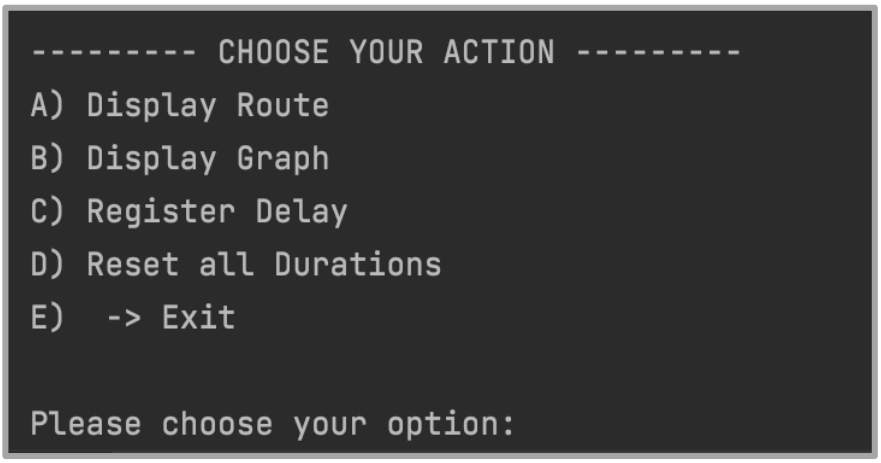
Arbeitsteilung Miniprojekt Modul 411 AirportRoutes

Projektbeschrieb



Wir haben ein Java Konsolenprogramm entwickelt, welches in der Lage ist, auf einem selbst gezeichneten Netz aus Flughäfen und Flugverbindungen die schnellste Verbindung vom Startflughafen zum Zielflughafen ausgeben zu können. Auch zeigt das Programm alle Zwischenflughäfen an, welche man anfliegen müsste, sodass man auf dem schnellsten Weg an seinen Zielflughafen kommt. Weiter kann man alle Flughäfen und die dazugehörigen Flugrouten ausgeben und anzeigen lassen. Weiter kann man die Flugzeit einer Flugroute über die Konsole anpassen. Dies dient dazu, um im Falle einer Verspätung die Flugzeit zu verlängern oder gegebenfalls zu verkürzen. Auch kann man über die Konsole alle Änderungen, die man an den Fluglinien vorgenommen hat mit einem Command wieder zurücksetzen.

Am Ende dieser Dokumentation hat es ein Bild, welches das Fluglinniennetz, welches wir in einem CSV gespeichert haben, visualisiert. Einige der Verbindungen zwischen den Flughäfen sind mit einem Pfeil gekennzeichnet, was soviel bedeutet, wie dass diese Fluglinien nur in eine Richtung geflogen werden.

Kurze Anleitung

In diesem Abschnitt wird erklärt, welche Möglichkeiten die Applikation bietet und wie sie funktionieren.

A)

Diese Methode kann anhand der Kürzel des Startflughafens und des Endflughafens mithilfe des Dijkstraalgorithmus die schnellste Route berechnen und wie folgt ausgeben:

```
--------- ENTER THE START AND END ---------
Enter the first airport abbreviation: ZRH
Enter the second airport abbreviation: ARN

> ZRH -- 0.5800 -- FRA-- 1.7000 -- OSL-- 1.0000 -- ARN
> Duration: 3h 16min
> END
```

B)

Diese Methode gibt zu jedem Flughafen die dazugehörigen Flugrouten und die Flugzeit aus. (Beispiel vom Flughafen Moskau):

```
Moskau-Scheremetjewo (SVO):
        >> 1.6300   Starptautisk? lidosta (RIX)
        >> 1.7500   Chopin-Flughafen Warschau (WMI)
        >> 4.5000   Adolfo Suárez Madrid-Barajas (MAD)
        >> 2.5000   Istanbul Airport (IST)
        >> 2.1200   Международный аэропорт Сыктывкар имени П.А.Истомина (SCW)
```

C)

Mit dieser Methode kann die Flugzeit einer bestimmten Fluglinie angepasst werden. Hierbei muss beachtet werden, dass diese Änderung nicht permanent gespeichert wird.

```
Please choose your option: C

--------- ENTER THE DATA ---------
Enter the first airport abbreviation: ZRH
Enter the second airport abbreviation: FRA
Enter the new duration: 2
Duration was successfully altered!
```

D)

Diese Option setzt alle Änderungen, welche an den Flugrouten vorgenommen wurden, zurück.

