🔹 TASK 01 – Add a New Hotel Role Class (25 marks)

1. Create a new Java class called `Receptionist` within the package `hotel_package`. (2 marks)
2. Ensure the class inherits from `HotelStaff` and uses the concept of inheritance. (4 marks)
3. The class should inherit common fields from `HotelStaff` (e.g., name, ID) and include additional fields: `shiftType` (e.g., morning, evening), `deskNumber`, and `languageSpoken`. (4 marks)
4. Implement a constructor that takes name and ID as parameters. (4 marks)
5. Add getter and setter methods for all the new variables. (8 marks)
6. Override `toString()` to return a readable description of the Receptionist object. (3 marks)

Answer

```
package hotel_package;


public class Receptionist extends HotelStaff {

    private String shiftType;

    private int deskNumber;

    private String languageSpoken;


    public Receptionist(String name, int staffID) {

        super(name, staffID);

    }

    public String getShiftType() { return shiftType; }

    public void setShiftType(String shiftType) { this.shiftType = shiftType; }
```

```
    public int getDeskNumber() { return deskNumber; }

    public void setDeskNumber(int deskNumber) { this.deskNumber = deskNumber; }


    public String getLanguageSpoken() { return languageSpoken; }

    public void setLanguageSpoken(String languageSpoken) { this.languageSpoken =
languageSpoken; }


    @Override

    public String toString() {

        return super.toString() + ", Shift: " + shiftType + ", Desk: " + deskNumber + ", Language: " +
languageSpoken;

    }

}
```

🔹 TASK 02 – Edit Guest Name (24 marks)

1. In the `HotelManager` interface, add an abstract method called `editGuestName()`. (2 marks)
    - Signature: `void editGuestName();`
2. Implement `editGuestName()` in the `WestminsterHotelManager` class. (2 marks)
3. Ask the user to input the Guest ID from the keyboard. (3 marks)
4. If the guest is found:
    - Print the current name and assigned room number. (4 marks)
    - Print the guest type (e.g., VIP, Regular) using class check. (3 marks)
5. Ask for the new guest name. (2 marks)

6. Save the updated name correctly. (2 marks)

7. Modify `runMenu()` in `WestminsterHotelManager` to include this new functionality. (6 marks)

Answer

```
public interface HotelManager {
    void editGuestName();
}
@Override
public void editGuestName() {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter Guest ID: ");
    String id = scanner.nextLine();

    for (Guest g : guestList) {
        if (String.valueOf(g.getGuestID()).equals(id)) {
            System.out.println("Name: " + g.getName());
            System.out.println("Room: " + g.getRoomNumber());
            System.out.println("Type: " + (g instanceof VIPGuest ? "VIP" : "Regular"));
            System.out.print("New Name: ");
            g.setName(scanner.nextLine());
            System.out.println("Updated.");
            return;
        }
    }
```

```
    System.out.println("Guest not found.");
}
```

🔹 TASK 04 – Update Table Model (10 marks)

1. Modify the `HotelTableModel.java` class to add a new column that shows the *staff id*(10 marks)

Answer:

```
private String[] columnNames = {"Name", "Surname", "Date of Birth",
"Role","staff id"};

else if(columnIndex == 4){

            temp = list.get(rowIndex).getStaffID();

        }

        return temp;

    }
```

🔹 TASK 05 – Staff Info Button Functionality (15 marks)

1. In `HotelTableGUI.java`, add an event handler to the `"Info"` button. (8 marks)
2. When clicked, the GUI should display:
    - Total number of Managers
    - Total number of HouseKeepers
    - Total number of Receptionists
      in a dialog or label area. (7 marks)

```
infoButton.addActionListener(e -> {

    int managers = 0, houseKeepers = 0, receptionists = 0;
```

```
    for (HotelStaff staff : staffList) {
        if (staff instanceof Manager) managers++;
        else if (staff instanceof HouseKeeper) houseKeepers++;
        else if (staff instanceof Receptionist) receptionists++;
    }

    JOptionPane.showMessageDialog(null, "Managers: " + managers +
        "\nHouseKeepers: " + houseKeepers +
        "\nReceptionists: " + receptionists);
});
```

