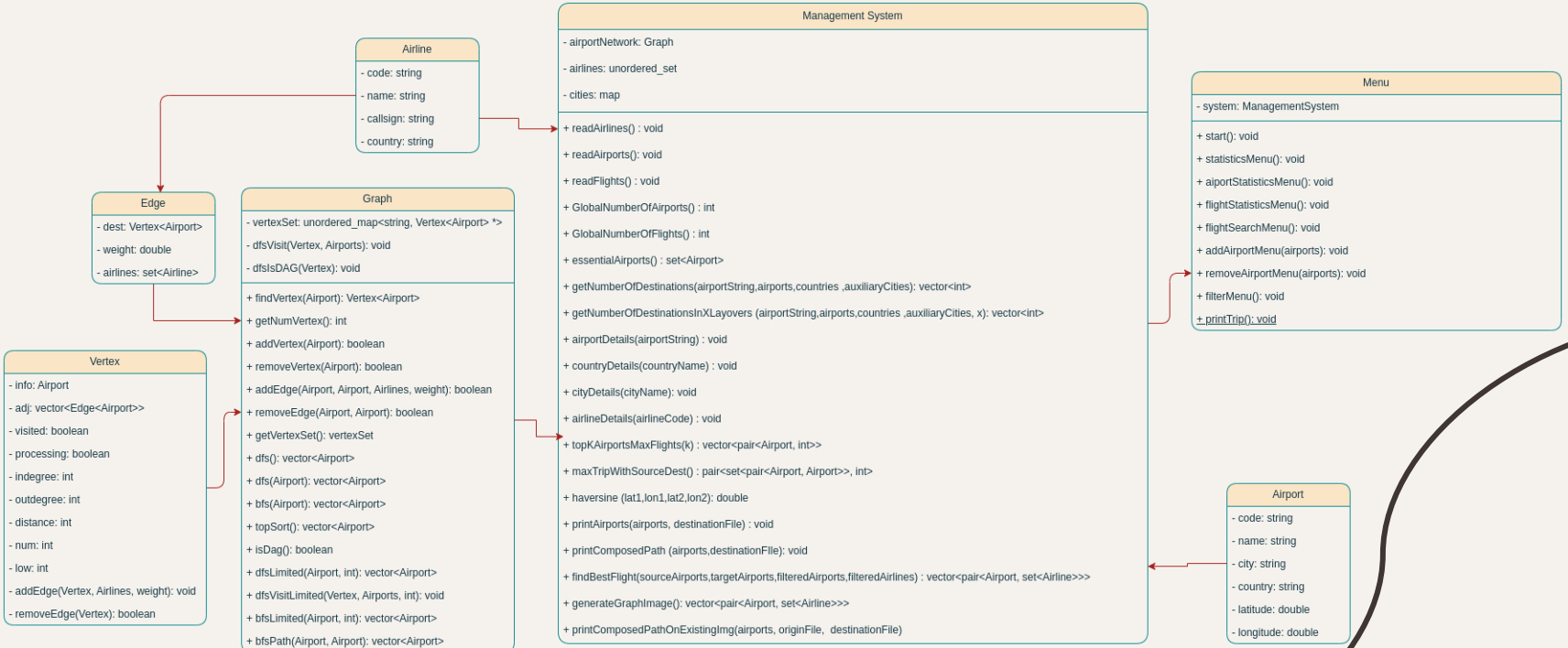
Air Travel Flight Management

Made by Henrique Fernandes, José Sousa and Leandro Martins

Table of contents

Introduction

Overview of the structure



Reading the dataset

-Read Airlines -> read airlines from airlines.csv and add to the unordered_set airlines

-Read Airports -> read airports form airports.csv, create a vertex with each airport and store the vertices in an unordered_set

-Read Flights -> read flights from flights.csv and a new edge if there isn't one between the airports already and if there is add the airline to the edge

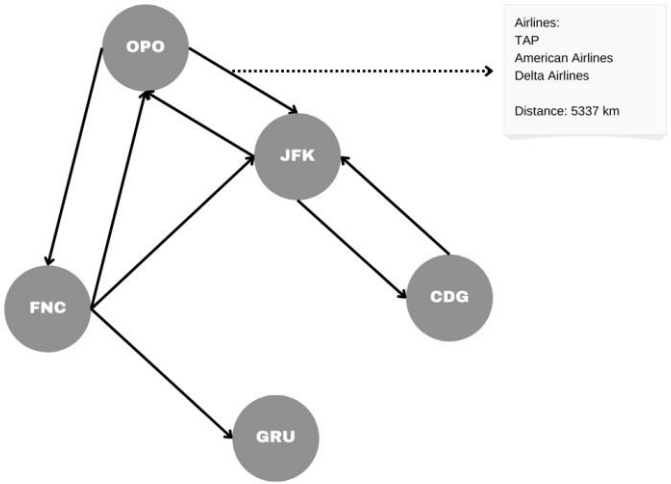
Graph description

-**Vertex** – Represents each Aiport in the network

-**Edge** – Represents each Flight from one Airport to another
In addition it holds the values of the different airlines that fly that route

In addition the Graph has many helper functions for BFS, DFS, sorting(TopSort), cycle detection , etc..

