GTU Department of Computer Engineering
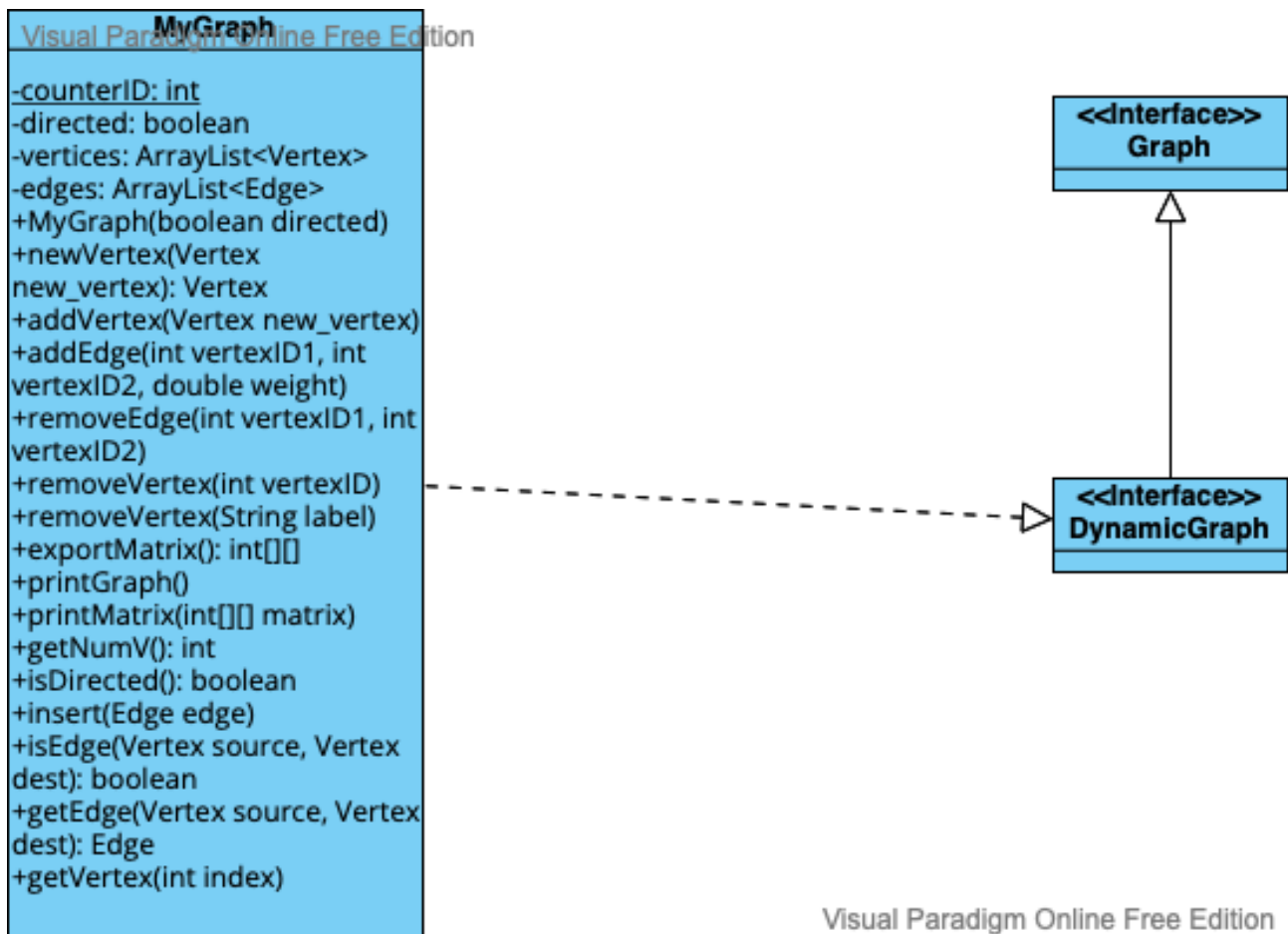
CSE 222/505 - Spring 2022

Homework 8 Report

Sena Özbelen
1901042601

1. System Requirements

- The graph should be created firstly.
- The methods should be called with right parameters.

2. Class Diagram

MyGraph

-counterID: int
-directed: boolean
-vertices: ArrayList<Vertex>
-edges: ArrayList<Edge>
+MyGraph(boolean directed)
+newVertex(Vertex new_vertex): Vertex
+addVertex(Vertex new_vertex)
+addEdge(int vertexID1, int vertexID2, double weight)
+removeEdge(int vertexID1, int vertexID2)
+removeVertex(int vertexID)
+removeVertex(String label)
+exportMatrix(): int[][]
+printGraph()
+printMatrix(int[][] matrix)
+getNumV(): int
+isDirected(): boolean
+insert(Edge edge)
+isEdge(Vertex source, Vertex dest): boolean
+getEdge(Vertex source, Vertex dest): Edge
+getVertex(int index)

<<Interface>>
Graph

<<Interface>>
DynamicGraph

3. Problem Solution Approach

For the first question, I implemented MyGraph class implementing the interface DynamicGraph. There are two array lists to store the vertices and edges. The program can find the selected vertex searching in the array list. Every vertex has a list of adjacent vertices. While adding a new vertex, the program also adds it to the adjacent vertices' lists considering if it is directed or not. While removing a vertex, the program finds the vertex and removes it. After that, it finds the connected edges and removes them as well.
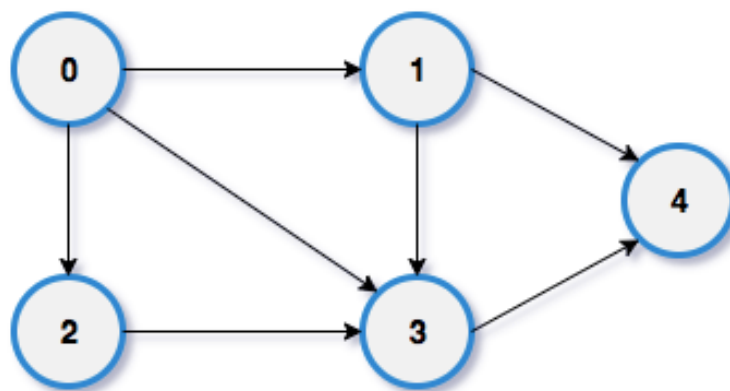
For the second question, the program uses the algorithm of BFS and DFS which are explained in the book.

For the third question, the program uses the algorithm of Dijkstra in the book as a base.

4. Test Cases

For the first question,
-Create this graph as directed using addVertex, newVertex and addEdge methods.

For the second question,
- Perform the BFS and DFS respectively
- Display the results

For the third question,
- Perform the modified Dijkstra's Algorithm

5. Running Command and Results

```
The graph:
Vertex: 0 0 1.0 -> Vertex: 1 1 1.0 -> Vertex: 2 2 1.0 -> Vertex: 3 3 1.0
Vertex: 1 1 1.0 -> Vertex: 3 3 1.0 -> Vertex: 4 4 1.0
Vertex: 2 2 1.0 -> Vertex: 3 3 1.0
Vertex: 3 3 1.0 -> Vertex: 4 4 1.0
Vertex: 4 4 1.0

Breadth First Search Order:
01234

Depth First Search Order:
43120

The BFS: 60
The DFS: 100

ID Distance Pred
1 8.0 0
2 12.0 0
3 14.0 1
4 18.0 1
```