#include <bits/stdc++.h>

using namespace std;

class employee

{

private:

struct EmployeeInfo

{

std::string name;

std::string role;

std::string licenseNumber;

};

std::vector<EmployeeInfo> employees;

public:

void addEmployee()

{

EmployeeInfo employee;

std::cout << "Enter name: ";

std::cin >> employee.name;

std::cout << "Enter role: ";

std::cin >> employee.role;

std::cout << "Enter license number: ";

std::cin >> employee.licenseNumber;

employees.push\_back(employee);

}

void deleteEmployee(const std::string &name)

{

for (auto it = employees.begin(); it != employees.end(); ++it)

{

if (it->name == name)

{

employees.erase(it);

std::cout << name << " deleted successfully." << std::endl;

return;

}

}

std::cout << name << " not found." << std::endl;

}

void editEmployeeDetails(const std::string &name)

{

for (auto &employee : employees)

{

if (employee.name == name)

{

std::cout << "Enter new role: ";

std::cin >> employee.role;

std::cout << "Enter new license number: ";

std::cin >> employee.licenseNumber;

std::cout << name << "'s details updated successfully." << std::endl;

return;

}

}

std::cout << name << " not found." << std::endl;

}

void displayPilotDetails()

{

for (const auto &employee : employees)

{

std::cout << "Name: " << employee.name << ", Role: " << employee.role << ", License Number: " << employee.licenseNumber << std::endl;

}

}

};

class Runway

{

private:

int number;

bool occupied;

public:

Runway(int n) : number(n), occupied(false) {}

void occupy()

{

occupied = true;

}

void release()

{

occupied = false;

}

bool isOccupied() const

{

return occupied;

}

};

class Staff

{

protected:

string name;

public:

Staff(string n) : name(n) {}

virtual void display() const

{

cout << "Name: " << name << endl;

}

};

class FlightAttendant

{

public:

FlightAttendant() {}

FlightAttendant(const string &name, const string &language, int flightID) : name(name), language(language), flightID(flightID) {}

string getName() const

{

return name;

}

string getLanguage() const

{

return language;

}

int getFlightID() const

{

return flightID;

}

void display() const

{

cout << "Name: " << name << endl;

cout << "Language: " << language << endl;

cout << "Flight ID: " << flightID << endl;

}

void input()

{

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Enter name of the flight attendant: " << endl;

getline(cin, name);

cout << "Enter language spoken by the flight attendant: " << endl;

getline(cin, language);

cout << "Enter flight ID: " << endl;

cin >> flightID;

cin.ignore();

}

private:

string name;

string language;

int flightID;

};

class Flight

{

private:

int flightId;

string departure;

string destination;

float price;

int capacity;

int bookedSeats;

int flightNumber;

int distance;

int duration;

public:

Flight(int id, string src, string dest, float p, int cap)

: flightId(id), departure(src), destination(dest), price(p), capacity(cap), bookedSeats(0) {}

int getFlightId() const { return flightId; }

string getdeparture() const { return departure; }

float getPrice() const { return price; }

int getCapacity() const { return capacity; }

int getBookedSeats() const { return bookedSeats; }

bool bookSeat()

{

if (bookedSeats < capacity)

{

bookedSeats++;

return true;

}

else

{

return false;

}

}

bool isFull() const

{

return bookedSeats >= capacity;

}

Flight() {}

Flight(int flightNumber, const string &departure, const string &destination, int distance, int duration)

: flightNumber(flightNumber), departure(departure), destination(destination), distance(distance), duration(duration) {}

int getFlightNumber() const

{

return flightNumber;

}

string getDeparture() const

{

return departure;

}

string getDestination() const

{

return destination;

}

int getDistance() const

{

return distance;

}

int getDuration() const

{

return duration;

}

void input()

{

cout << "Enter flight number: ";

cin >> flightNumber;

cin.ignore();

cout << "Enter departure city: ";

getline(cin, departure);

cout << "Enter destination city: ";

getline(cin, destination);

cout << "Enter distance (in miles): ";

cin >> distance;

cout << "Enter duration (in minutes): ";

cin >> duration;

cin.ignore();

