Relatório Trabalho AED II

José Cosgrove
a18826

André Cardoso
a18848

11/04/2020

Escola Superior de Tecnologia
Licenciatura em Engenharia de Sistemas Informáticos
Professor Alberto Simões

1 Introdução

Este relatório tem como principal função explicar quais os métodos usados e o porquê de serem usados na resolução deste trabalho.

O projeto foi implementado na linguagem de programação C para a cadeira de Algoritmos e Estrutura de Dados II em parceria com a cadeira de Estatistica.

O projeto é composto de 1 ficheiro para leitura, 6 ficheiros ".h", 6 ficheiros ".c", 1 makefile, 1 relatório em pdf e o respetivo ficheiro do relatório em latex.

2 Resumo

Foi pedida a criação de uma aplicação capaz de analisar um ficheiro já estruturado. Esse ficheiro tem várias linhas e cada linha é composta por uma palavra na sua forma original, a mesma palavra segundo a sua raiz, a análise morfológica e por último a certeza da aplicação quanto à análise realizada.

A aplicação deverá ainda ser capaz de:

- Ler todas as linhas do ficheiro e guardar a informação necessária numa ou mais estruturas de dados dinámicas;
- Apresentar uma tabela de frequências quanto à categoria gramatical;
- Apresentar uma tabela de frequências quanto ao tamanho das palavras existentes;
- Apresentar a média aritmética e desvio padrão de cada categoria gramatical;
- Apresentar as medidas de dispersão e localização relativamente ao tamanho das palavras.

3 Trabalho

3.1 Estrutura de Ficheiros

Este trabalho é composto de vários ficheiros:

- O ficheiro principal "main.c" que tem como função fazer a leitura do ficheiro, chamar funções e apresentar os valores necessários.
- uma série de ficheiros ".c" na raiz da pasta que tem diferentes funções para cada lista criada que são chamados através dos ficheiros headers.
- uma pasta com ficheiros ".h" que contem variáveis estáticas, estruturas de dados das diferentes listas criadas e as funções correspondentes a essas estruturas.
- uma pasta file que contem o ficheiro a ser lido.

3.2 Leitura do Ficheiro

A leitura do ficheiro é feita no ficheiro "main.c" através das funções de ficheiros existentes na linguagem de programação C.

É feita a leitura (r - read) do ficheiro com path definido num header file e, linha a linha, vai ser percorrido o ficheiro até ao seu fim (EOF). A cada linha percorrida vão sendo atribuidos os seus valores a várias variáveis e chamadas diversas funções.

Num primeiro momento essas funções apenas tem o dever de inserir os valores em listas, à exceção de sinais de pontuação e linhas em branco.

3.3 Criação de Listas

O tipo de listas usadas nesta aplicação são as listas simplesmente ligadas com inserção à cabeça. Foi usado este tipo pois não vimos necessidade de utilizar listas mais complexas.

Este projeto usa 3 listas distintas. Uma que guarda todos os elementos do ficheiro tal como pedido no ex.1, outra lista que guarda todas as categorias gramaticais e o numero de vezes que são listadas e por fim uma lista que guardas quantas palavras existem com um determinado número de caractéres e o respetivo número de caracteres.

Cada lista tem um ficheiro ".c" e ".h" onde estão presentes as estruturas e as funções para tratamento das listas e solução às funcionalidades a que a aplicação deve ser capaz de dar resposta.

As estruturas de dados são compostas por uma estrutura com variaveis unicamente do tipo da estutura à qual se destina e depois uma outra estrutura onde guarda a estrutura anteriormente descrita e ainda a lista seguinte.

As funções existentes são de inicialização, verificação de existência, inserção, ordenação e impressão. Existem ainda outras especificas a cada lista.

3.3.1 Lista Ficheiros

```
typedef struct _data{
char original[100];
char root[100];
char morphology[5];
double assurance;
}Data;

typedef struct _list{
Data dados;
struct _list *next;
}List;
```

3.3.2 Lista Categoria Gramatical

```
typedef struct _tabGra{
char morphology[5];
int frequency;
double assurance;
double variance;
}TabGra;

typedef struct _listTabGra{
TabGra dados;
struct _listTabGra *next;
}ListTabGra;
```

3.3.3 Lista Tamanho das Palavras

```
typedef struct _tabPal{
int size;
int frequency;
}TabPal;

typedef struct _listTabPal{
TabPal dados;
struct _listTabPal *next;
}ListTabPal;
```

3.3.4 Lista Palavras

```
typedef struct _countPal{
char original[100];
int frequency;
}CountPal;

typedef struct _listCountPal{
CountPal dados;
struct _listCountPal *next;
}ListCountPal;
```

3.3.5 Lista Histograma

```
typedef struct _histograma{
double inferior;
double superior;
int frequency;
}Histograma;

typedef struct _listHistograma{
Histograma dados;
struct _listHistograma *next;
}ListHistograma;
```

3.4 Funções Principais

3.4.1 NewList()

Esta função vai retornar uma lista onde nula.

