Alex Yang

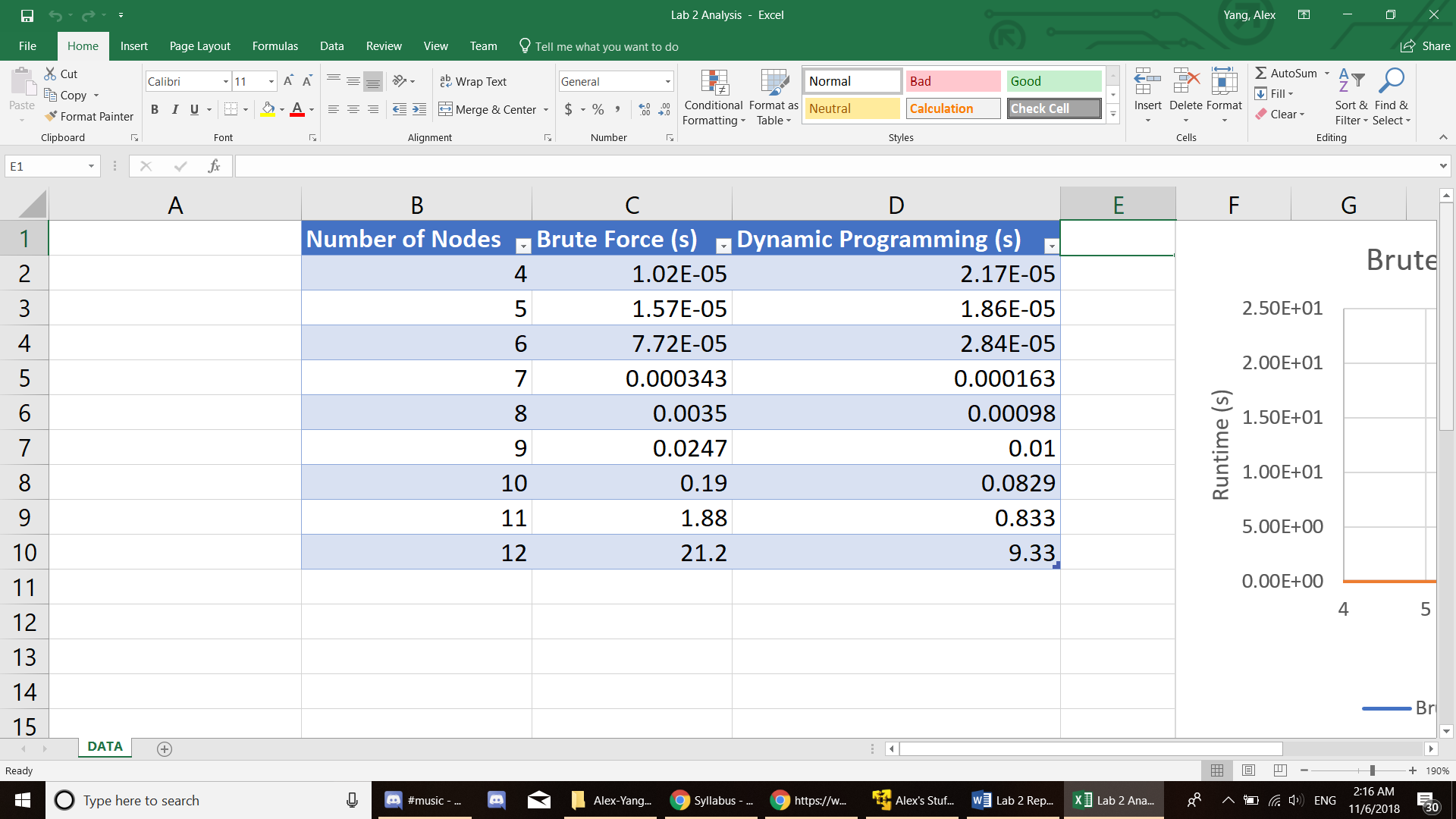
Professor Clark

CSE 3353

6 November 2018

Lab 3 Report

Raw Data:



Graphs:

Analysis

My Brute Force Algorithm had time complexity of O(n!) because every node in my adjacency list had to check every other node in the graph for distances (since brute force tries every possible permutation) and my Dynamic Programming Algorithm had a time complexity of O(2^n).

