Set 1 – Practical Exam Question

Smart Home Appliance Control System

Problem Statement

- Develop a console-based Smart Home Appliance Control System using Core Java. The system must manage smart appliances (lights, ACs, fans, heaters), track their power usage, control their ON/OFF status, and calculate total energy consumption. The application should follow OOP principles, implement interfaces for device operations, and use arrays and collections for data handling.

Functional Requirements

1. Add appliances with ID, type, room, wattage.
2. Turn appliances ON/OFF.
3. Record daily usage hours (array of size 7 for a week).
4. Calculate weekly energy consumption (units = watts × hours).
5. Display all appliances and their usage.
6. Maintain unique room names using HashSet.
7. Generate total energy consumption for the house.

Technical Requirements

- Interfaces:
  - Controllable: turnOn(), turnOff(), getStatus()
  - Searchable: matchesId(), displayDetails()
- Classes: Appliance (base), Light/AC/Fan subclasses, ControlSystem (main).
- OOP Concepts: inheritance, interfaces, overriding, encapsulation.
- Custom Exception: ApplianceNotFoundException (optional).

Data Structures Used ArrayList for appliances, HashSet for rooms, array for weekly usage.

Expected Output

- Status of all appliances
- Weekly energy units
- Total consumption

Code:

```
import java.util.*;

interface Controllable {
    void turnOn();

    void turnOff();

    String getStatus();
}

interface Searchable {
    boolean matchesId(String id);

    void displayDetails();
}

class ApplianceNotFoundException extends Exception {
    public ApplianceNotFoundException(String msg) {
        super(msg);
    }
}

abstract class Appliance implements Controllable, Searchable {
    protected String id;
    protected String type;
    protected String room;
    protected double wattage;
    protected boolean isOn;
    protected double[] weeklyHours = new double[7];

    public Appliance(String id, String type, String room, double wattage) {
        this.id = id;
        this.type = type;
        this.room = room;
        this.wattage = wattage;
        this.isOn = false;
    }

    public void setWeeklyHours(double[] hours) {
        if (hours != null && hours.length == 7)
            for (int i = 0; i < 7; i++)
                weeklyHours[i] = hours[i];
    }

    public void setDailyHours(int dayIndex, double hours) {
        if (dayIndex >= 0 && dayIndex < 7 && hours >= 0)
```

```
            weeklyHours[dayIndex] = hours;
    }

    public double weeklyEnergy() {
        double total = 0;
        for (double h : weeklyHours)
            total += wattage * h;
        return total;
    }

    public boolean matchesId(String id) {
        return this.id.equals(id);
    }

    public void turnOn() {
        isOn = true;
    }

    public void turnOff() {
        isOn = false;
    }

    public String getStatus() {
        if (isOn == true)
            return "ON";
        else
            return "OFF";
    }

    public void displayDetails() {
        System.out.println("ID: " + id + " | Type: " + type + " | Room: " +
room + " | Wattage: " + wattage
                + "W | Status: " + getStatus());
        System.out.print("Weekly hours: ");
        for (int i = 0; i < weeklyHours.length; i++)
            System.out.print(weeklyHours[i]);
        System.out.println();
        System.out.println("Weekly energy (watt-hours): " + weeklyEnergy());
    }
}

class Light extends Appliance {
    public Light(String id, String room, double wattage) {
        super(id, "Light", room, wattage);
    }
}

class AC extends Appliance {
```

```
    public AC(String id, String room, double wattage) {
        super(id, "AC", room, wattage);
    }
}

class Fan extends Appliance {
    public Fan(String id, String room, double wattage) {
        super(id, "Fan", room, wattage);
    }
}

class Heater extends Appliance {
    public Heater(String id, String room, double wattage) {
        super(id, "Heater", room, wattage);
    }
}

public class ControlSystem {

    private List<Appliance> appliances = new ArrayList<>();
    private Set<String> rooms = new HashSet<>();
    private Scanner sc = new Scanner(System.in);

    private void addAppliance() {
        System.out.print("Enter ID: ");
        String id = sc.nextLine();
        System.out.print("Enter type (Light/AC/Fan/Heater): ");
        String type = sc.nextLine();
        System.out.print("Enter room name: ");
        String room = sc.nextLine();
        System.out.print("Enter wattage (W): ");
        double watt = Double.parseDouble(sc.nextLine());
        Appliance a;

        switch (type) {
            case "Light":
                a = new Light(id, room, watt);
                break;
            case "AC":
                a = new AC(id, room, watt);
                break;
            case "Fan":
                a = new Fan(id, room, watt);
                break;
            case "Heater":
                a = new Heater(id, room, watt);
                break;
            default:
```

