

Algoritmos e Estrutura de Dados
Lista de Exercícios - Questões Desafios
Assunto: Grafos e Encaminhamentos em Grafos

Orientações Gerais

- 1) Para as duas questões abaixo, você deve enviar os dois arquivos diferentes do código utilizado para resolver cada questão. Faça também o carregamento dos 3 arquivos de entrada da primeira questão e dos 2 arquivos de entrada exemplo da segunda questão.
- 2) A entrada dos programas é sempre um grafo que é lido a partir de um arquivo .txt. O arquivo de entrada tem o formato discriminado abaixo. A saída (print na tela) está explicitada em cada questão.

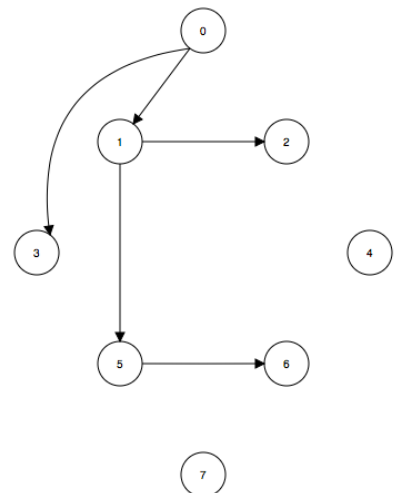
TEMPLATE DO ARQUIVO - PRIMEIRA QUESTÃO

N # quantidade de vertices (vertices sempre são rotulados com 0,..., N-1)
Q #quantidade de arestas
Vi Vj #dois numeros inteiros para sinalizar a conexão da aresta direcionada Vi->Vj
Vi Vj #dois numeros inteiros para sinalizar a conexão da aresta direcionada Vi->Vj
Vi Vj #dois numeros inteiros para sinalizar a conexão da aresta direcionada Vi->Vj

Exemplo de arquivo:

Dado o grafo ao lado, o arquivo de entrada da primeira questão é:

8
5
0 1
0 3
1 2
1 5
5 6



Para a segunda questão, é necessário informar dois vértices que representam onde sua busca inicia, e onde se deseja chegar pelo menos caminho. Logo o arquivo de entrada terá o seguinte formato

TEMPLATE DO ARQUIVO DO SEGUNDO ARQUIVO

```
N # quantidade de vertices (vertices sempre são rotulados com
0,..., N-1)
Q #quantidade de arestas
Vi Vj #dois numeros inteiros para sinalizar a conexão da
aresta direcionada Vi->Vj
...
Vi Vj #dois numeros inteiros para sinalizar a conexão da
aresta direcionada Vi->Vj
Vi Vj #dois numeros inteiros para sinalizar a conexão da
aresta direcionada Vi->Vj
Vi Vd #dois numeros inteiros para sinalizar que a busca deve
iniciar de Vi e o vértice onde se deseja chegar é o vértice Vd
```

Exemplo de arquivo:

Dado o grafo acima, o arquivo de entrada da segunda questão pode ser o seguinte abaixo, onde 0 é o vértice de origem da busca em largura e 6 é o destino.

```
8
5
0 1
0 3
1 2
1 5
5 6
0 6
```

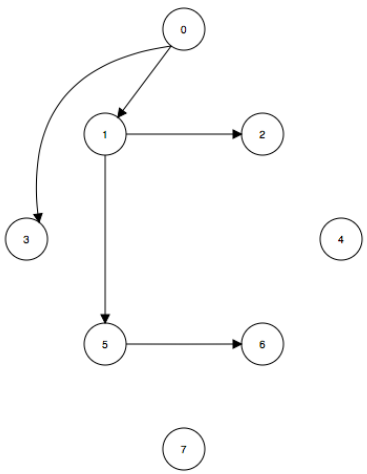
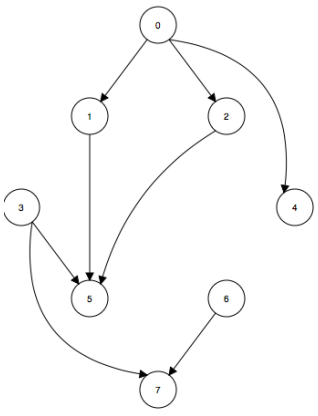
Questões

