

Міністерство освіти і науки України
Національний університет «Львівська політехніка»
Кафедра «Електронних обчислювальних машин»



Звіт
з лабораторної роботи № 1
з дисципліни: «Кросплатформенні засоби програмування»
на тему: «Класи та пакети»

Виконав:

студент групи КІ-306

Щирба Д.В.

Перевірив:

доцент кафедри ЕОМ

Іванов Ю. С.

Мета роботи: ознайомитися з базовими конструкціями мови Java та оволодіти навиками написання й автоматичного документування простих консольних програм мовою Java.

Завдання (варіант № 28) – реалізувати клас - Лампочка

1. Написати та налагодити програму на мові Java, що реалізує у вигляді класу предметну область згідно варіанту. Програма має задовольняти наступним вимогам:
 - програма має розміщуватися в пакеті `Група.Прізвище.Lab2`;
 - клас має містити мінімум 3 поля, що є об'єктами класів, які описують складові частини предметної області;
 - клас має містити кілька конструкторів та мінімум 10 методів;
 - для тестування і демонстрації роботи розробленого класу розробити клас-драйвер;
 - методи класу мають вести протокол своєї діяльності, що записується у файл;
 - розробити механізм коректного завершення роботи з файлом (не надіятися на метод `finalize()`);
 - програма має володіти коментарями, які дозволять автоматично згенерувати документацію до розробленого пакету.
2. Автоматично згенерувати документацію до розробленої програми.
3. Завантажити код на GitHub згідно методичних вказівок по роботі з GitHub.

4. Дати відповіді на контрольні запитання:

- Синтаксис визначення класу.
- Синтаксис визначення методу.
- Синтаксис оголошення поля.
- Як оголосити та ініціалізувати константне поле?
- Які є способи ініціалізації полів?
- Синтаксис визначення конструктора.
- Синтаксис оголошення пакету.
- Як підключити до програми класи, що визначені в зовнішніх пакетах?
- В чому суть статичного імпорту пакетів?
- Які вимоги ставляться до файлів і каталогів при використанні пакетів?

Вихідний код програми:

Файл Lightbulb.java

```
package KI306.Shchyrba.Lab2;

import java.io.FileNotFoundException;
import java.io.*;

/**
 * This class represents a Lightbulb with various attributes and functionality.
 */

public class Lightbulb {

    private String model;
    private int energy_consumption;
    private boolean isOn;
    private String color;
    private PrintWriter logFile;

    public Lightbulb() throws FileNotFoundException
    {
        model = "Voltic";
        energy_consumption = 10;
        isOn = false;
        color = "white";
        logFile = new PrintWriter(new File("Lightbulb_Log.txt"));
    }

    /**
     * Constructor with parameters initializes an object with specified values.
     *
     * @param model           The model of the Lightbulb.
     * @param energy_consumption The energy consumption of the Lightbulb (in watts).
     * @param isOn            The state of Lightbulb.
     * @param color           The color which Lightbulb emits.
     */

    public Lightbulb(String model, int energy_consumption, String color) throws FileNotFoundException
    {
        this.model = model;
        this.energy_consumption = energy_consumption;
        this.isOn = false; // Лампочка за замовчуванням вимкнена
        this.color = color;
        logFile = new PrintWriter(new File("Lightbulb_Log.txt"));
    }

    /**
     * Turns on lightbulb and logs the change.
     *
     */

    public void TurnOn ()
    {
        this.isOn = true;
        logFile.println("Лампочку ввімкнено.");
        logFile.flush();
    }

    /**
     * Turns off lightbulb and logs the change.
     */
}
```

```

*
*/

public void TurnOff ()
{
    this.isOn = false;
    logFile.println("Лампочку ввимкнено.");
    logFile.flush();
}

/**
 * Changes the color of lightbulb and logs the change.
 *
 ** @param newColor New color of the lightbulb .
 */

public void changeColor(String newColor) {
    color = newColor;
    logFile.println("Колір світла змінено на " + newColor);
    logFile.flush();
}

/**
 * Changes the model of the lightbulb and logs the change.
 *
 ** @param newModel New model of the lightbulb .
 */

public void ChangeModel(String newModel) {
    model = newModel;
    logFile.println("Модель змінено на " + newModel);
    logFile.flush();
}

/**
 *Returns model of the lightbulb
 *
 *@return model
 */

public String GetModel() {
    return model;
}

/**
 *Returns energy consumption of the lightbulb
 *
 *@return energy consumption value
 */

public int GetEnergyConsumption() {
    return energy_consumption;
}

/**
 * Changes energy consumption of the lightbulb and logs the change.
 *
 ** @param newEnergyConsumption New energy consumption of the lightbulb .
 */

public void ChangeEnergyConsumption(int newEnergyConsumption) {
    energy_consumption = newEnergyConsumption;
    logFile.println("Потужність змінено на " + newEnergyConsumption + " ватт");
    logFile.flush();
}

```

```

/**
 * Defines luminous efficiency of the lightbulb and logs the stats.
 *
 * @return Luminous efficiency value
 */

public int LuminousEfficiency ()
{
    int efficiency = 10 * energy_consumption;
    logFile.println("Енергоефективність лампочки " + efficiency + " люмен/ватт");
    logFile.flush();
    return efficiency;
}

/**
 * Displays info about the current lightbulb
 */

public void GetInfo() {
    System.out.println("Модель: " + model);
    System.out.println("Потужність (ватт): " + energy_consumption);
    System.out.println("Стан: " + (isOn ? "увімкнена" : "вимкнена"));
    System.out.println("Колір світла: " + color);
}

/**
 * Closes log file.
 */

public void CloseLogFile() {
    logFile.close();
}

}

