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

Звіт
з лабораторної роботи № 4
з дисципліни: «Кросплатформенні засоби програмування»
на тему: «СПАДКУВАННЯ ТА ІНТЕРФЕЙСИ »

Виконав:
студент групи КІ-35
Сабадаш Ю.А.
Прийняв:
доцент кафедри ЕОМ
Іванов Ю. С.

Львів – 2022

Мета: ознайомитися з спадкуванням та інтерфейсами у мові Java.

Завдання:

Написати та налагодити програму на мові Java, що розширює клас, що реалізований у лабораторній роботі №3, для реалізації предметної області заданої варіантом. Суперклас, що реалізований у лабораторній роботі №3, зробити абстрактним. Розроблений підклас має забезпечувати механізми свого коректного функціонування та реалізовувати мінімум один інтерфейс. Програма має розміщуватися в пакеті Група.Прізвище.Lab4 та володіти коментарями, які дозволять автоматично згенерувати документацію до розробленого пакету.

8. Цифрова відеокамера

Текст програми

Lab4.java

```
package Camera_package;

import java.io.FileNotFoundException;

public class Lab4 {

    public static void main (String [] args) throws FileNotFoundException {
        DigitalCamera dcamera1 = new
DigitalCamera(7.2,50,false,150.3,2.1,10.4,false);
        dcamera1.allInfo();
        System.out.println("\n\n");
        dcamera1.changeAccessories();
        dcamera1.SeatingOfTheRecording();
        dcamera1.DigitalCameraStartRecording();
        dcamera1.increaseLight(4);
        dcamera1.cameraCleanLens();
        System.out.println("\n\n");
        dcamera1.allInfo();
    }
}
```

DigitalCamera.java

```
package Camera_package;

import java.io.FileNotFoundException;
import java.io.FileOutputStream;
/**
 * Class <code>DigitalCamera</code>
 * @author  Sabadash Yurii KI-35
 * @version 1.0
 */

public class DigitalCamera extends Camera implements Seating{
```

```
    private double lightFilter;
    private boolean accessories;
    /**
     * Constructor
     * @throws FileNotFoundException
     * @param<code>accessories</code>
     * @param<code>lightFilter</code>
     */

    DigitalCamera() throws FileNotFoundException {
        this.accessories=false;
        this.lightFilter=0.5;
    }
    /**
     * Another Constructor
     * @throws FileNotFoundException
     * @param<code>valueZoom</code> value of zoom
     * @param<code>valueMegapixel</code> the number of megapixels
     * @param<code>lensStatus</code> lens status
     * @param<code>valuePrice</code> price of the camera
     * @param<code>valueWeight</code> weight of the camera
     * @param<code>valueLightFilter</code> valueLightFilter of filter
     * @param<code>valueAssessor</code> availability of the Assessors
     */

    DigitalCamera(double valueZoom, int valueMegapixel, boolean lensStatus,
double valuePrice, double valueWeight, double valueLightFilter, boolean
valueAssessor) throws FileNotFoundException {
        super(valueZoom,valueMegapixel,lensStatus,valuePrice,valueWeight);
        this.lightFilter=valueLightFilter;
        this.accessories=valueAssessor;
    }

    public double getLightFilter() {
        return this.lightFilter;
    }
    public boolean getAccessories(){
        return this.accessories;
    }
    public void setAccessories(boolean accessories) {
        this.accessories = accessories;
    }

    public void setLightFilter(double lightFilter) {
        this.lightFilter = lightFilter;
    }
    /**
     * method check availability of the Assessors
     */

    public void changeAccessories(){
    if (this.accessories){
        System.out.println("Now you have accessories ");
        fout.println("Now you have accessories");
        fout.flush();
    }
    else{
        this.accessories=true;
        System.out.println("You get an accessories");
        fout.println("You get an accessories");
        fout.flush();
    }
}
public void increaseLight(double valueLight){
```

```
        if (valueLight<0){
            System.out.println("You enter wrong number");
            fout.println("You enter wrong number");
            fout.flush();
        }
        else{
            this.lightFilter+=valueLight;
            System.out.println("You increase your light filter");
            fout.println("You increase your light filter");
            fout.flush();
        }

}
public void reduseLight (double valueLight){
        if (valueLight>this.lightFilter){
            System.out.println("You enter wrong number");
            fout.println("You enter wrong number");
            fout.flush();
        }
        else{
            this.lightFilter-=valueLight;
            System.out.println("You reduse your light filter");
            fout.println("You reduse your light filter");
            fout.flush();
        }
}

    public void DigitalCameraStartRecording() {
        System.out.println("Now you start recording");
        if(lens1.getMegapixel()>15){
            System.out.println("Now you recording 720p, but you can change
it");
        }
        else{
            System.out.println("You don't have opportunity change your
seating of the recording");
        }
    }

    public void SeatingOfTheRecording() {
        System.out.println("You change your seating of recording and now you
recording 1080p");
        this.lightFilter++;
        System.out.println("You change light filter");
        fout.print("You change your seating of recording and now you
recording 1080p\n");

    }
    public void allInfo(){
        System.out.println("Price is "+ charact1.getPrice() + "$");
        System.out.println("Weight is "+ charact1.getWeight() + "Kg");
        System.out.println("Camera has "+ lens1.getMegapixel() +"
megapixels");
        System.out.println("Camera status "+ lens1.getLensStatus() +" (If
true - you have clean lens, else your lens is untidy");
        System.out.println("Zoom of camera is "+ zoom1.getZoomValue());
        System.out.println("Light Filter is " + getLightFilter());
        System.out.println("You have accessions " + getAccessories());
        fout.println("You look all info");
        fout.flush();
    }
}
```

