Uber

# Project Description

An Uber driver is fulfilling a list of requests. Each request has a start location, an end location, and the time the request was made. The Uber driver fulfills these requests in the order that they were made. Upon completing a request, the driver must head to the location of the next request and fulfill that request. The wait time of each request is recorded. The Uber driver must minimize the cumulative wait time of all the requests as much as possible.

# Strategy

The network is built from the csv file. It is read in as a 2d array where the location and value represent the node connections and the edges between them. For example, [0][5] in the 2d array with a value of 1 would represent a connection between node 0 and 5, with a 1 time unit travel time. The goal of the algorithm is to find the shortest path between nodes given the connections between each node.

# Algorithm

The data is read from the csv files into a 2d array representing the city network. The array is then converted into a numpy matrix, which is then converted into a networkx graph object. Using this object, the shortest path can be found using Dijkstra’s algorithm.

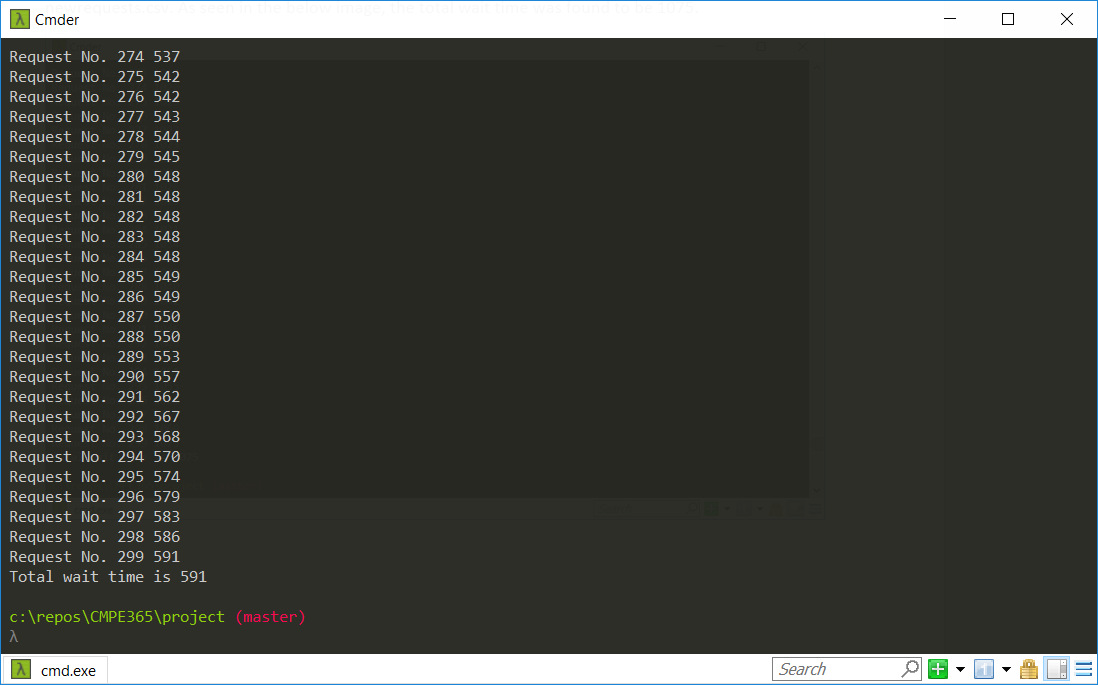
## Dijkstra’s Algorithm

Dijkstra’s algorithm is an algorithm for finding the shortest path between 2 nodes in a graph. Dijkstra’s algorithm initially marks every node with a tentative distance of infinity for every node except for the starting node. This is to note that the nodes have not yet been visited. For the node that the algorithm is currently checking, it will calculate the tentative distance between itself and all its neighbours and assign the smaller value to the neighbouring node. When checking the neighbouring nodes is completed, the current node will be marked as visited, and it will never be checked by the algorithm again. This algorithm will repeat itself until it has either marked the destination node as visited or the smallest tentative distance in the unvisited set is infinity, which signals that there is node connection from the start to the end node. The time complexity of this algorithm is O(V2) where v is the number of nodes.