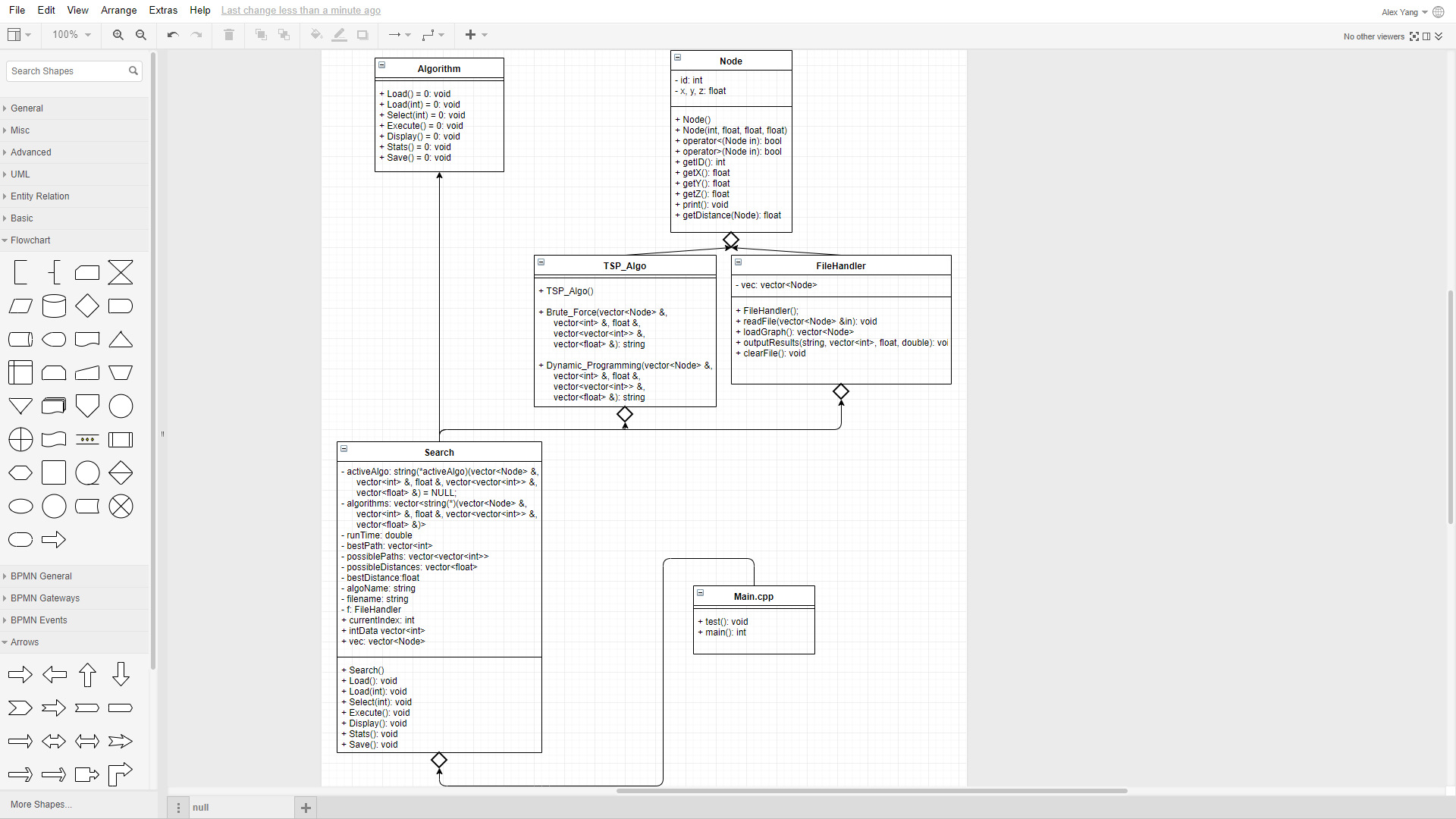
Design Decisions

I used the strategy design pattern as my design pattern. Strategy seemed the most appealing to me because I can just write new algorithms in my SearchAlgo class and be able to easily add them to my Search class’s vector of algorithms. Because I’m using the strategy design pattern, I can also run every algorithm the same way as seen in my test function in my main.cpp. I can test multiple different sized graphs and see the results. All I need to do is call the load, select, and execute functions no matter which algorithm and it will run them. If I need to add a new algorithm, I can just write a new one in my TSP\_Algo class as a function.

To describe the structure of my program, my main class includes from Search which inherits from the virtual base class Algorithm. Search runs all the Strategy pattern commands (load, select, execute, etc.). Search includes TSP\_Algo which contains the brute force and DP algorithms in the form of static functions. Search has a vector of function pointers (algorithms) which it adds to from TSP\_Algo in my Search classes constructor. My graph is represented as a vector of nodes called vec contained in my Search class. Each node contains the X, Y, and Z coordinates of each point in the graph.

I also implemented a fileHandler object that incorporates both the file loader and output system into a single interface. The Load() and Save() functions in my Search class now call the fileHandler object and it takes care of reading and writing to files. In addition, if I were to re-use this program for a different type of algorithm with different text files, I could just change how the object reads in the input without touching any other parts of my code (so I know problem will be isolated to that class alone).

UML Diagram



Dynamic Programming

The main subproblem I needed to accomplish in my dynamic programming algorithm was creating a 2-D Matrix filled with floats that represent the distance between 2 different points. I populated with 2 for loops which increment by 1. For instance matrix[i][j] would contain the float distance between node i and node j. This quick searching through the 2D Matrix drastically increased my algorithms efficiency.