Kathmandu University

Department of Computer Science and Engineering

Dhulikhel, Kavrepalanchowk

LAB REPORT 04 (Graph Implementation)

COMP202

Submitted By:

Rohan Dhakal (14)

CE II Year/I semester

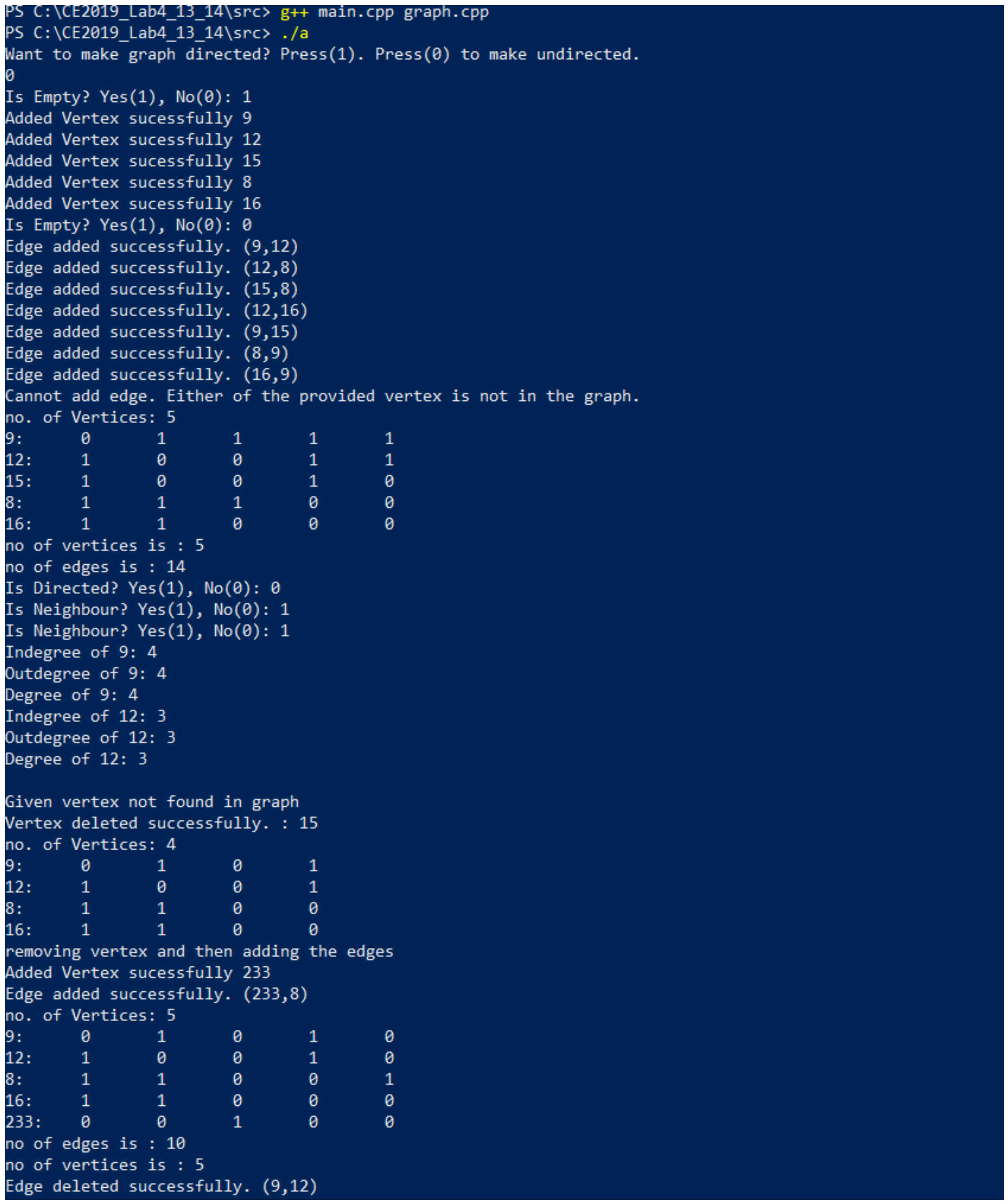
Submitted to:

