CSCE 221 Cover Page

Programming Assignment #6

Due December 4th at midnight to eCampus

First Name Anders Last Name Wallace UIN 925000221

User Name awallace2 E-mail address awallace2@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: Aggie Honor System Office

Type of sources			
People			
Web pages (provide URL)			
Printed material			
Other Sources			

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 Anders Wallace Date 12/4/17

Programming Assignment 6

1 Description of Data Structures Implemented

For this programming assignment, we were asked to read in a graph given the vertices that connected it, and then determine whether it could be drawn in one stroke without traversing any edges twice. In order to implement the graph, I used an adjacency matrix by using a vector of vectors. I also used an adjacency list using the std::list in order to do some other operations with the graph. The adjacency matrix was a matrix that had a 1 in the place of a position where there was an edge from i to j in an i x j matrix. The adjacency list was a list where each position showed all the other vertices it was connected to.

2 Necessary and Sufficient Conditions

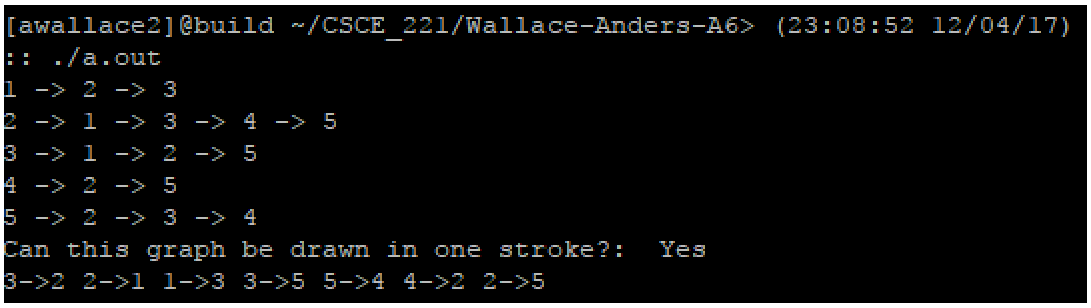
In order to see if a picture could be drawn in one stroke, we first needed to see if the graphs had Euler Paths or Circuits. To check if this was the case, we could look at the degrees of edges for all the vertices. If there were no vertices with odd degree edges, then there was an Euler Circuit. If there were 2 vertices with odd degrees, there was at least one Euler Path. Any other combination and no path existed such that we could draw the graph in one stroke. For this assignment, the graphs that could be drawn were 1, 2, and 5. 3, 4, and 6 could not be drawn without going over edges twice. Graphs 2 and 3 were Euler Circuits, so they could be drawn starting at any point you wanted. Graph 1 was an Euler Path, so you had to start at an odd degree vertex in order to successfully draw this picture (vertices 3 and 5).

3 Description of Algorithm

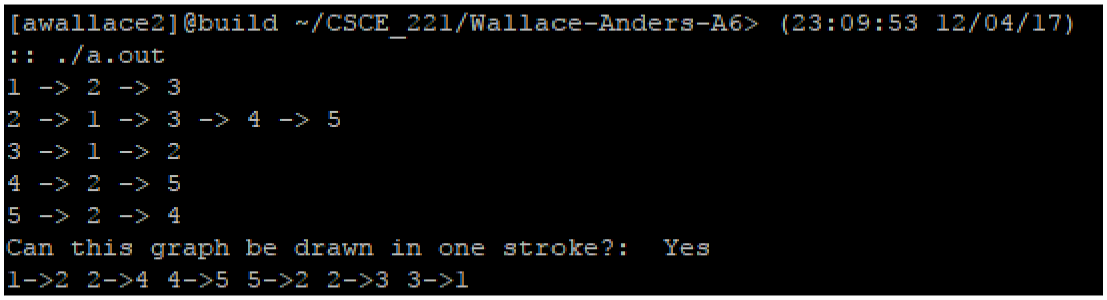
The algorithm used to find these paths was called Fleury's Algorithm. The way it found these paths was if it was an Euler Circuit it could start anywhere (at vertex 1 in our case), otherwise it was an Euler Path and it started at an odd degree vertex. It then traversed the graph, making sure to never cross a bridge (an edge where there was no return) if presented with the option of a bridge and a non-bridge. As it did this, it deleted edges as it went until there were no more edges left. This is where we used an adjacency list to add and delete edges for traversing our graphs, using a DFS variation to go through the graph. In this case, the runtime of the algorithm is $O(V + E)$ where V is the number of vertices and E is the number of edges.