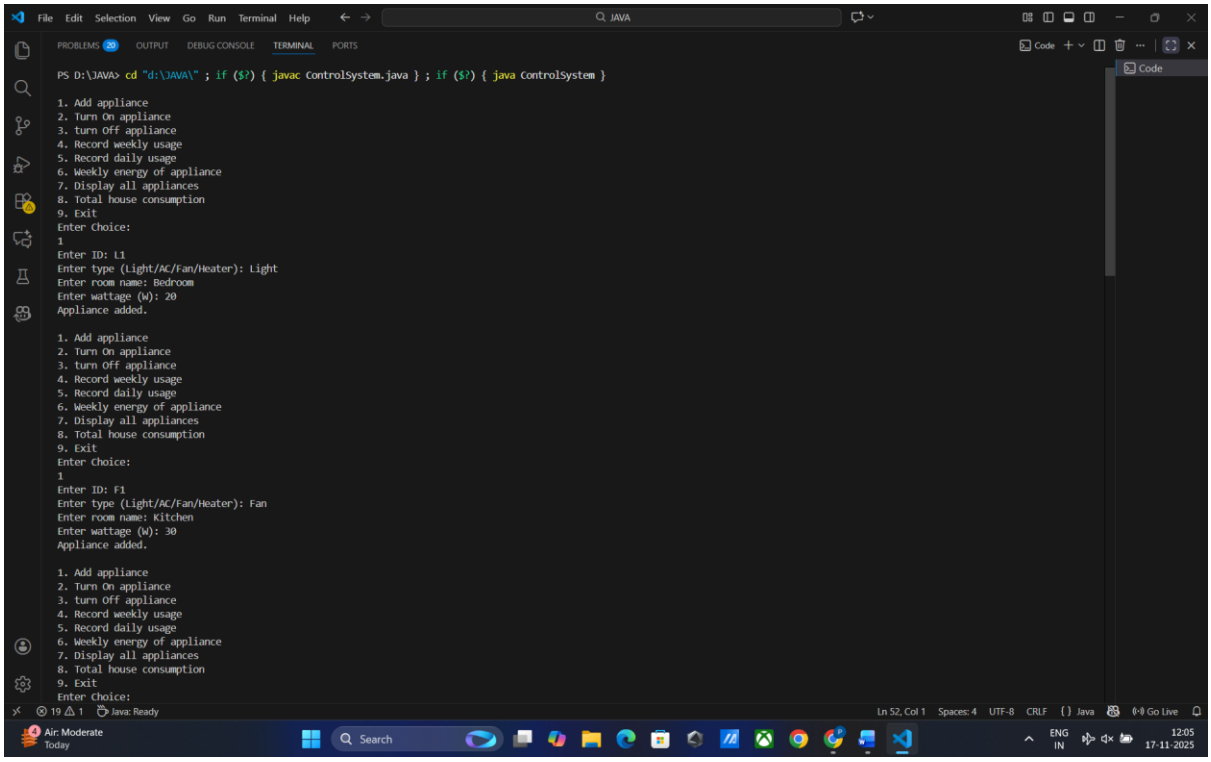
```
                System.out.println("Unknown type.");
                return;
            }

            appliances.add(a);
            rooms.add(room);
            System.out.println("Appliance added.");
        }

        Appliance findByApplianceId(String id) throws ApplianceNotFoundException {
            for (Appliance a : appliances)
                if (a.matchesId(id))
                    return a;
            throw new ApplianceNotFoundException("Appliance with ID " + id + " not
found.");
        }

        void turnAppliance(boolean on) {
            sc.nextLine();
            System.out.print("Enter appliance ID: ");
            String id = sc.nextLine();
            try {
                Appliance a = findByApplianceId(id);
                if (on)
                    a.turnOn();
                else
                    a.turnOff();
                System.out.println("Appliance " + id + " turned " + (on ? "ON" :
"OFF"));
            } catch (Exception e) {
                System.out.println(e.getMessage());
            }
        }

        void recordWeeklyUsage() {
            sc.nextLine();
            System.out.print("Enter appliance ID: ");
            String id = sc.nextLine();
            try {
                Appliance a = findByApplianceId(id);
                double[] hours = new double[7];
                for (int i = 0; i < 7; i++) {
                    System.out.print("Hours for day " + (i + 1) + ": ");
                    hours[i] = Double.parseDouble(sc.nextLine());
                }
                a.setWeeklyHours(hours);
                System.out.println("Weekly usage recorded.");
            } catch (Exception e) {
```

