Universidade Federal do Espírito Santo - CT - DI
Trabalho Computacional II - Programação II
Engenharia e Ciência da Computação - 2014/2
Data para entrega: 05/12/2014

Considere o esboço do estado do ES e $n = 30$ cidades enumeradas de $1, 2, \ldots, n$. Os nomes das cidades e as suas respectivas coordenadas x e y estão armazenadas no arquivo **nome-coord.txt**. Cada estrada entre duas cidades possui um custo por quilômetro que deve ser multiplicado pela distância para calcular o custo de viagem entre as cidades. Além disso, o hotel em cada cidade possui um valor para a diária. Os valores das diárias e os custos por quilômetro estão armazenados no arquivo **diaria-custo.txt**.

Faça um programa na linguagem C utilizando obrigatoriamente vetores, strings, matrizes, structs, arquivos e construção de biblioteca para:

1. Ler o arquivo de dados **nome-coord.txt**;

2. Ler o arquivo de dados **diaria-custo.txt**;

3. Construir uma matriz $D_{n \times n}$ de distâncias entre todas as cidades e armazenar em um arquivo que se chamará **distancia.txt**;

4. Construir uma matriz $C_{n \times n}$ de custos de viagem entre todas as cidades e armazenar em um arquivo que se chamará **custo.txt** com a expressão $CV_{ij} = c_{ij} \times d_{ij} + diaria_j$;

5. Imprimir na tela os nomes cidades mais ao norte, mais ao sul, mais a oeste, mais a leste e a mais central. Em caso de empate imprimir todas as cidades com mesmo valor nas grandezas avaliadas;

6. Construir (imprimir na tela) o caminho da **cidade 1** até a **cidade n**, passando por todas as cidades na ordem $1 \rightarrow 2 \rightarrow 3 \rightarrow \ldots \rightarrow 29 \rightarrow n$. Calcular (imprimir na tela) a distância total deste caminho (usando a matriz D) e o custo total desta viagem (usando a matriz C), considerando que o viajante dormirá uma noite em cada cidade, exceto a cidade 1;

7. Construir (imprimir na tela) um caminho da **cidade 1** até a **cidade n** seguindo a **Lei de Formação 1**: a cidade seguinte no caminho é a mais próxima e sempre considerando as cidades a frente na ordem fornecida no arquivo **nome-coord.txt** e ilustradas no gráfico. Exemplo: se estamos na **cidade 10** a cidade seguinte será a **cidade 15** e não a **cidade 8 ou 9**. Se houver empate com relação à distância, escolher a cidade que diminui o número total de cidades visitadas no caminho. Calcular (imprimir na tela) a distância total deste caminho e o custo total desta viagem;

8. Construir (imprimir na tela) um caminho da **cidade 1** até a **cidade n** seguindo a **Lei de Formação 2**: a cidade seguinte no caminho é a menos custosa e sempre considerando as cidades a frente na ordem fornecida no arquivo **nome-coord.txt** e ilustradas no gráfico. Se houver empate com relação ao custo, escolher a cidade que diminui o número total de cidades visitadas no caminho.

Calcular (imprimir na tela) a distância total deste caminho e o custo total desta viagem;

9. Fornecidos pelo teclado as cidades **origem** e **destino** com rótulo da **origem** sempre menor que o rótulo do **destino**:

   (a) Construir (imprimir na tela) um caminho da **cidade origem** até a **cidade destino** seguindo a **Lei de Formação 1**: a cidade seguinte no caminho é a mais próxima e sempre considerando as cidades a frente na ordem fornecida no arquivo **nome-coord.txt** e ilustradas no gráfico.
   Exemplo: se **origem= 10** e **destino=13**, observar no mapa que a cidade seguinte seria a **cidade 15** mas esta não é válida pois $15 > 13$. Então, a cidade seguinte no caminho será **cidade 11**, depois **cidade 12** e finalmente **destino=13**, seguindo a **Lei de Formação 1**. Se houver empate com relação à distância, escolher a cidade que diminui o número total de cidades visitadas no caminho. Calcular (imprimir na tela) a distância total deste caminho e o custo total desta viagem;

   (b) Construir (imprimir na tela) um caminho da **cidade origem** até a **cidade destino** seguindo a **Lei de Formação 2**: a cidade seguinte no caminho é a menos custosa e sempre considerando as cidades a frente na ordem fornecida no arquivo **nome-coord.txt** e ilustradas no gráfico. Caso a cidade seguinte no caminho for maior que **destino**, escolher cidade válida de acordo com a **Lei de Formação 2**. Se houver empate com relação ao custo, escolher a cidade que diminui o número total de cidades visitadas no caminho. Calcular (imprimir na tela) a distância total deste caminho e o custo total desta viagem.

Forma de Entrega:

1. Enviar o arquivo do código fonte do programa, o arquivo da biblioteca construída e makefile para o e-mail **crangel@inf.ufes.br**. Certifiquem-se que a versão enviada é a mais atual.

2. O assunto do e-mail deverá ser o seguinte:
   Trabalho2:ProgramacaoII:CC:nomes dos integrantes do grupo (Ciência da Computação)
   Trabalho2:ProgramacaoII:EC:nomes dos integrantes do grupo (Engenharia de Computação);

3. Os grupos que, por ventura, atrasarem a entrega terão 1(um) ponto descontado para cada dia de atraso. Apenas um integrante do grupo deve ser responsável por enviar o trabalho.

Veja abaixo um exemplo de um e-mail de envio do trabalho do grupo formado por alunos hipotéticos João da Silva e José Geraldo Castro do curso de Ciência da Computação (enviado por João da Silva).

Para: crangel@inf.ufes.br
De: João da Silva
Assunto: Trabalho2:ProgramacaoII:CC:JoaoSilva:JoseCastro

Outras Observações Importantes:

1. Em caso de deteção de cópia (parcial ou integral), todos os envolvidos recebem nota ZERO.

2. Trabalho que não compila recebe nota ZERO. Não adianta submeter.

3. Os trabalhos serão compilados e verificados usando o compilador gcc no sistema operacional Linux.

4. Os programas serão avaliados pela sua correção durante a execução e também pelo estilo de programação. Serão observados particularmente se os programas possuem os comentários apropriados, se usam nomes significativos para as variáveis e funções, se o código está indentado e modularizado corretamente.

A seguir, o esboço do ES com as $n = 30$ cidades.