**Individual Project One Reflection**

Cyril P. Kujar

Ashesi University

Course: Intermediate Computer Programming

September 30th, 2022

Instructor: David Ebo Adjepon Yamoah

Faculty Interns: Owusu-Banahene Osei

In navigating the route problem, initially a number of ideas came to mind as to how to solve this challenge and the different ways to go about it. However, my first concern was about how I would read the file input from the user and begin the search. After a thorough analysis, I decided that once I read the input from the user, for each line read I would add it to a list then assign the list indexes to variables. Now the issue was how to generate the airport in a particular city given the city name. The first data structure that came to mind was a HashMap due to its capabilities of storing key, value pairs.

I noticed that my program would need to be look up certain items quickly and so this structure would be most efficient. I stored data from each file in these Maps with a unique key being the city and country name for an airport, the source IATA code for a route and the airline IATA code for an airline. After doing this, I noticed that some cities possess multiple airports and so this might cause some of my airport values to be overridden and not added to the HashMap. For this reason, I decided to keep track of a list of airports and routes that leave a particular location as my values in my airport and route HashMap respectively. This enabled me to easily look up the airports given a start city, and once this airport object is attained, I can generate all the routes leaving the respective airports as a list.

Once I was able to obtain all the routes leaving a particular airport from my routes HashMap, I use a search algorithm termed depth-first search to find a route to the given destination. The algorithm takes the start position and checks if it is the goal. If it is not, then it adds it to a structure called a frontier and expands all the children of this position. In this context, the children are the routes which can be visited from a particular airport object. Once the solution is found, it returns the path taken to get this solution.