

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 - 26

Виконав:
студент групи КІ-34
Карплюк Ю.Б.
Прийняв:
Іванов Ю.С.

Львів 2022

Мета: ознайомитися з спадкуванням та інтерфейсами у мові Java.

ЗАВДАННЯ

1. 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.
2. Автоматично згенерувати документацію до розробленого пакету.
3. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації.
4. Дати відповідь на контрольні запитання.

Варіант завдання:

7. Ноутбук

Код програми:

File Laptop.java

```
package KI34.Karpluk.Lab4;

import java.io.FileNotFoundException;

/**
 * Class <code>Laptop</code> extends class Computer
 *
 * @author Yurii Karpluk
 * @version 1.0
 */
// оголошуємо клас, що реалізує інтерфейси Chargeable, Cursor, Displayable та
// наслідується від абстрактного класу Computer
public class Laptop extends Computer implements Chargeable, Cursor, Displayable {
    // Laptop battery capacity value
    private int batteryCapacity;

    /**
     * Constructor
     *
     * @param
     * <code>powerValue</code> PowerSupply power value
     * @throws FileNotFoundException
     */
    public Laptop(int powerValue) throws FileNotFoundException {
        super(powerValue);
        batteryCapacity=(int) ((Math.random() * (100 - 1)) + 1);
    }

    /**
     * Method simulates switching on laptop
     */
}
```

```

@Override
public void switchOn() {
    super.getPowerSupply().turnOn();
    super.getProcessor().startProcessorWorking();
    super.getOs().startOS();
    System.out.println("Laptop is switched on!");
    super.getFout().print("Laptop is switched on!" + "\n");
    super.getFout().print("Power supply is turned on! Status: " +
super.getPowerSupply().isTurnedOn() + "\n");
    super.getFout().print("Processor start's working : " +
super.getProcessor().getUsagePercentage() + "\n");
    super.getFout().print("OS start's working! Status: " +
super.getOs().isStarted() + "\n");
    super.getFout().flush();
}

/**
 * Method simulates switching off laptop
 */
@Override
public void switchOff() {
    super.getPowerSupply().turnOff();
    super.getProcessor().finishProcessorWorking();
    super.getOs().finishOS();
    System.out.println("Laptop is switched off!");
    super.getFout().print("Laptop is switched off! " + "\n");
    super.getFout().print("Power supply is turned off! Status: " +
super.getPowerSupply().isTurnedOn() + "\n");
    super.getFout().print("Processor finishes working : " +
super.getProcessor().getUsagePercentage() + "\n");
    super.getFout().print("OS finishes working! Status:" +
super.getOs().isStarted() + "\n");
    super.getFout().flush();
}

/**
 * Method simulates charging laptop
 */
@Override
public void charge() {
    batteryCapacity+=(int) ((Math.random() * (5 - 1)) + 1);
    System.out.println("Start charging battery! Capacity
is:"+batteryCapacity);
}

/**
 * Method simulates moving cursor
 */
@Override
public void moveCursor(int x, int y) {
    System.out.println("Cursor moved to coordinate: (" +x+";" +y+"");
}

/**
 * Method simulates displaying content on the laptop screen
 */
@Override
public void displayOnScreen(String content) {
    System.out.println("Information displayed on screen
successfully:"+"\n"+content);
}
}

```

File LaptopApplication.java

```

package KI34.Karpluk.Lab4;

import java.io.FileNotFoundException;

```

```

/**
 * Laptop Application class implements main method for Laptop
 * class possibilities demonstration
 *
 * @author Yuriy Karpliuk
 * @version 1.0
 */
public class LaptopApplication {
    /**
     * @param args
     * @throws FileNotFoundException
     */
    public static void main(String[] args) throws FileNotFoundException {
        Laptop laptop = new Laptop(850);
        laptop.switchOn();
        System.out.println("RAM size: "+ laptop.getRAMSize()+" GB");
        System.out.println("RAM name: "+laptop.getRAMName());
        System.out.println("Processor cores value:
"+laptop.getProcessorCoresValue());
        System.out.println("Processor name: "+laptop.getProcessorNameValue());
        System.out.println("Power Supply value:
"+laptop.getPowerSupplyPowerValue()+" W");
        System.out.println("Processor usage percentage:
"+laptop.getProcessorUsagePercentage()+" %");
        System.out.println("Operating System name: "+laptop.getOSTypeName());
        laptop.changeOSType(OSType.LINUX);
        laptop.charge();
        laptop.moveCursor(22,5);
        laptop.displayOnScreen("Application starts running!");
        laptop.switchOff();
        laptop.increaseRAM(8);
        laptop.dispose();
    }
}

```

File Chargeable.java

```

package KI34.Karpliuk.Lab4;
// оголошуємо інтерфейс Chargeable
public abstract interface Chargeable {
    void charge(); // прототип методу
}

```

File Cursor.java

```

package KI34.Karpliuk.Lab4;
// оголошуємо інтерфейс Cursor
public interface Cursor {
    void moveCursor(int x,int y); // прототип методу
}

