TEST PLAN

-Abhilash Datta

Unit Testing Plan:

Class Station:

- First Construction is tested.
- The Methods like GetName, GetDistance are tested.
- The friend function overloaded << operator is tested.
- Every output is compared to its Golden Value.

Class Date:

- First Construction and Validity is tested.
- The Methods like getDay, getMonth, getYear, ComputeAge are tested.
- Copy constructor is tested.
- Overloaded Equality operator is tested.
- The friend function overloaded << operator is tested.
- Every output is compared to its Golden Value.

Class Railways:

- Validity is checked.
- The Methods likeGetDistance are tested.
- The friend function overloaded << operator is tested.
- Every output is compared to its Golden Value.

Class Gender and Hierarchy:

- Methods like GetName, GetTitle, IsMale are being tested and are compared with their Golden outputs. If nothing gets printed, this implies no error.

Class Passenger:

- Good and bad passenger objects are constructed.
- Methods like GetGender, ValidatePassenger are being tested.
- We can see that the good passenger gets printed (using << overloaded operator) whereas the bad passenger(with some wrong input) doesn't.

Class BookingClass and Hierarchy:

- Testing << operator. This operator calls methods like GetName, IsSitting, IsAC, GetNumberOfTiers, and IsLuxury methods
- if the printed values match golden values, it works fine.
- Testing virtual GetLoadFactor Method: calling GetLoadFactor of every final concrete class and comparing them to golden values. If no error prints on screen, the method works fine
- Every inherited class object is called and printed.

Class BookingCategory and Hierarchy(Categories) :

- The whole hierarchy consists of singleton classes.
- Unit testing checking whether every singleton is properly implemented or not.
- Two instances of every inherited class is called, and checked for the same address.
- If nothing gets printed, it has no error.

Class Exceptions and Hierarchy:

- All kinds of exceptions are thrown and catched to check the proper implementation of Exceptions.

Class Booking:

- Testing of Application implies testing of this class.

Application Testing Plan

Approach: Testing every kind of exception by giving wrong input and then giving the right one, covering all booking Categories and classes. In all test cases a negative and a positive solution is there.

Test Case 1:

Scenario: Giving a Bad_Date exception by putting ambiguous dates like 30/2/2001. Booking Category: General
Output: Bad Date.

Then fixing it by exchanging them.
Output: prints the booking

Test Case 2:

Scenario: Giving Bad_Stations exception by putting wrong name of station. Booking Category: Senior Citizen. Age 60+, Male gender
Output: Bad_Stations.

Fixing it by making the spelling right.
Output: prints the booking

Test Case 3:

Scenario: Giving Name_Error exception by putting no name of passenger. Booking Category: Ladies .
Output: Bad_Name.

Fixing it by giving a name.
Output: prints the booking

Test Case 4:

Scenario: Giving Aadhar_Error exception by putting less than 12 characters in aadhar number of passenger.. Booking Category: Ladies .
Output: Bad_Aadhar.

Fixing it by giving a correct aadhar number.
Output: prints the booking

Test Case 5:

Scenario: Giving Mobile_Error exception by putting less than 10 characters in mobile number of passenger. Booking Category: Ladies.
Output: Bad_Mobile.

Fixing it by giving a correct mobile number.
Output: prints the booking

Test Case 6:

Scenario: Giving Date_Of_Booking_Error exception by giving dob < dor or dob = dor. Booking Category: Ladies.
Output: Bad Date of Booking

Fixing it by exchanging them.
Output: prints the booking

Test Case 7:

Scenario: Giving Senior_Citizen_Error exception by applying for the senior citizen booking category and keeping the age of passenger <60 for Male . Booking Category: Senior Citizen.
Output: Bad_Senior_Citizen.

Fixing it by making the age >60.

Output: prints the booking

Test Case 8:

Scenario: Giving Divyaang_Error exception by applying for any of the divyaang section booking categories but not giving any passenger info related to that . Booking Category: TB.
Output: Bad_Divyaang.

Fixing it by giving proper information.
Output: prints the booking

Test Case 9:

Scenario: Giving Tatkal_Error exception by applying for any of the Tatkal section booking categories but dob and dor are more than 1 day apart. Booking Category: Tatkal.
Output: Bad_Tatkaal.

Fixing it by keeping the days one day apart.
Output: prints the booking