**Name**: Ravi Jamanbhai Makwana

**Roll No**: 21BCP418

Div-6, G-12

**Assignment-10 (Decorator Pattern)**

**Aim:**

To implement Decorator Design Pattern using Accommodation management system.

**Code:**

**Room.java**

public interface Room{

    void allot();

}

**BasicRoom.java**

public class BasicRoom implements Room{

    @Override

    public void allot() {

        System.out.println("Added BASIC Room Features.");

    }

}

**RoomDecorator.java**

public class RoomDecorator implements Room{

    protected Room room;

    public RoomDecorator(Room r){

        this.room = r;

    }

    @Override

    public void allot() {

        this.room.allot();

    }

}

**GeneralRoom.java**

public class GeneralRoom extends RoomDecorator{

    public GeneralRoom(Room r) {

        super(r);

    }

    @Override

    public void allot(){

        //first, the super class method is called, so that features of super class get added.

        super.allot();

        System.out.println("Adding General Room Features.");

    }

}

**LuxuryRoom.java**

public class LuxuryRoom extends RoomDecorator{

    public LuxuryRoom(Room r) {

        super(r);

    }

    @Override

    public void allot(){

        //first, the super class method is called, so that features of super class get added.

        super.allot();

        System.out.println("Adding Luxury Room Features.");

    }

}

**Test.java**

public class Test {

    public static void main(String args[]){

        //creating a new General Room

        Room gr = new GeneralRoom(new BasicRoom());

        gr.allot();

        System.out.println();

        //creating a new Luxury Room

        Room lr = new LuxuryRoom(new BasicRoom());

        lr.allot();

        System.out.println();

        //creating a new General Luxury Room

        Room glr = new GeneralRoom(new LuxuryRoom(new BasicRoom()));

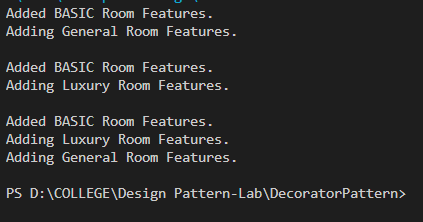
        glr.allot();

        System.out.println();

    }

}

**Output:**



**Class Diagram:**