Ms. Rajani Chulyadyo

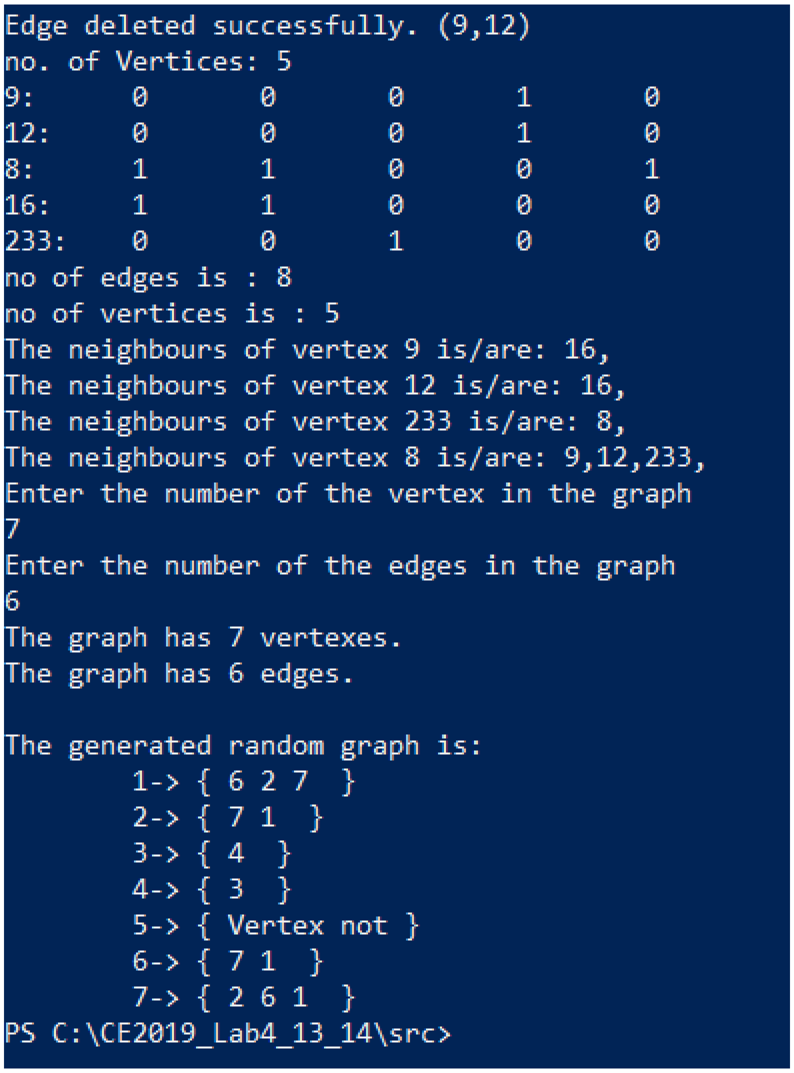
Department of Computer Science and Engineering