Camera.java

```
package Camera_package;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.PrintWriter;
/**
 * Class <code>Camera</code> all information about camera
 * @author  Sabadash Yurii
 * @version 1.0
 */

public abstract class Camera {
    protected Zoom zoom1;
    protected Lens lens1;
    protected OtherCharact charact1;
    protected PrintWriter fout;

    Camera() throws FileNotFoundException {
        zoom1 =new Zoom();
        lens1 = new Lens();
        charact1 = new OtherCharact();
        fout = new PrintWriter(new File("Result.txt"));
    }

    Camera(double valueZoom, int valueMegapixel, boolean lensStatus, double
valuePrice, double valueWeight) throws FileNotFoundException{
        zoom1 = new Zoom(valueZoom);
        lens1 = new Lens(valueMegapixel,lensStatus);
        charact1 = new OtherCharact(valuePrice, valueWeight);
        fout = new PrintWriter(new File("Result.txt"));
    }

    public void cameraSetZoom(double valueZoom){
        zoom1.setZoomValue(valueZoom);
        fout.print("You change Zoom Value\n");
        fout.flush();
    }
    public void cameraIncreaseZoom(double valueZoom){
        zoom1.increaseZoom(valueZoom);
        System.out.println("Now Zoom is "+ zoom1.getZoomValue());
        fout.print("You change Zoom Value\n");
        fout.flush();
    }
    public void cameraReduseZoom(double valueZoom){
        zoom1.reduseZoom(valueZoom);
        System.out.println("Now Zoom is "+ zoom1.getZoomValue());
        fout.print("You change Zoom Value\n");
        fout.flush();
    }

    public void cameraGetDiscount(){
        charact1.getDiscount();
        fout.println("You get discount");
        fout.flush();
    }

    public void cameraRecommendation(){
        charact1.recommendation();
    }
    public void cameraSetMegapixel(int valueMegapixel){
        lens1.setMegapixel(valueMegapixel);
```

```
    }
    public void cameraStartRecording(){
        lens1.startRecording();
        fout.print("Camera start recording\n");
        fout.flush();
    }
    public void cameraCleanLens(){
        lens1.cleanLens();
        fout.print("Now you have tidy lens\n");
        fout.flush();
    }
    public void cameraInfoLens(){
        lens1.showInfoAboutLens();
        fout.println("You look info");
        fout.flush();
    }

    public void allInfo(){
        System.out.println("Price is "+ charact1.getPrice() + "$");
        System.out.println("Weight is "+ charact1.getWeight() + "Kg");
        System.out.println("Camera has "+ lens1.getMegapixel() +"
megapixels");
        System.out.println("Camera status "+ lens1.getLensStatus() +" (If
true - you have clean lens, else your lens is untidy");
        System.out.println("Zoom of camera is "+ zoom1.getZoomValue());
        fout.println("You look all info");
        fout.flush();
    }
}
```

