Introduction

In this project exercise, I investigated numerous graph search algorithms in order to discover the best solution to the project problem. I used an informed heuristic search technique known as A* search to map out possible flight itineraries from various airports because it has advantages over other choices such as Dijkstra's, breadth first search, or depth first search. A*, in particular, is not a step-based search like depth or breadth first search. Its greedy nature enabled my program to calculate and generate the optimal solution paths in terms of distance. Again, the algorithms' capacity to compute the proximity of each neighboring node to the target provides it with an informed edge over other greedy algorithms such as Dijkstra's.

Approach

The approach is implemented using object instances of a custom Airport class. These are then represented as graph nodes and enqueued onto the search frontier, which is ideally a priority queue (min heap) to capitalize on its sorted nature. A link or route between airports is represented as a route object, which is utilized in the search to create successor airports (nodes). These frontier nodes are arranged in the frontier according to their distances from the target airport. This allows airports with shorter distances to the target airport to be expanded at the expense of others, ensuring that the algorithm always returns the shortest path.

Conclusion

This assignment taught me about the various types of search algorithms and how they fit different demands. When cost is immaterial to the search, breadth first search and depth first search are excellent graph search algorithms. However, when optimality by a cost component is necessary, greedy algorithms such as Dijkstra's and best first techniques like A* are undeniably superior.