SCC0216 – Modelagem Computacional em Grafos
Prof. Ricardo J. G. B. Campello
Estagiário PAE: Francisco A. R. Anjos
1º semestre 2016

Laboratório 4 – "Workaholic"

Introdução

Joãozinho é um engenheiro de produção que presta consultoria a várias fábricas de médio porte. Cada vez que recebe uma solicitação, ele viaja até a cidade de seu cliente para atendê-lo. Devido ao grande número de viagens, Joãozinho passa pouquíssimo tempo com sua família, fato este que está terminantemente disposto a mudar. Porém, antes de mais nada, Joãozinho precisa honrar os contratos já firmados. Pensando nisso ele decidiu terceirizar parte de suas consultorias passando-as para antigos colegas. Afim de escolher quais clientes continuará atendendo e quais serão terceirizados, ele estabeleceu os seguintes critérios:

1. Ele continuará atendendo todas as cidades em que o preço de sua viagem mais o preço da sua hospedagem seja menor do que o custo total de contratar qualquer um de seus amigos.
2. Para o colega escolhido, Joãozinho pagará um valor fixo X, que é igual para todos, mais o preço de se viajar da cidade do seu colega até a cidade do cliente (diária não é paga ao colega, pois esta já está inclusa no valor X).
3. O serviço sempre será passado ao amigo mais barato (ou seja, ao amigo mais "próximo" do cliente, considerando o custo total).

A família de Joãozinho gostou tanto da mudança que convenceram-no a trabalhar assim permanentemente. No entanto, de tempos em tempos, o valor das viagens, das hospedagens e dos amigos dispostos a trabalhar com Joãozinho mudam e ele é obrigado a replanejar tudo de novo. Pensando nisso, com o intuito de automatizar o processo de escolha, Joãozinho entrou em contato com você para que faça um "programinha" que dado as cidades dos seus clientes, os preços das viagens e as cidades de seus amigos, determine se ele repassa o serviço ou não.

Entrada

Os dados – fornecidos na entrada-padrão – contêm apenas um caso de teste. Na primeira linha é informado a quantidade **N** de cidades, na segunda, os valores **X** de se contratar um amigo e **Y** de uma diária (paga apenas se Joãozinho decidir viajar). A terceira linha contém a quantidade **M** de rotas entre essas cidades, seguidas por **M** linhas na forma ***<id_cidade>***, ***<id_cidade>*** e ***<valor_passagem>***, logo após é informada a quantidade **C** de amigos que estão dispostos a trabalhar com Joãozinho, seguido por **C** linhas que indicam as cidades de cada um deles. A última linha da entrada realiza uma consulta informando o tipo da saída (**1**, **2** ou **3**) que o programa deve produzir e a cidade consultada. Abaixo tem-se um exemplo de entrada:

7	número de cidades (int)
10.0 2.0	custo de terceirização & custo de hospedagem (double, double)
8	número de rotas possíveis (int)
1 3 10.0	rota entre as cidades 1 e 3 que possui custo 10.0 (int, int, double)
3 2 5.0	...
2 5 15.0	...
3 5 20.0	...
5 6 3.0	...
4 6 7.0	...
3 4 19.0	…
5 7 15.0	...
2	número de amigos (int)
5	cidade do primeiro amigo (int)
7	...
1 4	tipo da operação a ser realizada & cidade do cliente (int, int)

Observação 1: O número de cidades **N** é menor do que ou igual a **50**, o número de rotas **M** é menor do que ou igual a **200**, o número de consultores **C** é menor do que ou igual a **10**, o valor **X** de se contratar um consultor **SEMPRE** será maior do que o valor **Y** da diária de Joãozinho. A diária de Joãozinho só é paga na cidade do atendimento.

Observação 2: Os identificadores das cidades serão números inteiros **positivos** enquanto os custos das rotas, o valor **X** e o valor **Y** serão números reais **positivos** que devem ser modelados como **double** (ou seja, não existem custos nulos ou negativos).

Observação 3: Considere que Joãozinho mora na cidade de identificador 1 e que não existirão caminhos distintos entre duas cidades que possuam o mesmo custo.

Observação 4: As rotas entre cidades são bidirecionais (ou seja, podem ser percorridas nos dois sentidos e possuem o mesmo custo). Cada rota aparece apenas uma vez no caso de teste e os identificadores das cidades não estão ordenados.

Observação 5: Nenhuma cidade vizinha à de Joãozinho possui consultor.

Saída

O programa deverá possuir três tipos de saída diferentes: para a opção **“1”**, o programa deverá imprimir na tela o **custo total** da solução ótima para atender o cliente na cidade informada na última linha da entrada (a partir da cidade de quem fará o atendimento – Joãozinho ou um de seus amigos). O valor impresso deve possuir exatamente três casas de precisão e ser seguido por uma quebra de linha (**“\n”**), conforme exemplo a seguir:

20.000

Nota: Por simplicidade, para cálculo do custo total, considere apenas o valor de **ida** do custo da viagem (ou seja, não considere ida e volta), mais diária ou valor de consultoria.

Para a opção **“2”**, o programa deverá imprimir o caminho ótimo entre a cidade de quem fará o atendimento (Joãozinho ou um de seus amigos) e a cidade consultada. Os identificadores impressos devem estar separados por um único espaço em branco. Além disso, o identificador da última cidade deverá ser seguido por uma quebra de linha (**“\n”**), conforme o exemplo abaixo:

5 6 4

Para a opção **“3”**, o programa deverá imprimir a palavra **“Atende”** caso Joãozinho preste a consultoria pessoalmente ou **“Repassa”** caso contrate um amigo seguido do custo total do atendimento. O resultado **“Atende”** ou **“Repassa”** deve estar separado por um único espaço em branco do valor impresso, que deve possuir exatamente três casas de precisão e ser seguido por uma quebra de linha (**“\n”**). Veja o exemplo a seguir:

Repassa 20.000

Considerações Finais

1. O aluno deverá utilizar a linguagem de programação **C (padrão ANSI)**;
2. Os trabalhos deverão ser feitos individualmente;

3. O arquivo **makefile deve está na raiz do arquivo compactado**. Caso contrário, a plataforma run.codes não consegue encontrá-lo;
4. Serão anulados todos os trabalhos nos quais forem detectados qualquer tipo de cópia ou plágio, não importando a origem;
5. Qualquer aluno que esqueça de assinar a lista de presença terá o seu trabalho anulado;
6. Os trabalhos devem ser submetidos pela plataforma run.codes **(código da disciplina: 4ZGF)**;
7. O horário limite para a entrega é **23h59m** de **10 de maio**;