**Project 3**

Danny Foley

Computer Science and Engineering

Speed School of Engineering

University of Louisville, USA

[danny.foley@louisville.edu](mailto:danny.foley@louisville.edu)

**Introduction**

In Project 3, the assignment was to compute the shortest possible route from the Traveling Salesperson Problem. The traveling salesperson problem is a common problem where it is representing an actual real-life issue where a salesperson would need to figure out the best way to travel to each city in the area so that they can reduce time and costs.

**Approach**

For this project, a greedy heuristic method is to be taken. A function called greedy (Figure 1), the first city from the given documents (Figure 2) was made to be the starting city. The closest city to that was chosen as the second city and an edge was formed. After the edge was formed, the closest city to the two points forming the edge was found, creating a second and third edge, or a triangle route. Following the formation of the route, each edge and city were looked at and the closest city and edge pair were found. Upon finding these, the existing edge was removed, and two new edges were created that added the closest city to the route. This continued until all cities were added to the route. Upon finishing, the graph containing the edges and cities was returned so that it could be used in a graphical visual (Figure 3) for the user. Also, the route and the distance were printed to the terminal (Figure 4)

**Results**

The algorithm performed as expected. In comparison to the brute force approach or even the BFS or DFS approach, the run times are negligible. The greedy approach was able to complete in 890 microseconds with 30 cities and in 1790 microseconds with 40 cities.

**Discussion**

While this method is much faster, an absolute shortest path is not guaranteed. There are many places where distances could be saved without having certain edges crossing. Having some method of making sure that the crossings did not occur may have decreased the overall distance travelled. Having the greedy method start at different points and then finding the best route from those different start positions also could have led to possible better times. With C++ not having any built-in libraries for graphical interfaces, finding a working plot library was also difficult and limited the ability to graph the route in a way that could label. Moving forward, the next projects may need to be written in something like Python, which has countless GUI libraries.

**References**

<https://www.techiedelight.com/measure-elapsed-time-program-chrono-library/>

<https://www.nuget.org/packages/plot_client/>

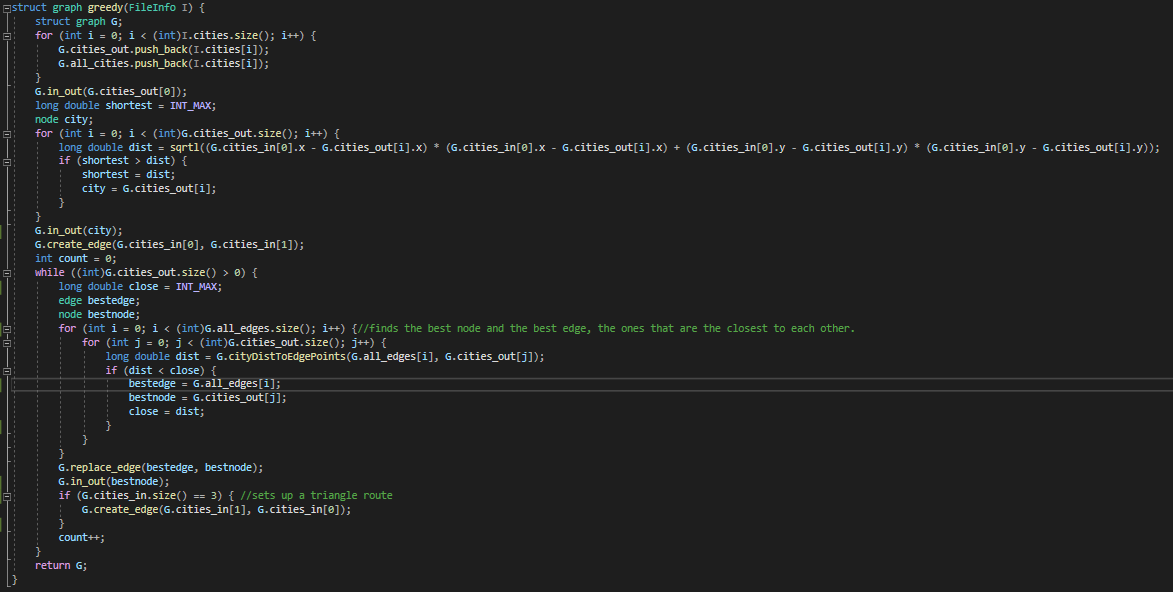
**FIGURES**

Figure 2

Figure 1

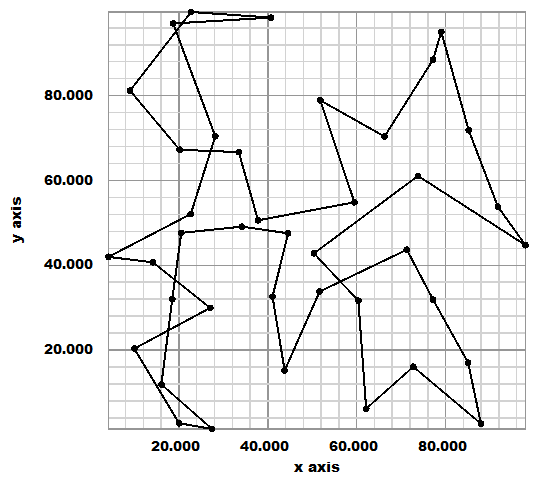
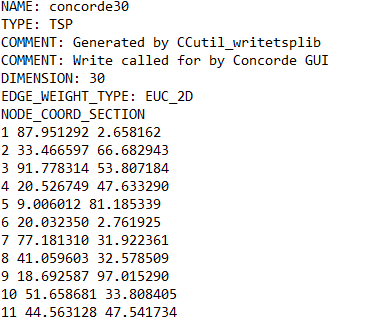


Figure 3

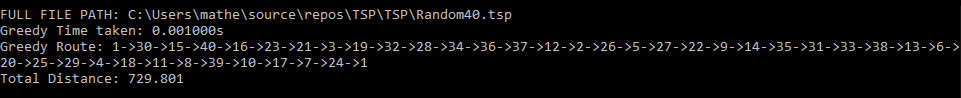


Figure 4