}

void display() const

{

cout << "Flight Number: " << flightNumber << endl;

cout << "Departure: " << departure << endl;

cout << "Destination: " << destination << endl;

cout << "Distance: " << distance << " miles" << endl;

cout << "Duration: " << duration << " minutes" << endl;

}

};

class AirportManagementSystem

{

private:

vector<Flight> flights;

vector<FlightAttendant> flightAttendants;

vector<Runway> runways;

public:

void addFlight()

{

Flight flight;

flight.input();

flights.push\_back(flight);

}

void addFlightAttendant()

{

FlightAttendant attendant;

attendant.input();

flightAttendants.push\_back(attendant);

}

void addRunway(const Runway &runway)

{

runways.push\_back(runway);

}

void displayFlights() const

{

cout << "Available Flights:\n";

cout << "---------------------------------------------\n";

for (const Flight &flight : flights)

{

flight.display();

cout << "---------------------------------------------\n";

}

}

void displayFlightAttendants() const

{

cout << "Available Flight Attendants:\n";

cout << "---------------------------------------------\n";

for (const FlightAttendant &attendant : flightAttendants)

{

attendant.display();

cout << "---------------------------------------------\n";

}

}

void displayRunways() const

{

cout << "Available Runways:\n";

cout << "---------------------------------------------\n";

for (const Runway &runway : runways)

{

cout << "Runway Number: " << runway.isOccupied() << endl;

cout << "---------------------------------------------\n";

}

}

void pay()

{

int d, e, f, amo, y;

cout << "Choose mode of payment :1.Net banking 2.Credit card 3.Debit card 4.Paytm" << endl;

cin >> f;

cout << "Enter amount to be paid" << endl;

cin >> amo;

cout << "Click 1. YES to proceed and 2.NO to go back\n";

cin >> y;

cout << endl;

}

bool bookTicket(int flightId)

{

for (Flight &flight : flights)

{

if (flight.getFlightId() == flightId)

{

if (!flight.isFull())

{

flight.bookSeat();

cout << "Your Ticket has been booked successfully!\n";

return true;

}

else

{

cout << "Sorry, the flight is fully booked!\n";

return false;

}

}

}

cout << "Flight not found!\n";

return false;

}

};

int main()

{

string s1, s2, s;

AirportManagementSystem ams;

ams.addRunway(Runway(1));

ams.addRunway(Runway(2));

employee e;

int choice;

do

{

cout << "\nAirport Management System\n";

cout << "1. Add airport employees\n";

cout << "2. Display Airport Employees\n";

cout << "3. Edit Airport Employees\n";

cout << "4. Delete Airport Employees\n";

cout << "5. View all Available Flights\n";

cout << "6. Book a Ticket\n";

cout << "7. View all Available Runways\n";

cout << "8. Add Flight Attendee\n";

cout << "9. View all Available Flight Attendants\n";

cout << "10. Add Flight \n";

cout << "11. Display FLight\n";

cout << "12. Exit\n";

cout << "Enter your choice: ";

cin >> choice;

switch (choice)

{

case 1:

e.addEmployee();

break;

case 2:

e.displayPilotDetails();

break;

case 3:

cout << "Enter name" << endl;

cin >> s;

e.editEmployeeDetails(s);

break;

case 4:

cout << "Enter Name" << endl;

cin >> s;

e.deleteEmployee(s);

break;

case 5:

ams.displayFlights();

break;

case 6:

int flightId;

int time;

cout << "Enter to and from location" << endl;

cin >> s1 >> s2;

cout << "Choose your preferred timing" << endl;

cout << "1.9:00 hrs 2.13:20 hrs 3.22:00 hrs" << endl;

cin >> time;

cout << "Enter the ID of the flight you want to book: ";

cin >> flightId;

ams.pay();

ams.bookTicket(flightId);

break;

case 7:

ams.displayRunways();

break;

case 8:

ams.addFlightAttendant();

break;

case 9:

ams.displayFlightAttendants();

break;

case 10:

ams.addFlight();

break;

case 11:

ams.displayFlights();

break;

case 12:

cout << "Thank you for using the Airport Management System. Have a nice day!\n";

break;

default:

cout << "Invalid choice. Please try again.\n";

}

} while (choice != 5);

return 0;

}­­­­­­­­­