Hacettepe University

Department Of Artificial Intelligence

BBM 104 Assignment 2 Report

Yusuf Emir Cömert – 2220765023

21.04.2023

Indexes

Problem:

Humankind is a lazy thing indeed, according to our evolution progress, we always tried to do something with less and less effort. Smart devices do automation and make our lives easier. The problem in this project is there are few smart devices as smart plugs, smart cameras, smart lamps, smart color lamps. These devices have features of its own such as these devices have status (on/off button), smart color lamp has color value in hexadecimal format so on…

My Approach:

The approach to this project should be starting from creating the device classes first. Then I started reading the input file line by line. You can see that there are different if, else if statements which does the job of starts with command. You can do these functions (I'll call it functions, but they are in main method) separately, but I opened the input file and applied the functions from top to bottom because if I added it separately, I could've missed something. My first function was set initial time function because it must be at the top when the program is executed. Then I created device constructors differently and started creating add function. If device might get 4,3,2 variables and the input have 2 variables, the other 2 variable would be settled default. I added the devices to the arrayLists at the end such as smartPlugs, smartCameras… I created allDevices arrayList to add these arrayLists. I also created the name of devices list such as nameOfSmartPlugs which keeps the devices name because I used it for checking if are there a same named device. These kinds of things go on till the end of the assignment. At the zReport part, I used 2 for loops using i and j indexes. At i index, I searched for allDevices list and at j index, I searched for devices.

This assignment did not seem hard but then there would be a realization time that you get the hardest things at this assignment were the errors. Every device has errors of its own. For example, brightness value should be at interval between 0 and 100 if it is not, the program executes the error part. Hexadecimal data must be between 0x0-0xFFFFFF. If this value is not in range this interval, the program must execute the required error.

Problems That I Faced and My Solutions:

I challenged with time problems a lot. Got infinitely many errors and I might solve it with strange way but code works, that's all that matters.

For example, in this project's set Time part, where the "There is nothing to change" part, Program should've return the time as yyyy-MM-dd_HH:mm:ss format but in this program it returns yyyy-MM-ddTHH:mm:ss . There is a nonsense "T" in the middle of the date. I cast this date with .toString() method and then replaced "T" with "_". Then when printing this, I figured that the seconds which ":ss" part is not visible. I gave different inputs to test it and I figured that when date ends with 00 which seconds are 0, program doesn't write it. (Writes at yyyy-MM-dd_HH:mm format) To solve this, I added :00 as string value of time if seconds are 0.

When calculating Hexadecimal string is valid or not, I put ConvertHexString method to the program which takes string parameter of HexString input. This method returns Boolean. The program is surrounded by try catches and if some error occurs, program prints the required error and returns false if and error occurs.

If input of any date would be in the wrong format, for example writing hh:m:ss instead of hh:mm:ss. To make this error not occur, I used Date Time format but appended the yyyy-M-d_H:m:s format to the date time format builder.

Set initial time command must be at the beginning of the code as you know. In the program, there is integer line count variable and program is adding one to this variable at every line ending. Program checks if this command at the beginning or not. If not, the program is going to terminate.

The other problem is when reading file, there are empty lines at the input1.txt (example input). It was the easiest problem for me because I noticed there is a .isEmpty function at java's built in functions.

Benefits Of The System:

The goal of the program is to learn Java's Object-Oriented Programming better. This system is like not so abstract. You can correlate it with real life and it was fun to code.

Benefits Of OOP:

1. Modularity: OOP enables you to break down a large program into smaller, more manageable objects that can be easily reused and modified. This modular approach makes it easier to understand, develop, and maintain code.

2. Encapsulation: OOP provides encapsulation, which means that the internal workings of an object are hidden from the outside world. This makes it easier to manage complexity and reduces the likelihood of errors.

3. Inheritance: OOP allows you to create new classes based on existing ones. This feature makes it easier to reuse code and build more complex systems.

4. Polymorphism: OOP enables you to use the same interface for different types of objects. This allows you to write more generic code that can be applied to a wider range of scenarios.

5. Code reusability: With OOP, you can easily reuse code across different parts of your program or in different programs altogether. This saves time and effort and reduces the likelihood of errors.

