Assignment 1: Space Travel System using C++

CS253 - 2023

IIT Kanpur

Gone are the days when space travel was a unique privilege for a few highly skilled astronauts. We are entering into an age of commercial space travel with the advent of SpaceX, Virgin Galactic etc.

In this problem, you will implement a **Space Travel System (STS)** in C++ language using Objected Oriented Programming Concepts. In this STS, the traveller should be able to book a ticket to a planet from a source planet on a specified date. The traveller should also have an optional choice of booking a return ticket.

Each ticket will have a default validity of ten years. The traveller should be able to book, delete and update any ticket. The traveller should set his identification number. Each travel is assigned a unique identity number on creation. Implement a functionality to print or view the ticket. Traveller name and travelled ID should be printed by calling getName() and getId() methods respectively.

There should be three types of travellers i.e. Astronaut, Passenger, and Commander. Astronaut and Commander should have an infinity validity. The Astronaut should have Years of Experience and License ID (type: int / string, unique), flying license/ Space travel license. Commander should also have authority field. Commander should be able to check upcoming travel details.

The following classes should be implemented:

- Space Travel Class :
  All the data members of this class are private. It should have the following attributes:
  - List of travellers: Up to 10
  - Astronaut: One per travel
  - Commander: One per travel

  The class should implement the following methods.
  - Add travellers

- List all travellers
- Set the Astronaut ID
- Set the Commander ID
- Update the astronaut ID, while changing the astronaut.
- Change the commander ID, while changing the commander.

A space travel class object should only be created when more than 2 passengers book the ticket for the same travel from the same source station to the same destination nation on the same specific date.

- Ticket class :
  Check the validity of the passenger before booking the ticket. When a ticket is created, the price is determined based on how far the travel date is from the current date i.e. (date of travel + 1) * price_of_ticket = K (K is a constant). Consider Euclidean distance to calculate the distance between two stations.

- Planet class :
  The class should have the name and x,y and z coordinate position as member elements. It should implement getCoordinates() member function, which will return the coordinates of the planet. It should also create different planets for the whole universe so that travel can be possible.

  [Note as a bonus you may also implement different attributes of various planets. See the class diagram. But this part is purely optional.]

In the main code() first build the universe with at least 3 planets, 15 passengers (out of which 2 are astronauts and 2 are commanders).

Each travel must have one astronaut and one commander assigned. For now, there is no need to check whether the astronaut and the commander would have travel conflict. [Bonus: make sure there is no conflict, else the travel is cancelled, if there are no available astronauts or commanders.]

The classes shown here are just exemplary. Feel free to be a bit creative by adding your own data members or member functions such that the system is implemented.

Each attribute of the class must have a member function to either update the attribute or reset the value.

Each attribute of a class must have a member function, that prints or shows the value of the attribute.

Feel free to decide which data member of a class is either public or private. See the class diagram for reference.

Astronauts and commanders do not have the validity attribute.

Feel free to choose default value(s) for other parameters.

Any queries regarding this assignment should be directed to **Debkanta Chakraborty (debkanta@cse.iitk.ac.in)**. Please mention **CS253** in the subject line while sending any email.

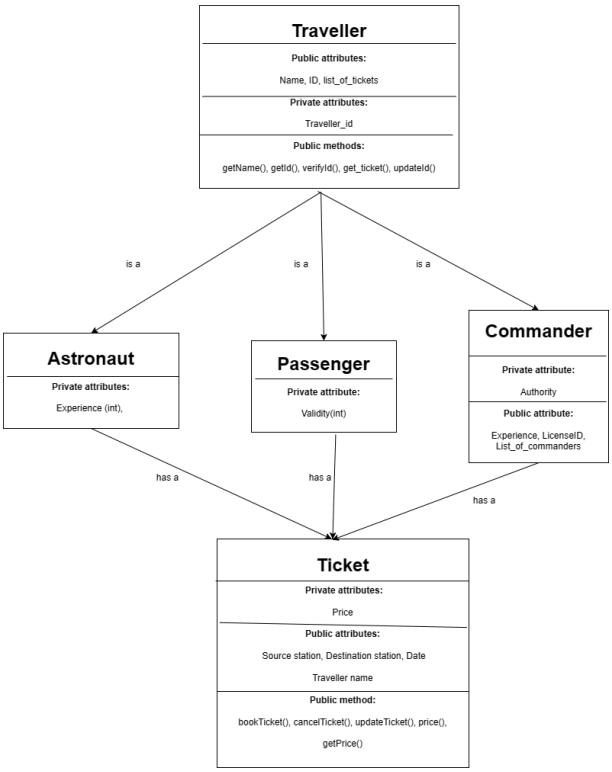

Figure 1: List of Traveller and Ticket classes

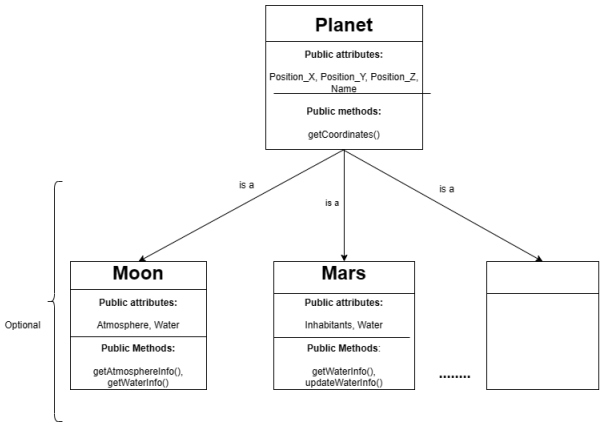

Figure 2: List of Planet class