Name : Nayanathara P.M.C.

Index No : 210417X

Lab 10 – Graphs ADT

➢ Section 1 – Implementing Graph ADT

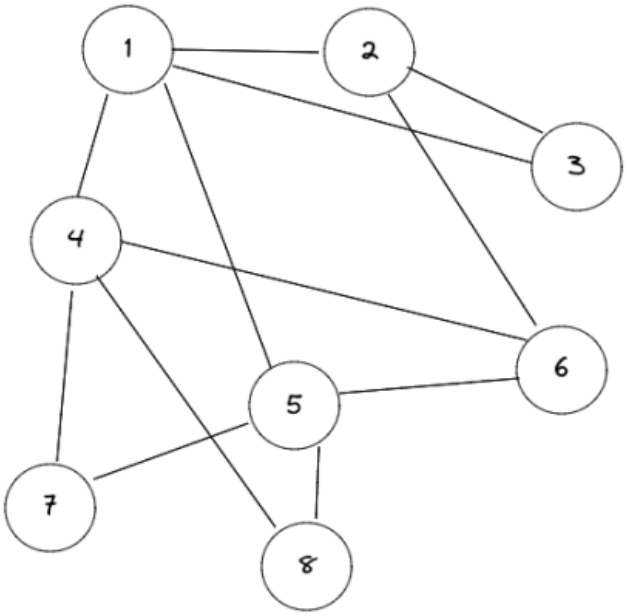
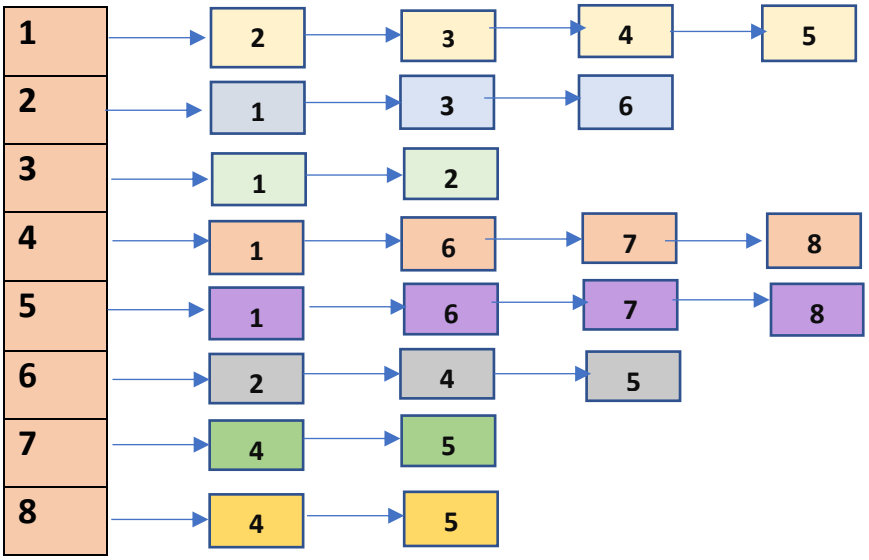
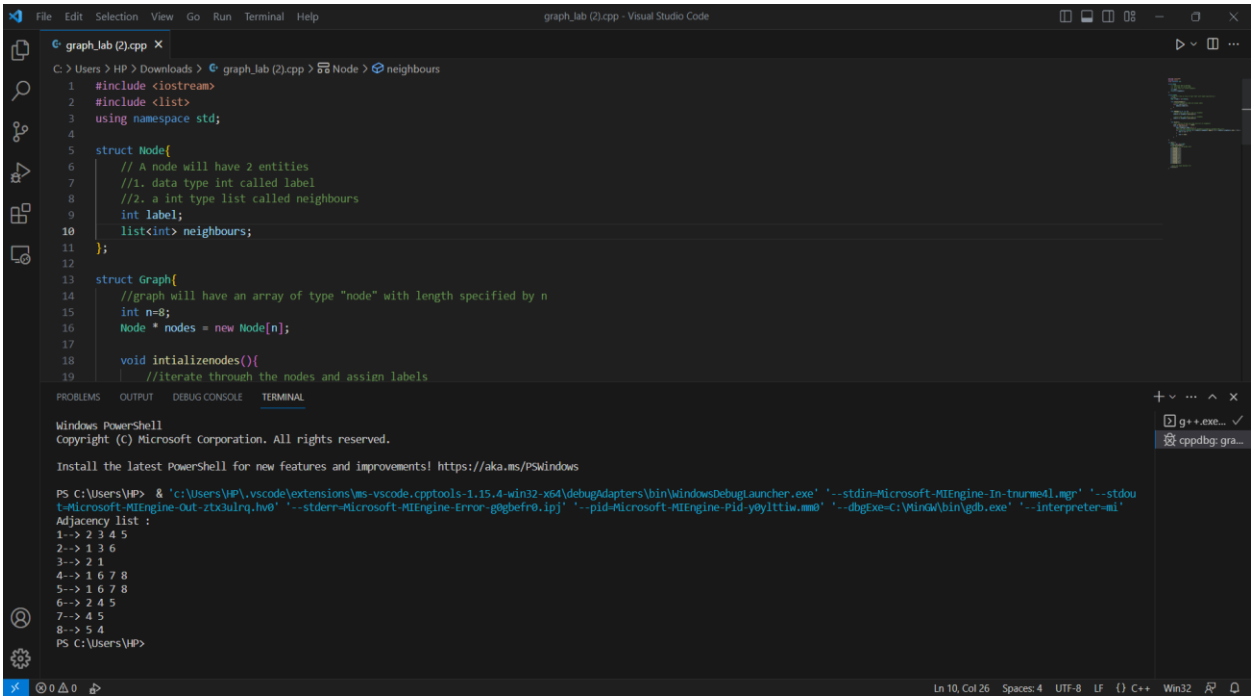


