idsNews;
        int numAux;
     struct ltags *prox;
} *Tags;

char **idsNoticias;
    int NumNews = 0;

    FILE *out, *jornal, *tags, *indiceTags;

    Tags listTags;
```

```
int countTags = 0;
int flag  = 0;
int stop  = 0;
int aux   = 0;

void initNoticia(){
    noticia.title = (char*) calloc(64,sizeof(char*));
    noticia.idNoticia = (char*) calloc(64,sizeof(char*));
    noticia.date = (char*) calloc(64,sizeof(char*));
    noticia.tags = (char**) calloc(64,sizeof(char**));
    noticia.text = (char**) calloc(1024,sizeof(char**));
    noticia.category = (char*) calloc(64,sizeof(char*));
    noticia.countText = 0;
}

void initTags(){
    listTags = (Tags) malloc(sizeof(struct ltags));
    listTags->tag = (char*) calloc(64,sizeof(char*));
    listTags->titles = (char**) calloc(4096,sizeof(char**));
    listTags->idsNews = (char**) calloc(4096,sizeof(char**));
    listTags->valor = 0;
    listTags->numAux = 0;
    listTags->prox = NULL;
}

int searchNoticia(char *id){
    if(NumNews==0){
        idsNoticias[0] = strdup(id);
        idsNoticias[1] =  NULL;
        NumNews++;
        return 0;
    } else {
        int i=0;
        while(idsNoticias[i] != NULL){
            if(strcmp(idsNoticias[i],id)==0){
                return 1;
            }
            i++;
        }
        idsNoticias[i] = strdup(id);
        idsNoticias[i+1] = NULL;
        NumNews++;
        return 0;
    }
}

void putListTags(char *tag){
```

```
 int flag=0;
 Tags *aux = &listTags;

 while((*aux) && !flag){
 if(strcmp((*aux)->tag,tag)==0){
            (*aux)->idsNews[(*aux)->numAux] = strdup(noticia.idNoticia);
            (*aux)->titles[(*aux)->numAux] = strdup(noticia.title);
            (*aux)->numAux++;
 flag=1;
 (*aux)->valor++;
 }
 aux = &((*aux)->prox);
 }
 if(flag==0){
  Tags new = (Tags) malloc (sizeof (struct ltags));
 new->valor = 1;
 new->tag = strdup(tag);
         new->numAux = 0;
         new->numAux = 0;
         new->idsNews = (char**) calloc(4096,sizeof(char**));
         new->titles = (char**) calloc(4096,sizeof(char**));
         new->idsNews[new->numAux] = strdup(noticia.idNoticia);
         new->titles[new->numAux] = strdup(noticia.title);
         new->numAux++;
  new->prox = *aux;
  *aux = new;
         countTags++;
 }
}

void printIndiceTags(){
    Tags *aux = &listTags;
    char *temp;
    char *file;
    NumNews--;

    fprintf(indiceTags,"<p></p>\n");
    fprintf(indiceTags,"<p>Number of News: %d</p>\n",NumNews);

    while(*aux){
        fprintf(indiceTags,"<p></p>\n");
        if((*aux)->valor >0){
            fprintf(indiceTags,"<b>Tag: %s</b>\n",(*aux)->tag);
            fprintf(indiceTags,"<p></p>\n");
            fprintf(indiceTags,"<ul>");
            for(int i=0; i<(*aux)->numAux; i++){
                temp = (char*) malloc(sizeof(char)*64);
                file = (char*) malloc(sizeof(char)*64);
```

```
                strcpy(file,"Noticias/");
                sprintf(temp,"%s.html",(*aux)->idsNews[i]);
                file = strcat(file,temp);
                if(strcmp("",(*aux)->titles[i])==0){
                   fprintf(indiceTags,"  <li><a href='%s'>%s</a></li>\n",file,"[No Title]");
                } else{
                   fprintf(indiceTags,"  <li><a href='%s'>%s</a></li>\n",file,(*aux)->titles
                }
            }
            fprintf(indiceTags,"</ul>");
        }
        aux = &((*aux)->prox);
    }
}

void beginTable(){
 fprintf(tags,"<!DOCTYPE html>\n<html>\n<head>\n<meta charset = \"UTF-8\"/>\n<style>\ntable,
}

void finishTable(){
 fprintf(tags,"</table>\n</body>\n</html>");
}

void htmlInit(FILE* file, char *name){
    if(strcmp("jornal",name)==0){
        fprintf(file,"<html>\n<h2>Indice Jornal</h2>\n<head>\n<meta charset = \"UTF-8\"/>\n<
    } else{
        fprintf(file,"<html>\n<h2>Indice Tags</h2>\n<head>\n<meta charset = \"UTF-8\"/>\n</he
    }
}

void htmlFinit(FILE* file, char *name){
    if(strcmp("jornal",name)==0){
        fprintf(file,"</ul>\n</body>\n</html>");
    } else{
        fprintf(file,"</body>\n</html>");
    }
}

void printTags(){
  Tags *aux = &listTags;
  beginTable();
    while(*aux){
     if((*aux)->valor >0){
         fprintf(tags,"<tr>\n");
         fprintf(tags,"<th>%s</th>\n",(*aux)->tag);
         fprintf(tags,"<th>%d</th>\n",(*aux)->valor);
         fprintf(tags,"</tr>\n");
```

