Individual Project Reflection

In this project exercise, I explored various graph search algorithms to find the optimal solution to the project challenge posed. I settled on using an informed heuristic search strategy known as A\* search, to map out various flight paths from different airports, given it had certain advantages of over other alternatives like Dijkstra’s, breadth first search or depth first search. In particular, A\* is not step based like either depth first search or breadth first search. Its greedy nature provided my program the ability to calculate and generate the solution paths that were optimal, distance wise.

Again, the algorithms’ ability to calculate the closeness of each neighboring node to the destination, gives it an informed advantage over the likes of other greedy algorithms like Dijkstra’s.

My program’s implementation of the algorithm makes use of object instances of a custom Airport class. These are further represented as graph nodes and enqueued unto the search frontier which is ideally a priority queue (min heap) to take advantage of its sorted nature. A link or route between airports is represented as a route object, which is used to generate successor airports(nodes) in the search. These nodes on the frontier are arranged in the frontier based on a their distances to the destination airport. This enables airports with shorter distances to the destination airport to be expanded at the expense of others, hence enforcing that the shortest path is always returned by the algorithm.

Through this activity, I became well versed with the variations of searching algorithms and how they suite different needs. Breadth first search and depth first search are great graph search algorithms when cost is irrelevant to the search under question. However, in instances where optimality by a cost factor is required, greedy algorithms like Dijkstra’s and best first approaches are undisputably better options.