Vedant Kaushik

I encountered a variety of problems when tackling this project. I initially tried using the template code files provided by the professor, however I had trouble getting these files to run together through the main file. I then opted to simply write the entirety of the project into one file. This solved most of my errors when trying to compile the code. Another difficulty was writing this code without the <vector> library. Restrictions like this always end up complicating what would be a trivial section of code. The way I took care of this was writing my code using the vector library, then solving backwards to what it would be like without using the vector library. This helped me visualize what the code needed to do the most.

There aren't many bugs with my implementation that I know of. The weirdest one is a printing error that I get with my output. Initially the shortest path lengths were all printed excluding the source vertex that dijkstras was being run on. Now, they all print but the source vertex is not in order. Im not sure where this issue is arising from but for an example here is the expected output versus my output:
Expected:

```
Single source shortest path lengths from node 2
  1: 1
  2: 0
  3: 1
  4: 1
  5: 2
  6: 2
  7: 3
  8: 3
```

Mine:

```
Single source shortest path lengths from node 2
 2: 0
 1: 1
 3: 1
 4: 1
 5: 2
 6: 2
 7: 3
 8: 3
```

The path values are still correct, these snippets are taken from the small-network.txt file.

In this code all of my classes are kept within one file. I found this to be the easiest way to implement the code. My Edge struct defines the characteristics of the edge class. Each edge has a starting and ending vertex, along with a weight. The weight is assumed to be 1 for most of the inputs. The graph class and creating adjacency matrix is where a bulk of the code is written. With the adjacency matrix the other two requirements of the project are possible so this is where I spent a lot of time. When the graph class is called the first two values of the input file are taken as the number of vertices and edges respectively. Then the matrix is initialized with 0s. The

class also gets information about the edges through the pairs of values below V and E. The respective cells in the matrix are then updated to show that there is an edge between vertices. To find the degree I simply use a double for loop and iterate through each row. Then I find the sum of each row.

To implement dijkstras there are two arrays. One is visited and one is dist or distance. The algorithm iterates 1 less than the number of vertices amount of times and every iteration chooses the vertex with the shortest distance from the source. The arrays are updated accordingly.