4 Evidence

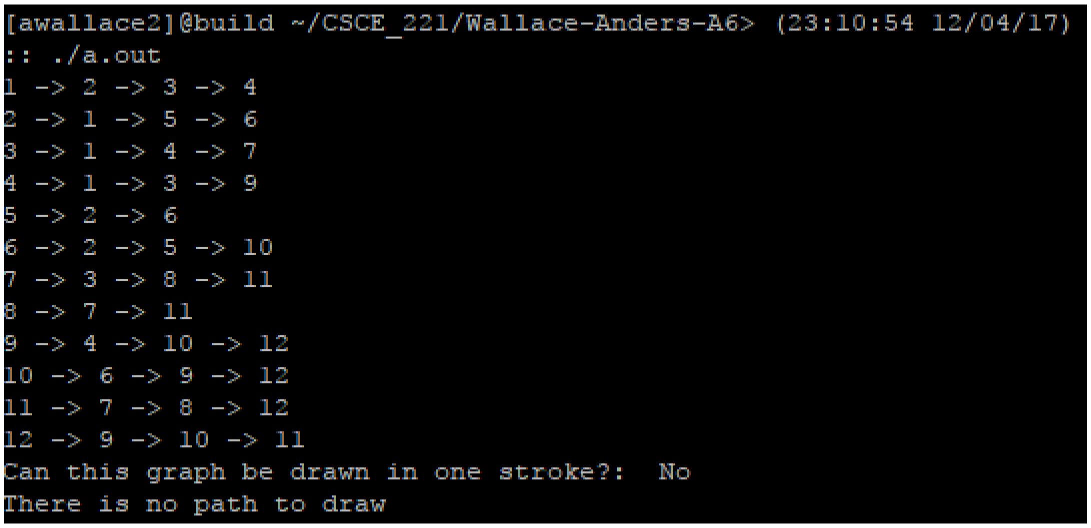
Below you will find evidence of testing. Contained are screenshots of running my program on the first 3 graphs:



```
[awallace2]@build ~/CSCE_221/Wallace-Anders-A6> (23:08:52 12/04/17)
:: ./a.out
1 -> 2 -> 3
2 -> 1 -> 3 -> 4 -> 5
3 -> 1 -> 2 -> 5
4 -> 2 -> 5
5 -> 2 -> 3 -> 4
Can this graph be drawn in one stroke?:  Yes
3->2 2->1 1->3 3->5 5->4 4->2 2->5
```



```
[awallace2]@build ~/CSCE_221/Wallace-Anders-A6> (23:09:53 12/04/17)
:: ./a.out
1 -> 2 -> 3
2 -> 1 -> 3 -> 4 -> 5
3 -> 1 -> 2
4 -> 2 -> 5
5 -> 2 -> 4
Can this graph be drawn in one stroke?:  Yes
1->2 2->4 4->5 5->2 2->3 3->1
```



```
[awallace2]@build ~/CSCE_221/Wallace-Anders-A6> (23:10:54 12/04/17)
:: ./a.out
1 -> 2 -> 3 -> 4
2 -> 1 -> 5 -> 6
3 -> 1 -> 4 -> 7
4 -> 1 -> 3 -> 9
5 -> 2 -> 6
6 -> 2 -> 5 -> 10
7 -> 3 -> 8 -> 11
8 -> 7 -> 11
9 -> 4 -> 10 -> 12
10 -> 6 -> 9 -> 12
11 -> 7 -> 8 -> 12
12 -> 9 -> 10 -> 11
Can this graph be drawn in one stroke?:  No
There is no path to draw
```