Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра ЕОМ



Звіт

до лабораторної роботи № 4

з дисципліни: «Кросплатформні засоби програмування»

«Спадкування та інтерфейси»

Варіант - 19

Виконав:

студент групи КІ - 35

Мороз М.І.

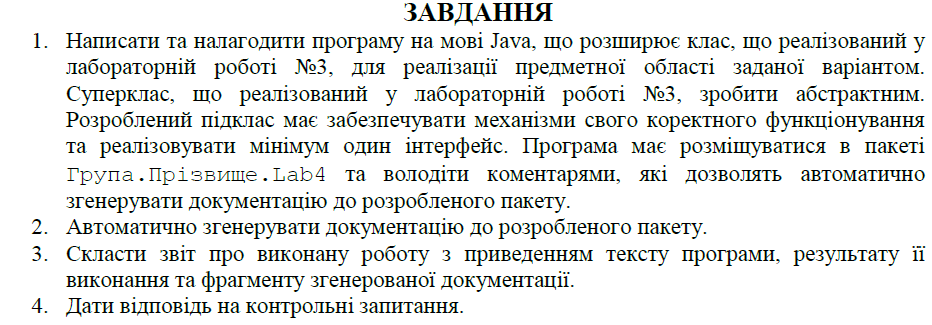
Прийняв:

доцент кафедри ЕОМ

Іванов Ю. С.

**Львів 2022**

**Мета:** ознайомитися з спадкуванням та інтерфейсами у мові Java.



**Варіант завдання:**



**Код програми:**

**File App.java**

package KI35.Moroz.lab4;  
  
*/\*\*  
 \* Class App  
 \** ***@version*** *1.0  
 \*/*public class App {  
 */\*\*  
 \** ***@param*** *args  
 \** ***@throws*** *Exception  
 \*/* public static void main(String[] args) throws Exception  
 {  
 MultifunctionalDevice device = new MultifunctionalDevice();  
  
 device.printText("Some text");  
  
 Paper paper = device.getPaper();  
  
 device.getRemainPaint();  
  
 device.getCartridgePollution();  
  
 device.addPapers(1);  
  
 device.scan(paper);  
  
 device.printScanned();  
  
 device.dispose();  
 }  
}

**File Cartridge.java**

package KI35.Moroz.lab4;  
  
*/\*\*  
 \* Class Cartridge  
 \** ***@version*** *1.0  
 \*/*public class Cartridge {  
  
 private Color color;  
 private int remainPaint = 100;  
 private int pollutionValue = 0;  
 private int MAX\_POLLUTION = 100;  
 */\*\*  
 \* Constructor  
 \** ***@param*** *color cartridge color  
 \*/* public Cartridge(Color color)  
 {  
 this.color = color;  
 }  
 */\*\*  
 \* Method prints text  
 \** ***@throws*** *Exception  
 \*/* public void print() throws Exception  
 {  
 if(MAX\_POLLUTION == pollutionValue)  
 {  
 throw new Exception("Cartridge is polluted!");  
 }  
  
 remainPaint -= 1;  
 }  
 */\*\*  
 \* Method refuels the cartridge  
 \*/* public void refuel()  
 {  
 remainPaint = 100;  
 }  
 */\*\*  
 \* Method return remain pain value  
 \*/* public int getRemainPaint()  
 {  
 return remainPaint;  
 }  
 */\*\*  
 \* Method clears the cartridge  
 \*/* public void clear()  
 {  
 pollutionValue = 0;  
 }  
 */\*\*  
 \* Method returns cartridge pollution  
 \*/* public int getPollution()  
 {  
 return pollutionValue;  
 }  
}

**File Color.java**

package KI35.Moroz.lab4;  
*/\*\*  
 \* Class Color  
 \** ***@version*** *1.0  
 \*/*public class Color {  
  