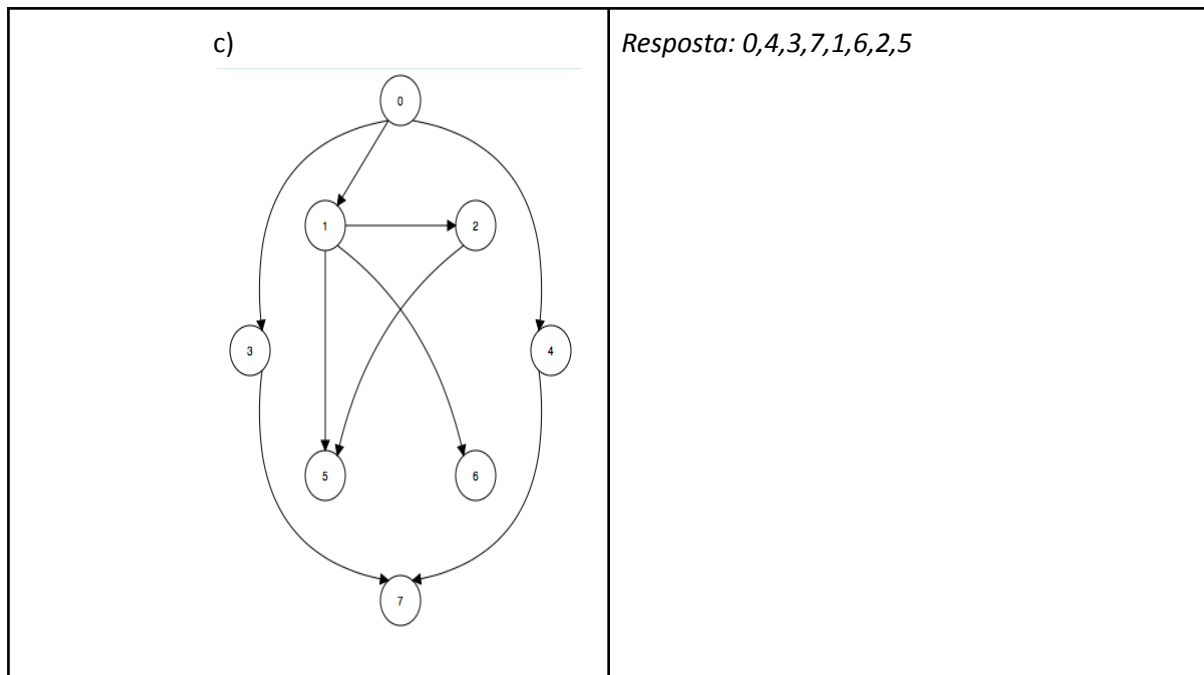
- 1) Em um projeto de sistemas, há várias tarefas a serem realizadas. Essas tarefas possuem dependência entre si, porque muitas vezes, é preciso terminar mais de uma tarefa para realizar alguma outra. Você foi contratado pelo Google para gerar a lista de tarefas que vão ser executadas em um projeto de melhoria do Youtube. Nós sabemos que, para cada tarefa, pode existir dependência de que outras tarefas tenham sido finalizadas anteriormente. Este é um problema conhecido como ordenação topológica. Utilizando necessariamente o encaminhamento em profundidade, implemente a lista de tarefas dado um grafo das tarefas e suas dependências do projeto.



