Міністерство освіти і науки України

Національний університет „Львівська політехніка”

Кафедра ЕОМ



**Звіт**

з лабораторної роботи №2

з дисципліни: “Кросплатформні засоби програмування”

на тему: “Класи та пакети”

Виконав:

ст. гр. КІ-305

Шевчук М.О.

Прийняв:

Іванов Ю.С.

Львів – 2023

**Мета:** ознайомитися з процесом розробки класів та пакетів мовоюJava.

**Індивідуальне завдання:** написати та налагодити програму на мові Java, що реалізує у вигляді класу предметну область згідно варіанту:

**28. Лампочка**

**Хід роботи:**

1. Запустив середовище IntelliJ IDEA та написав програму згідно індивідуального завдання:

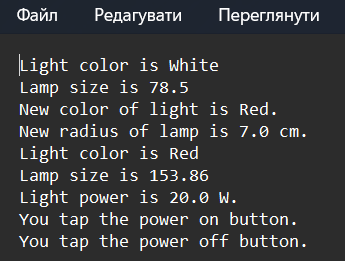
***Lab2ShevchoukKI305.java***

package pack;import java.io.FileNotFoundException; */\*\*  
 \* Class App realize example program to lab work 2  
 \*   
 \* @author Shevchouk Maxim  
 \* @version 1.1  
 \* @since version 1.0  
 \*/*public class Lab2ShevchoukKI305 { */\*\*  
 \*  
 \* @throws FileNotFoundException  
 \*/* public static void main(String[] args) throws Exception { Lamp lamp = new Lamp(23232232); lamp.getInfoLamp(); lamp.repaintLight("Red"); lamp.resizeLamp(7); lamp.getInfoLamp(); lamp.getLightPower(); lamp.powerOn(); lamp.powerOff(); }}

***Lamp.java***

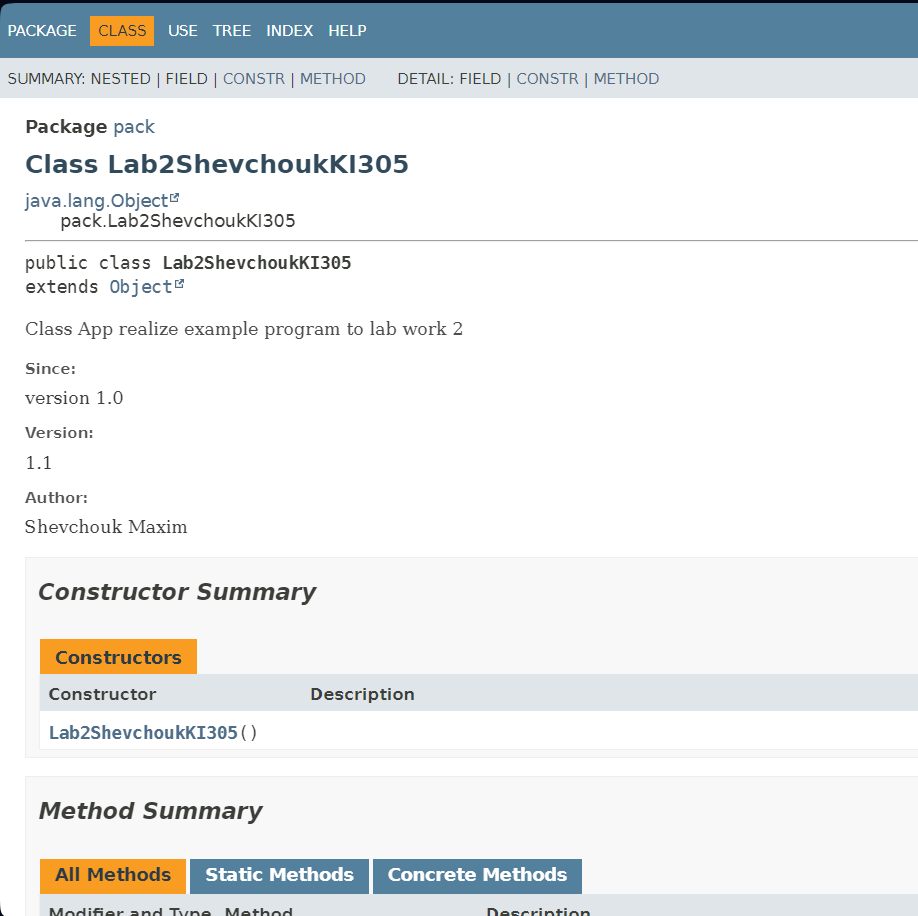
package pack;  
  
import java.io.\*;  
  
*/\*\**  
 *\* Class <code>StoveForCooking</code> implements stove for cooking*  
 *\*/*  
public class Lamp {  
 private Type TypeOfLamp;  
 private Light LightOfLamp;  
 private Button PowerOn;  
 private Button PowerOff;  
 private PrintWriter fout;  
 static int *power*;  
  
