Hacettepe University

Department Of Computer Engineering

BBM 103 Assignment 2 Report

Mert Can Köseoğlu – 2220356055

21.04.2023

CONTENTS

Problem:

The purpose of the system is to be able to control and manage smart home devices, including Smart Lamp, Smart Lamp with Color, Smart Plug, and Smart Camera. The settings of these devices can be adjusted by users, and they can be sorted by switch times in ascending order. Energy consumption for the Smart Plug and storage usage for the Smart Camera can be calculated by the system. The main goal is to provide a user-friendly interface for controlling and managing these devices, while following the principles of Object-Oriented Programming (OOP) and adhering to the given rules and requirements.

Good object-oriented programming principles, including the four pillars of OOP: Inheritance, Polymorphism, Abstraction, and Encapsulation, should be implemented by the system. Appropriate error messages for invalid user inputs, such as values outside of specified ranges or non-integer values, should also be provided by the system.

Solution:

The solution of the system is provided in the form of a Java program. The program includes a method called "write_output" that is used to write data into an output file. The input data is read from an input file named "input.txt" and is stored in an ArrayList of ArrayLists of Strings called "input_list". The program also contains an object of the "Time" class and an object of the "SmartDevices" class.

The first command in the input file is checked to see if it is either "SetInitialTime" or empty. If it is, the program proceeds to execute the remaining commands in the input file. Each command in the input file is processed using a series of if-else statements to

determine the appropriate action to take. These actions include setting the initial time, adding a new device, switching a device on or off, plugging in or unplugging a device, setting the time, skipping minutes, setting the kelvin, brightness, white, color code, or color of a device, removing a device, performing a ZReport, changing the name of a device, setting the switch time, or performing a NOP (no operation).

If an erroneous command is encountered, the program writes an error message to the output file. If the ZReport command is not executed, the program automatically performs a ZReport at the end. Finally, if the first command in the input file is not "SetInitialTime", the program writes an error message to the output file and terminates.

Overall, the program is designed to manage and control various smart home devices using an object-oriented programming approach and adhering to the given rules and requirements.

Benefits of The System:

The benefits of the system can be described using the passive voice as follows:

Smart home devices can be controlled and managed efficiently, allowing users to adjust their settings and sort them by switch times in ascending order.

The system calculates energy consumption for the Smart Plug and storage usage for the Smart Camera, providing users with valuable information on their device usage.

The user interface is designed to be user-friendly, making it easy for users to interact with and control their devices.

The system follows good object-oriented programming principles, including Inheritance, Polymorphism, Abstraction, and Encapsulation, which makes the code more organized and easier to maintain.

Error messages are provided for invalid user inputs, helping users to correct their mistakes and avoid potential problems.

Overall, the system provides a reliable and efficient solution for controlling and managing smart home devices, with a focus on user-friendliness and good programming practices.

Benefits of OOP:

Reusability of code is increased through encapsulation, inheritance, and polymorphism.

Maintenance and debugging are easier due to modularity and abstraction.

Collaboration between developers is facilitated through the use of class hierarchies and well-defined interfaces.

Complexity is managed through encapsulation and abstraction, leading to simpler and more organized code.

Flexibility is improved, as changes to one part of the code do not necessarily affect other parts.

Security is enhanced by controlling access to the internal data of an object through encapsulation.