3.4.2 Exists(Lista, Dados)

Esta função vai verificar se o elemento em estudo já existe na lista, no caso de existir a sua ocorrência vai ser incrementada, se não, vai ser inserida.

3.4.3 Insert(Lista, Dados)

Esta função vai criar um novo elemento para ser colocado na lista com os dados que lhe foram enviados e o proximo elemento da lista vai ser a lista que foi enviada.

3.4.4 Order(Lista, Dados)

Esta função vai ser chamada depois da lista estar completamente carregada e é chamada elemento a elemento, verificando se o elemento onde se encontra está de encontro com o tipo de ordenação desejado, percorrendo a lista até ser possível a sua colocação numa nova lista que vai ser retornada.

3.4.5 Show(Lista)

Esta função vai imprimir no ecrã os dados que se encontram em cada elemento da lista, usando um ciclo for para percorrer todos os elementos da lista.

3.5 Funções Especificas

3.5.1 CalcularGramatical(Lista)

Esta função percorre toda a lista, imprimindo no monitor a morfologia da palavra, a média e o desvio padrão.

A soma dos valores e a soma de cada valor ao quadrado já estão a ser guardados na lista.

A média calcula-se através da divisão da soma dos valores com a sua frequência (quantidade desses valores).

O desvio padrão calcula-se através da seguinte fórmula: $\sqrt{X - x^2}$ (raiz quadrada da variância menos média ao quadrado).

3.5.2 CalcularTamanhoPalavras(Lista, Total)

Esta função percorre toda a lista, imprimindo no monitor a média, a mediana, a moda e o desvio padrão.

A soma dos valores e a soma de cada valor ao quadrado já estão a ser guardados na lista.

A média e o desvio padrão calculam- se da mesma forma do anterior.

A mediana é o valor do elemento que está no elemento central da lista, uma vez que já se encontra ordenada.

A moda é o elemento que ocorre mais vezes, ou seja, o que tem maior frequência.

3.5.3 CalcularQuartil(Lista, Total, Palavra)

Esta função vai receber como parâmetros uma lista com todas as palavras e o número de vezes que aparecem, o total de palavras que existem e uma palavra (introduzida pelo utilizador).

A lista já está ordenada da palavra com mais ocorrência para a com menos e vai ser percorrida até encontrar a palavra definida. À medida que vai sendo percorrida a lista vão sendo acumulados os valores de frequência de cada palavra.

Quando a palavra for encontrada, vai ser feita a divisão do somatório de frequências pelo número total de palavras, que vai dar um resultado de 0 a 1. No caso de ser menor ou igual a 0.25 está no 1º quartil, entre 0.25 e 0.50 no 2º quartil, entre 0.50 e 0.75 no 3º quartil e maior de 0.75 está no 4º quartil.

3.5.4 Histograma(Lista, Total)

Esta função vai ter como responsabilidade calcular o número de classes que vão ser precisas, criando uma lista que contem para cada classe o intervalo e o número de elementos que existem nesse intervalo.

O primeiro passo a tomar é precisar quais os limites dos valores encontrados. Esses valores foram obtidos após percorrer toda a lista que foi enviada em parâmetro e que contém todos os valores.

De seguida, é preciso determinar o número de classes. O método para determinar esse valor foi a Regra de Sturges ($k = 1 + log(Total)/log(2)$).

O terceiro passo é determinar a amplitude das classes ($h = (maior - menor)/k$).

Por fim é percorrida para k vezes a lista, contados quantos valores existem para cada classe e a cada repetição da lista são atualizados os valores superiores e inferiores e são introduzidos os valores no elemento da lista.

4 Conclusão

Com este trabalho pudemos pôr em prática tudo um pouco daquilo que sabemos em C, desde o mais básico como escrever e ler palavras, ciclos ou calculos até leitura de ficheiros e manipulação de listas.

Foi bastante produtivo num ponto de vista pessoal, pois tivemos de trabalhar em equipa para manter sempre o mesmo foco, dando e recebendo conselhos, planeando a forma como as coisas iriam ser geridas.

Para além disso, pôs-nos também à prova pois no início este projeto pode ter parecido algo simples, mas a verdade é que no seu desenrolar nos foi dando algumas dores de cabeça.

Para finalizar, apenas dizer que a cada dia que passa nos vamos sentindo mais capazes de realizar qualquer projeto, pois é nesse sentido que todos os dias trabalhamos.