 */\*\**  
 *\* Constructor*  
 *\**   
 *\* @throws FileNotFoundException*  
 *\*/*  
public Lamp() throws FileNotFoundException {  
 TypeOfLamp = new Type();  
 LightOfLamp = new Light();  
 PowerOn = new Button();  
 PowerOff = new Button();  
 fout = new PrintWriter(new File("Log.txt"));  
 }  
  
 */\*\**  
 *\* Constructor*  
 *\**   
 *\* @param <code>resource</code> Lamp resource*  
 *\* @throws FileNotFoundException*  
 *\*/*  
public Lamp(int resource) throws FileNotFoundException {  
 TypeOfLamp = new Type();  
 LightOfLamp = new Light();  
 PowerOn = new Button(resource);  
 PowerOff = new Button(resource);  
 fout = new PrintWriter(new File("Log.txt"));  
 }  
  
  
 */\*\**  
 *\* Method change color of light lamp*  
 *\*/*  
public void repaintLight(String uCollor) {  
 TypeOfLamp.setColor(uCollor);  
 System.*out*.println("New color of light is " + uCollor + ".");  
 fout.println("New color of light is " + uCollor + ".");  
 fout.flush();  
 }  
  
 */\*\**  
 *\* Method change radius of lamp*  
 *\*/*  
public void resizeLamp(double uRadius) {  
 TypeOfLamp.setRadius(uRadius);  
 System.*out*.println("New radius of lamp is " + uRadius + " cm.");  
 fout.println("New radius of lamp is " + uRadius + " cm.");  
 fout.flush();  
 }  
  
 */\*\**  
 *\* Method provides information about the appearance of the lamp*  
 *\*/*  
public void getRadiusLamp() {  
 System.*out*.println("Radius of lanp: " + TypeOfLamp.getRadius() + " cm.");  
 fout.println("Radius of lanp: " + TypeOfLamp.getRadius() + " cm.");  
 fout.flush();  
 }  
  
 */\*\**  
 *\* Method provides information about the light power of Lamp*  
 *\*/*  
public void getLightPower() {  
 System.*out*.println("Light power is " + LightOfLamp.getPower() + " W.");  
 fout.println("Light power is " + LightOfLamp.getPower() + " W.");  
 fout.flush();  
 }  
  
 */\*\**  
 *\* Method provides information about the lamp*  
 *\*/*  
public void getInfoLamp() {  
 System.*out*.println("Light color is " + TypeOfLamp.getColor());  
 System.*out*.println("Lamp size is " + TypeOfLamp.getSize());  
 fout.println("Light color is " + TypeOfLamp.getColor());  
 fout.println("Lamp size is " + TypeOfLamp.getSize());  
 fout.flush();  
 }  
  
 */\*\**  
 *\* The method provides information about the remaining lamp life*  
 *\*/*  
public void PowerOnResource() {  
 System.*out*.println("The lamp will work for a while " + PowerOn.getResource() + " time units.");  
 fout.println("The lamp will work for a while " + PowerOn.getResource() + " time units.");  
 fout.flush();  
 }  
  
 */\*\**  
 *\* The method provides information about the remaining lamp life*  
 *\*/*  
public void PowerOffResource() {  
 System.*out*.println("The lamp will work for a while " + PowerOff.getResource() + " time units.");  
 fout.println("The lamp will work for a while " + PowerOff.getResource() + " time units.");  
 fout.flush();  
 }  
  
 */\*\**  
 *\* Method simulate the power on tap*  
 *\*/*  
public void powerOn() {  
 PowerOn.clickButton(true);  
 System.*out*.println("You tap the power on button.");  
 fout.println("You tap the power on button.");  
 fout.flush();  
 }  
  
 */\*\**  
 *\* Method simulate the power off tap*  
 *\*/*  
public void powerOff() {  
 PowerOff.clickButton(false);  
 System.*out*.println("You tap the power off button.");  
 fout.println("You tap the power off button.");  
 fout.flush();  
 }  
  
 */\*\**  
 *\* The method provides information about the number of lamp switching on and off*  
 *\*/*  
public void OnOffCount() {  
 System.*out*.println("You tap the on/off button " + PowerOn.getCountClicks());  
 fout.println("You tap the on/off button " + PowerOn.getCountClicks());  
 fout.flush();  
 }  
}  
  
  
  
  
  
  
  
*/\*\**  
 *\* Class <code>Type</code> implements type of lamp*  
 *\**   
 *\* @author Shevchouk Maxim*  
 *\* @version 1.0*  
 *\*/*  
class Type {  
 private String LightColor;  
 private double radius;  
  
 */\*\**  
 *\* Constructor*  
 *\*/*  
public Type() {  
 LightColor = "White";  
 radius = 5;  
 }  
  