TASK 05 – Staff Info Button Functionality(Get all stuff number) (15 marks)

3. When clicked, the GUI should display:
    - Total number of staff

answer

```
JButton button = new JButton("Info");


button.addActionListener(e ->{
   int totalstaff=list.size();
   StringBuilder message = new StringBuilder("Total number of staff: " +
totalstaff + "\n\n");
   JOptionPane.showMessageDialog(this, message, "Staff Info",
JOptionPane.INFORMATION_MESSAGE);
});
```

🔹 TASK 07 – Staff Removal Functionality (9 marks)

1. Add removeStaffByID() method to WestminsterHotelManager.java.

2. Update the menu system to include the new option and adjust existing option numbers

Answer

```
case 5:
   this.removeStaffbyid();


   break;


public boolean removeStaffbyid() {
   Scanner s = new Scanner(System.in);
   System.out.print("Enter the staff ID to remove: ");
   String staffID = s.nextLine();
   boolean removed = removeStaffByID(staffID);
   if (removed) {
       System.out.println("Staff member with ID '" + staffID + "' was
successfully removed.");
   } else {
       System.out.println("No staff member found with ID '" + staffID + "'.");
   }
   return removed;
}
```

TASK 08 (BONUS) – Save Guest List to File

Question:

Implement a method in `WestminsterHotelManager` to save the guest list to a text file using `FileWriter`.

```
public void saveGuestListToFile() {
   try (FileWriter writer = new FileWriter("guests.txt")) {
      for (Guest guest : guestList) {
```

```
            writer.write(guest.toString() + "\n");
        }
        System.out.println("Guest list saved to file.");
    } catch (IOException e) {
        System.out.println("Error saving guest list: " + e.getMessage());
    }
}
```

TASK 08 (BONUS) – Handle Invalid Input with Custom Exception

Question:

Create a custom exception `InvalidRoomNumberException` that is thrown if a user tries to assign a room number < 0. Catch it in your guest-adding logic.

```
public class InvalidRoomNumberException extends Exception {
    public InvalidRoomNumberException(String message) {
        super(message);
    }
}
```

```
public void assignRoom(Guest guest, int roomNumber) {
    try {
        if (roomNumber < 0) throw new InvalidRoomNumberException("Room number cannot be
negative");
        guest.setRoomNumber(roomNumber);
    } catch (InvalidRoomNumberException e) {
        System.out.println("Error: " + e.getMessage()); }}
```

Task 9(Bonus)- Print a list of guests option in cli

Answer:

```
case 2:
                this.printHotelStaffList();
                Break;
```

```
@Override
    public void guestList() {

        if (!hotelGuestList.isEmpty()) {
            for (Guest member : hotelGuestList) {
                System.out.println(member.toString());
            }
        } else {
            System.out.println("There are no guests in the system.");
        }
    }
```

All the classes

1.Guest.java

```
package com.mycompany.hotelmanagementsystem;

public class Guest{

    private String name;
    private String surname;
    private int roomNumber;
    private int nightsStayed;

    public Guest(String name, String surname) {
        this.name = name;
        this.surname = surname;
        roomNumber = 0;
        nightsStayed = 0;
    }

    // Setter and Getter methods
    public void setRoomNumber(int roomNumber) {
        this.roomNumber = roomNumber;
    }

    public void setNightsStayed(int nightsStayed) {
        this.nightsStayed = nightsStayed;
    }

    public int getRoomNumber() {
```

```
        return roomNumber;
    }

    public int getNightsStayed() {
        return nightsStayed;
    }

    public String getName() {
        return name;
    }

    public String getSurname() {
        return surname;
    }
```

