

In this question, you will implement two methods for a class `Hotel` that is part of a hotel reservation system. The hotel class uses the `Reservation` class shown below. A `Reservation` is for the person and room number specified when the `Reservation` is constructed.

```
public class Reservation
{
    public Reservation(String guestName, int roomNumber)
    {    /* implementation not shown */    }

    public int getRoomNumber()
    {    /* implementation not shown */    }

    // private data and other methods not shown.
}
```

An incomplete declaration for the `Hotel` class is shown below. Each hotel in the hotel reservation system has rooms numbered 0, 1, 2, ..., up to the last room number in the hotel. For example, a hotel with 100 rooms would have rooms numbered 0, 1, 2, ..., 99.

```
public class Hotel
{
    private Reservation[] rooms;
    // each element corresponds to a room in the hotel;
    // if rooms[index] is null, the room is empty;
    // otherwise, it contains a reference to the Reservation for that room,
    // such that rooms[index].getRoomNumber() returns index

    private List<String> waitList;
    // contains names of guests who have not yet been assigned a room because all rooms are full
    // created as an ArrayList

    // if there are any empty rooms (rooms with no reservation), then create a reservation for an empty
    // room for the specified guest and return the new Reservation;
    // otherwise, add the guest to the end of waitList and return null
    public Reservation requestRoom(String guestName)
    {    /* to be implemented in part (a) */    }

    // release the room associated with parameter res, effectively canceling the reservation;
    // if any names are stored in waitList, remove the first name and create a Reservation for this person
    // in the room reserved by res; return that Reservation;
    // if waitList is empty, mark the room specified by res as empty and return null
    // precondition: res is a valid Reservation for some room in this hotel
    public Reservation cancelAndReassign(Reservation res)
    {    /* to be implemented in part (b) */    }

    // .. constructors and other methods not shown.
}
```

- a) Write the Hotel method `requestRoom`. Method `requestRoom` attempts to reserve a room in the hotel for a given guest. If there are any empty rooms in the hotel, one of them will be assigned to the named guest and the newly created reservation is returned. If there are no empty rooms, the guest is added to the end of the waiting list and `null` is returned.

Complete the method `requestRoom` below.

```
// if there are any empty rooms (rooms with no reservation), then create a reservation for an empty room for
// the specified guest and return the new Reservation;
// otherwise, add the guest to the end of waitList and return null
public Reservation requestRoom(String guestName) {
    for (int i = 0; i < rooms.length; i++) {
        if (rooms[i] == null) {
            rooms[i] = new Reservation(guestName, i);
            return rooms[i];
        }
    }
    waitList.add(guestName);
    return null;
}
```

- b) Write the Hotel method `cancelAndReassign`. Method `cancelAndReassign` releases a previous reservation. If the waiting list for the hotel contains any names, the vacated room is reassigned to the first person at the beginning of the list. That person is then removed from the waiting list and the newly created reservation is returned. If no one is waiting, the room is marked as empty and `null` is returned.

In writing `cancelAndReassign` you may call any accessible methods in the `Reservation` and `Hotel` classes. Assume that these methods work as specified.

Complete method `cancelAndReassign` below.

```
// release the room associated with parameter res, effectively canceling the reservation;
// if any names are stored in waitList, remove the first name and create a Reservation for this person in the
// room reserved by res; return that Reservation;
// if waitList is empty, mark the room specified by res as empty and return null
// precondition: res is a valid Reservation for some room in this hotel
public Reservation cancelAndReassign(Reservation res) {
    Reservation someone = null;

    if (waitList.isEmpty()) {
        someone = new Reservation((String) waitList.get(0), res.getRoomNumber());
        waitList.remove(0);
    }

    rooms[res.getRoomNumber()] = someone;

    return someone;
}
```