CSCE 221 Assignment 5 Cover Page

First Name Cody Last Name Williams UIN 924008283

User Name will77868 E-mail address will77868@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	Bailey Bauman	Samantha Ray	Chris Ridley	Abhishek Joshi
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 Cody Williams Date 4/30/17

- – The description of the data structures implemented in your program
  - ∗ Graph: Used to represent data formed from adjacency lists
  - ∗ Adjacency list: Used to information about edges and adjacent edges
  - ∗ Vector: used to implement the adjacency list and keep track of verticies
  - ∗ stl List: used to store information about edges
  - ∗ queue: used to store information when traversing graph in search of shortest path
- – The description of the algorithms implemented in your program
  - ∗ Breadth First Search: Used to travers graph in search of shortest path
- – The evidence of testing your program for correctness.
  - ∗ When testing, I was unable to obtain a succesful working function for finding the shortest path from one point a to another point b in the graph. However, I understand the concept of Breadth First Search and how it could be used.
  - ∗ Breadth First Search uses a queue to keep track of the nodes that have been visited. Once an adjacent Node has been visited, BFS dequeues the node and visits the next adjacent node.
  - ∗ In order to find the shortest path, BFS is used to find all paths from a to b while keeping track of the number of edges and returning the path with the least number of edges and outputting the verticies as they are visited.
  - ∗

```
[will77868]@build ~/final> (18:14:13 04/30/17)
:: make
c++ -std=c++14 -c Graph.cpp
c++ -std=c++14 -o main main.o Graph.o Vertex.o Edge.o

[will77868]@build ~/final> (18:43:30 04/30/17)
:: ./main input1.txt
0: 1 3
1: 2 0
2: 1 3
3: 0 2

List 1 (with duplicates): 1 3 1 3
List 2 (with duplicates): 2 0 0 2
[will77868]@build ~/final> (18:43:32 04/30/17)
:: ./main input2.txt
0: 1 2
1: 0 2
2: 0 1 3 4
3: 2 4
4: 2 3

NO GROUPING POSSIBLE

[will77868]@build ~/final> (18:43:35 04/30/17)
:: ./main input3.txt
0: 1 2
1: 0 3 4
2: 0 5 6
3: 1
4: 1
5: 2
6: 2

List 1 (with duplicates): 1 2 1 1 2 2
List 2 (with duplicates): 0 3 4 0 5 6
[will77868]@build ~/final> (18:43:39 04/30/17)
```