Visual Representation of Graph



Implemented Functionality (3)

- Display **Global number of airports** by returning the number of vertices **O(1)** and the **number of flights** by iterating over the vertices and then over the edges and adding for each edge the number of airlines **O(|V|*|E|))**

- Print the **essential airports** using the algorithm for finding articulation points presented in the theorical classes based on the Tarjan algorithm **O(|V|*|E|*K))**

-Print number of **possible destinations** from an airport **O(|V|+|E|) and reachable destinations** from an airport in **X layovers O(|V|+|E|)** both implemented with using DFS

Implemented Functionality (3)

- Display **details** of a given **airport** for instance number of **flights, countries and airlines O(|E|*K)**

- Display **details** of a given **country** for instance number of **flights, countries and airlines O(|E|*K*L)**

- Display **details** of a given **city** for instance number of **flights , countries and airlines O(|E|*K*L)**

- Display **details** of a given **airline** for instance number of **flights and airports O(|V||E|)**

-

Implemented Functionality (3)

- Display the **top k airports** with the maximum number of flights **O(|V|*|E|+log(V))**

- Find **max Trip** (diameter of the graph) using BFS **O(|V|*(|V|+|E|))**

-

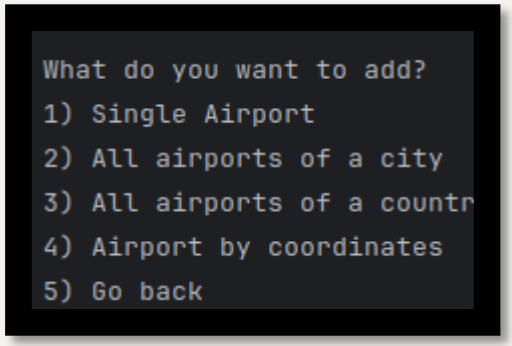
Implemented Functionality (4-5)

-Function uses the **haversine distance** to calculate distance between 2 points on a sphere.

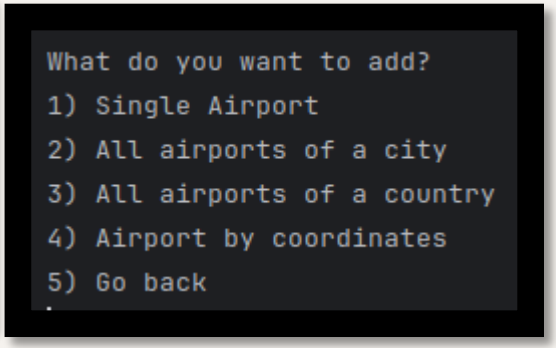
-Display the **best flight options** for a trip with filtering functionality using BFS **O(|V|*(|V|+|E|))**

Implemented Functionality (4-5)