2.GuestTableModel.java

```
import javax.swing.table.AbstractTableModel;
import java.util.List;

public class GuestTableModel extends AbstractTableModel {
    private final String[] columnNames = { "Name","Surname", "Room Number", "Nights Stayed"
};
    private final List<Guest> guestList;

    public GuestTableModel(List<Guest> guestList) {
        this.guestList = guestList;
    }

    @Override
    public int getRowCount() {
        return guestList.size();
    }

    @Override
    public int getColumnCount() {
        return columnNames.length;
    }

    @Override
    public Object getValueAt(int rowIndex, int columnIndex) {
        Guest guest = guestList.get(rowIndex);
        switch (columnIndex) {
```

```
            case 0: return guest.getName();
            case 1: return guest.getSurname();
            case 2: return guest.getRoomNumber();
            case 3: return guest.getNightsStayed();
            default: return null;
        }
    }

    @Override
    public String getColumnName(int column) {
        return columnNames[column];
    }
}
```

3.HotelManagerInterface

```
package com.mycompany.hotelmanagementsystem;

/**
 *
 * @author b.villarini
 */
public interface HotelManager {

    boolean runMenu(); // Run Menu - main function

    void addHotelStaff(); // Add hotel staff member

    void printHotelStaffList(); // Print hotel staff member list

    void addGuest();

    boolean removeStaffbyid();// Add hotel guest

    void runGUI(); // Run main GUI
}
```

4.HotelStaff.java

```
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;

public abstract class HotelStaff {
```

```
private String name;
private String surname;
private LocalDate dateOfBirth;
private String staffID;

public HotelStaff(String name, String surname) {
    this.name = name;
    this.surname = surname;
}

// Setter and Getter methods
public void setName(String name) {
    this.name = name;
}

public void setSurname(String surname) {
    this.surname = surname;
}

public void setDateOfBirth(LocalDate dateOfBirth) {
    this.dateOfBirth = dateOfBirth;
}

public void setStaffID(String staffID) {
    this.staffID = staffID;
}

public String getName() {
    return name;
}

public String getSurname() {
    return surname;
}

public LocalDate getDateOfBirth() {
    return dateOfBirth;
}

public String getStaffID() {
    return staffID;
}
```

```
    public String getStringDate() {
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("dd/MM/yyyy");
        return dateOfBirth.format(formatter);
    }

    @Override
    public String toString() {
        return name + " " + surname + ", ID: " + staffID + ", DOB: " + getStringDate();
    }}
```

5.HotelTableGui

```
import java.awt.BorderLayout;
import java.awt.Dimension;
import java.util.ArrayList;
import javax.swing.*;

public class HotelTableGUI extends JFrame {

    JTable staffTable;
    JTable guestTable;
    HotelTableModel staffTableModel;
    GuestTableModel guestTableModel;
    ArrayList<HotelStaff> staffList;
    ArrayList<Guest> guestList;

    public HotelTableGUI(ArrayList<HotelStaff> staffList, ArrayList<Guest> guestList) {
        this.setTitle("Hotel Management System");
        this.staffList = staffList;
        this.guestList = guestList;
        staffTableModel = new HotelTableModel(staffList);
        guestTableModel = new GuestTableModel(guestList);
        staffTable = new JTable(staffTableModel);
        guestTable = new JTable(guestTableModel);

        setBounds(20, 20, 800, 600);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        staffTable.setAutoCreateRowSorter(true);
        guestTable.setAutoCreateRowSorter(true);

        JScrollPane staffScrollPane = new JScrollPane(staffTable);
        staffScrollPane.setPreferredSize(new Dimension(380, 280));
        JScrollPane guestScrollPane = new JScrollPane(guestTable);
        guestScrollPane.setPreferredSize(new Dimension(380, 280));
```

```
        JTabbedPane tabbedPane = new JTabbedPane();
        tabbedPane.addTab("Staff", staffScrollPane);
        tabbedPane.addTab("Guests", guestScrollPane);

        JButton button = new JButton("Info");
        button.addActionListener(e -> {
            StringBuilder info = new StringBuilder("Staff List:\n");
            for (HotelStaff staff : staffList) {
                info.append(staff.toString()).append("\n");
            }
            info.append("\nGuest List:\n");
            for (Guest guest : guestList) {
                info.append(guest.toString()).append("\n");
            }
            JOptionPane.showMessageDialog(this, info.toString(), "Information",
JOptionPane.INFORMATION_MESSAGE);
        });


        add(tabbedPane, BorderLayout.CENTER);
        add(button, BorderLayout.SOUTH);
    }
}
```

