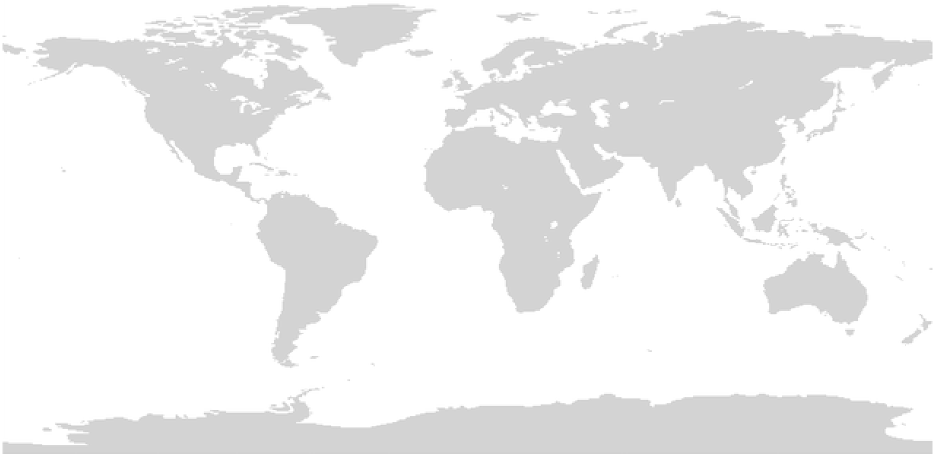
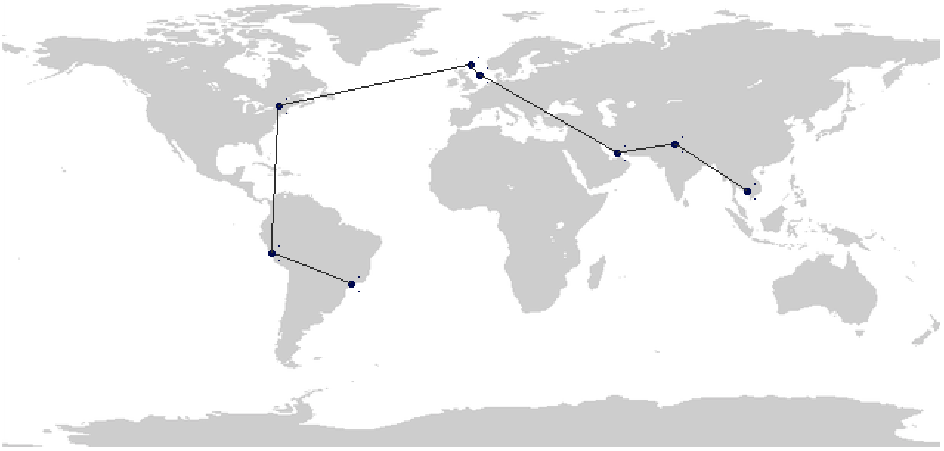
RESULTS

For our project, we set out to make a program that can find the shortest path between any two destinations and their corresponding flights, as well as find a way to visit the top 40 destinations in any starting location and traverse them all. We would then take the selected path and animate them in a gif which shows step-by-step traversals as well as show the final completed path in a png picture format.

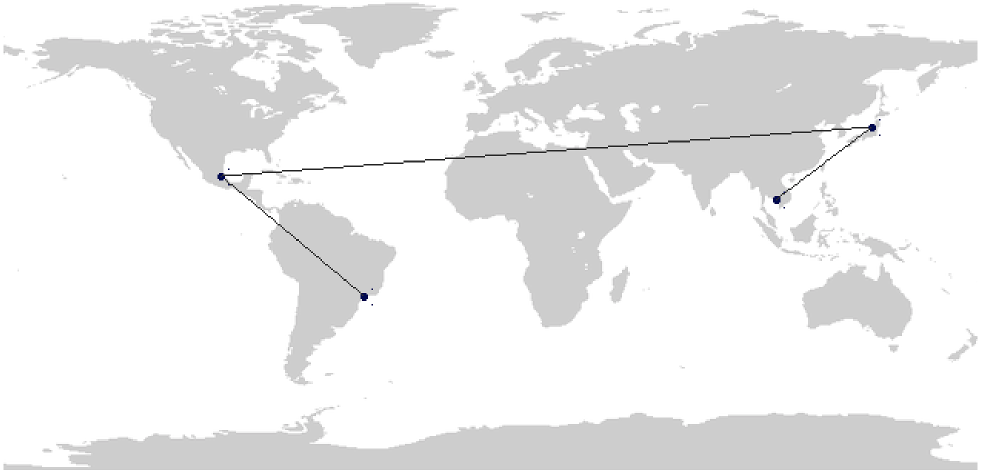
The approach that we took in order to combat this issue was through the use of Dijkstra and BFS in order to generate two unique paths in order to get from a starting point to the desired destination that are both inputted by the user. In this sense, we did finish and accomplish our goals as we did the Dijkstra algorithm to find the minimum time it takes to travel between two cities by plane. In which case, it generated a relatively straight path along each step just like the algorithm is supposed to. This also fits alongside the real-world examples as a direct flight is usually always faster than making layovers at certain places. In addition, we also did BFS to pick out the route from one city to another with minimum transfer times. (Results shown below) We were successful in our attempt to animate and visualize the graph traversals as well as the shortest path algorithm from any two cities that are in the list of 40. Even going as far as to add one city at a time on the gif which shows a step progression rather than just a simple all complete printed list/picture. In addition, we double-checked and were able to print out the flight path in an array that matched that of the correct corresponding solution which we cross-referenced by hand using the real CSV data.

DJISTRAK GIF AND PNG:





BFS TRAVERSAL PNG:



Along the way, some of the interesting discoveries we found were the limitations of the C++ as a language which prohibited more complicated algorithm implementations, such as its complicated nature when we were coding data breakdown and parsing. In addition to parsing data, we were intrigued by the actual usefulness of the algorithms we learned in class and their real-life applications. We were interested that the graphs could be constructed and traced just like that of predetermined data structures that we encounter during coding practices. Turning real data into computer problems is truly fascinating.

Some additional features we wished we had more time for included but not limited to, a formatting strategy that allowed the user to type in a continent and it would get the resulting cities in said continent. Another feature would be using feature selection with more data like costs and number of passengers to get a more realistic flight approach that takes in more than just simply how long it requires to fly to a destination.