DFS and Topological Sort

Homework #11

By

John Kiyak

CS303

November 6 2018

**Problem Specification**

The goal of this assignment was to implement the DFS algorithm to work on a directed graph and doing a topological sort of the vertices given from the “tinyDG.txt” text file.

**Program Design**

For my program in python, I used the sample code from geeksforgeeks and modified it a little bit. The graph class was initialized with a dictionary and vertices. The addEdge function added vertex (v) to the graph array [u]. For the topologicalSort, it has a boolean variable of visited, and an array for stack. It gets all of the vertexes in the array and marks them as not ivisted, and then goes through each one and visits them.

**Testing Plan**

First, I would maek sure that the input files were being read in, and correctly being output into the arrays. Then, I would make sure that the graph was being created with the correct number of vertices and values from the max value from the input file. Then, I would check that the BFS algorithm was displaying correctly, and had the accurate values for a sample size that I deemed fit.

**Analysis and Conclusions**

When testing a topological sort of miniscule numbers, it is hard to put a time complexity, since the test that I did had a time of 0.0 seconds. However, according to GeeksforGeeks, it has the same time complexity as DFS, since the topological sort is basically the same, except for adding one additional stack. Since the time complexity of DFS is O(V+E), Topological Sort is the same.

When the code is run, the code returns:

**Following is a Topological Sort of the given graph**

**[0, 4, 5, 1, 7, 3, 6, 2]**