6.HotelTableModel

```
import java.util.ArrayList;
import javax.swing.table.AbstractTableModel;

public class HotelTableModel extends AbstractTableModel {

    private String[] columnNames = {"Name", "Surname", "Date of Birth", "Role"};
    private ArrayList<HotelStaff> list;

    public HotelTableModel(ArrayList<HotelStaff> staffList) {
        list = staffList;
    }

    @Override
    public int getRowCount() {
        return list.size();
    }

    @Override
```

```
    public int getColumnCount() {
        return columnNames.length;
    }

    @Override
    public Object getValueAt(int rowIndex, int columnIndex) {
        Object temp = null;
        if (columnIndex == 0) {
            temp = list.get(rowIndex).getName();
        } else if (columnIndex == 1) {
            temp = list.get(rowIndex).getSurname();
        } else if (columnIndex == 2) {
            temp = list.get(rowIndex).getStringDate();
        } else if (columnIndex == 3) {
            if (list.get(rowIndex) instanceof Manager) {
                temp = "Manager";
            } else if (list.get(rowIndex) instanceof HouseKeeper) {
                temp = "HouseKeeper";
            }
        }
        return temp;
    }

    // Needed to show column names in JTable
    @Override
    public String getColumnName(int col) {
        return columnNames[col];
    }
}
```