OUTPUT: graph data structure implementation using adjacency matrix

I. Output for Undirected graph

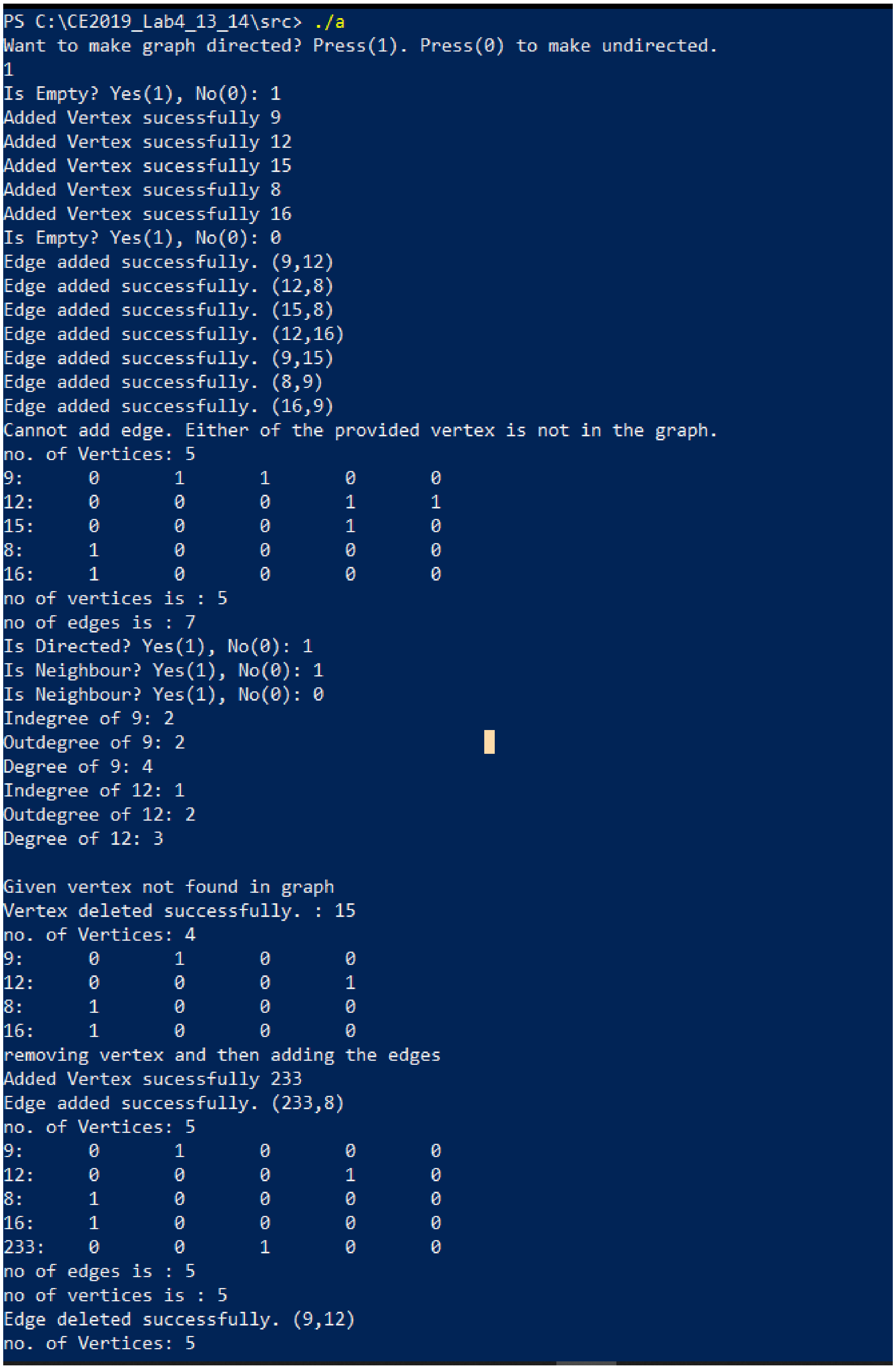


```
PS C:\CE2019_Lab4_13_14\src> g++ main.cpp graph.cpp
PS C:\CE2019_Lab4_13_14\src> ./a
Want to make graph directed? Press(1). Press(0) to make undirected.
0
Is Empty? Yes(1), No(0): 1
Added Vertex sucessfully 9
Added Vertex sucessfully 12
Added Vertex sucessfully 15
Added Vertex sucessfully 8
Added Vertex sucessfully 16
Is Empty? Yes(1), No(0): 0
Edge added successfully. (9,12)
Edge added successfully. (12,8)
Edge added successfully. (15,8)
Edge added successfully. (12,16)
Edge added successfully. (9,15)
Edge added successfully. (8,9)
Edge added successfully. (16,9)
Cannot add edge. Either of the provided vertex is not in the graph.
no. of Vertices: 5
9:      0       1       1       1       1
12:     1       0       0       1       1
15:     1       0       0       1       0
8:      1       1       1       0       0
16:     1       1       0       0       0
no of vertices is : 5
no of edges is : 14
Is Directed? Yes(1), No(0): 0
Is Neighbour? Yes(1), No(0): 1
Is Neighbour? Yes(1), No(0): 1
Indegree of 9: 4
Outdegree of 9: 4
Degree of 9: 4
Indegree of 12: 3
Outdegree of 12: 3
Degree of 12: 3

Given vertex not found in graph
Vertex deleted successfully. : 15
no. of Vertices: 4
9:      0       1       0       1
12:     1       0       0       1
8:      1       1       0       0
16:     1       1       0       0
removing vertex and then adding the edges
Added Vertex sucessfully 233
Edge added successfully. (233,8)
no. of Vertices: 5
9:      0       1       0       1       0
12:     1       0       0       1       0
8:      1       1       0       0       1
16:     1       1       0       0       0
233:    0       0       1       0       0
no of edges is : 10
no of vertices is : 5
Edge deleted successfully. (9,12)
```

