PIMPRI CHINCHWAD EDUCATION TRUST's.

PIMPRI CHINCHWAD COLLEGE OF ENGINEERING

(An Autonomous Institute)

Class : SY BTech **Acad. Yr. 2025-26** **Semester :** I

Name of the student: Samruddhi Ramswami Bansode. **PRN :** 124B1B019

Department: Computer Engineering **Division :** A

Course Name : Data Structures

Course Code: BCE23PC02

Completion Date :

Assignment No. 10

Problem Statement:

Develop a task scheduler for smart devices (e.g., lights, AC, alarms) using a circular queue. Tasks are scheduled in time slots and repeated in a cyclic order.

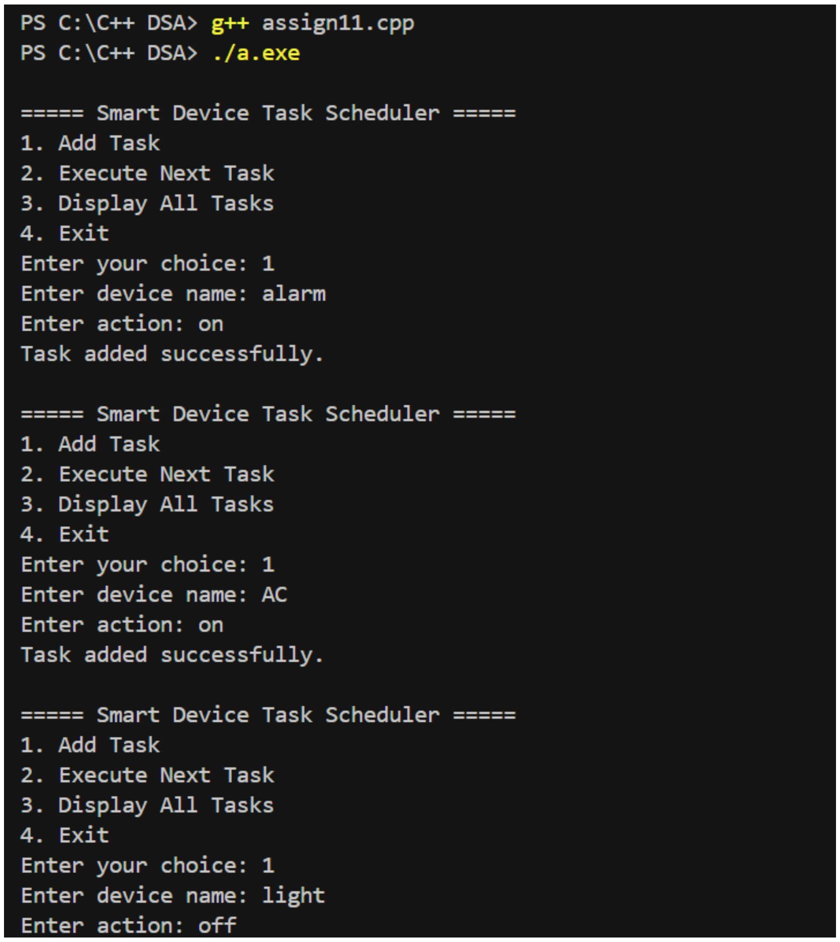
Source Code :

```
#include <iostream>
#include <string>
using namespace std;
#define SIZE 5

class Task {
public:
    string device;
    string action;
    void input() {
        cout << "Enter device name: ";
        cin >> device;
        cout << "Enter action: ";
        cin >> action;
    }
```