Figure 1: Graph for Section 1

Adjacency list



Terminal Output



Changing the implementation to accept directed graphs

```
void addedge(int u, int v){
        //select node u and push v into u's neighbour
        nodes[u].neighbours.push_back(v);
}
```

Here is the ‘addedge’ altered function to add only directed graphs to the Graph structure. This can be obtained by removing the code relevant to adding the reverse edge between the two nodes.

After modifying the code in this manner, only the edge directing from node ‘u’ to ‘v’ is added to the graph. The edge directing from node ‘v’ to ‘u’ will not be added. This results in implementation of a directed graph data structure.

➢ Section 2 – Working out link prediction

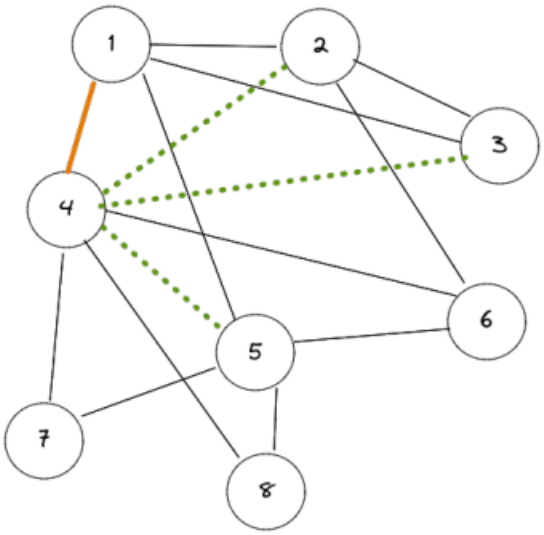


Figure 2: Graph for Section 2

Node 1 have three friends(neighbors) denoted by the nodes 2, 3, and 5 other than 4 who just became the friend.

Let's calculate the similarity score of each of these nodes with node 4.

Sim(2,4) = (No. of shared neighbors between 2,4) / (Total neighbors in 2,4)

= 2/5 = 0.4

Sim(3,4) = (No. of shared neighbors between 3,4) / (Total neighbors in 3,4)

= 1/5 = 0.2

Sim(5,4) = (No. of shared neighbors between 5,4) / (Total neighbors in 5,4)

= 4/4 =1.0

According to the above similarity scores, we can see that all the friends of 5 are also friends of 4. So, there is a very high probability for 4 and 5 to become friends. Therefore, 5 will be a best friend suggestion for 4.

➢ Git-Hub Repository link

https://github.com/ChethmiNayanathara/Data-Structures-and-Algorithms