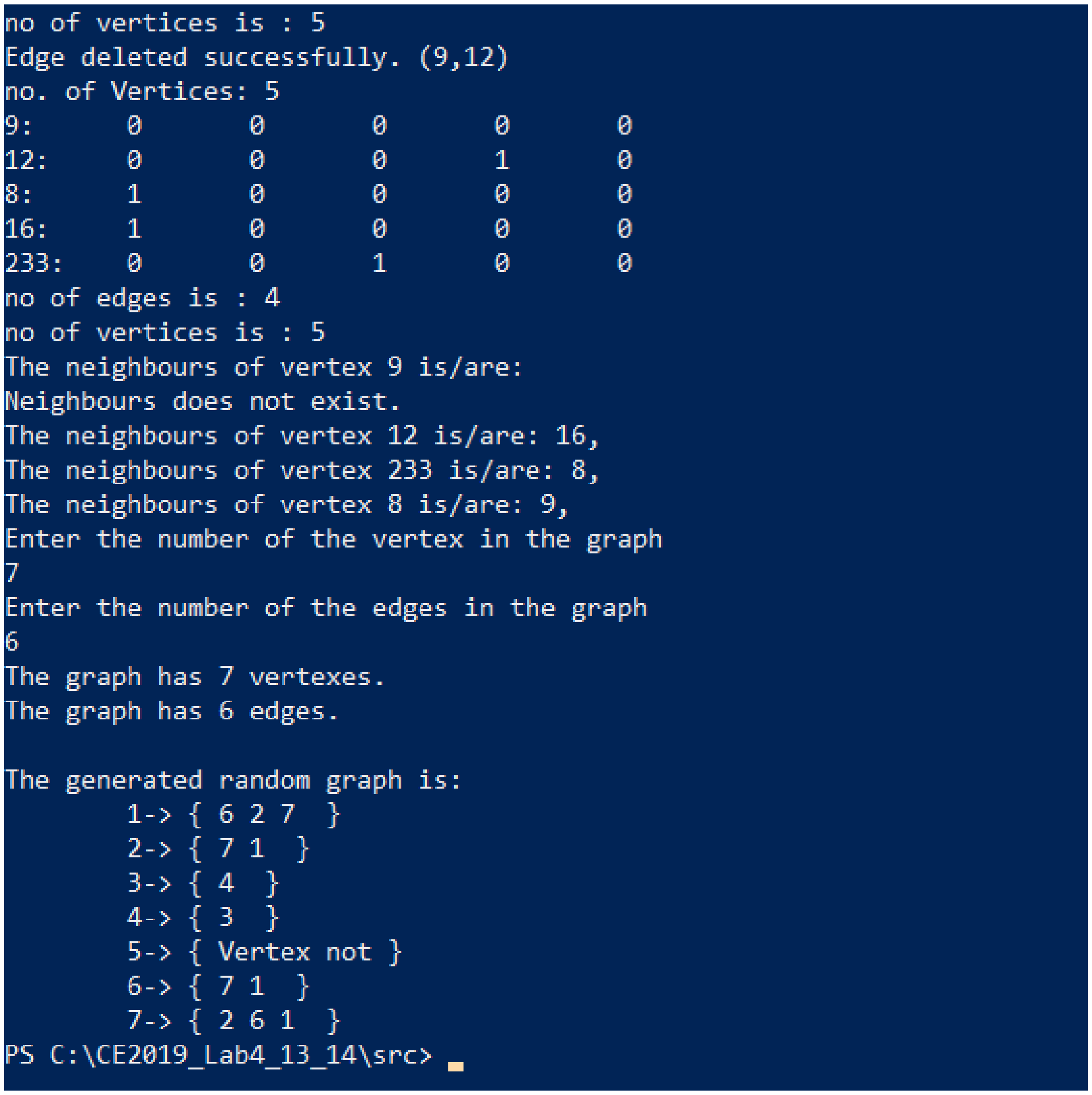


```
Edge deleted successfully. (9,12)
no. of Vertices: 5
9:      0       0       0       1       0
12:     0       0       0       1       0
8:      1       1       0       0       1
16:     1       1       0       0       0
233:    0       0       1       0       0
no of edges is : 8
no of vertices is : 5
The neighbours of vertex 9 is/are: 16,
The neighbours of vertex 12 is/are: 16,
The neighbours of vertex 233 is/are: 8,
The neighbours of vertex 8 is/are: 9,12,233,
Enter the number of the vertex in the graph
7
Enter the number of the edges in the graph
6
The graph has 7 vertexes.
The graph has 6 edges.

The generated random graph is:
        1-> { 6 2 7  }
        2-> { 7 1  }
        3-> { 4  }
        4-> { 3  }
        5-> { Vertex not }
        6-> { 7 1  }
        7-> { 2 6 1  }
PS C:\CE2019_Lab4_13_14\src>
```