 String color;  
 */\*\*  
 \* Constructor  
 \** ***@param*** *color cartridge color  
 \** ***@throws*** *Exception  
 \*/* public Color(String color) throws Exception  
 {  
 if(!isValid(color))  
 {  
 throw new Exception("Color is invalid!");  
 }  
  
 this.color = color;  
 }  
  
 private boolean isValid(String color)  
 {  
 if(!color.equals("BLACK") && !color.equals("BLUE") && !color.equals("YELLOW") && !color.equals("RED"))  
 {  
 return false;  
 }  
 return true;  
 }  
}

**File MultifunctionalDevice.java**

package KI35.Moroz.lab4;  
  
interface IMultifunctionalDevice{  
 void scan(Paper paper);  
 void printScanned() throws Exception;  
}  
  
public class MultifunctionalDevice extends Printer implements IMultifunctionalDevice{  
  
 private String dataStorage;  
  
 */\*\*  
 \* Constructor  
 \** ***@throws*** *Exception  
 \*/* public MultifunctionalDevice() throws Exception {}  
 */\*\*  
 \* Method for scanning the paper  
 \*/* public void scan(Paper paper)  
 {  
 dataStorage = paper.getText();  
 }  
 */\*\*  
 \* Method prints the data  
 \** ***@throws*** *Exception  
 \*/* public void printScanned() throws Exception  
 {  
 super.printText(dataStorage);  
 }

**File Paper.java**

package KI35.Moroz.lab4;  
*/\*\*  
 \* Class Paper  
 \** ***@version*** *1.0  
 \*/*public class Paper {  
 private String text = "";  
 */\*\*  
 \* Method prints a text  
 \*/* public void printText(String input)  
 {  
 text = input;  
 }  
 */\*\*  
 \* Method returns a text on paper  
 \*/* public String getText()  
 {  
 return text;  
 }  
}

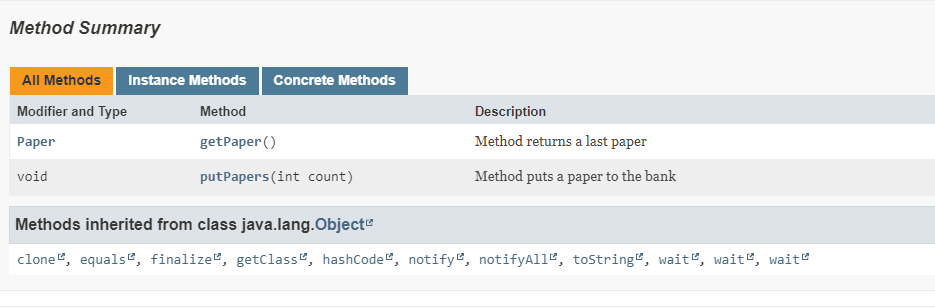
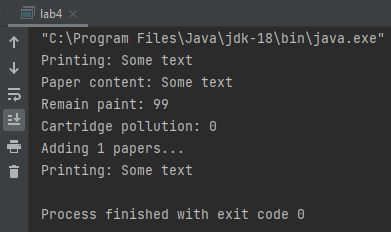
**File PaperBank.java**

package KI35.Moroz.lab4;  
  
*/\*\*  
 \* Class PaperBank  
 \** ***@version*** *1.0  
 \*/*public class PaperBank {  
  
 int papers;  
 int MAX\_COUNT = 100;  
  
 */\*\*  
 \* Constructor  
 \*/* public PaperBank()  
 {  
 this.papers = MAX\_COUNT;  
 }  
 */\*\*  
 \* Method puts a paper to the bank  
 \** ***@throws*** *Exception  
 \*/* public void putPapers(int count) throws Exception  
 {  
 if(count > MAX\_COUNT)  
 {  
 throw new Exception("Too many papers!");  
 }  
  
 this.papers = count;  
 }  
 */\*\*  
 \* Method returns a last paper  
 \** ***@throws*** *Exception  
 \*/* public Paper getPaper() throws Exception  
 {  
 if(papers == 0)  
 {  
 throw new Exception("No paper!");  
 }  
  
 papers -= 1;  
  
 return new Paper();  
 }  
  
}

**File Printer.java**

package KI35.Moroz.lab4;  
  
import java.io.File;  
import java.io.PrintWriter;  
import java.util.ArrayList;  
*/\*\*  
 \* Class App  
 \** ***@version*** *1.0  
 \*/*public abstract class Printer {  
  