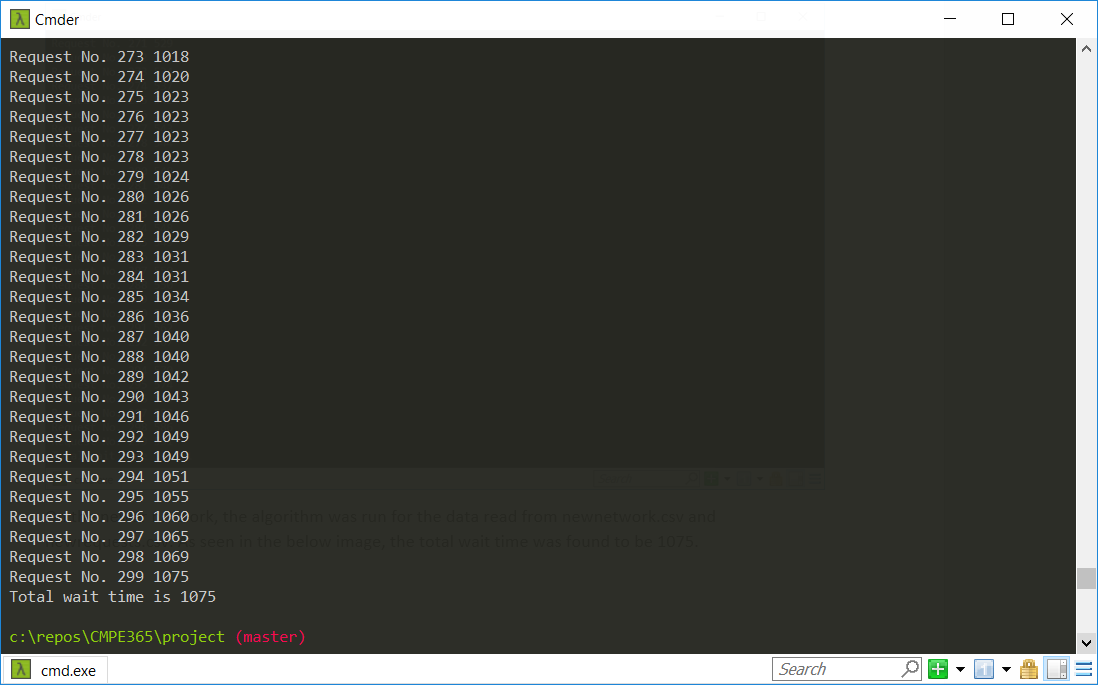
After using Dijkstra’s algorithm to find the shortest path from the start node to the end node, the time taken to travel to that node is recorded. If the current request being processed is not the last one, the algorithm will be run once again from the node at the end of the current request to the node at the beginning of the next request. The time it takes to travel from those nodes will also be recorded and added to the total time elapsed. If the total time elapsed is less than the time that the next request was made, that means that the Uber driver got to the next location early. The Uber driver will then have to wait for the request to be made before it can be processed. If the total time elapsed is greater than the request time, that means the Uber driver was late getting to the location. The difference in time will then be added to the total wait time. This algorithm repeats until every request in the list has been completed. The total wait time will then be outputted as a result. With the number of results included and represented by R, the complexity of the algorithm will be O(V2R).

# Results

When the algorithm is run for network.csv and requests.csv, the total wait time was found to be 591, as seen below.



On the newer network, the algorithm was run for the data read from newnetwork.csv and newrequests.csv. As seen in the below image, the total wait time was found to be 1075.



# Conclusion

Overall, the algorithm is able to determine the shortest path from each node. However, there are slight alterations that can be made to the algorithm that would make it slightly more efficient. For example, requests that are submitted at the same time or around the same time could be done first, rather than being placed in the queue. This would help lower the cumulative wait time for certain scenarios. For example, if two requests were made at the same time, selecting the closer node to where the Uber driver is currently, may lower the total wait time.