Programming Assignment Six

Nathaniel Sigafoos

4/29/17

The purpose of this program is to find the minimum spanning tree from an adjacency matrix. A minimum spanning tree is a tree that connects all the nodes in such a way that the total weight of the tree is minimized. The total weight of the tree is the sum of the weights of the connections between the nodes. In other words, we want to find the "lightest" connection to another node that each node has, such that all the nodes are connected.

The algorithm works by finding an unchecked node that is "closest" (i.e. the edge between the two nodes has the lowest weight) to a node that has already been checked, until it has check all the nodes. In order to do this efficiently, the program uses a LIFO (Last In, First Out) stack to optimize the order in which the node are checked. The nodes are put into the stack such that the closer the node is to the current node, the higher on the stack it will be, and thus the sooner it will be checked.

The problem of finding the minimum spanning tree is very similar to another problem know as the Traveling Salesperson Problem. In the TSP, you are given a collection of interconnected "cities" or nodes, with each edge between the cities having a cost/weight assigned to it. The problem then is to find a circuit through the cities that has the lowest cost. This is very similar to finding the minimum spanning tree, because in both cases you want to find the edges between cities that have the lowest cost. However, in the TSP, the edges must form a path that goes though each city/node exactly once, and ends back at the starting city/node. In the MSP the edges do not have to form a circuit, there only has to be a path from any given node to another. Because of this, the weight of the MSP will almost always be lower than the minimal cost of the TSP.