Lens.java

```
package Camera_package;

import javax.swing.*;
/**
 * Class <code>Lens</code> counts the number of megapixels
 * @author  Sabadash Yurii KI-35
 * @version 1.0
 */

public class Lens {
    private int megapixel;
    private boolean lensStatus;
    /**
     * Constructor
     */

    Lens(){
        this.megapixel=5;
        this.lensStatus=true;
    }
    /**
     *  Another Constructor
     */

    Lens(int megapixel, boolean lensStatus){
        this.megapixel=megapixel;
        this.lensStatus=lensStatus;
    }
    /**
     * Method returns Megapixel
```

```
     * @return Megapixel
     */

    public int getMegapixel(){
        return this.megapixel;
    }
    public void setMegapixel(int valueMegapixel){
        if (valueMegapixel<0){
            System.out.println("Megapixel cannot be less than 0");
        }else{
            this.megapixel=valueMegapixel;
        }
    }
    /**
     * Method returns LensStatus
     * @return LensStatus
     */
    public boolean getLensStatus(){
        return lensStatus;
    }
    public void setLensStatus(boolean valueLensStatus){
        this.lensStatus=valueLensStatus;
    }
    public void startRecording(){
        if (megapixel<10){
            System.out.println("You start recording 720p ");
        }
        else{
            System.out.println("You start recording 1080p ");
        }
    }

    public void cleanLens(){ // продебажити
        if (lensStatus){
            System.out.println("You have tidy lens ");
        }
        else{
            lensStatus=true;
            System.out.println("You clean your lens ");
        }
    }

    public void showInfoAboutLens(){
        System.out.println("Your lens have " +getMegapixel() + " megapixel");
        System.out.println("Your lens status " +getLensStatus() + " (If true
- you have clean lens, else your lens is untidy");
    }
}
```

Location.java

```
package Camera_package;

import javax.swing.plaf.synth.SynthOptionPaneUI;
/**
 * Class <code>OtherCharact</code> basic information about Camera in general
 * @author  Sabadash Yurii KI-35
 * @version 1.0
 */
public class OtherCharact {
    private double price;
```

```
    private double weight;

    OtherCharact(){
        this.price=100;
        this.weight= 1.0;
    }
    OtherCharact(double price, double weight){
        this.price=price;
        this.weight=weight;
    }
    public double getWeight() {
        return weight;
    }
    public double getPrice() {
        return price;
    }

    public void setPrice(int price) {
        this.price = price;
    }
    public void setWeight(double weight) {
        this.weight = weight;
    }

    public void getDiscount(){
        if (price<1000){
            System.out.println("You don't get a discount");
        }
        else{
            double discount=this.price/10;
            this.price-=discount;
        }
    }
    public void recommendation(){
        if(this.weight>1){
            System.out.println("You must buy case");
        }
        else{
            System.out.println("You must buy strap");
        }
    }
}
```

OtherCharacter.java

```
package Camera_package;

import javax.swing.*;
/**
 * Class <code>Lens</code>
 * @author  Sabadash Yurii KI-35
 * @version 1.0
 */

public class Lens {
    private int megapixel;
    private boolean lensStatus;
    /**
     * Constructor
     */

    Lens(){
```

```
        this.megapixel=5;
        this.lensStatus=true;
    }
    /**
     *  Another Constructor
     */

    Lens(int megapixel, boolean lensStatus){
        this.megapixel=megapixel;
        this.lensStatus=lensStatus;
    }
    /**
     * Method returns Megapixel
     * @return Megapixel
     */

    public int getMegapixel(){
        return this.megapixel;
    }
    public void setMegapixel(int valueMegapixel){
        if (valueMegapixel<0){
            System.out.println("Megapixel cannot be less than 0");
        }else{
            this.megapixel=valueMegapixel;
        }
    }
    /**
     * Method returns LensStatus
     * @return LensStatus
     */
    public boolean getLensStatus(){
        return lensStatus;
    }
    public void setLensStatus(boolean valueLensStatus){
        this.lensStatus=valueLensStatus;
    }
    public void startRecording(){
        if (megapixel<10){
            System.out.println("You start recording 720p ");
        }
        else{
            System.out.println("You start recording 1080p ");
        }
    }

    public void cleanLens(){ // продебажити
        if (lensStatus){
            System.out.println("You have tidy lens ");
        }
        else{
            lensStatus=true;
            System.out.println("You clean your lens ");
        }
    }

    public void showInfoAboutLens(){
        System.out.println("Your lens have " +getMegapixel() + " megapixel");
        System.out.println("Your lens status " +getLensStatus() + " (If true
- you have clean lens, else your lens is untidy");
    }
}
```