```
         }
            aux = &((*aux)->prox);
        }
        finishTable();
    }

    void beginNoticia(){
        char *temp = (char*) malloc(sizeof(char)*64);
char *file = (char*) malloc(sizeof(char)*64);
        int i = 0;

strcpy(file,"Noticias/");

sprintf(temp,"%s.html",noticia.idNoticia);

file = strcat(file,temp);

fprintf(jornal,"  <li><a href='%s'>%s</a></li>\n", file, noticia.idNoticia);

    out = fopen(file, "w");

    if(out != NULL){
     // Init Hmtl
     fprintf(out,"<!DOCTYPE html>\n");
     fprintf(out,"<html>\n");
     fprintf(out,"<head>\n");
     fprintf(out,"<meta charset = \"UTF-8\"/>\n");
     fprintf(out, "<style>\n#title {\ndisplay: inline;\n}\n</style>\n");
     fprintf(out,"</head>\n<body>\n");

     // Init Noticia
     fprintf(out," <pub id=\"%s\">\n", noticia.idNoticia);
     fprintf(out," <b>Title:</b>\n");
            if(strcmp("",noticia.title)==0){
                fprintf(out,"           <title id=\"title\">%s</title>\n","[No Title]");
            } else{
                fprintf(out,"           <title id=\"title\">%s</title>\n", noticia.title);
            }
            fprintf(out," <p></p>\n");
            fprintf(out," <b>Date:</b>\n");
            if(strcmp("",noticia.date)==0){
                fprintf(out,"          <author_date>%s</author_date>\n","[No Date]");
            } else{
                fprintf(out,"          <author_date>%s</author_date>\n", noticia.date);
            }
            fprintf(out," <p></p>\n");
            fprintf(out," <b>Tags:</b>\n");
            fprintf(out, " <tags>\n");
```

```
            while(noticia.tags[i] != NULL){
             if(i==0){
             fprintf(out," <tag>%s</tag> ",noticia.tags[i]);
             } else{
                  fprintf(out,"<tag> || %s</tag> ",noticia.tags[i]);
             }
                i++;
            }
            fprintf(out,"\n");
            fprintf(out," </tags>\n");
            fprintf(out," <p></p>\n");
            fprintf(out," <b>Category:</b>\n");
            if(strcmp("",noticia.category)==0){
                fprintf(out,"           <category>%s</category>\n","[No Category]");
            } else{
                fprintf(out,"           <category>%s</category>\n",noticia.category);
            }
            fprintf(out," <p></p>\n");
            fprintf(out," <text>\n");
            for(int j=0; j<noticia.countText; j++){
                if(noticia.text[j]!=NULL){
                    fprintf(out,"           %s\n",noticia.text[j]);
                }
            }
            fprintf(out," </text>\n");
            fprintf(out," </pub>\n");
            fprintf(out, "</body>\n</html>\n");
    }

        fclose(out);
    }

    void rmSubstr(char *str, const char *toRemove){
        size_t length = strlen(toRemove);
        while((str = strstr(str,toRemove))){
            memmove(str, str + length, 1 + strlen(str + length));
        }
    }

    char* trim(char *s) {
        while(isspace(*s)) s++;
        char *back = s + strlen(s);
        while(isspace(*--back));
        *(back+1) = '\0';
        return s;
    }

%}
```

```
%x NOTICIA IGNORE

%%

<*>\<pub\> { initNoticia(); BEGIN IGNORE; BEGIN NOTICIA; }

<IGNORE>{

<*>.* {;}

<*>\<!--.* {;}

<*>Etiquetas:.* {;}

<*>PARTILHE.* {;}
}

<NOTICIA>\<\/pub\> {
                        if(!searchNoticia(noticia.idNoticia) && noticia.idNoticia[0]=='p'){
                            int i=0;
                            while(noticia.tags[i]!=NULL){
                                putListTags(noticia.tags[i]);
                                i++;
                            }
                            beginNoticia();
                        }
                    }

<NOTICIA>{

   <*>#TAG:.*       {
                       char *input = strdup(yytext+6);
                       rmSubstr(input,"{");
                       rmSubstr(input,"}");
                       rmSubstr(input,"tag");
                       char *token = strtok(input,":");
                       int i = 0;
                       while(token != NULL){
                            trim(token);
                            noticia.tags[i] = strdup(token);
                            i++;
                            token = strtok(NULL,":");
                       }
                       noticia.tags[i] = NULL;
                    }

   <*>#ID:.*        {
```

```
                        noticia.idNoticia = strdup(strtok(yytext+5," "));
                      }

    <*>#DATE:.*       {
                        noticia.date = strdup(yytext+15);
                      }

<*>[^.\n].*      {
if(flag<2){
if(flag==0){
noticia.category = strdup(yytext);
}
if(flag==1){
noticia.title = strdup(yytext);
}
}
else if(flag>2){
noticia.text[aux] = strdup(yytext);
aux++;
}
flag++;
     }

    <*>\n\n\n\n\n\n\n.*    {
     noticia.countText = aux;
      flag = 0;
      aux = 0;
       }

  <*>.|\n              {;}

}

%%

int main(int argc, char* argv[]){
    if(argc>1){
        indiceTags = fopen("Indice-Tags.html","w");
        jornal = fopen("Indice-Jornal.html","w");
        tags = fopen("Tags.html","w");
        htmlInit(jornal,"jornal");
        htmlInit(indiceTags,"tags");
        initTags();
        idsNoticias = (char**) calloc(10000,sizeof(char**));

        yyin = fopen(argv[1], "r");

        yylex();
```

```
    printTags();

        htmlFinit(jornal,"jornal");

        printIndiceTags();

        htmlFinit(indiceTags,"tags");

        fclose(yyin);
        fclose(jornal);
        fclose(tags);
        fclose(indiceTags);
    }
    else {
        return -1;
    }
    return 1;
}
```