E)

Das Programm wird beendet.

Schwierigkeiten

Während des ganzen Projektes sind wir unterschiedlichen Schwierigkeiten begegnet. Im folgenden Abschnitt werden wir unsere Probleme auflisten und erklären.

Funktionierende Klassen für Fluglinie und Flughafen:

Um funktionierende Klassen für alle Flughäfen und Fluglinien zu erstellen, mussten wir uns zuerst alle Attribute, welche eine Klasse haben kann, überlegen, sodass wir später eine grobe Idee hatten wie diese Klassen aufgebaut werden können. Danach bestand die Schwierigkeit darin, dass die Klassen für die Flughäfen (Airport) gut mit der Klasse AirNavigationGraph, welche für die Flugverbindungen zuständig ist, zusammenarbeiten muss.

Implementierung Dijkstraalgorithmus:

Um die schnellste Route zwischen zwei Flughäfen herauszufinden, mussten wir uns über den Dijkstraalgorithmus schlau machen. Dies haben wir mit dem Youtubevideo, welches im README File verlinkt ist, gelernt. Hier ist zu erwähnen, dass wir keinen Pseudocode verwendet haben und so den Algorithmus selbst anhand unseres Wissens designt haben. Als wir das Prinzip des Algorithmus verstanden haben, mussten wir eine Funktion schreiben, welche mit unserer Flughafenklasse, sowie der AirNavigationGraph Klasse auskommt. Sobald der

Algorithmus funktioniert hatte, kümmerten wir uns um das ausgeben der Flugroute, was auch eine ziemlich grosse Herausforderung war.

Auslesen der Daten aus dem CSV:

Die Daten für die Flughäfen und Flugrouten holt sich unser Programm aus den jeweiligen CSV Files. Dies stellte für uns am Anfang eine Herausforderung dar. Jedoch hat uns die Klasse, welche wir während des Modules erstellt haben, die ein File auslesen kann, sehr stark geholfen.

