JS Advanced Exam

Problem 02. Flight Booking System

You are developing a Flight Booking System for a travel agency. Create a class called "FlightBookingSystem" to manage flight bookings and passenger records

Flight Booking System

```
class FlightBookingSystem {
    //TODO: Implement this class
    }
```

Implement a **class `FlightBookingSystem`**, which supports the functionality described below.

Functionality

Constructor

The constructor has **4** properties:

- **`agencyName - a string`**
- **`flights - an empty array`**
- **`bookings - an empty array`**
- **`bookingsCount – 0`**

`At the initialization` of the **`FlightBookingSystem`** class, the **constructor** accepts only the **`agencyName`!**

Methods

addFlight (flightNumber, destination, departureTime, price)

This method adds a new flight to the system. It accepts 4 arguments:

- **`flightNumber - a unique identifier for the flight.`**
- **`destination - a string representing the flight's destination.`**
- **`departureTime - a string representing the departure time.`**
- **`price - a number representing the ticket price.`**

If a flight with the same flight number already exists in the flights array, **return** the following message:

`` `Flight ${flightNumber} to ${destination} is already available.` ``

Otherwise, add the new flight to the flights array as an object and **return** the following message:

`` `Flight ${flightNumber} to ${destination} has been added to the system.` ``

bookFlight (passengerName, flightNumber)

This method allows a passenger to book a flight. It accepts 2 arguments:

- **`passengerName - a string representing the name of the passenger.`**
- **`flightNumber - the flight number for booking.`**

  If the flight with the specified flight number is not found in the flights array, **return** the following message:

  `` `Flight ${flightNumber} is not available for booking.` ``

Otherwise, add the booking to the bookings array as an object and **increment `bookingsCount`** by **`1`**. **Return** the following message:

`` `Booking for passenger ${passengerName} on flight ${flightNumber} is confirmed.` ``

cancelBooking (passengerName, flightNumber)

This method allows a passenger to cancel a flight booking. It accepts 2 arguments:

- **`passengerName - a string representing the name of the passenger.`**
- **`flightNumber - the flight number to cancel.`**

If the booking with the specified passenger name and flight number is not found in the bookings array, **throw error** with the following message:

`` `Booking for passenger ${passengerName} on flight ${flightNumber} not found.` ``

Otherwise, remove the booking from the bookings array, decrement **`bookingsCount`** by **`1`** and **return** the following message:

`` `Booking for passenger ${passengerName} on flight ${flightNumber} is cancelled.` ``

showBookings (criteria)

Accept 1 argument:

- **`criteria –a string representing the booking criteria ("all", "cheap", "expensive").`**

This method **returns** information based on the booking **criteria**:

If the bookings array is **empty**, **throw error** with the following message:

`` `No bookings have been made yet.` ``

If the criteria is "**all**", **return** a list of all bookings in the following format:

- On first line show the following message:

`All bookings(${bookingsCount}):`

- On the following lines, display information about each booking:

`${passengerName} booked for flight ${flightNumber}.`

If the criteria is "**cheap**", **return** a list of all bookings under or exactly the price of $100 in the following format:

- On first line show the following message:

"Cheap bookings:"

- On the following lines, display information about each booking:

`${passengerName} booked for flight ${flightNumber}.`

If the criteria is "**expensive**", **return** a list of all bookings over the price of $100 in the following format:

- On first line show the following message:

"Expensive bookings:"

- On the following lines, display information about each booking:

`${passengerName} booked for flight ${flightNumber}.`

Examples

Input 1

`const system = new FlightBookingSystem("TravelWorld");`
`console.log(system.addFlight("AA101", "Los Angeles", "09:00 AM", 250));`
`console.log(system.addFlight("BB202", "New York", "10:30 AM", 180));`
`console.log(system.addFlight("CC303", "Chicago", "11:45 AM", 120));`
`console.log(system.addFlight("AA101", "Los Angeles", "09:00 AM", 250));`

Output 1

`Flight AA101 to Los Angeles has been added to the system.`
 	
`Flight BB202 to New York has been added to the system.`
 	
`Flight CC303 to Chicago has been added to the system.`

```
Flight AA101 to Los Angeles is already available.
```

Input 2

```
const system = new FlightBookingSystem("TravelWorld");
console.log(system.addFlight("AA101", "Los Angeles", "09:00 AM", 250));
console.log(system.addFlight("BB202", "New York", "10:30 AM", 180));
console.log(system.bookFlight("Alice", "AA101"));
console.log(system.bookFlight("Bob", "BB202"));
console.log(system.bookFlight("Charlie", "CC303"));
```

Output 2

```
Flight AA101 to Los Angeles has been added to the system.

Flight BB202 to New York has been added to the system.

Booking for passenger Alice on flight AA101 is confirmed.

Booking for passenger Bob on flight BB202 is confirmed.

Flight CC303 is not available for booking.
```

Input 3

```
const system = new FlightBookingSystem("TravelWorld");
console.log(system.addFlight("AA101", "Los Angeles", "09:00 AM", 250));
console.log(system.addFlight("BB202", "New York", "10:30 AM", 180));
console.log(system.bookFlight("Alice", "AA101"));
console.log(system.bookFlight("Bob", "BB202"));
console.log(system.cancelBooking("Alice", "AA101"));
```

Output 3

```
Flight AA101 to Los Angeles has been added to the system.

Flight BB202 to New York has been added to the system.

Booking for passenger Alice on flight AA101 is confirmed.

Booking for passenger Bob on flight BB202 is confirmed.

Booking for passenger Alice on flight AA101 is cancelled.
```

Input 4

```
const system = new FlightBookingSystem("TravelWorld");
console.log(system.addFlight("AA101", "Los Angeles", "09:00 AM", 250));
console.log(system.addFlight("BB202", "New York", "10:30 AM", 180));
console.log(system.bookFlight("Alice", "AA101"));
console.log(system.bookFlight("Bob", "BB202"));
console.log(system.showBookings("all"));
```

Output 4

```
Flight AA101 to Los Angeles has been added to the system.

Flight BB202 to New York has been added to the system.

Booking for passenger Alice on flight AA101 is confirmed.

Booking for passenger Bob on flight BB202 is confirmed.

All bookings(2):

Alice booked for flight AA101.

Bob booked for flight BB202.
```

Input 5

```
const system = new FlightBookingSystem("TravelWorld");
console.log(system.addFlight("AA101", "Los Angeles", "09:00 AM", 250));
console.log(system.addFlight("BB202", "New York", "10:30 AM", 180));
console.log(system.bookFlight("Alice", "AA101"));
console.log(system.bookFlight("Bob", "BB202"));
```

```
console.log(system.showBookings("expensive"));
console.log(system.showBookings("cheap"));
```

Output 5

```
Flight AA101 to Los Angeles has been added to the system.

Flight BB202 to New York has been added to the system.

Booking for passenger Alice on flight AA101 is confirmed.

Booking for passenger Bob on flight BB202 is confirmed.

Expensive bookings:

Alice booked for flight AA101.

Bob booked for flight BB202.

No cheap bookings found.
```