Instituto Superior Técnico
DEEC
Programação 2019/2020 – 1º S
(MEAer e MEEC)
Trabalho Final

1 Introdução

Com este trabalho pretende-se que seja desenvolvida uma aplicação com a finalidade de permitir a escolha de rotas aéreas, entre um aeroporto de início de viagem e um aeroporto de fim de viagem. Este tipo de problema ocorre sempre que um turista recorre a uma agência de viagens, ou sempre que uma empresa de despacho de mercadorias (DHL, …) necessita enviar um produto.

Na decisão para a escolha de uma dada rota podem ser utilizados diversos critérios, como, o custo da viagem, a duração da viagem, a hora de partida e/ou a hora de chegada ao destino e, também o número de ligações entre voos (pode não existir uma ligação directa entre o aeroporto inicial de viagem e o aeroporto de final de viagem).

2 Dados a serem processados pela aplicação

Aeroportos

A aplicação começa por ler um ficheiro (**aeroportos.txt**) que contém em cada linha a informação de um aeroporto, exemplo para o aeroporto de Lisboa

LPPT LIS 38 46 52 N 9 8 9 W LISBON 0

LEMD MAD 40 29 36 N 3 34 0 W MADRID 1

Atenção, neste ficheiro o formato das linhas é o seguinte:

<identificado ICAO> <identificador IATA> <Latitude> <Longitude> <Cidade> <TimeZone>

Nota: Nos campos da Latitude e da Longitude, em vez do formato 38º46´52” passa a ser utilizado o formato 38 46 52. Os separados º, ´, “ são substituídos por espaços.

O ficheiro aeroportos.txt deve ter os aeroportos dos países que fazem parte da União Europeia (incluindo o Reino Unido – o Brexit ainda não ocorreu!). Para construir o ficheiro deve consultar o site https://www.greatcirclemapper.net/ ou outro site. No ficheiro devem ser utilizadas letras maiúsculas. A “TimeZone” de um aeroporto pode ser obtida através do site https://www.timeanddate.com/time/map/. Lisboa está na ”time zone” GMT 0H, MADRID está em GMT+1H.

Companhias Aéreas

A aplicação também deve ler um ficheiro (**rotas.txt**) que contém informação das companhias e das rotas que são realizadas pelas companhias.

Considera-se que o ficheiro rotas.txt contém unicamente ligações directas, i.e., a aeronave levanta do aeroporto A e aterra no aeroporto de destino B, sem realizar qualquer aterragem entre A e B. Como exemplo do formato do ficheiro, apresentam-se as linhas seguintes:

AIRLINE: TAP

TP831 ROME 11:40 ~~AM~~ LISBON ~~1:45 PM~~ 13:45

TP846 LISBON 10:00 ~~AM~~ ROME ~~2:00 PM~~ 14:00

…

AIRLINE IBERIA

…

Nota: A palavra AIRLINE: é obrigatória e tem como finalidade facilitar a indicação do nome da companhia aérea. As linhas seguintes indicam os voos directos dessa companhia, em que o formato é dado por:

<Código do voo> <Aeroporto de partida> <hora de partida – local> <Aeroporto de chegada> <hora de chegada – local>

Formato da hora 00:00 até 23:59. O dia começa às 00:00!

A aplicação deve ler os ficheiros e deve guardar a informação em estruturas ligadas usando memória dinâmica. Nota: Não é permitida a utilização de vectores dinâmicos para guardar essa informação.

3 Funcionamento do programa

A interacção entre o utilizador e o programa é realizada através da linha de comando. O programa tem acesso aos pedidos do utilizador através dos argumentos da função main. Assume-se neste documento que o programa executável tem o nome de **rota2019**. A aplicação deve apresentar, no ecrã, uma listagem da informação correspondente ao comando que é especificado pelo utilizador. Exemplificação dos comandos:

1. Visualização da informação dos aeroportos

   rota2019 -aeroportos

2. Visualização da informação dos voos directos. D*eve incluir a distância entre os aeroportos*.

   rota2019 -voos

3. Pedido de uma rota directa entre um aeroporto A (LISBON) e um aeroporto B (ROME), ligação directa (-L 0)

   rota2019 LISBON ROME -L 0

4. Idêntico ao ponto anterior mas a listagem é ordenada por hora de partida crescente (-TC)

   ou decrescente (-TD)

rota2019 LISBON ROME -L 0 -TC

5. Pedido de uma rota com uma ligação (mas sem a preocupação de ligação horária)

   rota2019 LISBON ROME -L 1

6. Pedido de uma rota com uma ligação (com preocupação de ligação horária)

   rota2019 LISBON ROME -L 1 -TC

7. Pedido de uma rota com ligação (com preocupação de ligação horária) e com a menor distância

   rota2019 LISBON ROME -L 1 -TC -D

Nota: "Ligação horária" significa que a hora de chegada de um voo a um aeroporto deve ser inferior à hora de partida do voo de ligação.

