Skypay Technical Test 2 - Hotel Reservation System

Summary

In this test, you will implement a simplified Hotel Reservation System using Java. The system manages three main entities:

We have 2 main entities : rooms and users. Your task is to complete the functions defined below, and create the Room and User entities.

User entity :

- A User is defined by its balance.

Room entity :

- A room is defined by its type and its price of booking per night.
- Room types are 3 : standard suite, junior suite and master suite.
- Two rooms can have the same type but different prices per night.

You will also need to create the Booking Entity. For this entity, Design and implement it yourself. If in your design you want the Booking entity to have relations with other entities, you are free to do so, but make sure that it respects the technical requirements.

Technical Requirements

- A User can book a room for a specific period if he has enough balance for the specified period and the room is free on that period. If the booking is successful, the user balance is updated.
- **The function setRoom(...) should not impact the previously created bookings.**
- The function setRoom(...) creates a room if it does not already exist.

- The function setUser(...) creates a user if it does not already exist.
- The function printAll(...) should print all rooms data and bookings data **both from the latest created to the oldest created**. The booking data should contain all the information about the room and user when the booking was done.
- The function printAllUsers(...) prints all user data **from the latest created to the oldest created**.
- Do not use repositories, update the ArrayLists.
- In Date checkIn, Date checkOut, consider only the year, month and day.
- Handle Exceptions whenever needed (invalid inputs,...).

Service

```
public class Service {

    ArrayList<Room> rooms;
    ArrayList<Users> users;

    void setRoom(int roomNumber, RoomType roomType, int roomPricePerNight);

    void bookRoom(int userId, int roomNumber, Date checkIn, Date checkOut);

    void printAll();

    void setUser(int userId, int balance);

    void printAllUsers();

}
```

Test Case

- Create 3 rooms :

Room 1	Room 2	Room 3
• ID : 1 • Type : standard • Price/night : 1000	• ID : 2 • Type : junior • Price/night : 2000	• ID : 3 • Type : suite • Price/night : 3000

- Create 2 users, with IDs 1 and 2 and balance 5000, 10000
- User 1 tries booking Room 2 from 30/06/2026 to 07/07/2026 (7 nights).
- User 1 tries booking Room 2 from 07/07/2026 to 30/06/2026.
- User 1 tries booking Room 1 from 07/07/2026 to 08/07/2026 (1 night).
- User 2 tries booking Room 1 from 07/07/2026 to 09/07/2026 (2 nights).
- User 2 tries booking Room 3 from 07/07/2026 to 08/07/2026 (1 night).
- setRoom(1, suite, 10000).
- Give screenshots of printAll(...) and printAllUsers(...) of the end result.

Design Questions (Bonus)

1/- Suppose we put all the functions inside the same service. Is this the recommended approach ? Please explain.

2/- In this design, we chose to have a function setRoom(..) that should not impact the previous bookings. What is another way ? What is your recommendation ? Please explain and justify.