CENG 218 2023-2024 Spring

Lab 1

In this programming lab, the task is to design and implement a C++ program that serves as a reservation system for a specific flight. The program is centered around a **Flight** class, which encapsulates a flight reservation system's essential attributes and functionalities. The user interacts with the system through a **menu**-driven interface, where they can perform operations like reserving a seat, canceling a reservation, and obtaining information about the current passengers. Programs starts with the following **menu** :

Flight Menu
1. Reserve a Ticket
2. Cancel Reservation
3. Check Reservation
4. Display Passengers
5. Exit

Consider the following member variables for Flight class:

-flight no: string
-maximum number of seats: 25
-number of passengers: int
-list of passengers

and following methods

+ reserve a seat (first available seat): this method takes one parameter, the the passenger making a reservation. This method returns true for successful reservation, otherwise false
+ cancel a reservation: this method takes one parameter, the the passenger canceling the reservation. This method returns true for successful cancellation; otherwise, it is false.
+ numberOfPassengers: This function returns the current number of passengers who have reserved seats.
+ printPassengers: This method outputs an alphabetically sorted list of passengers with reservations (by surname)

The program uses a Passenger class to manage individual passenger details, that are name, surname, and gender, along with constructor(s), getters, setters, and potentially overloaded operators.

To have a better user experience, the program incorporates input validation to ensure that user inputs conform to expected formats. Specifically, it verifies that names consist of letters only, preventing unintended input errors.

Moreover, the program includes robust error-handling mechanisms to address various edge cases. Examples of these edge cases include attempting to reserve a ticket when the flight is already full, trying to cancel a reservation for a non-existent passenger, and preventing multiple cancellations for the same reservation.

The overall objective is to create a user-friendly flight reservation system to provide a reliable and efficient solution for managing flight reservations for a single flight through the design and implementation of the Flight and Passenger classes, along with thoughtful consideration of input validation and error handling.