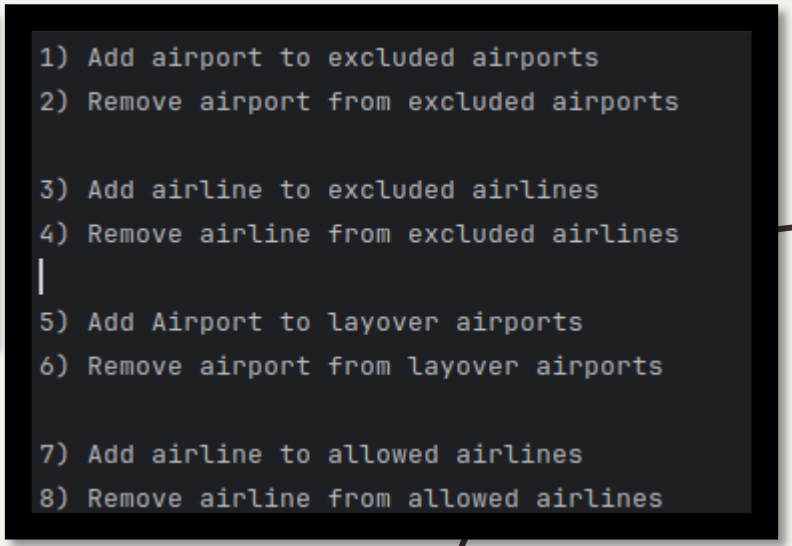
Source



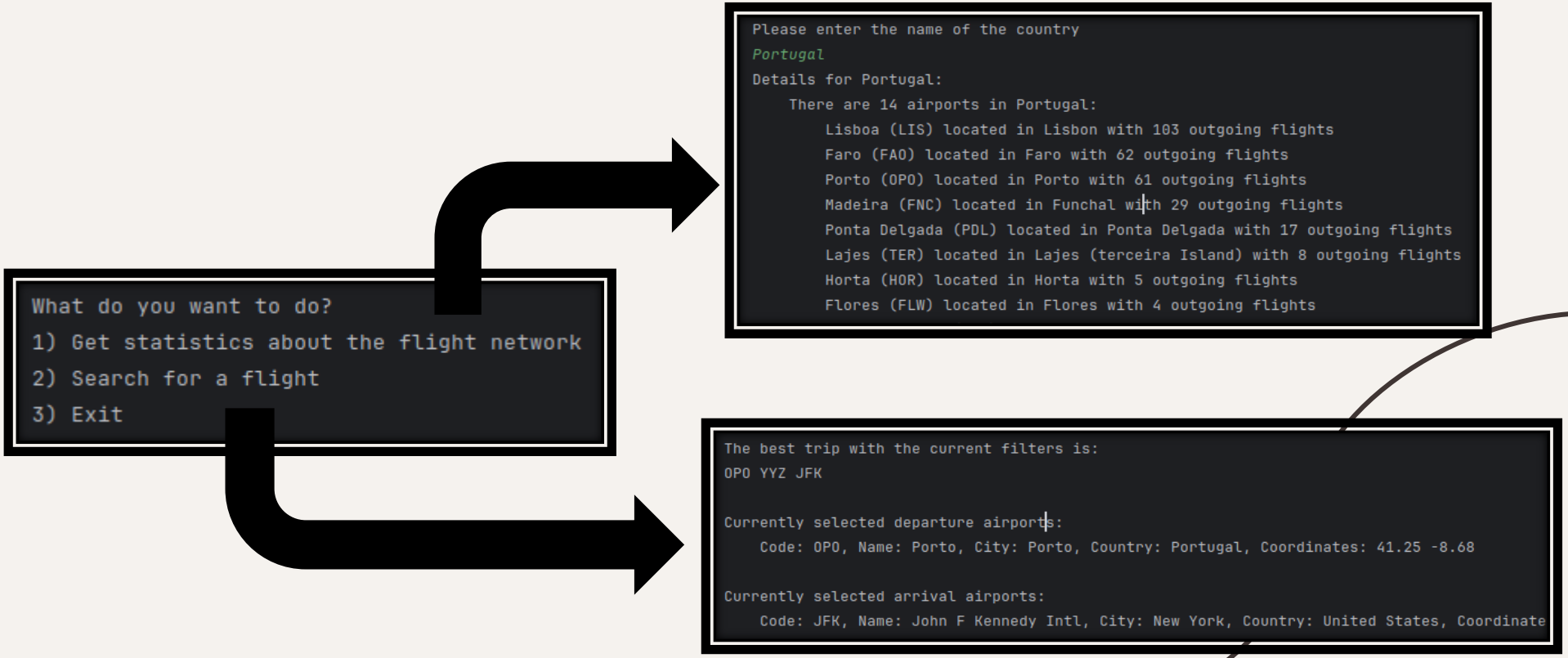
Destination



Network of flights



User Interface



Proud Features- Image System Interface

Valorization tasks – Modifying Images

-We wanted a way to actualy visualize the data we were dealing with in order to make it more appealing and easier to understand

-In the 1st year we developped another project for modifying PNG files so we simplified and integrated it into our current project

Valorization tasks – Mapping coordinates to Image

-Started with basic linear approximation between the physical coordinates and the image coordinates (worked well for longitude but not that well for latitude)

-Used a function based on the Miller Projection which fixed the problem with longitude and made the mapping quite accurate

Valorization tasks – Print Path

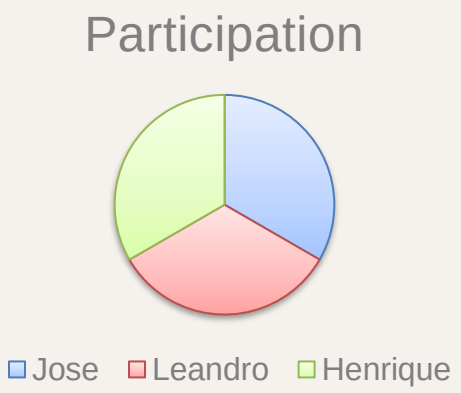
-We also developped functionality that prints a path from one place to another therefore allowing the user to visualize the path from the find the best path functionality, this path adjust for the curvature of the earth

Problems we faced

-Displaying the path

-Naming

-Confirming our results



Conclusion!