UNIVERSITY OF WESTMINSTER

INFORMATICS INSTITUTE OF TECHNOLOGY

Algorithms: Theory, Design and Implementation
5SENG003C.2

Implementation Report

Name : K.L.T Sathsara

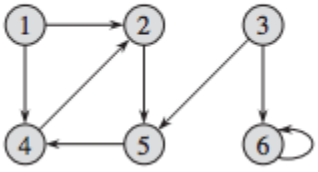
IIT NO : 20210083

UOW : W1866999

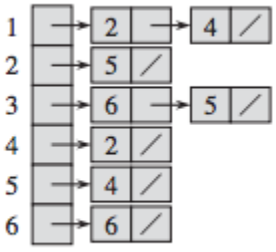
Contents

A. Data Structures

1. AdjacencyMatrix,



(a)



(b)

	1	2	3	4	5	6
1	0	1	0	1	0	0
2	0	0	0	0	1	0
3	0	0	0	0	1	1
4	0	1	0	0	0	0
5	0	0	0	1	0	0
6	0	0	0	0	0	1

(c)

This is the code for AdjacencyMatrix, implementation.

```
private int[][] adjMatrix;

public DirectedGraph(int[][] adjMatrix) {
   this.adjMatrix = adjMatrix;
}
```

The main data structure used is a 2D array called adjMatrix, which stores information about the edges in a directed graph. Each element of the array represents an edge from vertex i to vertex j, where a value of 1 indicates that there is an edge from i to j, and a value of 0 indicates that there is no such edge.

The adjMatrix is utilized in the findSink(), removeVertex(), and printEdges() methods, which all perform operations on the graph. For example, findSink() searches the adjMatrix to find a sink vertex, while removeVertex() modifies the adjMatrix by removing a specified vertex from the graph. The printEdges() method also uses adjMatrix to display the edges of the graph.

B. Algorithms

1. FindSink():

The findSink() method searches for a sink vertex in the graph. It does this by iterating over each vertex in the graph, considering each vertex as a potential sink. For each potential sink vertex, it checks if it has an outgoing edge to all other vertices in the

graph and no incoming edge from any other vertex. If it satisfies these conditions, it is considered a sink vertex and returned. If no sink vertex is found, the method returns -1.

2. RemoveVertex():

The removeVertex() method removes a given vertex from the graph by deleting all its incoming and outgoing edges to other vertices in the graph. To do this, the method sets the corresponding entries in the adjacency matrix to 0.

3. Parsing a graph input file:

The parseGraphInputFile() method reads a text file containing edges of a directed graph. It first scans the file to determine the number of vertices in the graph by finding the highest vertex number among all edges. It then initializes an adjacency matrix to represent the graph, where each entry in the matrix indicates whether there is a directed edge between two vertices. The method then reads the file again to populate the adjacency matrix with the edges from the file. Finally, the method returns a DirectedGraph object representing the parsed graph.hg

4. Printing the edges of a graph:

The printEdges() method prints out the edges of the graph by iterating over each entry in the adjacency matrix. If an entry has a value of 1, it indicates the presence of a directed edge between the corresponding vertices, so the method prints out the edge in the format "source_vertex -> target_vertex".