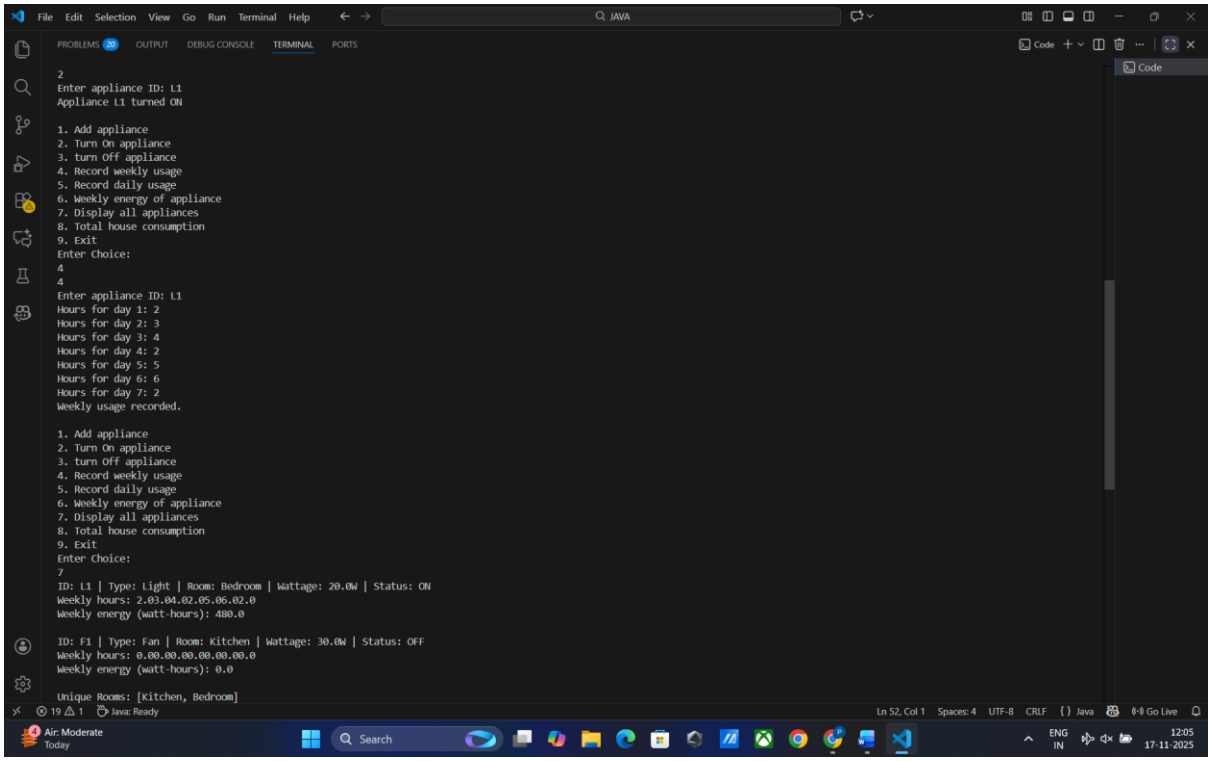
```
            System.out.println(e.getMessage());
        }
    }

    void recordDailyUsage() {
        sc.nextLine();
        System.out.print("Enter appliance ID: ");
        String id = sc.nextLine();
        try {
            Appliance a = findByApplianceId(id);
            System.out.print("Enter day (1-7): ");
            int day = Integer.parseInt(sc.nextLine()) - 1;
            System.out.print("Enter hours: ");
            double hrs = Double.parseDouble(sc.nextLine());
            a.setDailyHours(day, hrs);
            System.out.println("Daily usage updated.");
        } catch (Exception e) {
            System.out.println("Error: " + e.getMessage());
        }
    }

    void displayAll() {
        if (appliances.isEmpty()) {
            System.out.println("No appliances added.");
            return;
        }
        for (Appliance a : appliances) {
            a.displayDetails();
            System.out.println();
        }
        System.out.println("Unique Rooms: " + rooms);
    }

    void applianceWeeklyEnergy() {
        sc.nextLine();
        System.out.print("Enter appliance ID: ");
        String id = sc.nextLine();
        try {
            Appliance a = findByApplianceId(id);
            System.out.println("Weekly Energy: " + a.weeklyEnergy());
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }

    void totalConsumption() {
        double total = 0;
        for (Appliance a : appliances)
```

