MC202AB - Estrutura de Dados

Lab 11 - Grafos

Data da Primeira Chance: 5 de dezembro de 2022
Peso: 5

A empresa AMZ, reconhecida no mercado nacional por fazer suas entregas de forma rápida e barata, está tentando expandir seus negócios e oferecer o mesmo atendimento a áreas mais remotas nos interiores de cada estado do Brasil. Para isso, estão sendo estudados o alcance de cada centro de distribuição ***C***, os quais são responsáveis por distribuir encomendas a pontos de recebimento ***P***, responsáveis por receber encomendas e também por reencaminhá-las a outros ***P***.

Esse problema pode ser modelado como um grafo não-direcionado não ponderado, em que cada vértice pode ser do tipo ***C*** ou do tipo ***P*** ($0 \leq P \leq C \leq 20$), como mostra a imagem a seguir, onde se tem um centro de distribuição A e 5 pontos de recebimento (0 a 4):

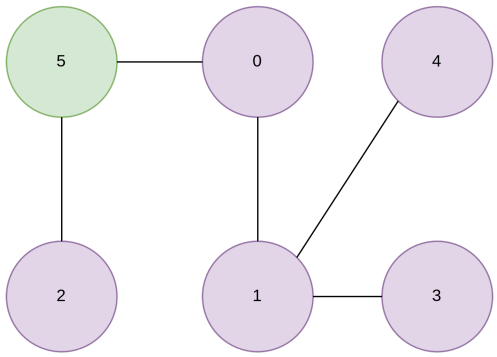

Figura 1: Um centro de distribuição **5** e 5 pontos de recebimento (**0 a 4**)

A nomeação dos vértices é feita da seguinte forma:

- A cada vértice do tipo ***P*** adicionado, este deve ser nomeado com um valor inteiro partindo do valor 0. Por exemplo: 5 vértices do tipo ***P*** serão adicionados, então estes vértices serão nomeados com os valores `"0", "1", "2", "3"` e `"4"`, respectivamente;
- A cada vértice do tipo ***C*** adicionado, este deve ser nomeado com um inteiro sequencialmente partindo do último valor de ***P***. Por exemplo: se houve 5 vértices do tipo ***P*** serão adicionados e em seguida são adicionados 2 vértices do tipo ***C***, então estes vértices serão nomeados com valores `"5" e "6"` respectivamente.

A ideia deste estudo da empresa é que, dado um vértice do tipo ***P***, seja calculada a distância entre ele e o vértice do tipo ***C*** mais próximo, isto é, o caminho mínimo entre um dado ponto de recebimento e o centro de distribuição mais próximo dele. Sempre haverá caminho quando a operação de distância for solicitada.

Caso essa distância calculada seja maior que um dado inteiro ***X*** (2 ≤ X ≤ 10), então um novo vértice do tipo ***C*** deve ser adicionado, o qual terá uma conexão direta para o vértice ***P*** da distância calculada, e também com todos os vértices adjacentes a ***P*** (isto é, vértices conectados a ***P***). Por exemplo, supondo que `X = 2`, e queremos calcular o número de arestas entre os vértices `0` e `25`:

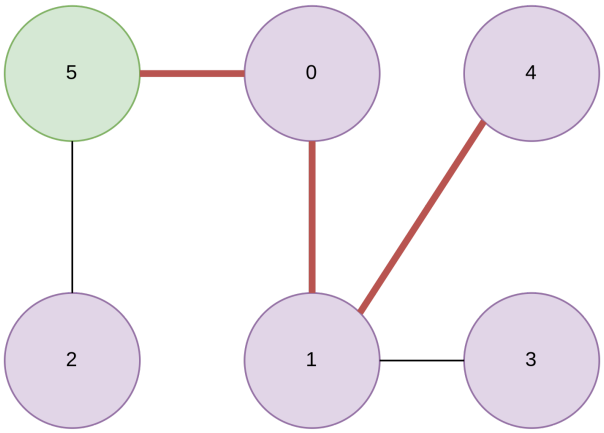

Figura 2: O número de arestas entre **5** e **4** tem tamanho 3, maior que X = 2

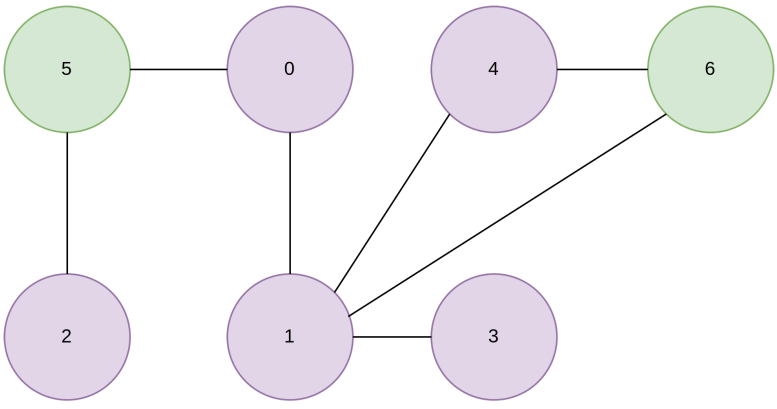

Figura 3: Vértice **6** adicionado, se conectando ao vértice 4 (o qual foi calculada a distância) e também ao vértice **1** (que é adjacente a **4**)

Também pode ser solicitado que determinada aresta de um vértice do tipo ***C*** seja removida durante a execução do estudo da empresa, por razões de corte de custos em locais pouco utilizados, analisados pelo setor de planejamento da AMZ. **Lembre-se:** se for solicitada a remoção da aresta “2 -> 3”, a aresta “3 -> 2” também precisa ser removida.

