Hotel Bookings Problem

A simple hotel booking system keeps track of the rooms in a hotel. A guest can book a room for individual nights and the booking system maintains the state of these bookings.

Guests are identified by their surname which, for the purposes of this exercise, can be considered unique.

Rooms are identified by their room number taken from an arbitrary, potentially non-sequential set of numbers. For example, a hotel might have four rooms {101, 102, 201, 203}.

The booking system may be used by a number of hotel staff at once, so must be thread-safe.

# Part 1)

Implement the following interface to provide the functionality for the booking manager. Try to keep your code simple and handle errors in a sensible fashion.

public interface BookingManager {

/\*\*

\* Return true if there is no booking for the given room on the date, \* otherwise false

\*/

**public** **boolean** isRoomAvailable(Integer room, Date date);

/\*\*

\* Add a booking for the given guest in the given room on the given date.

\*/

**public** **void** addBooking(String guest, Integer room, Date date);

}

# Example usage

Assuming a hotel with four rooms, {101, 102, 201, 203}:

BookingManager bm = // create your manager here;

Date today = java.sql.Date.*valueOf*("2012-03-28");

System.*out*.println(bm.isRoomAvailable(101, today)); // outputs true

bm.addBooking("Smith", 101, today);

System.*out*.println(bm.isRoomAvailable(101, today)); // outputs false

# Part 2)

Good news! The hotel staff love your system. However, they find it rather tedious to check the availability of each room separately.

Copy your solution to **Part 1**, and modify it to implement the method below:

/\*\*

\* Return a list of all the available room numbers for the given date

\*/

public Iterable<Integer> getAvailableRooms(Date date);