CSCE 221 Assignment 6 Cover Page

First Name: Matthew Last Name: Stevens UIN: 924000693

User Name: Mattizrawr E-mail address: Mattizrawr@tamu.edu

Please list all sources in the table below including web pages, which you used to solve or implement the current homework. If you fail to cite sources, you can get a lower number of points or even zero, read more on Aggie Honor System Office website: http://aggiehonor.tamu.edu/

Types of Sources			
People	Joshua Langley	Nicholas Wong	Other lab members
Web Pages	www.piazza.com	http://courses.cs.tamu.edu/teresa/cpsc211/BFS-DFS/DFS/DFS.html	http://www.geeksforgeeks.org/s path-for-directed-acyclic-graphs
Printed Material	N/A		
Other Sources	N/A		

I certify that I have listed all the sources that I used to develop the solutions/codes to the submitted work. On my honor as an Aggie, I have neither given nor received any unauthorized help on this academic work.

Your Name: Matthew Stevens Date: 4/24/2016

– Assignment number.

Assignment: CSCE 221 Assignment 6

– Description of C++ features used, assumptions on your data.

<vector> - for vectors to store the tree's branches

<stack> - for stack functions to transpose and morph the graph

<fstream> - mapping the input file to the main

<iostream> - general c++ functions

<stdio.h> - for c++ commands

Classes: to have data stored in order to access the data globally

Iterators: to traverse the vectors

– At least 3 different testing cases for checking correctness of the code.

Input	Expected output	output

1 2 4 5 -1
2 3 4 7 -1
3 4 -1
4 6 7 -1
5 4 -1
6 5 -1
7 6 -1
	1: 2 4 5 -1
	2: 3 4 7 -1
	3: 4 -1
	4: 6 7 -1
	5: 4 -1
	6: 5 -1
	7: 6 -1
		1: 2 4 5 -1
		2: 3 4 7 -1
		3: 4 -1
		4: 6 7 -1
		5: 4 -1
		6: 5 -1
		7: 6 -1

1 2 4 5 -1
2 3 4 -1
3 4 -1
4 6 -1
5 4 -1
6 5 -1
	1: 2 4 5 -1
	2: 3 4 7 -1
	3: 4 -1
	4: 6 -1
	5: 4 -1
	6: 5 -1
		Dfs
		0: 4222640, 0
		1: 4222640, 0
		2: 4222640, 0
		3: 4222640, 0
		4: 4222640, 0
		5: 4222640, 0
		6: 4222640, 0
		7: 4222640, 0

1 2 4 5 -1
2 3 4 7 -1
3 4 -1
4 7 -1
5 4 -1
7 -1
	1: 2 4 5 -1
	2: 3 4 7 -1
	3: 4 -1
	4: 7 -1
	5: 4 -1
	7: -1
		0: -1
		1: -1
		2: -1
		3: -1
		4: -1
		5: -1
		7:-1

Note: My code ran based on the transposed graph, which I could not get to work correctly. If resubmitted late, it should have the corrected version of the code, meanwhile, this is the data I received while testing my code.

– Running time expressed in terms of big-O notation for each function with justification

Graph::buildGraph – O(n) – it traverses through n elements as the graph is being built (uses one iterator)

Graph::Graph – O(n) – constructs the graph by pushing back the vertices

Graph::displayGraph O(n^2) – has a nested forloop in a for loop which both have n elements.

Graph::getDFS(int start) – O(1) sets timestamps as it traverses the graph. If including the traverse, it will be n vertices so O(n).

Graph::getDFS(Vertex&v, int &time) – O(n) sets colors to each vertice.

– Description of a real life application where your program can be used.

This program can be used for navigational systems. The input would be the start and end destinations, the edges would be the streets, and the vertices would be street intersections. The graph would transpose the data and find the fastest acyclic route that may be taken. This would help people navigate from one point to another.