```

File Displayable.java

```

package KI34.Karpliuk.Lab4;
// оголошуємо інтерфейс Displayable
public interface Displayable {
    void displayOnScreen(String screenContent); // прототип методу
}

```

File Computer.java

```

package KI34.Karpliuk.Lab4;

import java.io.FileNotFoundException;
import java.io.PrintWriter;

```

```

/**
 * Class <code>Computer</code> implements computer
 *
 * @author Yuriy Karpliuk
 * @version 1.0
 */
// оголошуємо абстрактний клас Computer
public abstract class Computer{
    private PrintWriter fout;
    private RAM ram;
    private Processor processor;
    private PowerSupply powerSupply;

    private OS os;

    /**
     * Constructor
     *
     * @throws FileNotFoundException
     */
    public Computer() throws FileNotFoundException {
        ram = new RAM();
        powerSupply = new PowerSupply();
        processor = new Processor();
        os = new OS();
        fout = new PrintWriter("Log.txt");
    }

    /**
     * Constructor
     *
     * @param
     * <code>powerValue</code> PowerSupply power value
     * @throws FileNotFoundException
     */
    public Computer(int powerValue) throws FileNotFoundException {
        ram = new RAM();
        powerSupply = new PowerSupply(powerValue);
        processor = new Processor();
        os = new OS();
        fout = new PrintWriter("Log.txt");
    }

    /**
     * Method simulates switching on computer
     */
    public void switchOn() {
        powerSupply.turnOn();
        processor.startProcessorWorking();
        os.startOS();
        System.out.println("Computer is switched on!");
        fout.print("Computer is switched on!" + "\n");
        fout.print("Power supply is turned on! Status: " +
powerSupply.isTurnedOn() + "\n");
        fout.print("Processor start's working : " +
processor.getUsagePercentage() + "\n");
        fout.print("OS start's working! Status: " + os.isStarted() + "\n");
        fout.flush();
    }

    /**
     * Abstract method
     */
    public abstract void switchOff();

    /**

```

```

    * Method simulates RAM increasing size value
    */
    public void increaseRAM(int increaseValue) {
        fout.print("RAM size before increasing : " + ram.getSize() + "\n");
        ram.increaseRAMSize(increaseValue);
        fout.print("RAM size after increasing : " + ram.getSize() + "\n");
    }

    /**
     * Method returns info about RAM size value
     *
     * @return RAM size value
     */
    public int getRAMSize() {
        return ram.getSize();
    }

    /**
     * Method returns info about RAM name value
     *
     * @return RAM name value
     */
    public String getRAMName() {
        return ram.getName();
    }

    /**
     * Method returns info about Processor usage percentage
     *
     * @return Processor usage percentage
     */
    public int getProcessorUsagePercentage() {
        return processor.getUsagePercentage();
    }

    /**
     * Method returns info about Processor name value
     *
     * @return Processor name value
     */
    public String getProcessorNameValue() {
        return processor.getNameOfProcessor();
    }

    /**
     * Method returns info about Processor cores value
     *
     * @return Processor cores value
     */
    public int getProcessorCoresValue() {
        return processor.getAmountOfCores();
    }

    /**
     * Method returns info about PowerSupply power value
     *
     * @return PowerSupply power value
     */
    public int getPowerSupplyPowerValue() {
        return powerSupply.getPowerValue();
    }

    /**
     * Method returns info about OS type name
     *

```

```

    * @return OS type name
    */
    public String getOSTypeName() {
        return os.getOsType().name();
    }

    /**
     * Method simulates OS type changing
     */
    public void changeOSType(OSType osTypeValue) {
        os.changeOS(osTypeValue);
        fout.print("OS type changed to : " + os.getOsType().name() + "\n");
        fout.flush();
    }

    /**
     * Method returns PrintWriter object
     *
     * @return PrintWriter object
     */
    public PrintWriter getFout() {
        return fout;
    }

    /**
     * Method returns RAM object
     *
     * @return RAM object
     */
    public RAM getRam() {
        return ram;
    }

    /**
     * Method returns Processor object
     *
     * @return Processor object
     */
    public Processor getProcessor() {
        return processor;
    }

    /**
     * Method returns PowerSupply object
     *
     * @return PowerSupply object
     */
    public PowerSupply getPowerSupply() {
        return powerSupply;
    }

    /**
     * Method returns OS object
     *
     * @return OS object
     */
    public OS getOs() {
        return os;
    }

    /**
     * Method releases used resources
     */
    public void dispose() {
        fout.close();
    }

```

```
}
```

File OS.java

```
package KI34.Karpliuk.Lab4;

/**
 * Class <code>OS</code> implements operating system
 *
 * @author Yurii Karpliuk
 * @version 1.0
 */
public class OS {
    // OS type value
    private OSType osType;

    //OS working mode
    private boolean isStarted;

    /**
     * Constructor
     */
    public OS() {
        osType = OSType.WINDOWS;
        isStarted=false;
    }

    /**
     * Method simulates OS start's working
     */
    public void startOS(){
        isStarted=true;
    }

    /**
     * Method simulates OS finishes working
     */
    public void finishOS(){
        isStarted=false;
    }

    /**
     * Method returns OS type value
     * @return OS type value of
     * <code>OS.OSType</code>
     */
    public OSType getOsType(){
        return osType;
    }

    /**
     * Method simulates OS changing
     */
    public void changeOS(OSType osTypeValue){
        osType=osTypeValue;
    }

    /**
     * Method returns OS mode
     * @return The OS mode
     */
    public boolean isStarted() {
        return isStarted;
    }
}
```