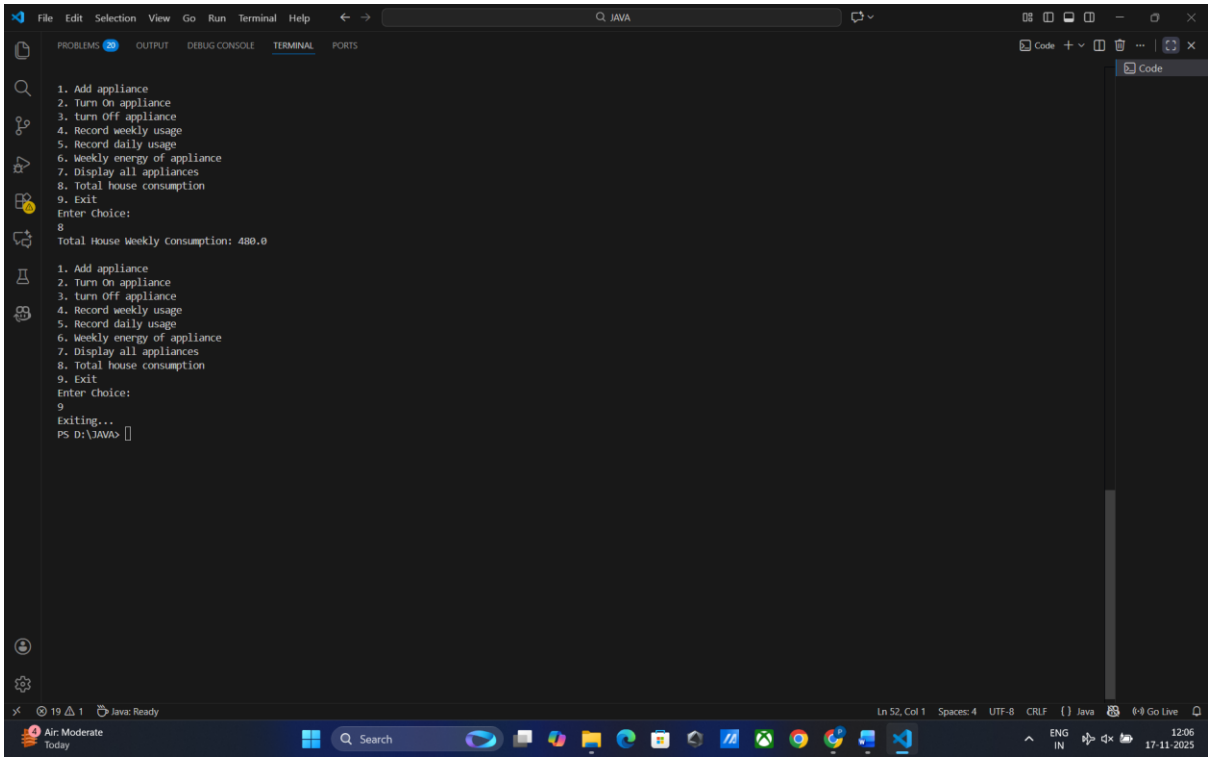
```
            total += a.weeklyEnergy();
        System.out.println("Total House Weekly Consumption: " + total);
    }

    public static void main(String[] args) {
        ControlSystem cs = new ControlSystem();
        int choice = 0;

        while (choice != 9) {
            System.out.println();
            System.out.println("1. Add appliance");
            System.out.println("2. Turn On appliance");
            System.out.println("3. turn Off appliance");
            System.out.println("4. Record weekly usage");
            System.out.println("5. Record daily usage");
            System.out.println("6. Weekly energy of appliance");
            System.out.println("7. Display all appliances");
            System.out.println("8. Total house consumption");
            System.out.println("9. Exit");
            System.out.println("Enter Choice:");

            try{
                choice = Integer.parseInt(cs.sc.nextLine().trim());
            }
            catch(Exception e) {
                choice = 0;
            }

            switch (choice) {
                case 1:
                    cs.addAppliance();
                    break;
                case 2:
                    cs.turnAppliance(true);
                    break;
                case 3:
                    cs.turnAppliance(false);
                    break;
                case 4:
                    cs.recordWeeklyUsage();
                    break;
                case 5:
                    cs.recordDailyUsage();
                    break;
                case 6:
                    cs.applianceWeeklyEnergy();
                    break;
                case 7:
```

```
                    cs.displayAll();
                    break;
                case 8:
                    cs.totalConsumption();
                    break;
                case 9:
                    System.out.println("Exiting...");
                    break;
                default:
                    System.out.println("Invalid choice. Please enter valid
choice.");
                    break;
            }
        }
    }
}
```

