**CSE1242 TERM PROJECT**

Project Name: Travel

Authors: Alperen Burak Koçyiğit 150121035

Aşkın Yavuz Tuna 150121016

Cihat Emre Vardiş 150121037

Lecture: CSE1242 Computer Programming II, Spring 2023

Date Submitted: 25.05.2023

1. **PROBLEM DEFINITION**

Our Project is a game called Travel. Aim of the game is transferring passengers between cities with as few moves as possible. Players try to get highest score until they transfer all passenger.

Game serves a gameboard and game rules then leaves players to create their own levels so the game is also has sandbox features.

1. **IMPLEMENTATION DETAILS**

|  |  |
| --- | --- |
| Fixed | |
| + | CellId : int |
| + | Fixed(CellId:int) |

* The Fixed class represents the cells that cannot be used to roads.
* The data field CellId represents the coordinate of the fixed cell.

|  |  |
| --- | --- |
| Vehicle | |
| +  + | CurrentCityId: int  PassengerCapacity : int |
| + | Vehicle(currentCityId: int, passengerCapacity: int) |

* The Vehicle class represents the vehicles which is used for transporting passengers from one city to another.
* The data field CurrentCityId represents the city’s Id which is the vehicle is present at that time.
* The data field PassengerCapacity represents how many people can travel in the vehicle at the same time.

|  |  |
| --- | --- |
| Passenger | |
| +  +  +  +  + | numberOfPassenger: int  startingCityId: int  destinationCityId: int  startingCityName: String  destinationCityName: String |
| + | Passenger(numberOfPassenger: int, startingCityId: int, destinationCityId: int, startingCityName: String, destinationCityName: String) |

* The Passenger class represents the passengers that travel from a city to another together.
* The data field numberOfPassenger represents the number of passengers that travel together.
* The data field startingCityId represents the passengers starting city’s Id.
* The data field destinationCityId represents destination city’s Id.
* The data field startingCityId represents name of the city which is the passengers start the journey.
* The data field destinationCityId represents name of the city which is the passengers arrive.

|  |  |
| --- | --- |
| City | |
| +  +  +  +  + | cityName: String  coordinate: int  cityNumber: int  passengers: ArrayList< Passenger>  arrivals: ArrayList< Passenger> |
| + | City(cityName String,coordinate: int,cityNumber: int) |

* The City class represents a city in the game.
* The data field cityName represents the name of that city.
* The data field coordinate represents the exact location of the city.
* The data field cityNumber represnt the city’s Id. (cityNumber isn’t represent the location of the city. Just a number to seperate the city from others.)
* The data field passengers represents an ArrayList object. This list contains the passengers which are travel from that city.
* The data field arrivals represents an ArrayList object. This list contains the passengers which are come this city from another.

|  |  |
| --- | --- |
| Main | |
| +  +  +  +  +  +  +  +  +  +  +  +  + | *level: int*  cities: ArrayList<City>  vehicle: Vehicle  polyline: Polyline  list: ObservableList<double>  vehicleimage: Circle  fullscreenMod: boolean  number: int  score: int  currentPassenger: int  drive: Button  nextLevel: Button  ender: boolean |
| +  +  +  +  +  +  +  +  +  + | start(primaryStage: Stage): void  addCity1(cityName: String, x: int, pane: Pane, center: Pane, bottomLeft: FlowPane, bottomRight: StackPane): void  addFixed(x: int, pane: Pane, center: Pane): void  addVehicle(cityId: int, pane: Pane, center: Pane, capacity: int): void  levelInitializer(filename: String, pane: Pane, center: Pane, bottomLeft: FlowPane, bottomRight: StackPane, topRight: StackPane): void  addItem(item: String, pane: Pane, center: Pane, bottomLeft: FlowPane, bottomRight: StackPane): void  createCircle(centerX: double, centerY: double, radius: double, imagePath: String): Circle  getInfo(city: City, bottomLeft: FlowPane): void  distance(first: int, second: int): int  addLine(pane: Pane, center: Pane, endingCityId: int, bottomRight: StackPane): Polyline |

In the implementation phase, we have tried to implement some methods to draw the lines which show the vehicle’s route but we can’t managed to do that. There were so many possibilites for each route and the lines had to show the best route without using fixed cells. As a result, we couldn’t figure it out.

The greatest difficulty was seperating tasks between us to make it faster. We made some distributions during implementation phase sometimes but generally we implemented together. Actually, this problem had some advantages and disadvantages. We solved many problems very quickly because we were together during that sessions. On the other hand, we had to finish that project and we were a bit slow.

1. **TEST CASES**

**A screenshot of a video game

Description automatically generatedThere are some screenshots that are taken during gameplay.**

**There is a test case for the first level above**

**A screenshot of a computer game

Description automatically generated with medium confidenceWhen player click a city, Drive button appears and informations about that city is shown**

**When user clicks drive, the vehicle moves on the line**

**A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedScreenshots above are taken without fullscreen mod**

**A screenshot of a video game

Description automatically generated with medium confidence**

**Another screenshot from level 3**

**A screenshot of a game

Description automatically generated with medium confidenceA screenshot of a game

Description automatically generated with medium confidence Last level of the game Ending scene and message**