Relatório do Trabalho

Disciplina de Estruturas de Dados

e Algoritmos I

– Serviço de Mensagens Curtas Inteligente –

20/06/2021

Membros do grupo:

-Diogo Mestre, nº48973;

-Rodrigo Alves, nº 48681.

Introdução

Este trabalho baseia-se na implementação da tecnologia de escrita de mensagens curtas inteligente T9 (Text on 9 Keys). Como todos sabemos esta tecnologia é usada nos telemóveis que todos conhecemos, os antigos Nokias, entre outros. Esta permite a escrita de palavras através da utilização de teclas numéricas em que cada tecla apenas está reservada para um certo número de caracteres, ou seja, cada tecla pode corresponder até 8 caracteres (neste caso a tecla 2), inclusive os acentuados. Foi necessário a utilização de dicionários para a procura das palavras possíveis, consoante as teclas numéricas escritas pelo utilizador.

Porquê a escolha desta estrutura de dados: De entre todas as estruturas de dados abordadas em aula, decidimos utilizar as Hash Tables Abertas, desta forma conseguimos agrupar a informação retirada do dicionário de forma eficiente e ao mesmo tempo facilitar a inserção, uma vez que, as colisões são resolvidas usando uma estrutura extra, no nosso caso optámos por usar as listas ligadas, o que simplificou ainda mais o nosso trabalho, pois já estávamos a reunir todas as palavras que se escrevem com a mesma sequência de algarismos, numa só estrutura de fácil acesso. Isto tudo permitiu de certa forma chegar ao objetivo do trabalho.

Em primeiro lugar o programa devolve palavra a palavra até à aceitação pelo utilizador. Estas palavras serão sugeridas ao utilizador conforme a sequência de algarismos inseridas pelo mesmo. No caso da palavra não se encontrar no dicionário, esta será então escrita pelo próprio e guardada no variável texto e num novo ficheiro (dicionário-updated.txt), de seguida esta será inserida no array que irá devolver todas as palavras aceites ou escritas pelo utilizador.

Este começa por abrir o dicionário indicado, em seguida percorre-o e ao mesmo tempo converte palavra a palavra para uma sequência de números. Para facilitar esta leitura decidimos utilizar um tipo de dados de caracter, wchar_t, o que nos permite ler todas as strings (acentuadas e não acentuadas).

A função encarregue de fazer esta conversão encontra-se no ficheiro “hash.h” com o nome de conv. Esta irá correr a string caracter a caracter, inicialmente começa por convertê-lo num caracter minúsculo e de seguida este irá ser submetido a um conjunto de condições até encontrar o algarismo equivalente.

Estas sequências numéricas serão utilizadas como as chaves na hashtable e de imediato iremos chamar a função Insert para cada uma das chaves, que irá inserir na lista ligada correspondente a respetiva string do dicionário.

Caso a chave já exista na estrutura de dados já referidas, estamos na presença de uma colisão, e, portanto, sabemos que estas são resolvidas no nosso caso através de uma lista ligada.

Se a lista estiver vazia a string será inserida à cabeça da lista, caso contrário será inserida na primeira posição vazia, ou seja, a seguir da última posição ocupada e desta forma as palavras na lista irão ficar por ordem alfabética.

Outputs do Terminal

A compilação do trabalho é feita através dos seguintes comandos:

- gcc -std=c99 -Wall hashsep.c TB.c -o TB
- ./TB nome do dicionário.txt

A primeira mensagem que aparece no terminal é a indicação do tempo demorado pelo programa a carregar o dicionário na estrutura de dados.

Depois de ser mostrado no terminal o tempo, irá ser mostrada uma tabela com a informação para o utilizador digitar os algarismos correspondentes às letras, após ser apresentada esta tabela é pedido ao utilizador uma sequência de algarismos. De seguida, o programa recebe essa mesma sequência e vai procurar a lista correspondente a essa chave e guarda a posição do início dessa lista. Depois percorre essa lista e vai mostrando ao utilizador as sugestões, até se aceitar uma ou até ao fim da lista. Caso não se aceite nenhuma o programa pede ao utilizador que escreva a sua palavra a partir do teclado. Todas as palavras aceites ou escritas pelo próprio serão guardadas numa variável à qual chamámos de array1. Iremos estar sempre a atualizar esta variável através de concatenações realizadas pela função: wcscat.

Quando é introduzida a palavra 1 (um) termina a escrita da mensagem criada pelo utilizador, que vai ser apresentada no final pelo programa, ou seja vai imprimir o conteúdo da variável array1. Se for introduzida a palavra 0 (zero) o programa pede ao utilizador se deseja ou não sair e sai se a resposta for s (sim).

Hash.h e Hash.c

Para a realização deste programa foi necessário efetuar alterações aos ficheiros hash.h e hash.c para que pudessem trabalhar com variáveis do tipo wchar_t, nomeadamente, o ElementType passou a receber variáveis long e algumas funções passaram a retornar variáveis unsigned long e a receberem argumentos do tipo wchar_t. Foi também necessário a criação de novas funções como a conv (já explicada anteriormente), para que o mesmo pudesse funcionar.

Conclusão

Inicialmente neste trabalho sentimos muitas dificuldades em entender que estrutura de dados utilizar e como a alterar para receber as strings do dicionário.

Após a escolha da mesma, ainda sentimos dificuldades na utilização dos wchar_t, uma vez que inicialmente o programa não reconhecia os caracteres acentuados, demorou algum tempo até percebermos minimamente como trabalhar com as mesmas. No programa ainda nos surgiram vários erros de compilação que nos ajudaram a perceber melhor o funcionamento do mesmo. Decidimos por falta de tempo não implementar todas as funcionalidades opcionais, pelo que apenas implementámos a parte de fazer uptade ao dicionário, para tal criámos um novo dicionário, no qual foram inseridas as palavras introduzidas pelo utilizador.

Em suma, este trabalho veio-nos aprofundar o conhecimento já adquirido nas aulas e ainda nos fez explorar novos conteúdos desafiadores. Permitiu-nos perceber melhor como funciona a linguagem C e como esta estrutura de dados que utilizámos (HashTable) é operada.