 PrintWriter fout;  
 private String file = "lab4.txt";  
  
 PaperBank paperBank = new PaperBank();  
 Cartridge cartridge;  
 ArrayList<Paper> output = new ArrayList<>();  
 */\*\*  
 \* Constructor  
 \** ***@throws*** *Exception  
 \*/* public Printer() throws Exception  
 {  
 fout = new PrintWriter(new File(file));  
 cartridge = new Cartridge(new Color("BLACK"));  
 }  
 */\*\*  
 \* Method prints text  
 \** ***@param*** *text input text  
 \** ***@throws*** *Exception  
 \*/* public void printText(String text) throws Exception  
 {  
 printMessage("Printing: " + text);  
 Paper paper = paperBank.getPaper();  
 paper.printText(text);  
 cartridge.print();  
 output.add(paper);  
 }  
 */\*\*  
 \* Method returns last paper  
 \** ***@throws*** *Exception  
 \*/* public Paper getPaper() throws Exception  
 {  
 if(output.size() == 0)  
 {  
 throw new Exception("No papers in output!");  
 }  
  
 Paper paper = output.get(output.size() - 1);  
  
 printMessage("Paper content: " + paper.getText());  
 return paper;  
 }  
 */\*\*  
 \* Method adds papers to the bank  
 \** ***@param*** *count papers count  
 \** ***@throws*** *Exception  
 \*/* public void addPapers(int count) throws Exception  
 {  
  
 printMessage("Adding " + count + " papers...");  
 paperBank.putPapers(count);  
 }  
 */\*\*  
 \* Method closes the file  
 \*/* public void dispose()  
 {  
 fout.flush();  
 fout.close();  
 }  
 */\*\*  
 \* Method changes the cartridge  
 \*/* public void changeCartridge(Color color)  
 {  
 printMessage("Changing the cartridge");  
 cartridge = new Cartridge(color);  
 }  
 */\*\*  
 \* Method refuels the cartridge  
 \*/* public void refuelCartridge()  
 {  
 printMessage("Refueling the cartridge");  
 cartridge.refuel();  
 }  
 */\*\*  
 \* Method return remain pain value  
 \*/* public int getRemainPaint()  
 {  
 printMessage("Remain paint: " + cartridge.getRemainPaint());  
  
 return cartridge.getRemainPaint();  
 }  
 */\*\*  
 \* Method clears the cartridge  
 \*/* public void clearCartridge()  
 {  
 printMessage("Cleaning the cartridge");  
 cartridge.clear();  
 }  
 */\*\*  
 \* Method returns cartridge pollution  
 \*/* public int getCartridgePollution()  
 {  
 printMessage("Cartridge pollution: " + cartridge.getPollution());  
 return cartridge.getPollution();  
 }  
  
 private void printMessage(String message)  
 {  
 System.*out*.println(message);  
 fout.println(message);  
 }  
  
}

**Результат виконання програми:**



**Відповіді на контрольні запитання**

* **Синтаксис реалізації спадкування:**

class Підклас extends Суперклас

{

Додаткові поля і методи

}

* В термінах мови Java базовий клас найчастіше називається суперклассом, а похідний клас – підкласом. Дана термінологія запозичена з теорії множин, де підмножина міститься у супермножині.

**Висновок**:

На цій лабораторній роботі я ознайомився із спадкуванням та інтерфейсами у мові Java. В результаті виконання роботи, я написав та налагодив програму, що розширює клас, що реалізований у лабораторній роботі №3.