CSCE 221 Cover Page
Programming Assignment #6

First Name Elizabeth Last Name Ho UIN 526003059

User Name lizzyholi E-mail address lizzyholi@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/

Type of sources				
People				
Web pages (provide URL)	http://discrete.openmathbooks.org/dmoi2/sec_paths.html			
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	Elizabeth Ho			Date	04/24/2019

1. The description of the data structures implemented in your program
   A vector<vector<int>> was used to hold the adjacency matrix and a second vector<int> to hold the solution.
2. The necessary and sufficient conditions for drawing one-stroke pictures.
   There must be either zero or two vertices that have an odd number of edges. If there are no vertices with an odd number of edges it can start anywhere and if there are two it must start a one of the vertices with off edges. From there it is easier to draw by tracing the edges
3. Description of the algorithm and its running time.
   For canDraw there are two for loops so that the number of edges at each vertex can be counted. It also calculates the starting vertex if it can be drawn.
   For findNext it is a recursive algorithm that has a for loop for each edge that it finds, but it also finds other possible paths that are not solutions. The findNext function finds the first value on a given row that has a value of 1, representing an existing edge. If there is not an existing edge on the row provided the goBack function deletes the last step on the path and calls findNext to find another edge from that vertex.
   Runtime: canDraw $O(n^2)$
   findNext recursive $O(n^2)$
4. The evidence of testing your program for correctness.

```
:: ./main graph1.data
1 -> 2 -> 3
2 -> 1 -> 3 -> 4 -> 5
3 -> 1 -> 2 -> 5
4 -> 2 -> 5
5 -> 2 -> 3 -> 4
True
Start at: 3
Path: 3 1 2 3 5 2 4 5
3 -> 1
1 -> 2
2 -> 3
3 -> 5
5 -> 2
2 -> 4
4 -> 5
```

```
:: ./main graph2.data
1 -> 2 -> 3
2 -> 1 -> 3 -> 4 -> 5
3 -> 1 -> 2
4 -> 2 -> 5
5 -> 2 -> 4
True
Start at: 1
Path: 1 2 4 5 2 3 1
1 -> 2
2 -> 4
4 -> 5
5 -> 2
2 -> 3
3 -> 1
```

```
:: ./main graph3.data
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
False
```

```
:: ./main graph4.data
1 -> 2 -> 3
2 -> 1 -> 4 -> 7
3 -> 1 -> 4 -> 5 -> 6
4 -> 2 -> 3 -> 5 -> 6
5 -> 3 -> 4 -> 6
6 -> 3 -> 4 -> 5 -> 7
7 -> 2 -> 6
True
Start at: 2
Path: 2 1 3 4 2 7 6 3 5 4 6
2 -> 1
1 -> 3
3 -> 4
4 -> 2
2 -> 7
7 -> 6
6 -> 3
3 -> 5
5 -> 4
4 -> 6
6 -> 5
```

```
:: ./main graph6.data
1 -> 2 -> 3 -> 4
2 -> 1 -> 3 -> 4 -> 5
3 -> 1 -> 2 -> 5
4 -> 1 -> 2 -> 5
5 -> 2 -> 3 -> 4
False
```

```
:: ./main graph5.data
1 -> 2 -> 3
2 -> 1 -> 3 -> 4 -> 5
3 -> 1 -> 2
4 -> 2 -> 5
5 -> 2 -> 4
True
Start at: 1
Path: 1 2 4 5 2 3 1
1 -> 2
2 -> 4
4 -> 5
5 -> 2
2 -> 3
3 -> 1
```