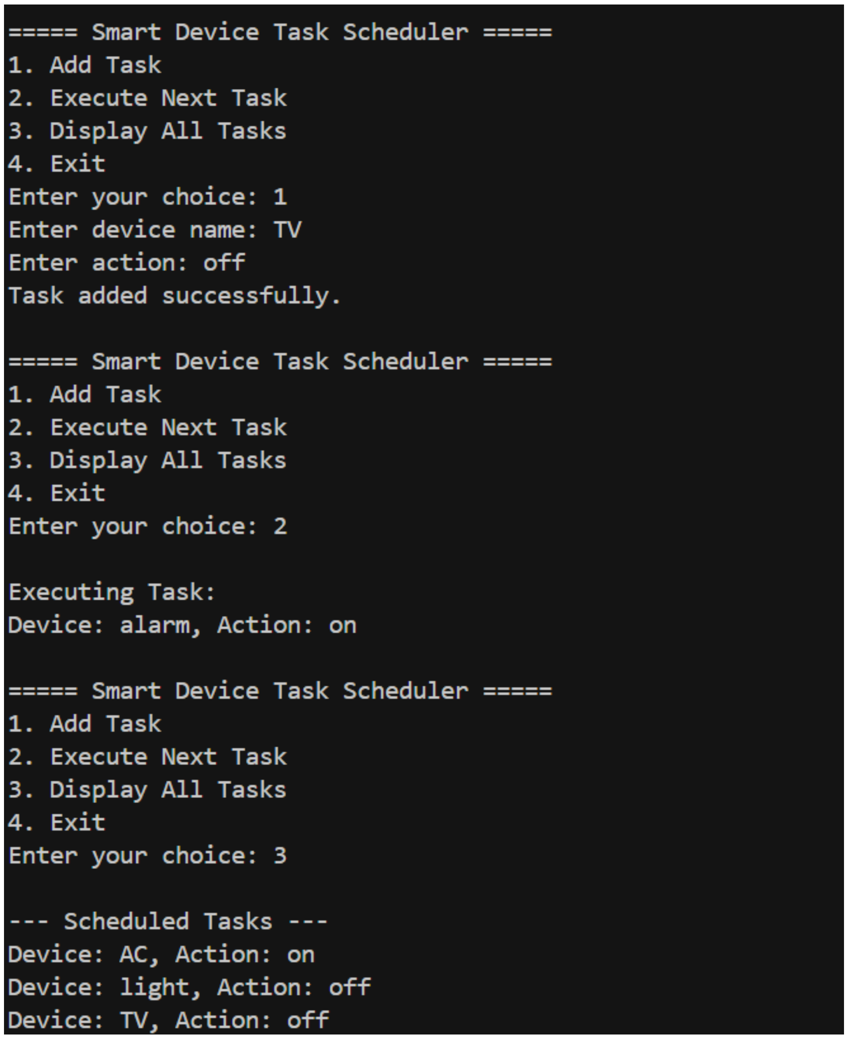
```cpp
    void display() {
        cout << "Device: " << device << ", Action: " << action << endl;
    }
};

class CircularQueue {
    Task tasks[SIZE];
    int front;
    int rear;
public:
    CircularQueue() {
        front = -1;
        rear = -1;
    }
    //insertion in queue
    void enqueue(Task t) {
        // Check if queue is full
        if ((front == 0 && rear == SIZE - 1) || (rear + 1) % SIZE == front) {
            cout << "Task queue is full! Cannot add task.\n";
            return;
        }
        if (front == -1) {
            front = rear = 0;
        } else {
            rear = (rear + 1) % SIZE;
        }
        tasks[rear] = t;
        cout << "Task added successfully.\n";
    }
    void serveTask() {
        if (front == -1) {
            cout << "No tasks to serve.\n";
            return;
        }
        cout << "\nExecuting Task:\n";
        tasks[front].display();
        // Only one task was present
        if (front == rear) {
            front = rear = -1;
        } else {
            front = (front + 1) % SIZE;
        }
    }
    void displayTasks() {
        if (front == -1) {
            cout << "No tasks in queue.\n";
            return;
        }
        cout << "\n--- Scheduled Tasks ---\n";
        int i = front;
        while (true) {
            tasks[i].display();
            if (i == rear) break;
            i = (i + 1) % SIZE;
        }
    }
};

int main() {
    CircularQueue scheduler;
    int choice;

    do {
        cout << "\n===== Smart Device Task Scheduler =====\n";
        cout << "1. Add Task\n";
        cout << "2. Execute Next Task\n";
        cout << "3. Display All Tasks\n";
        cout << "4. Exit\n";
        cout << "Enter your choice: ";
        cin >> choice;

        switch(choice) {
            case 1: {
                Task t;
                t.input();
                scheduler.enqueue(t);
                break;
            }
            case 2:
                scheduler.serveTask();
```

```
                    break;
                case 3:
                    scheduler.displayTasks();
                    break;
                case 4:
                    cout << "Exiting scheduler...\n";
                    break;
                default:
                    cout << "Invalid choice! Try again.\n";
            }
    } while(choice != 4);
    return 0;
}
```