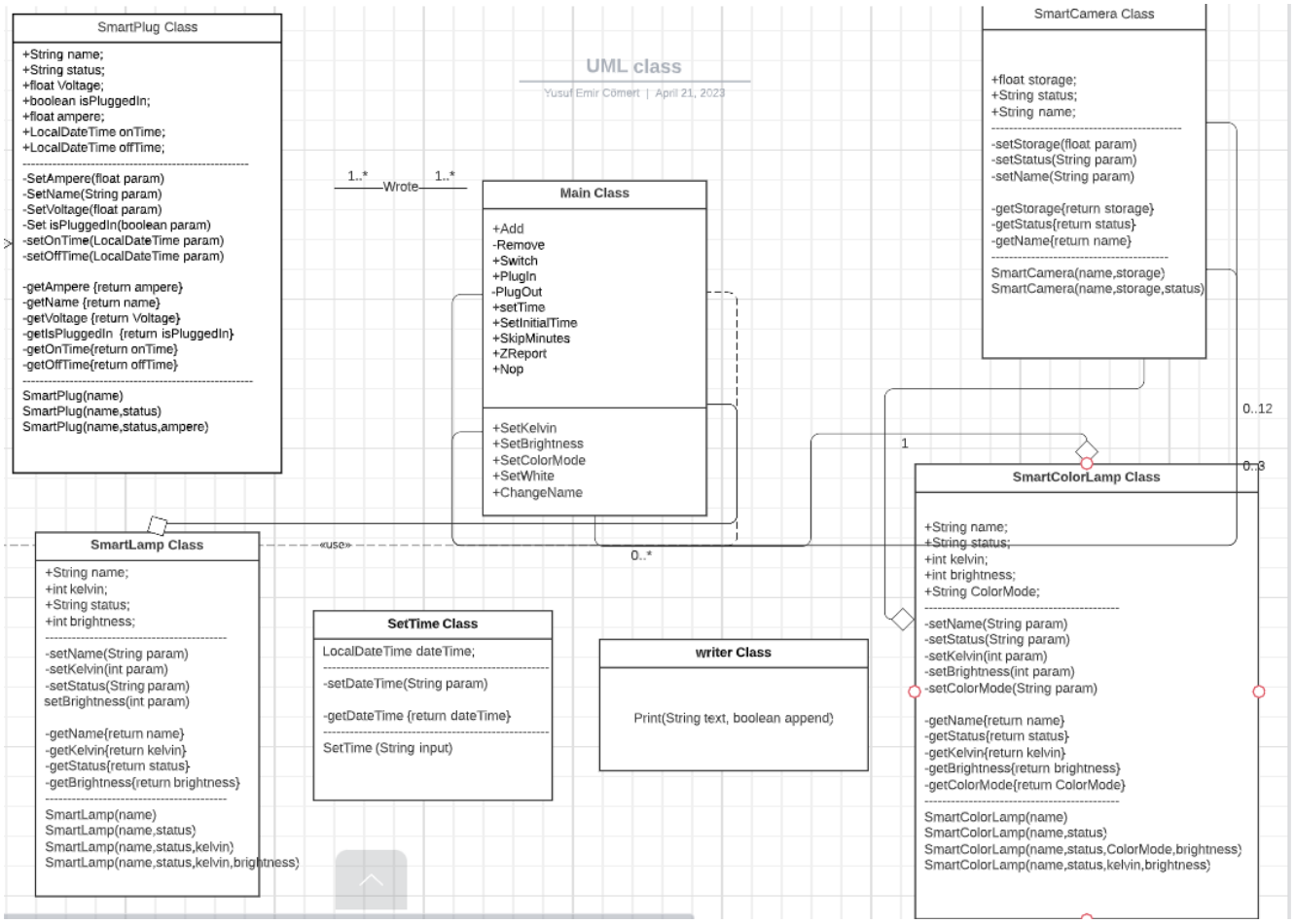
6. Improved productivity: OOP allows you to work faster and more efficiently by providing a clearer structure and more intuitive way of organizing and managing code.

Four Pillars Of OOP:

Encapsulation, Inheritance, Abstraction, Polymorphism

1. Encapsulation: Encapsulation is the practice of hiding the internal details of an object from the outside world, and instead exposing only a well-defined interface or set of behaviors. This is achieved through the use of access modifiers such as public, private, and protected, which control the visibility of an object's properties and methods.
2. Abstraction: Abstraction is the practice of representing complex real-world systems with simplified models. In OOP, this is typically achieved by defining abstract classes or interfaces that specify a set of behaviors without providing a specific implementation. Concrete classes can then implement these interfaces or inherit from abstract classes to provide specific behavior.
3. Inheritance: Inheritance is the practice of creating new classes from existing ones and inheriting their properties and behaviors. This allows you to reuse code and create more specialized classes without having to rewrite everything from scratch.
4. Polymorphism: Polymorphism is the practice of using the same interface to represent different types of objects. This allows you to write code that can work with objects of different types and is often achieved through method overriding and overloading.

UML:



References:

https://www.roberthalf.com/blog/salaries-and-skills/4-advantages-of-object-oriented-programming

https://www.geeksforgeeks.org/benefits-advantages-of-oop/

https://www.geeksforgeeks.org

https://stackoverflow.com