Zoom.java

```
package Camera_package;

public class Zoom {
    /**
     * Class <code>Engine</code> implements the operation of the Camera zoom
     * @author  Sabadash Yurii KI-35
     * @version 1.0
     */

    private double zoomValue;
    Zoom(){
        this.zoomValue=0;
    }
    Zoom(double zoomValue){
        this.zoomValue=zoomValue;
    }

    public double getZoomValue() {
        return zoomValue;
    }
    public void setZoomValue(double zoomValue){
        this.zoomValue=zoomValue;
    }

    public void increaseZoom(double valueZoom){
        if (valueZoom<0){
            System.out.println("You enter wrong value");
        }
        else{
            this.zoomValue+=valueZoom;
            System.out.println("Zoom ++");
        }
    }
    public void reduseZoom(double valueZoom){
        if (valueZoom>this.zoomValue){
            System.out.println("You enter wrong value");
        }
        else{
            this.zoomValue-=valueZoom;
            System.out.println("Zoom --");
        }
    }


}
```

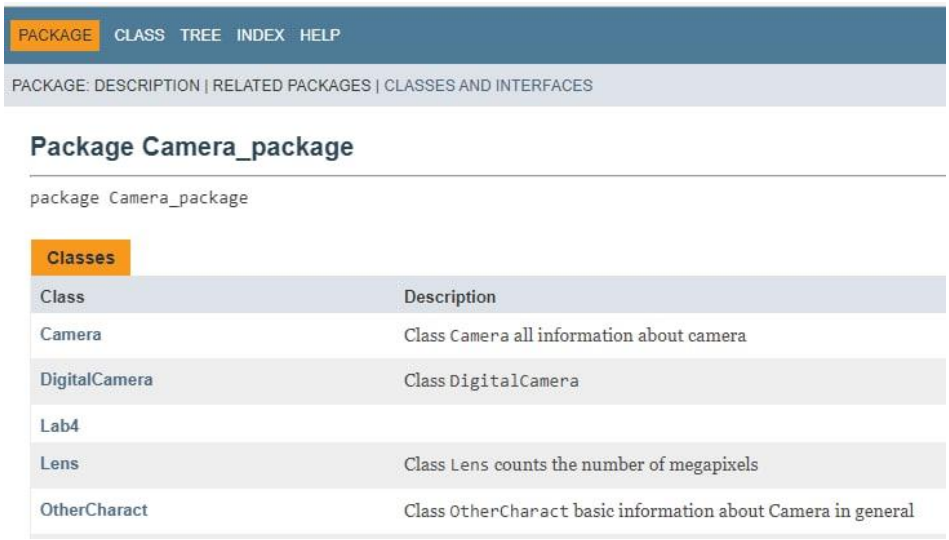
Interface.java

```
package Camera_package;

interface DigitalRecording{
    void DigitalCameraStartRecording();
}
interface Seating extends DigitalRecording{
    void SeatingOfTheRecording();
}
```

Результат роботи програми

```
C:\Users\Yurii\.jdks\corretto-17.0.4.1\bin\java.exe "-javaagent:C:\Program File
Price is 150.3$
Weight is 2.1Kg
Camera has 50 megapixels
Camera status false (If true - you have clean lens, else your lens is untidy
Zoom of camera is 7.2
Light Filter is 10.4
You have accessions false



You get an accessories
You change your seating of recording and now you recording 1080p
You change light filter
Now you start recording
Now you recording 720p, but you can change it
You increase your light filter
You clean your lens



Price is 150.3$
Weight is 2.1Kg
Camera has 50 megapixels
Camera status true (If true - you have clean lens, else your lens is untidy
Zoom of camera is 7.2
Light Filter is 15.4
You have accessions true
```

Фрагмент згенерованої документації



PACKAGE CLASS TREE INDEX HELP

PACKAGE: DESCRIPTION | RELATED PACKAGES | CLASSES AND INTERFACES

Package Camera_package

package Camera_package

Classes

Class	Description
Camera	Class Camera all information about camera
DigitalCamera	Class DigitalCamera
Lab4	
Lens	Class Lens counts the number of megapixels
OtherCharact	Class OtherCharact basic information about Camera in general

Відповіді на контрольні запитання:

1. Що таке абстрактний клас та як його реалізувати?

Абстрактні класи призначені бути основою для розробки ієрархій класів та не дозволяють створювати об’єкти свого класу. Вони реалізуються за допомогою ключового слова abstract.

2. Що таке інтерфейс?

Інтерфейси вказують що повинен робити клас не вказуючи як саме він це повинен робити. Інтерфейси покликані компенсувати відсутність множинного спадкування у мові Java та гарантують визначення у класах оголошених у собі прототипів методів

Висновок:на цій лабораторній роботі я ознайомився з спадкуванням та інтерфейсами у мові Java.