Rodolphe PONTHON
Groupe A – A17

AG44 Project 2 – Ariadne's Thread

Questions

1) The problem given corresponds to the shortest path seen in lessons. We choose to solve it with the Dijkstra's algorithm.

2) We choose to run again the Dijkstra's algorithm with a different graph (the same as the original without the type of edges we don't want to use). Given that we operate Dijkstra's algorithm the exact same way we saw in the lessons, the reachable points are only the ones the don't have an infinite weight.

Data Structures

- **Point.java** : this is the equivalent of a vertex and contains a number, a name, an altitude and an ArrayList of Point that represent his successors, all those data given in the input text file.

- **Route.java** : this is the equivalent of an edge and contains a number, a name, a type, a start and an arrival (directed graph), all those data given in the input text file. It also contains a float time variable, calculated from the type and the altitude difference of the start and arrival.

- **Station.java** : this is the equivalent of the graph and contains an ArrayList of Point and an ArrayList of Route. We use this class to read the file and build up the graph consequently.

Used Algorithms

As said in the Question part, we used the Dijkstra's algorithm, perfectly adapted to our problems (both questions) : there are no negative edges/routes so there no need to perform the Bellman-Ford algorithm. I encoded the algorithm with Boris BROGLE and Yann PONCET and we based it on the exact same way as we saw in lessons.

Dijkstra's algorithm is inside the Dijkstra.java file and to perform well this algorithm, we created another class inside dijkstra's one called WeightPred. WeightPred class is composed of a float that contains the weight of the vertex and an integer that contains the predecessor of the vertex. Dijkstra.java only has as attributes a Point (that corresponds to the starting point) and an ArrayList of WeightPred elements that contains for each Point the actual weight and the predecessor we used to have this weight (at the beginning, the starting Point has a weight of 0 and all other Point has an infinite value).

Rodolphe PONTHON
Groupe A – A17

Graphical Interface

Like many other students, I combined the swing library for the interface and the JGraphX library to print the graph because this one allows us to directly create vertices (with a position) and edges, both with a modifiable color. I implemented the graphical interface in the GFrame.java file and it only need a Station (graph class) to be build.

The only thing to know is that reachable points are purple, unreachable points are blue and the shortest path is yellow.