Centro Federal de Educação Tecnológica de Minas Gerais - Departamento de Computação

Laboratório de Algoritmos e Estruturas de Dados II (2ECOM.025)

Lista de Exercícios 7

Professores: Amadeu Almeida e Thiago Rodrigues **Data de entrega:** 14 de Outubro de 2019

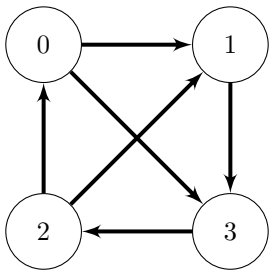
Instruções: leia com atenção **todas** as orientações abaixo antes de começar a lista.

- Os exercícios deverão ser feitos individualmente ou em dupla e valem 5 pontos.
- Submeta no SIGAA um arquivo compactado chamado **Lista6.zip**. Ele deverá conter:
  - As classes em Java que implementam as soluções das tarefas.
  - Um relatório em **PDF** contendo os grafos resultantes de cada experimento e as suas respectivas explicações.
- Caso a tarefa seja feita em dupla, apenas um aluno precisa submetê-la.
- O prazo de entrega é até às 23:59 do dia 14 de Outubro.
- Códigos fonte em PDF e submissões em atraso serão desconsiderados.
- O enunciado desta lista possui 3 exercícios e 3 páginas.

Exercício 1: classe Grafo

Esta tarefa objetiva implementar uma classe que manipula um grafo orientado que é representado por uma matriz de adjacências. Um exemplo deste tipo de grafo contendo 4 vértices e 6 arestas e sua matriz de adjacências são apresentados na figura abaixo. O peso de cada aresta do grafo é exibido na matriz.

Grafo:



Matriz de adjacência:

Vértice	0	1	2	3
0	0	5	0	3
1	0	0	0	3
2	6	6	0	0
3	0	0	1	0

1. Abra o projeto **ListasDeGrafos** e o arquivo Grafo.java.
2. Adicione o método a seguir na classe **Grafo**:
   - public int numVertices() - retorna a quantidade de vértices no grafo.

Exercício 2: classe AlgoritmosEmGrafos

O objetivo deste exercício é acrescentar o algoritmo de Dijkstra para obtenção de caminhos mais curtos à biblioteca de procedimentos em grafos que está sendo construída na classe **AlgoritmosEmGrafos**.

1. Acesse o arquivo **AlgoritmosEmGrafos.java**.
2. Insira os seguintes atributos na classe **AlgoritmosEmGrafos**:
   - **private final int[] distanciasCMC;** - guarda a distância entre o vértice inicial do algoritmo de Dijkstra e cada um dos outros nós do grafo.
   - **private final int[] verticeAntecessorCMC;** - armazena o vértice antecessor de cada nó pertencente a pelo menos um dos caminhos encontrados durante a execução do algoritmo.
3. Elabore os métodos a seguir:
   - **public AlgoritmosEmGrafos(int vertices)** - o método construtor deve inicializar os novos atributos da classe juntamente com os antigos.
   - **public int iniciaCaminhoMaisCurto(int verticeInicial, int verticeFinal)** - calcula o caminho mínimo entre dois vértices do grafo.
   - **public int[] iniciaCaminhoMaisCurto(int verticeInicial)** - calcula o caminho mais curto entre o vértice inicial e **todos** os outros.
     (a) Nos dois métodos acima, a distância mínima entre o vértice de partida e os demais deve ser iniciada com o valor ∞ (utilize a constante **Integer.MAX_VALUE** para representar este número), enquanto o caminho mínimo entre o vértice inicial e ele mesmo é igual 0. Já o valor de todos os vértices antecessores é se inicia com o valor -1.
   - **private void caminhoMaisCurto(int verticeInicial)** - implementa o algoritmo de Dijkstra.
   - **public int[] getVerticeAntecessorCMC()**
4. Explique o objetivo dos métodos implementados no item 3 por meio de comentários no próprio código.

Exercício 3: experimentos computacionais

Este exercício visa testar o funcionamento do algoritmo de Dijkstra em diferentes tipos de grafos orientados.

1. Antes de começar os experimentos, faça o download dos dois grafos que estão anexados junto a esta lista no SIGAA. A primeira linha de cada arquivo possui, respectivamente, a quantidade de vértices e arestas dos grafos, enquanto as demais contém uma das arestas do grafo. O vértice de saída, de entrada e o peso da aresta são retratados, respectivamente, no primeiro, no segundo e no terceiro número de cada linha.

   Por exemplo, o arquivo relativo ao grafo mostrado na página 1 deste documento é apresentado da seguinte forma:

   $$\begin{array}{ccc} 4 & 6 & \\ 0 & 1 & 5 \\ 0 & 3 & 3 \\ 1 & 3 & 3 \\ 2 & 1 & 6 \\ 2 & 2 & 6 \\ 3 & 2 & 1 \end{array}$$

2. Experimento 1:
   (a) Execute o algoritmo de Dijkstra para calcular o caminho mais curto entre os vértices 0 e $V - 1$ de cada um dos dois grafos, onde V é a quantidade de vértices do grafo.

(b) Para cada grafo, retorne:

i. O valor da distância mínima entre os nós 0 e $V-1$.

ii. Os vértices que estão no menor caminho entre 0 e $V-1$.

3. Experimento 2:

(a) Rode o procedimento de Dijkstra em cada um dos dois grafos para computar a distância mínima entre o vértice 0 e todos os outros.

(b) Para cada grafo, devolva:

i. Os custos dos caminhos mais curtos entre o vértice 0 e todos os outros.

ii. Os nós que estão nos caminhos mínimos entre o vértice 0 e todos os outros.

4. Crie duas tabelas que retratam os resultados do experimento 2 para ambos os grafos. Cada tabela possui três colunas contendo respectivamente:

- O número de cada vértice.
- O valor do caminho mais curto entre o vértice 0 e o atual.
- O vértice anterior ao atual no caminho entre o nó 0 e o corrente.

Modelo de Tabela:

Vértice	Caminho Mínimo	Anterior

Tabela relativa ao grafo exemplificado no início desta lista:

Vértice	Caminho Mínimo	Anterior
0	0	-1
1	5	0
2	4	3
3	3	0

5. Desenhe quatro **grafos**:

(a) Um que apresente, para cada grafo testado, os vértices que estão no caminho entre os nós 0 e $V-1$ e os valores das distâncias entre cada par de vértices consecutivos nesta rota. Em seguida, responda seguintes questões:

i. Por que nem todos os vértices dos grafos estão nestes caminhos?

ii. Os grafos que retratam as duas rotas contém ciclos? Por quê?

iii. Os grafos que contêm as rotas entre os nós 0 e $V-1$ são subgrafos dos originais? Por quê?

(b) Um que exiba, para cada grafo original, os vértices que estão em todos os caminhos entre o nó 0 e todos os outros e os custos das distâncias entre os pares de vértices consecutivos em cada uma destas rotas. Depois, responda as perguntas a seguir:

i. Por que alguns caminhos apresentados nestes grafos estão contidos em outros?

ii. Os grafos que contêm os caminhos entre os vértices 0 e $V-1$ são subgrafos daqueles que retratam as rotas entre o nó 0 e todos os outros? Por quê?

iii. Os grafos gerados acima são árvores? Por quê?

Guarde o projeto contendo as classes Grafo e AlgoritmosEmGrafos para as aulas seguintes, pois ele será utilizado nas próximas listas de exercícios.