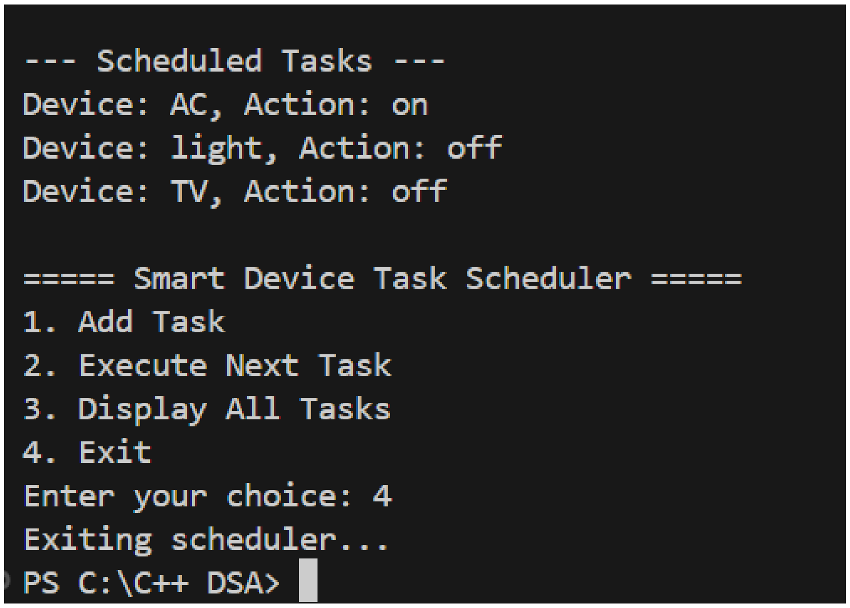
Screen Shot of Output :



```
PS C:\C++ DSA> g++ assign11.cpp
PS C:\C++ DSA> ./a.exe

===== Smart Device Task Scheduler =====
1. Add Task
2. Execute Next Task
3. Display All Tasks
4. Exit
Enter your choice: 1
Enter device name: alarm
Enter action: on
Task added successfully.

===== Smart Device Task Scheduler =====
1. Add Task
2. Execute Next Task
3. Display All Tasks
4. Exit
Enter your choice: 1
Enter device name: AC
Enter action: on
Task added successfully.

===== Smart Device Task Scheduler =====
1. Add Task
2. Execute Next Task
3. Display All Tasks
4. Exit
Enter your choice: 1
Enter device name: light
Enter action: off
```



```
===== Smart Device Task Scheduler =====
1. Add Task
2. Execute Next Task
3. Display All Tasks
4. Exit
Enter your choice: 1
Enter device name: TV
Enter action: off
Task added successfully.

===== Smart Device Task Scheduler =====
1. Add Task
2. Execute Next Task
3. Display All Tasks
4. Exit
Enter your choice: 2

Executing Task:
Device: alarm, Action: on

===== Smart Device Task Scheduler =====
1. Add Task
2. Execute Next Task
3. Display All Tasks
4. Exit
Enter your choice: 3

--- Scheduled Tasks ---
Device: AC, Action: on
Device: light, Action: off
Device: TV, Action: off
```



```
--- Scheduled Tasks ---
Device: AC, Action: on
Device: light, Action: off
Device: TV, Action: off

===== Smart Device Task Scheduler =====
1. Add Task
2. Execute Next Task
3. Display All Tasks
4. Exit
Enter your choice: 4
Exiting scheduler...
PS C:\C++ DSA>
```

Conclusion:

This project demonstrates a smart device task scheduler using a circular queue. Tasks are executed in **FIFO order** and repeated cyclically, allowing continuous scheduling. The circular queue efficiently manages tasks without shifting elements, showing a practical application of data structures in smart home automation.