CL-210
Data Structures

Objectives:

- Graph
- Adjacency Matrix
- Adjacency List

Note: Carefully read the following instructions (*Each instruction contains a weightage*)

1. There must be a block of comments at start of every question's code by students; the block should contain brief description about functionality of code.
2. Comment on every function and about its functionality.
3. Mention comments where necessary such as comments with variables, loop, classes etc to increase code understandability.
4. Use understandable name of variables. 5. Proper indentation of code is essential.
6. Write a code in C++ language.
7. Make a Microsoft Word file and paste all of your C++ code with all possible screenshots of every task **outputs in Microsoft Word and submit word file. Do not submit .cpp file.**
8. First think about statement problems and then write/draw your logic on copy.
9. After copy pencil work, code the problem statement on MS Studio C++ compiler.
10. At the end when you done your tasks, attached C++ created files in MS word file and make your submission on Google Classroom. (Make sure your submission is completed).
11. Please submit your file in this format **19F1234_L11**.
12. Do not submit your assignment after deadline. Late and email submission is not accepted.
13. Do not copy code from any source otherwise you will be penalized with negative marks.

Problem 1 | Implement Adjacency Matrix

Given a undirected Graph of N vertices 1 to N and M edges in form of 2D array. Array [][] whose every row consists of two numbers X and Y which denotes that there is an edge between X and Y, the task is to write C++ program to create Adjacency Matrix of the given Graph.

Input: N = 8, M = 7, array [] [] = {{1, 2}, {2, 3}, {4, 5}, {1, 5}, {6, 1}, {7, 4}, {3, 8}}

output will be

0	1	0	0	1	1	0
1	0	1	0	0	0	0
0	1	0	0	0	0	0
0	0	0	0	1	0	1
1	0	0	1	0	0	0
1	0	0	0	0	0	0
0	0	0	1	0	0	0
0	0	1	0	0	0	0

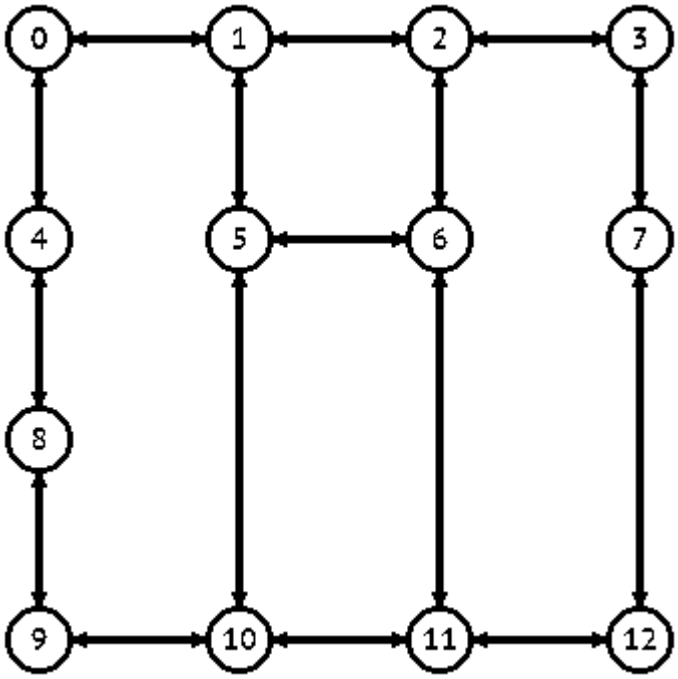
Problem 2 | Implement Adjacency Matrix

The following list of edges are given to you by using this edge list show the adjacency matrix and count that how many edges each node has.

[0,1], [0,6], [0,8], [1,4], [1,6], [1,9], [2,4], [2,6], [3,4], [3,5], [3,8], [4,5], [4,9], [7,8], [7,9]

Problem 3 | BFS and DFS

Consider the below graph and write a C ++ code to traverse whole graph by considering “0” as start node. Use BFS and DFS strategy.



Best of luck