II. Output for Directed graph



```
PS C:\CE2019_Lab4_13_14\src> ./a
Want to make graph directed? Press(1). Press(0) to make undirected.
1
Is Empty? Yes(1), No(0): 1
Added Vertex sucessfully 9
Added Vertex sucessfully 12
Added Vertex sucessfully 15
Added Vertex sucessfully 8
Added Vertex sucessfully 16
Is Empty? Yes(1), No(0): 0
Edge added successfully. (9,12)
Edge added successfully. (12,8)
Edge added successfully. (15,8)
Edge added successfully. (12,16)
Edge added successfully. (9,15)
Edge added successfully. (8,9)
Edge added successfully. (16,9)
Cannot add edge. Either of the provided vertex is not in the graph.
no. of Vertices: 5
9:      0       1       1       0       0
12:     0       0       0       1       1
15:     0       0       0       1       0
8:      1       0       0       0       0
16:     1       0       0       0       0
no of vertices is : 5
no of edges is : 7
Is Directed? Yes(1), No(0): 1
Is Neighbour? Yes(1), No(0): 1
Is Neighbour? Yes(1), No(0): 0
Indegree of 9: 2
Outdegree of 9: 2
Degree of 9: 4
Indegree of 12: 1
Outdegree of 12: 2
Degree of 12: 3

Given vertex not found in graph
Vertex deleted successfully. : 15
no. of Vertices: 4
9:      0       1       0       0
12:     0       0       0       1
8:      1       0       0       0
16:     1       0       0       0
removing vertex and then adding the edges
Added Vertex sucessfully 233
Edge added successfully. (233,8)
no. of Vertices: 5
9:      0       1       0       0       0
12:     0       0       0       1       0
8:      1       0       0       0       0
16:     1       0       0       0       0
233:    0       0       1       0       0
no of edges is : 5
no of vertices is : 5
Edge deleted successfully. (9,12)
no. of Vertices: 5
```



```
no of vertices is : 5
Edge deleted successfully. (9,12)
no. of Vertices: 5
9:      0       0       0       0       0
12:     0       0       0       1       0
8:      1       0       0       0       0
16:     1       0       0       0       0
233:    0       0       1       0       0
no of edges is : 4
no of vertices is : 5
The neighbours of vertex 9 is/are:
Neighbours does not exist.
The neighbours of vertex 12 is/are: 16,
The neighbours of vertex 233 is/are: 8,
The neighbours of vertex 8 is/are: 9,
Enter the number of the vertex in the graph
7
Enter the number of the edges in the graph
6
The graph has 7 vertexes.
The graph has 6 edges.

The generated random graph is:
        1-> { 6 2 7  }
        2-> { 7 1  }
        3-> { 4  }
        4-> { 3  }
        5-> { Vertex not }
        6-> { 7 1  }
        7-> { 2 6 1  }
PS C:\CE2019_Lab4_13_14\src> _
```

My Implementatios:

1. isDirected();
2. addEdge(int, int);
3. indegree(int);
4. outDegree(int);
5. inDegree(int);
6. degree(int);
7. neighbour (int, int);
8. removeVertex(int);
9. helperFunction(i.e. checkVertex)

Why adjacency Matrix?

The basic operations like adding an edge, removing an edge and checking whether there is an edge from vertex i to vertex j are extremely time efficient, constant time operations. Also, it is simpler to implement.

Github Link:

anishadhakal/CE2019_Lab4_13_14: LAB-4 (github.com)