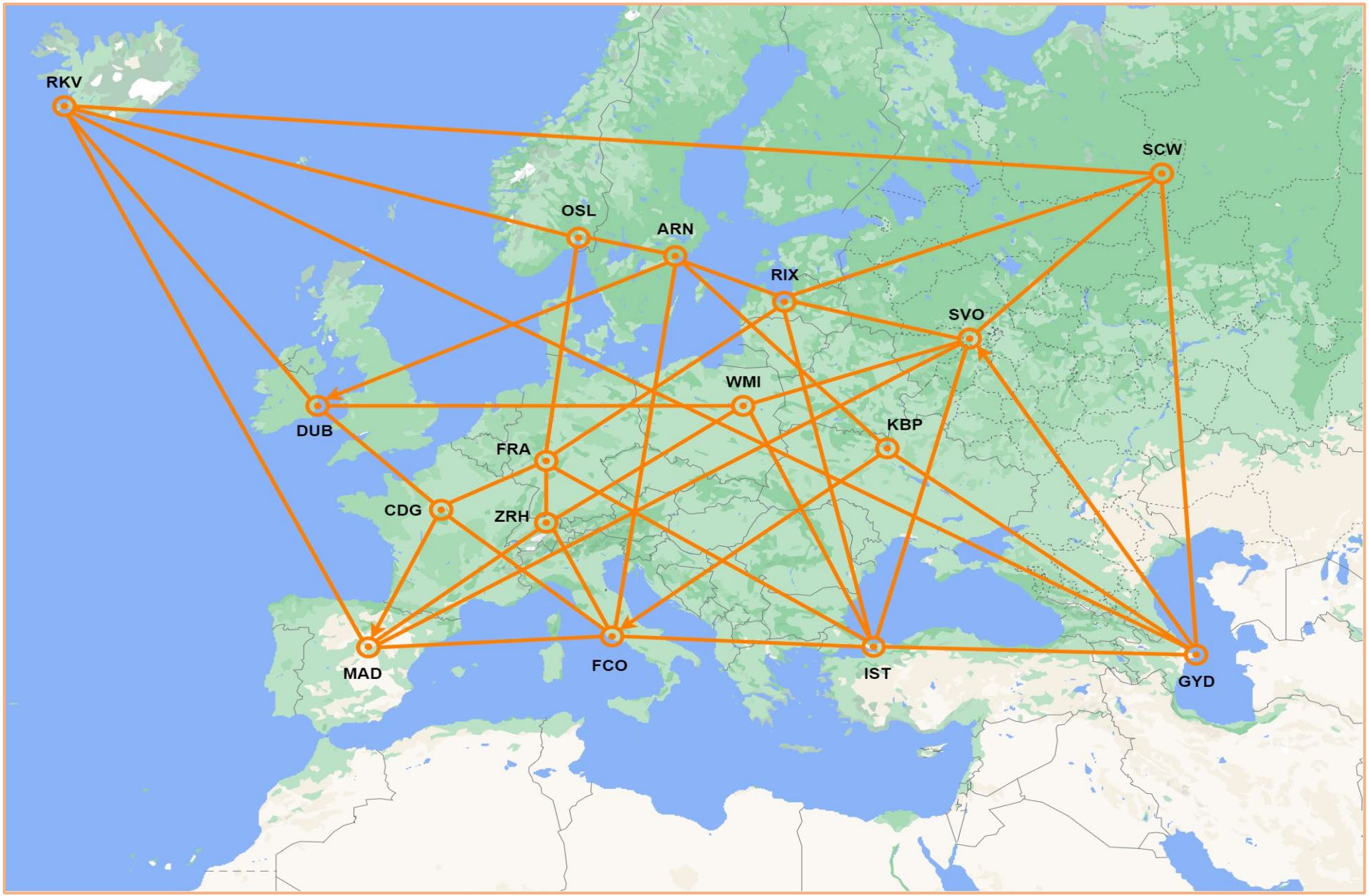
Arbeitsteilung / Arbeitsweise

In diesem Abschnitt dokumentieren wir unsere Arbeitsaufteilung. Meistens haben wir immer wieder zusammen an einem Task gearbeitet. Um nicht die Übersicht zu verlieren haben wir die Tasks immer jemandem zugewiesen, der für diesen zuständig ist. Oftmals haben wir in der Schule in der Zeit, die uns zur Verfügung stand am Miniprojekt gearbeitet. Gegen das Ende, als wir das Projekt noch Zuhause abschliessen mussten, haben wir uns in einem Discordmeeting getroffen und zusammen am Projekt gearbeitet.

Task	Umgesetzt von	Zeitaufwand	Status
Air Routes Class	David	1.5h	Done
Air Navigation Graph Class	David	4h	Done
ConsoleInteraction Class	David	1h	Done
FileHandler Class	David	30min	Done
QueueBucket Class	Joris	30min	Done
Documentation	Joris	2h	Done
Create Flightroutes Graphic	Joris	1h	Done
Create CSV with Airportinformations	David	40min	Done
Create CSV with Flightrouteinformations	Joris	1h	Done

Übersicht Routenplan

Auf der folgenden Grafik kann man eine Übersicht zu allen Flughäfen und Flugrouten anschauen. Dieser Routenplan dient einerseits zur Übersicht, sowie auch zur Hilfe, wenn man einen Flug planen möchte.


RKV
SCW
OSL
ARN
RIX
SVO
WMI
DUB
KBP
FRA
CDG
ZRH
MAD
FCO
IST
GYD