7. westminsterHotelManager

```
package com.mycompany.hotelmanagementsystem;

/**
 *
 * @author b.villarini
 */
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;
import java.time.format.DateTimeParseException;
import java.util.ArrayList;
import java.util.Scanner;

public class WestminsterHotelManager implements HotelManager {
```

```
private ArrayList<HotelStaff> hotelStaffList;
private ArrayList<Guest> hotelGuestList;
private int staffLimit;

public WestminsterHotelManager(int maxMembersNumber) {
    hotelStaffList = new ArrayList<>();
    hotelGuestList = new ArrayList<>();
    staffLimit = maxMembersNumber;
}

// New method to remove staff by ID
public boolean removeStaffByID(String staffID) {
    for (int i = 0; i < hotelStaffList.size(); i++) {
        if (hotelStaffList.get(i).getStaffID().equals(staffID)) {
            hotelStaffList.remove(i);
            return true;
        }
    }
    return false;
}

@Override
public boolean runMenu() {
    boolean exit = false; // Exit flag
    System.out.println("\n-- HOTEL MANAGEMENT SYSTEM CONSOLE MENU--");
    System.out.println("To save and exit, press 0");
    System.out.println("To Add a new staff member, press 1");
    System.out.println("To Print the list of staff members press 2");
    System.out.println("To Add a guest, press 3");
    System.out.println("To Open GUI, press 4");
    System.out.println("To Remove staff member, press 5");

    Scanner s = new Scanner(System.in);
    int choice = s.nextInt();
    s.nextLine(); // consume newline

    switch (choice) {
        case 0:
            exit = true;
            break;
        case 1:
            this.addHotelStaff();
            break;
        case 2:
```

```
            this.printHotelStaffList();
            break;
        case 3:
            this.addGuest();
            break;
       case 4:
            this.runGUI();
            break;
        case 5:
            this.removeStaffbyid();

            break;
    }
    return exit;
}

@Override
public void addHotelStaff() {
    Scanner s = new Scanner(System.in);
    if (hotelStaffList.size() < staffLimit) {
        System.out.println("Press 1 if you want to add a Manager");
        System.out.println("Press 2 if you want to add a Housekeeper");

        int choiceStaff = s.nextInt();
        s.nextLine();

        // Common questions
        System.out.println("Enter the first name");
        String name = s.nextLine();

        System.out.println("Enter the last name");
        String surname = s.nextLine();

        System.out.println("Enter the staff ID");
        String staffID = s.nextLine();
        while(isStaffIDTaken(staffID)) {
            System.out.println("Error: Staff ID already exists. Please use a unique ID.");

            staffID=s.nextLine();

        }

        System.out.println("Enter the date of birth (dd/MM/yyyy format only!)");
        LocalDate date = null;
```

```
String dob = null;
boolean parsingSucceeds = false;
while (!parsingSucceeds) {
   dob = s.nextLine();
   DateTimeFormatter formatter = DateTimeFormatter.ofPattern("dd/MM/yyyy");
   try {
      date = LocalDate.parse(dob, formatter);
      parsingSucceeds = true; // If parsing succeeds, the format is correct
   } catch (DateTimeParseException e) {
      System.out.println("Enter the correct format. It should be dd/MM/yyyy!");
      parsingSucceeds = false;
   }
}


// Check if the staff is a manager or a guest
switch (choiceStaff) {
   case 1:
      // It is a manager
      System.out.println("Enter the license number");
      String licenseNum = s.nextLine();

      // Create a new Manager and add to the list
      Manager manager = new Manager(name, surname);
      manager.setLicenseNumber(licenseNum);
      manager.setDateOfBirth(date);
      manager.setStaffID(staffID);
      this.addStaffToList(manager);
      break;

   case 2:
      // It is a HouseKeeper
      System.out.println("Enter the years of experience");
      int yearsOfExperience = s.nextInt();
      s.nextLine();


      HouseKeeper housekeeper = new HouseKeeper(name, surname);
      housekeeper.setYearsOfExperience(yearsOfExperience);
      housekeeper.setDateOfBirth(date);
      housekeeper.setStaffID(staffID);
      this.addStaffToList(housekeeper);
      break;
```

```
        }
    } else {
        System.out.println("No more space in the system");
    }
}

public void addStaffToList(HotelStaff staff) {
    if (this.hotelStaffList.size() < staffLimit) {
        hotelStaffList.add(staff);
    } else {
        System.out.println("No more space in the list");
    }
}

@Override
public void printHotelStaffList() {
    if (!hotelStaffList.isEmpty()) {
        for (HotelStaff member : hotelStaffList) {
            System.out.println(member.toString());
        }
    } else {
        System.out.println("There are no staff members in the system.");
    }
}

 @Override
public void addGuest() {
    Scanner s = new Scanner(System.in);

    System.out.println("Enter the first name");
    String name = s.nextLine();

    System.out.println("Enter the last name");
    String surname = s.nextLine();

    int roomNum = 0;
    while (true) {
        System.out.println("Enter the room number (1-999)");
        try {
            roomNum = s.nextInt();
            if (roomNum >= 1 && roomNum <= 999) {
                break;
            } else {
                System.out.println("Room number must be between 1 and 999.");
```

```
            }
        } catch (Exception e) {
            System.out.println("Invalid input. Please enter a valid room number.");
            s.nextLine();
        }
    }
    s.nextLine();

    int nightsStayed = 0;
    while (true) {
        System.out.println("Enter the number of nights stayed (0 or more)");
        try {
            nightsStayed = s.nextInt();
            if (nightsStayed >= 0) {
                break;
            } else {
                System.out.println("Nights stayed cannot be negative.");
            }
        } catch (Exception e) {
            System.out.println("Invalid input. Please enter a valid number of nights.");
            s.nextLine();
        }
    }
    s.nextLine();

    Guest guest = new Guest(name, surname);
    guest.setRoomNumber(roomNum);
    guest.setNightsStayed(nightsStayed);
    hotelGuestList.add(guest);
}

private boolean isStaffIDTaken(String staffID) {
    for (HotelStaff staff : hotelStaffList) {
        if (staff.getStaffID().equals(staffID)) {
            return true;
        }
    }
    return false;
}

@Override
public void runGUI() {
    HotelTableGUI table = new HotelTableGUI(hotelStaffList, hotelGuestList);
    table.setVisible(true);
```

```
    }
    public boolean removeStaffbyid() {
        Scanner s = new Scanner(System.in);
        System.out.print("Enter the staff ID to remove: ");
        String staffID = s.nextLine();
        boolean removed = removeStaffByID(staffID);
        if (removed) {
            System.out.println("Staff member with ID '" + staffID + "' was successfully removed.");
        } else {
            System.out.println("No staff member found with ID '" + staffID + "'.");
        }
        return removed;
    }
}
```