4 Distância entre dois aeroportos

O cálculo da distância entre dois aeroportos deve ser realizada utilizando o conceito de grande círculo (pesquisar na Internet). Assume-se que a Terra é esférica, raio R0 = 6371 Km, e que há um sistema de eixos solidário com a Terra, ortogonal, com origem no centro da Terra. O eixo Z está alinhado com o eixo de rotação da Terra, aponta para a estrela Polar, o eixo X passa no ponto de intersecção do meridiano de Greenwich e o equador. O eixo Y é ortogonal aos eixos X e Z. O procedimento para o cálculo consiste nos passos seguintes:

1. O voo decorre ao longo de uma arco de circunferência de raio R = R0 + 10 km, em que o valor de 10km representa a altitude do voo.
2. Transformar as coordenadas geográficas (latitude e longitude) de um aeroporto (A) nas coordenadas (xa, ya, za), eixos X, Y e Z, o que permite obter o vector OA (O representa a origem do sistema de eixos XYZ). Aplicar as equações seguintes:

$$x = R\cos(latitude)\cos(longitude)$$
$$y = R\cos(latitude)\sin(longitude)$$
$$z = R\sin(latitude)$$

3. Transformar as coordenadas geográficas (latitude e lobgitude) de um aeroporto (B) nas coordenadas (xb, yb, zb), eixos X, Y e Z, o que permite obter o vector OB (O – origem).
4. Calcular o produto interno entre os vectores OA e OB para determinar o ângulo entre eles.
5. Utilizar o ângulo (em radianos) e o raio R para calcular a distância correspondente ao voo.

5 Aplicação de conhecimentos

A resolução deste trabalho implica a aplicação de conhecimentos dos pontos seguintes do programa da unidade curricular:

1. Definição de novos tipos, tipos estruturados e definição de funções

2. Organização do trabalho em ficheiros. Compilação de módulos separados.
3. Organização de dados em estruturas ligadas e memória dinâmica. **Não é permitida a utilização de vectores dinâmicos.**

6 Organização de dados e do código

Neste trabalho o programa deve ser desenvolvido com base **na norma ansi**.

O código do programa deve estar organizado em funções, de acordo com as boas práticas seguintes:

1. As funções devem estar documentadas e as instruções devem estar bem alinhadas.
2. Uma função não deve ter mais de 50 a 60 linhas, tamanho da fonte 12pt.
3. Uma função deve ter no máximo 5 argumentos.

7 Entrega e avaliação

A aplicação será avaliada nos computadores do laboratório em ambiente (sistema operativo) Gnu-Linux. Os alunos devem testar o programa nos computadores de laboratório ANTES de realizarem o *upload* da versão final no sistema fénix.

Devem colocar num ficheiro zip os ficheiros com o código fonte (ficheiros *.c e *.h) conjuntamente com os ficheiros de dados que foram utilizados no teste do programa. Não deve incluir o programa executável.

Elementos a serem avaliados nesta fase:

- Organização de dados.
- Organização do código.
- Acesso às opções do utilizador através dos argumentos da função main.
- Leitura do ficheiro aeroportos.txt.
- Leitura do ficheiro rotas.txt.
- Funcionamento das opções que estão indicadas na secção “Funcionamento do Programa”.

- **Data de entrega:** Consultar a página da disciplina a data correspondente a **Entrega final**.

8 Dúvidas

As dúvidas podem ser esclarecidas no início e no fim das aulas, e no horário de esclarecimento de dúvidas.

Nota: Na definição inicial das funcionalidades de uma aplicação, existem elementos que não estão completamente especificados. Nessas situações, o programador pode especificar os elementos em falta utilizando para o efeito justificações lógicas.

Votos de um bom processamento!