public class Flight {

private String flightNumber;

private String destination;

private String departureTime;

public Flight(String flightNumber, String destination, String departureTime) {

this.flightNumber = flightNumber;

this.destination = destination;

this.departureTime = departureTime;

}

public String getFlightNumber() {

return flightNumber;

}

public String getDestination() {

return destination;

}

public String getDepartureTime() {

return departureTime;

}

@Override

public String toString() {

return "Flight{" +

"flightNumber='" + flightNumber + '\'' +

", destination='" + destination + '\'' +

", departureTime='" + departureTime + '\'' +

'}';

}

}

public class Reservation {

private int reservationId;

private String passengerName;

private Flight flight;

public Reservation(int reservationId, String passengerName, Flight flight) {

this.reservationId = reservationId;

this.passengerName = passengerName;

this.flight = flight;

}

public int getReservationId() {

return reservationId;

}

public String getPassengerName() {

return passengerName;

}

public Flight getFlight() {

return flight;

}

@Override

public String toString() {

return "Reservation{" +

"reservationId=" + reservationId +

", passengerName='" + passengerName + '\'' +

", flight=" + flight +

'}';

}

}

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

public class AirlineReservationSystem {

private List<Flight> flights;

private List<Reservation> reservations;

private int nextReservationId;

public AirlineReservationSystem() {

flights = new ArrayList<>();

reservations = new ArrayList<>();

nextReservationId = 1;

// Adding some dummy flights

flights.add(new Flight("FL001", "New York", "10:00 AM"));

flights.add(new Flight("FL002", "Los Angeles", "12:00 PM"));

flights.add(new Flight("FL003", "Chicago", "02:00 PM"));

}

public void addReservation(String passengerName, String flightNumber) {

Flight flight = findFlightByNumber(flightNumber);

if (flight != null) {

Reservation reservation = new Reservation(nextReservationId++, passengerName, flight);

reservations.add(reservation);

System.out.println("Reservation added: " + reservation);

} else {

System.out.println("Flight not found.");

}

}

public void listReservations() {

if (reservations.isEmpty()) {

System.out.println("No reservations found.");

} else {

System.out.println("Reservations List:");

for (Reservation reservation : reservations) {

System.out.println(reservation);

}

}

}

public void removeReservation(int reservationId) {

Reservation toRemove = null;

for (Reservation reservation : reservations) {

if (reservation.getReservationId() == reservationId) {

toRemove = reservation;

break;

}

}

if (toRemove != null) {

reservations.remove(toRemove);

System.out.println("Reservation removed: " + toRemove);

} else {

System.out.println("Reservation with id " + reservationId + " not found.");

}

}

private Flight findFlightByNumber(String flightNumber) {

for (Flight flight : flights) {

if (flight.getFlightNumber().equals(flightNumber)) {

return flight;

}

}

return null;

}

public static void main(String[] args) {

AirlineReservationSystem system = new AirlineReservationSystem();

Scanner scanner = new Scanner(System.in);

while (true) {

System.out.println("\nAirline Reservation System");

System.out.println("1. Add Reservation");

System.out.println("2. List Reservations");

System.out.println("3. Remove Reservation");

System.out.println("4. Exit");

System.out.print("Enter your choice: ");

int choice = scanner.nextInt();

scanner.nextLine(); // consume the newline

switch (choice) {

case 1:

System.out.print("Enter passenger name: ");

String passengerName = scanner.nextLine();

System.out.print("Enter flight number: ");

String flightNumber = scanner.nextLine();

system.addReservation(passengerName, flightNumber);

break;

case 2:

system.listReservations();

break;

case 3:

System.out.print("Enter reservation ID to remove: ");

int reservationId = scanner.nextInt();

system.removeReservation(reservationId);

break;

case 4:

System.out.println("Exiting...");

scanner.close();

System.exit(0);

default:

System.out.println("Invalid choice. Please try again.");

}

}

}

}