Um applet que mostra a execução da ordenação topológica para alguns grafos pode ser encontrado aqui: <https://www.cs.usfca.edu/~galles/visualization/TopoSortDFS.html>

Execute seu algoritmo para os grafos abaixo:

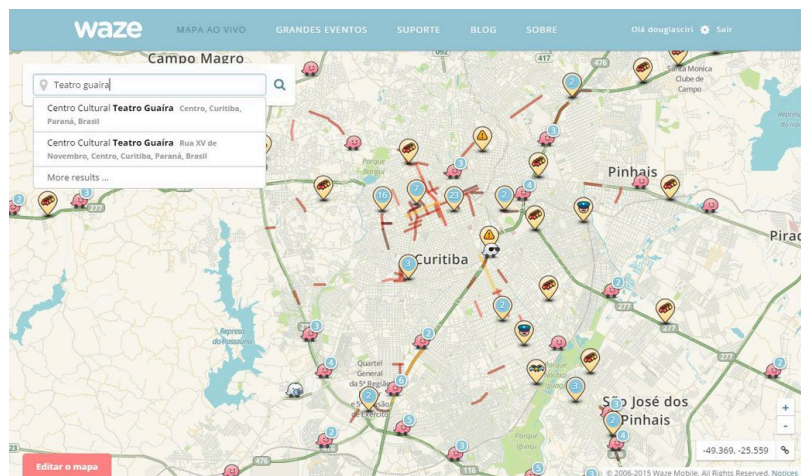
<p>a)</p> 	<p>Resposta: 7,4,0,3,1,5,6,2</p>
<p>b)</p> 	<p>Resposta: 6,3,7,0,4,2,1,5</p>



2) Como encontrar o melhor caminho entre dois pontos? Como ir de um ponto a outro na cidade gastando o menor tempo?

Para modelar esse problema, o Waze considera diversas variáveis: tamanho das ruas, engarrafamentos nas ruas, velocidade máxima nas ruas, etc.

Se cada rua fosse "igual" (ou seja, possuísse o mesmo tamanho, o mesmo nível de engarrafamento, a mesma velocidade da via, etc), poderíamos resolver o problema do Waze com o algoritmo por encaminhamento por largura.

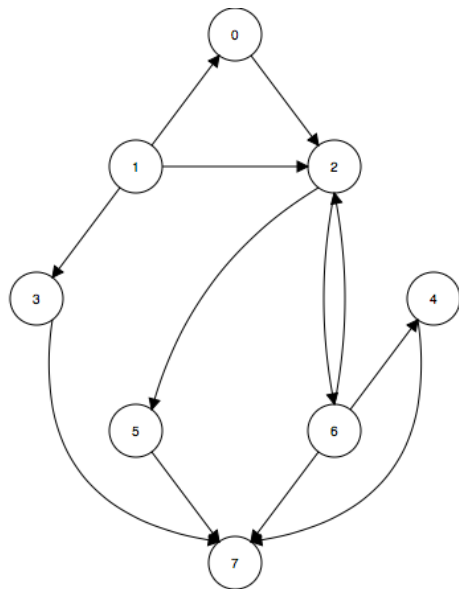


Vamos implementar o Waze considerando essas condições. Cada aresta representa uma rua possível de se trafegar na cidade. Entre dois pontos, qual o melhor caminho (menor sequência de arestas) que eu posso tomar?

Receba como entrada um grafo, G , e dois vértices desse grafo, A , B , e verifique qual o menor caminho (se existir) entre A e B neste grafo.

Um applet que mostra a execução da busca em largura para alguns grafos pode ser encontrado aqui: <https://www.cs.usfca.edu/~galles/visualization/BFS.html>

a)



A=0, B=7:

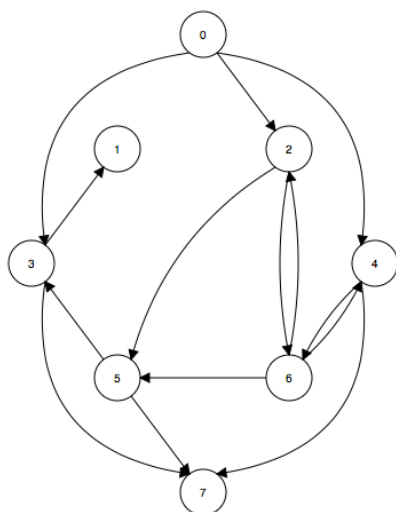
0,2,5,7

A=1, B=4

1,2,6,4

A=4, B=3

b)



A=0, B=7

0,3,7

A=4, B=1

4,6,5,3,1