Atividade Prática de Programação - AP12

Caminho Mínimo em Grafo com Restrição

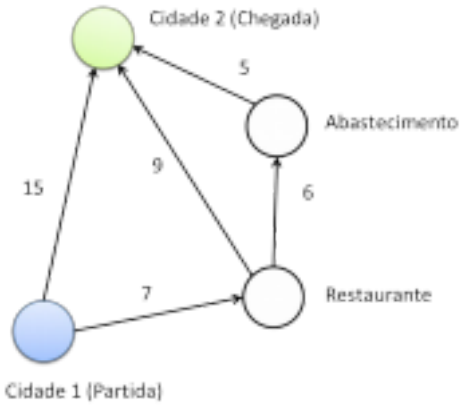
Algoritmos e Estruturas de Dados II

Tempo limite: 1s

Instruções:

1. E/S: tanto a entrada quanto a saída de dados devem ser “secas”, ou seja, não devem apresentar frases explicativas.
2. Identificadores de variáveis: escolha nomes apropriados;
3. Documentação: inclua cabeçalho, comentários e indentação no programa;
4. Plágios serão punidos na atribuição da nota;
5. Submeta o programa no sistema Judge.

Em determinadas viagens muito longas, na maioria das vezes, torna-se indispensável realizar determinadas pausas para alimentação, abastecimento do veículo, descanso, entre outras coisas. Dessa forma, nem sempre o caminho mais curto é o melhor caminho a ser seguido nesses trajetos. Isso pode ser observado na figura abaixo.



A partir disso, encontrar o caminho mínimo de um ponto até outro, exige em alguns casos o uso de restrições. Transpondo isso para um grafo, é possível afirmar, de forma simplificada, que na busca do caminho mínimo entre dois nós/vértices, deve-se considerar arestas com custo **menor ou igual** a um valor **R** (restrição).

Deve-se escrever um programa que exiba o caminho mínimo de um determinado **grafo orientado**, levando em consideração uma restrição nesse caminho mínimo

encontrado, ou seja, o caminho mínimo que percorre arestas que tem, cada uma, custo menor ou igual à restrição informada. Essa operação deve ser realizada considerando apenas um nó de partida, em relação a todos os nós (vértices) do grafo. Por exemplo, se o grafo tem 4 nós/vértices, e o nó inicial for 2, então a saída vai ser composta por quatro números, que representam a distância (caminho mínimo com restrição) do vértice 2 para alcançar os demais nós do grafo. Além disso, a saída também deverá exibir o caminho (rota) iniciando no vértice de origem e chegando a um vértice de destino determinado nos dados de entrada.

Deve-se usar o Algoritmo de Bellman-Ford para buscar o caminho mínimo com as adaptações necessárias para se trabalhar com restrições.

Entrada

A entrada para cada teste se inicia informando:

A. Um valor inteiro representando o nó onde será iniciada a busca do caminho mínimo. O nó inicial tem valor **V** ($0 \leq \mathbf{V} \leq 100$).
B. Um valor inteiro representando o valor da restrição aplicado as arestas, representado por **R** ($-1024 < \mathbf{R} < 1024$).
C. Um par de inteiros informando o número total de nós **N** ($0 \leq \mathbf{N} \leq 100$) e o número de arestas **M** ($1 \leq \mathbf{M} \leq 1024$).
D. Uma série de **M** linhas representando todas as arestas que serão criadas juntamente com o peso atribuído a cada uma. Dessa forma, cada linha terá 3 valores inteiros, como, por exemplo, 2 7 -3, a qual indica que existe uma aresta ligando o vértice 2 ao vértice 7 (2 -> 7) com peso -3. Os pesos **P** das arestas estão no intervalo $-1024 < \mathbf{P} < 1024$. Os vértices são representados por valores numéricos que variam entre 0 e 100.
E. Um último e único valor inteiro representando uma aresta válida, a qual servirá para determinação e exibição da rota do vértice de início (primeira linha) até esse vértice.

Saída

A saída do programa é exibida em duas linhas:

A. A primeira linha mostra o custo mínimo para cada vértice, a partir do caminho mínimo computado pelo algoritmo com restrição, em relação ao vértice de

origem determinado.

B. A segunda linha representa o caminho (rota) do vértice de origem até atingir o vértice de destino determinado, caso exista o caminho.

Dicas e restrições:

- Pode-se utilizar as estruturas de dados do tipo Pilha e Fila como mecanismos auxiliares;
- O programa deve ser e escrito em C/C++;
- Arestas podem ter custo negativo;
- Os vértices são numerados de 0 até N-1, onde N é a quantidade de vértices;
- O custo do caminho mínimo que um nó leva para chegar a si próprio é igual à zero;
- Deve-se ter no código as funções de: inicialização, inserção e busca;
- Deve-se utilizar uma Lista de Adjacência como estrutura de dados para organização do Grafo;
- Não existem arestas com laços (arestas cujo vértice aponta para si próprio).
- Quando um vértice não for alcançado pelo vértice de origem, a saída referente a primeira linha deve exibir, apenas para o vértice inatingível, a palavra "`INF`", relativo ao custo até o vértice inatingível. Para mais informações, veja o exemplo;
- Caso o vértice de destino definido não seja atingível pelo vértice de origem, ou seja, não há caminho até ele, deve-se exibir, na segunda linha, o texto: `Destino nao alcancado`. Mais informações nos exemplos.

Observação importante: deve-se considerar a **ordem crescente** quando for inserido um novo nó, representando uma aresta, na lista de adjacência. Por exemplo, considere que o nó/vértice 9 aponta (tem aresta de ligação) para 5, 6 e 8. Caso insira uma nova aresta 9 -> 3, o no/vértice 3 deve ser inserido antes do nó/vértice 5 na lista de adjacência. Caso insira a aresta 9- > 4, então o no/vértice 4 deve ser inserido antes do nó/vértice 5 e depois do no/vértice 3 na lista de adjacência.

Exemplos de Entrada e Saída

Exemplo de Entrada	Exemplo de Saída
0 8 8 9 0 1 4 0 3 -2 1 4 9 1 5 5 4 6 3 6 0 -5 5 2 7 2 7 5 2 4 -2 7	0 4 16 -2 14 9 17 21 0 1 5 2 7

Exemplo de Entrada	Exemplo de Saída
2 20 5 8 0 1 2 1 2 3 2 3 4 3 0 2 4 0 6 4 2 7 1 4 25 3 4 29 0	6 8 0 4 INF 2 3 0

Exemplo de Entrada	Exemplo de Saída
0 12 8 9 0 1 4 0 3 -2 1 4 9 1 5 5 4 6 23 6 0 -5 5 2 7 2 7 5 2 4 -2 6	0 4 16 -2 13 9 INF 21 Destino nao alcancado