Output:


PS D:\JAVA> cd "d:\JAVA\" ; if ($?) { javac ControlSystem.java } ; if ($?) { java ControlSystem }
1. Add appliance
2. Turn On appliance
3. turn Off appliance
4. Record weekly usage
5. Record daily usage
6. Weekly energy of appliance
7. Display all appliances
8. Total house consumption
9. Exit
Enter Choice:
1
Enter ID: L1
Enter type (Light/AC/Fan/Heater): Light
Enter room name: Bedroom
Enter wattage (W): 20
Appliance added.
1. Add appliance
2. Turn On appliance
3. turn Off appliance
4. Record weekly usage
5. Record daily usage
6. Weekly energy of appliance
7. Display all appliances
8. Total house consumption
9. Exit
Enter Choice:
1
Enter ID: F1
Enter type (Light/AC/Fan/Heater): Fan
Enter room name: Kitchen
Enter wattage (W): 30
Appliance added.
1. Add appliance
2. Turn On appliance
3. turn Off appliance
4. Record weekly usage
5. Record daily usage
6. Weekly energy of appliance
7. Display all appliances
8. Total house consumption
9. Exit
Enter Choice:


2
Enter appliance ID: L1
Appliance L1 turned ON
1. Add appliance
2. Turn On appliance
3. turn Off appliance
4. Record weekly usage
5. Record daily usage
6. Weekly energy of appliance
7. Display all appliances
8. Total house consumption
9. Exit
Enter Choice:
4
4
Enter appliance ID: L1
Hours for day 1: 2
Hours for day 2: 3
Hours for day 3: 4
Hours for day 4: 2
Hours for day 5: 5
Hours for day 6: 6
Hours for day 7: 2
Weekly usage recorded.
1. Add appliance
2. Turn On appliance
3. turn Off appliance
4. Record weekly usage
5. Record daily usage
6. Weekly energy of appliance
7. Display all appliances
8. Total house consumption
9. Exit
Enter Choice:
7
ID: L1 | Type: Light | Room: Bedroom | Wattage: 20.0W | Status: ON
Weekly hours: 2.03.04.02.05.06.02.0
Weekly energy (watt-hours): 480.0
ID: F1 | Type: Fan | Room: Kitchen | Wattage: 30.0W | Status: OFF
Weekly hours: 0.00.00.00.00.00.00.0
Weekly energy (watt-hours): 0.0
Unique Rooms: [Kitchen, Bedroom]


File Edit Selection View Go Run Terminal Help
JAVA
PROBLEMS 20 OUTPUT DEBUG CONSOLE TERMINAL PORTS
Code
1. Add appliance
2. Turn On appliance
3. turn Off appliance
4. Record weekly usage
5. Record daily usage
6. Weekly energy of appliance
7. Display all appliances
8. Total house consumption
9. Exit
Enter Choice:
8
Total House Weekly Consumption: 480.0
1. Add appliance
2. Turn On appliance
3. turn Off appliance
4. Record weekly usage
5. Record daily usage
6. Weekly energy of appliance
7. Display all appliances
8. Total house consumption
9. Exit
Enter Choice:
9
Exiting...
PS D:\JAVA>
Java: Ready
Ln 52, Col 1 Spaces: 4 UTF-8 CRLF Java Go Live
Air: Moderate
Today
Search
ENG
IN
12:06
17-11-2025