Indian Institute of Technology Mandi
February-June 2017 Semester

CS202: Data Structure and Algorithms

Programming Assignment 6 Problem Statements

Last date of submission of code: 24th April, 2017

Implement one of the following problems using C++ programing language.
Note:

1. Implement undirected graph and directed graph using the interfaces UndirectedGraph.hpp and DirectedGraph.hpp respectively. Both these classes are implemented as a subclass of AbstractGraph.hpp, which is an interface to represent any type of graph.
2. Implement undirected and directed graphs using both adjacency matrix and adjacency list representations. Use AdjacencyMatrix.hpp to implement adjacency matrix representation and AdjacencyList.hpp to implement adjacency list representation. Both these classes are implemented as a subclass of base class GraphAdjacencyBase.hpp, which can denote any Graph Adjacency representation.
3. Implement **BFS** and **DFS**. Both BFS and DFS are the functions in undirected and directed graphs. Use adjacent list representation for implementing these graph search algorithms.
4. BFS must be implemented using **queue** data structure and DFS must be implemented using **Stack** data structure. Use the same queue and stack interfaces used in assignment-3 (stack.hpp and queue.hpp)**.**
5. Write a separate main programs to evaluate the functions (including DFS and BFS) in undirected and directed graphs represented using both the representations. DFS and BFS should be there only when the representation is adjacency matrix.
6. Each main program can take the input either from keyboard or from a file. A sample file format for an input is given in the moodle. The graph data in the input file is in the form of an adjacency matrix. Your program must be general enough to take input from a file for any graph by providing the file-name along with its path.
7. These main programs must display the resulting graphs or subgraphs in an effective manner one edge at a time. (Use your programming skill for this. Also this display is not the part of any graph class).
8. Write a separate main programs to simulate the problem given below which is an application of BFS and DFS.

Problem: *Flood-fill algorithm*

Flood fill algorithm helps in visiting each and every point in a given area. It determines the area connected to a given cell in a multi-dimensional array. Flood fill algorithm can be simply modeled as graph traversal problem, representing the given area as a matrix of *n* x *m* and considering every cell of that matrix as a vertex that is connected to points above it, below it, to right of it, and to left of it and in case of 8-connections, to the points at both diagonals also. Now that the given area has been modeled as a graph, a DFS or BFS can be applied to traverse that graph. *Given a matrix, a source cell, a destination cell, some cells which cannot be visited, and some valid moves*, check if the destination cell can be reached from the source cell. Choose the direction to move randomly at each step. Use a built in random number generator to generate a number between 1 and 8 to choose one of the 8 directions. Print the path traversed from source to destination. (Use your programming skills to print the output effectively).

References:

1.https://www.hackerearth.com/practice/algorithms/graphs/flood-fill-algorithm/tutorial/
2.https://en.wikipedia.org/wiki/Flood_fill
3.https://youtu.be/QlcwfGo8W_E