Homework 7

Out: 11.12.20
Due: 11.23.20

1. [Graph algorithms, 10 Points]
   What is the runtime of BFS for a graph represented as an adjacency matrix?

2. [Graph algorithms, 10 Points]
   Design an efficient algorithm that computes a path in a given connected, undirected graph G that traverses every edge in G exactly once in each direction. Analyze its runtime.

3. [Graph algorithms, 10 Points]
   A directed graph is "almost connected" if for any two vertices there is a path in at least one direction between the them. Provide an efficient algorithm to determine if a graph is "almost connected", and analyze its runtime.

4. [MST, 10 points]
   Provide an implementation of Prim's algorithm for a graph G=(V,E), represented as an adjacency matrix. What is the runtime of your algorithm?

5. [MST, 10 points]
   Give an algorithm to find a maximum spanning tree in a connected undirected graph.

6. [Maze, 50 points]
   Consider a 2D array of 1s and 0s of size NxN. The array is a "maze." You must start from the top left (index (0,0)) and find the length of the shortest path to the bottom right (index (n,n)). You may only travel across elements that have value "1". The top left and bottom right indices are guaranteed to be "1". You may only move horizontally or vertical one index with each step.

   For example, for the following input maze, a shortest path is along bolded numbers and has a length of 9:

   11001
   0**1**011
   0**11**00
   1**111**0
   110**11**

   Your program will open an input file, called *maze.txt*, containing a maze. Your program, when run with no command-line arguments in the same directory as *maze.txt*, should read the provided maze from the file, determine N, the dimension of the maze, and output the length of the shortest path to the screen. If no path exists, your program should output '0'. You may use the provided *maze.txt* file to test your code.

Submit your code in a single file named *Problem6.cpp*.