```

Файл LightbulbApp.java

```

package KI306.Shchyrba.Lab2;
import java.io.*;

public class LightbulbApp {

    public static void main(String[] args) throws FileNotFoundException {

        Lightbulb A = new Lightbulb ("Blitz", 15, "yellow");
        A.GetInfo();
        A.TurnOn();
        A.GetModel();
        A.GetEnergyConsumption();
        A.LuminousEfficiency();
        A.ChangeEnergyConsumption(20);
        A.LuminousEfficiency();
        A.ChangeModel("Lumos");
        A.TurnOff();
        A.GetInfo();
        A.CloseLogFile();

    }

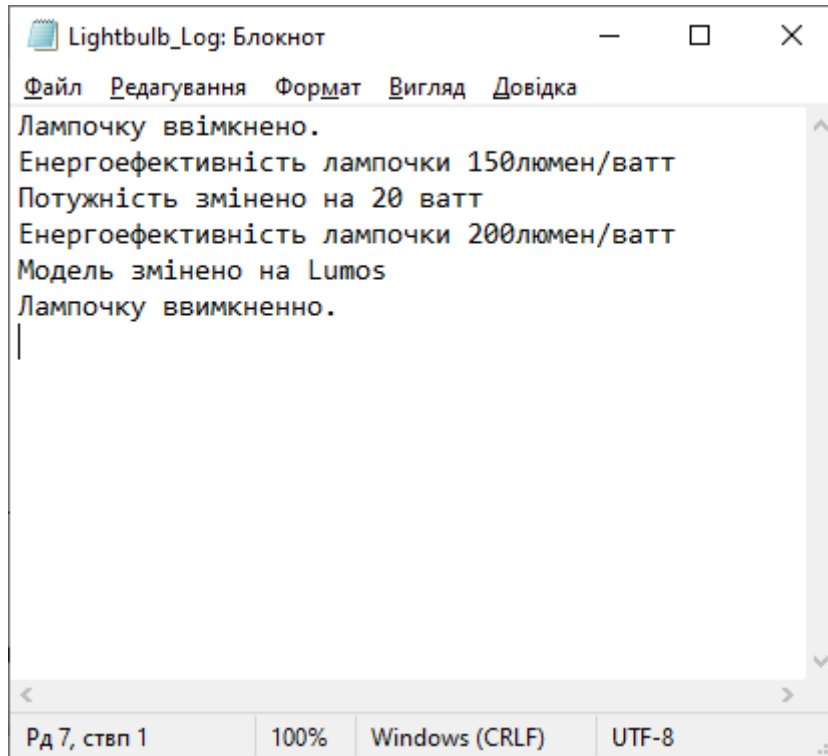
}

```

Результат виконання програми:

```
Модель: Blitz
Потужність (ватт): 15
Стан: вимкнена
Колір світла: yellow
Модель: Lumos
Потужність (ватт): 20
Стан: вимкнена
Колір світла: yellow
```

Текстовий файл з результатом виконання програми:



Фрагмент згенерованої документації:

Package KI306.Shchyrba.Lab2

Class Lightbulb

java.lang.Object[Ⓜ]
KI306.Shchyrba.Lab2.Lightbulb

```
public class Lightbulb
extends ObjectⓂ
```

This class represents a Lightbulb with various attributes and functionality.

Constructor Summary

Constructors	
Constructor	Description
<code>Lightbulb()</code>	Default constructor initializes an object with default values.
<code>Lightbulb(String[Ⓜ] model, int energy_consumption, String[Ⓜ] color)</code>	Constructor with parameters initializes an object with specified values.

Відповіді на контрольні запитання:

1. Синтаксис визначення класу.

- ```
public class ClassName {
 // Class members (fields, methods, constructors)
}
```

### 2. Синтаксис визначення методу.

- ```
public returnType methodName(parameters) {  
    // Method body  
}
```

3. Синтаксис оголошення поля.

- ```
accessModifier dataType fieldName;
```

### 4. Як оголосити та ініціалізувати константне поле?

- ```
public static final dataType CONSTANT_NAME = initial_value;
```

5. Які є способи ініціалізації полів?

- Явна ініціалізація при оголошенні поля.
- Ініціалізація у конструкторі класу.
- Ініціалізація у блоку ініціалізації (конструкторі, статичному або звичайному).

6. Синтаксис визначення конструктора.

- ```
public ClassName(parameters) {
 // Constructor body
}
```

### 7. Синтаксис оголошення пакету.

- ```
package packageName.subpackage;
```

8. Як підключити до програми класи, що визначені в зовнішніх пакетах?

- Вказати повне ім'я класу перед використанням (наприклад, `java.util.Date`
`today = new java.util.Date();`).

- Використовувати оператор `import` для підключення класів з інших пакетів, щоб уникнути повторення повного імені класу.

9. В чому суть статичного імпорту пакетів?

- Статичний імпорт дозволяє підключити статичні методи і поля класів без повного імені класу.
- Завдяки статичному імпорту, можна використовувати статичні члени класу, не додаючи перед ними ім'я класу.

10. Які вимоги ставляться до файлів і каталогів при використанні пакетів?

- Назви пакетів повинні відповідати структурі каталогів.
- Назви загальнодоступних класів повинні співпадати з назвами файлів, де вони розміщені.
- Після компіляції ієрархія каталогів проекту повинна відповідати ієрархії пакетів.
- Для компіляції та запуску програми слід використовувати шляхи до файлів та пакетів.

Висновок: У ході виконання даної лабораторної роботи, я отримав цінні навички розробки класів та пакетів у мові програмування Java. Ця лабораторна робота надала мені можливість ознайомитися з базовими конструкціями Java, такими як оголошення класів, методів та полів. Я навчився правильно структурувати свій код, визначати доступ до класів та їх членів, а також використовувати модифікатори доступу для керування видимістю.