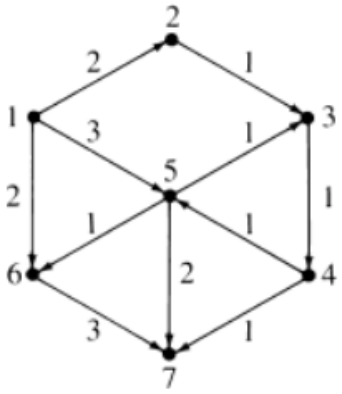
Algoritmos e Estruturas de Dados II

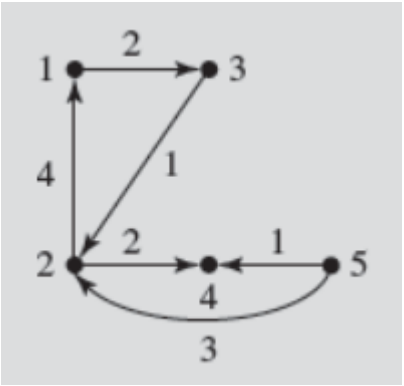
Exercícios – Caminho mais Curto (parte 2)

1) Dado os seguintes grafos, aplique e desenhe as matrizes D e P de cada passo do algoritmo de Floyd-Warshall.

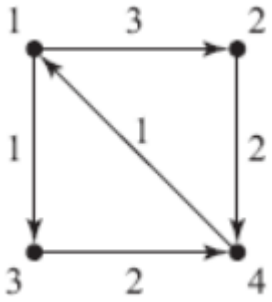
a)



b)



c)



2) Existem muitas técnicas utilizadas na construção de soluções algorítmicas para problemas computacionais. Podemos citar: algoritmos de divisão-e-conquista, algoritmos gulosos e

programação dinâmica, entre outros. Pesquise a definição de “Programação Dinâmica” e cite outros algoritmos que utilizam a mesma técnica.

3) Para este exercício, será utilizado o código-fonte (projeto Eclipse) disponibilizado no Moodle (ExerciciosGrafos7.zip).

a) Implemente o algoritmo de Floyd-Warshall, complementando a classe fornecida em *FloydWarshall.java*. As seguintes alterações de código devem ser implementadas:

- *public FloydWarshall(AdjMatrixEdgeWeightedDigraph G)*: construtor, que executa o algoritmo em si.
- *public ArrayList<Integer> caminho(int s, int t)*: se houver, retorna uma lista com o caminho (nodos) entre s e t, incluindo os mesmos.
- O atributo *temCicloNegativo* deve ser setado como true, se o algoritmo encontrar um ciclo negativo.

Ao final, teste a aplicação com o método *main* fornecido na classe *FloydWarshall*: o resultado deve ser igual a Dijkstra para o vértice 0.

b) Experimente usar a aplicação *AppShortestPathComparison*, que executa ambos os algoritmos e obtém o tempo de cada um. A aplicação por padrão carrega um arquivo com a descrição do grafo, mas é possível também gerar um grafo internamente, através da classe *DigraphGenerator* (veja no código). Você consegue descobrir como é o comportamento dos dois algoritmos em cada caso? Um deles é melhor SEMPRE? Se não é, qual é o motivo?