Train Travel Inc.

You must create a console-based application that helps a travel agency sell train tickets. The agency has a number of **Routes** it has tickets for and wants to keep track of the number of available seats for each route as well as their income from selling tickets. The income is obtained by adding the prices for all tickets sold. The price of train tickets is based on a flat hourly rate, and is calculated as the product between the hourly rate and the travel time of the route, with minute precision. The functionalities that you must implement are:

1. Add a new train route. Each train route has a unique ***number***, a ***departure city*** and ***time***, an ***arrival city*** and ***time***, and a ***number of available tickets***. The departure and arrival cities must be distinct, the departure time must precede the arrival time (assume the same day), and the number of available tickets must be a positive integer (***1.5p***).
2. Sell a ticket. The user enters the train number and the program calculates and displays the price of the ticket. If the user accepts, the price is added to the company's income and the number of available seats on the route is decreased by 1. If the user declines, the number of seats available is not changed (***1.5p***).
3. Show the total income. This will show the total income on-screen (***1p***).
4. Show report. This will show an ordered list of train routes, sorted in descending order by the number of tickets sold for each route. The number of tickets sold must also be displayed (***2p***).

Non-functional requirements:

- Implement a layered architecture solution with Repository, Controller and UI (***1.5p***).
- Provide specification and tests for all non-UI methods (***1.5p***).

Observations!

- The list of train routes is loaded from/saved to a text-file that must initially hold at least 5 entries.
- Solutions that store train routes in memory are graded at 50% for all functional requirement (maximum is 3p).
- The exam cannot be passed without working functionalities!
- The first number in the example below is the value of the flat hourly rate

routes.txt example:

```
15
1742,Satu Mare,10:20,Bucuresti,20:30,100
1743,Bistrita,12:00,Cluj,14:15,50
1856,Constanta,00:00,Bucuresti,03:30,220
1900,Tulcea,05:30,Galati,06:45,90
1901,Galati,09:00,Tulcea,10:15,100
```