Entrada

A entrada do programa consiste em, primeiramente, informar a quantidade nós ***C*** e nós ***P*** no grafo em uma única linha. Depois disso, será dado, em uma linha, um valor inteiro ***E*** representando a quantidade de arestas no grafo, seguido ***E*** linhas com pares de vértices representando as arestas do grafo no formato ***[V1] [V2]***, em que $V_1 \neq V_2$.

Após estes dados para construção do grafo, será dado em uma linha um valor inteiro ***Q***, representando a quantidade de operações de cálculo de distância entre vértices, no formato ***"D [C] [P] X"*** ou de remoção de arestas, no formato ***"R [C] [P]"***.

Caso a distância entre os vértices ***[C]*** e ***[P]*** seja maior que ***X***, então um novo vértice do tipo ***C*** deve ser adicionado, como explicado no enunciado acima. A nomenclatura deste novo vértice será o valor inteiro seguinte do último adicionada, isto é, se o último vértice do tipo ***C*** adicionado tem o valor *"3"*, então este novo vértice terá o nome *"4"*.

Atenção: a adição deste vértice e suas conexões afeta o resultado das operações seguintes a essa.

Saída

A saída deve conter os seguintes itens:

- A mensagem ***"GRAFO AMZ CONSTRUIDO!"*** assim que todas as arestas do grafo forem lidas e armazenadas;
- A distância ***D*** entre vértices quando o comando de distância for requisitado, no formado ***"DISTANCIA [C] -> [P] = [resultado]"***
    - Caso seja o caso de um vértice do tipo ***C*** ser adicionado após esse cálculo, imprimir a mensagem ***"[C] ADICIONADO E CONECTADO A [P1] [P2] ..."***
    - A impressão destes valores ***"[P1] [P2] …"*** na mensagem acima deve ser em ordem crescente. Por exemplo: se um vértice F se conectou aos vértices 3, 2 e 4, então a mensagem a ser impressa é: ***"F ADICIONADO E CONECTADO A 2 3 4"***

- A mensagem ***"ARESTA [C] -> [P] REMOVIDO"***, quando o comando de remoção de aresta for executado

Exemplos

Exemplo 1:

Entrada

```
1 5
5
5 0
5 2
0 1
1 4
1 3
2
D 5 4 2
R 1 6
```

Saída

```
GRAFO AMZ CONSTRUIDO!
DISTANCIA 5 -> 4 = 3
6 ADICIONADO E CONECTADO A 1 4
ARESTA 1 -> 6 REMOVIDO
```

Exemplo 2:

Entrada

```
2 8
7
6 0
1 2
1 5
0 7
7 5
5 4
2 3
4
D 6 3 4
D 7 4 2
R 2 3
D 7 1 2
```

Saída

```
GRAFO AMZ CONSTRUIDO!
```

```
DISTANCIA 6 -> 3 = 6
10 ADICIONADO E CONECTADO A 2 3
DISTANCIA 7 -> 4 = 2
ARESTA 2 -> 3 REMOVIDO
DISTANCIA 7 -> 1 = 2
```

Regras e Avaliação

Importante: é obrigatório o uso de listas de adjacência para implementar o grafo da solução deste lab.

Seu código será avaliado não apenas pelos testes do CodePost, mas também pela qualidade. Dentre os critérios subjetivos de qualidade de código iremos analisar: o uso apropriado de funções, de comentários, e de structs; a escolha de bons nomes de funções, variáveis e de structs e seus campos; o correto uso de Tipos Abstratos de Dados e a correta separação em vários arquivos; a ausência de diversos trechos de código repetidos, e o tempo de execução e uso de memória dos algoritmos projetados. Note, porém, que essa não é uma lista exaustiva, pois outros critérios podem ser analisados dependo do código apresentado visando mostrar ao aluno como o código poderia ser melhor.

Submissão

Você deverá submeter no CodePost, na tarefa Lab 11 - Grafos, os arquivo ***grafo.h***, ***grafo.c*** e ***lab11.c***. No arquivo ***grafo.h*** é esperado que você declare a(s) estrutura(s) com os devidos campos e funções necessárias para implementar as operações com o *Grafo* desta atividade. No arquivo ***grafo.c*** é esperado que você implemente as funções declaradas no arquivo ***grafo.h***. Por fim, no arquivo ***lab11.c*** é esperado que você faça a leitura da entrada do programa e execute as chamadas das funções por meio do TAD implementado.

Após o prazo estabelecido para a atividade, será aberta uma tarefa Lab 11 - Segunda Chance, com prazo de entrega até o fim do semestre.