 */\*\**  
 *\* Method sets new light color of lamp*  
 *\**   
 *\* @param <code>newColor</code> The color value*  
 *\*/*  
public void setColor(String newLightColor) {  
 LightColor = newLightColor;  
 }  
  
 */\*\**  
 *\* Method sets new radius of type of lamp*  
 *\**   
 *\* @param <code>newSize</code> The Size value*  
 *\*/*  
public void setRadius(double newRadius) {  
 radius = newRadius;  
 }  
  
 */\*\**  
 *\* Method gets light color of type of lamp*  
 *\*/*  
public String getColor() {  
 return LightColor;  
 }  
  
 */\*\**  
 *\* Method gets radius of type of radius in cm*  
 *\*/*  
public double getRadius() {  
 return radius;  
 }  
  
 */\*\**  
 *\* Method gets size of type of lamp in cm^2*  
 *\*/*  
public double getSize() {  
 return 3.14 \* radius \* radius;  
 }  
}  
  
  
  
  
  
  
  
*/\*\**  
 *\* Class <code>Light</code> implements Light of stove for cooking*  
 *\**   
 *\* @author Shevchouk Maxim*  
 *\* @version 1.0*  
 *\*/*  
class Light {  
 private double power;  
  
 */\*\**  
 *\* Constructor*  
 *\*/*  
public Light() {  
 power = 20;  
 }  
  
 */\*\**  
 *\* Constructor*  
 *\**   
 *\* @param <code>uPower</code> The power value*  
 *\*/*  
public Light(int uPower) {  
 power = uPower;  
 }  
  
 */\*\**  
 *\* Method returns power*  
 *\**   
 *\*/*  
public double getPower() {  
 return power;  
 }  
  
 */\*\**  
 *\* Method sets new power of lamp*  
 *\**   
 *\* @param <code>newPower</code> The power value*  
 *\*/*  
public void setPower(double newPower) {  
 power = newPower;  
 }  
}  
  
  
  
  
  
  
*/\*\**  
 *\* Class <code>Button</code> implements buttons of audio player*  
 *\**   
 *\* @author Shevchouk Maxim*  
 *\* @version 1.0*  
 *\*/*  
class Button {  
 *// remaining work resource*  
private int remainingWork, count;  
  
 */\*\**  
 *\* Constructor*  
 *\*/*  
public Button() {  
 remainingWork = 99999999;  
 count = 0;  
 }  
  
 */\*\**  
 *\* Constructor*  
 *\**   
 *\* @param <code>res</code> remaining work resource*  
 *\*/*  
public Button(int res) {  
 remainingWork = res;  
 count = 0;  
 }  
  
 */\*\**  
 *\* Method simulates Button clicking*  
 *\*/*  
public void clickButton(boolean type) {  
 if (type) {  
 remainingWork--;  
 count++;  
 } else {  
 remainingWork--;  
 count--;  
 }  
 }  
  
 */\*\**  
 *\* Method returns resource of lamp*  
 *\**   
 *\*/*  
public int getResource() {  
 return remainingWork;  
 }  
  
 */\*\**  
 *\* Method returns how many times the user clicked on the button at one time*  
 *\**   
 *\* @return Button clicks*  
 *\*/*  
public int getCountClicks() {  
 return count;  
 }  
  
}

1. Після виконання програми переглянув створений файл Log.txt:



*Рис.1. Результат виконання програми*

1. Згенерував документацію

****

*Рис.2. Згенерований html-файл*

**Контрольні питання:**

1. Синтаксис визначення класу.

public class ClassName {

// Властивості, методи і конструктори класу

}

1. Синтаксис визначення методу.

returnType methodName(parameterType parameterName) {

// Тіло методу

}

1. Синтаксис оголошення поля.

dataType fieldName;

1. Як оголосити та ініціалізувати константне поле?

public static final dataType CONSTANT\_NAME = value;

1. Які є способи ініціалізації полів?

Ініціалізація під час оголошення поля, в конструктор класу, в методах класу.

1. Синтаксис визначення конструктора.

public ClassName(parameterType parameterName) {

// Тіло конструктора

}

1. Синтаксис оголошення пакету.

package packageName;

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

Використовуйте ключове слово import для імпортування класу з іншого пакету:

import packageName.ClassName.

1. В чому суть статичного імпорту пакетів?

Статичний імпорт дозволяє користуватися статичними методами та полями класу без звертання до самого класу. Статичний імпорт дозволяє викликати статичні методи без вказівки імені класу, наприклад:

import static packageName.ClassName.staticMethod;

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

Файли класів повинні розміщуватися у відповідних пакетах, відображаючи структуру пакетів. Ім’я файлу повинно відповідати імені класу. Файли класів повинні бути збережені у відповідних каталогах, які відповідають ім’ям пакетів.

**Висновок:**

На даній лабораторній роботі ознайомився з процесом розробки пакетів та класів мовою Java.