# Assignment on Graphs:

You must implement a graph abstract data structure (ADT) using an adjacency list and an adjacency matrix. For simplicity, we will consider only undirected, unweighted graphs. The graph ADT should have the following functionalities:

1. AddNode(int v): Add a new node v to the graph
2. AddEdge(int u, int v): Add a new edge (u, v) in the graph
3. CheckEdge(int u, int v): Check whether there is an edge (u, v) between node u and node v
4. RemoveNode(int v): Remove node v from the graph
5. RemoveEdge(int u, int v): Remove an edge (u, v) from the graph
6. CheckPath(int u, int v): Check whether a path exists between node u and node v
7. GetNeighbors(int u): Returns all the neighbors of u
8. FindShortestPath(int u, int v): Find the shortest path between node u and node v
9. FindShortestPathLength(int u, int v): Find the length of the shortest path between node u and node v

# You Tasks:

You are given four files as follows:

a. ***GraphADT.h*** : You don't need to do anything in this file
b. ***AdjacencyListGraph.h***: You need to implement the functions in the file whenever you will find a TODO.
c. ***AdjacencyMatrixGraph.h***: You need to implement the functions in the file whenever you will find a TODO
d. ***GraphTest.cpp*** : This is the main file to test your implementation. You may add more tests in this file. However, your implementation should allow the given test to run for both AdjacencyListGraph.h and AdjacencyMatrixGraph.h

The GraphTest.cpp file will take a graph as input from a file, as it is given in our *input.txt*. **If you need any data structure like a list, stack, or queue, you must need to use your own implementation.** You can't use list, stack, or queue from the standard library.

# Sample I/O:

See the given input.txt and output.txt files.

## Marks Distribution:

For an adjacency list-based implementation, you have 50 marks. Same mark distribution for the adjacency matrix-based implementation. The following is given for a single case.

| Functionality | Percentage of Marks |
|---|---|
| 1 | 5 |
| 2 | 5 |
| 3 | 5 |
| 4 | 5 |
| 5 | 5 |
| 6 | 5 |
| 7 | 5 |
| 8 | 5 |
| 9 | 10 |
| Total (Adjacency List) | 50 |

Total marks considering both cases: 50x2 =100