File OSType.java

```
package KI34.Karpliuk.Lab4;

public enum OSType {
    WINDOWS, LINUX, MAC, MS_DOS,
}
```

File PowerSupply.java

```
package KI34.Karpliuk.Lab4;
/**
 * Class <code>PowerSupply</code> implements power supply
 *
 * @author Yurii Karpliuk
 * @version 1.0
 */
public class PowerSupply {
    // PowerSupply mode
    private boolean isTurnedOn;
    // PowerSupply power value
    private int powerValue;

    /**
     * Constructor
     */
    public PowerSupply() {
        isTurnedOn = false;
    }

    /**
     * Constructor
     * @param
     * <code>pValue</code> PowerSupply power value
     */
    public PowerSupply(int pValue) {
        powerValue = pValue;
    }

    /**
     * Method simulates PowerSupply turning on
     */
    public void turnOn(){
        isTurnedOn=true;
    }

    /**
     * Method simulates PowerSupply turning off
     */
    public void turnOff(){
        isTurnedOn=false;
    }

    /**
     * Method returns power supply power value
     * @return The PowerSupply power value
     */
    public int getPowerValue() {
        return powerValue;
    }

    /**
     * Method returns power supply mode
     * @return The PowerSupply mode
     */
    public boolean isTurnedOn() {
        return isTurnedOn;
    }
}
```

```
}  
}
```

File Processor.java

```
package KI34.Karpliuk.Lab4;  
/**  
 * Class Processor implements processor  
 *  
 * @author Yurii Karpliuk  
 * @version 1.0  
 */  
public class Processor {  
    // Processor amount of cores value  
    private int amountOfCores;  
    // Processor name value  
    private String nameOfProcessor;  
  
    // Processor usage percentage value  
    private int usagePercentage;  
  
    /**  
     * Constructor  
     * @param  
     * <code>pAmountOfCores</code> Processor amount of cores value  
     * @param  
     * <code>pNameOfProcessor</code> Processor name value  
     */  
    public Processor(int pAmountOfCores, String pNameOfProcessor) {  
        amountOfCores = pAmountOfCores;  
        nameOfProcessor = pNameOfProcessor;  
        usagePercentage = 0;  
    }  
  
    /**  
     * Constructor  
     */  
    public Processor() {  
        nameOfProcessor = "Intel Core i5";  
        amountOfCores = 6;  
    }  
  
    /**  
     * Method returns the Processor amount of cores value  
     * @return The Processor amount of cores value  
     */  
    public int getAmountOfCores() {  
        return amountOfCores;  
    }  
  
    /**  
     * Method returns the Processor name value  
     * @return The Processor name value  
     */  
    public String getNameOfProcessor() {  
        return nameOfProcessor;  
    }  
  
    /**  
     * Method simulates Processor start working  
     */  
    public void startProcessorWorking() {  
        usagePercentage= (int) ((Math.random() * (100 - 5)) + 5);  
    }  
}
```

```

/**
 * Method simulates Processor finish working
 */
public void finishProcessorWorking() {
    usagePercentage= 0;
}

/**
 * Method returns the Processor usage percentage value
 * @return The Processor usage percentage value
 */
public int getUsagePercentage() {
    return usagePercentage;
}
}

```

File RAM.java

```

package KI34.Karpliuk.Lab4;

/**
 * Class <code>RAM</code> implements random-access memory
 *
 * @author Yurii Karpliuk
 * @version 1.0
 */
public class RAM {
    // RAM size value
    private int size;

    // RAM name value
    private String name;

    /**
     * Constructor
     *
     * @param
     * <code>psSize</code> RAM size value
     * @param
     * <code>pName</code> RAM name value
     */
    public RAM(int psSize, String pName) {
        size = psSize;
        name = pName;
    }

    /**
     * Constructor
     */
    public RAM() {
        size = 8;
        name="Kingston DDR4";
    }

    /**
     * Method returns the RAM size value
     *
     * @return The RAM size value
     */
    public int getSize() {
        return size;
    }

    /**
     * Method returns the RAM name value
     *
     * @return The RAM name value
     */
}

```

```

public String getName() {
    return name;
}

/**
 * Method simulates RAM increasing size value
 */
public void increaseRAMSize(int increaseValue) {
    size += increaseValue;
}
}

```

Результат виконання програми:

```

Laptop is switched on!
RAM size: 8 GB
RAM name: Kingston DDR4
Processor cores value: 6
Processor name: Intel Core i5
Power Supply value: 850 W
Processor usage percentage: 51%
Operating System name: WINDOWS
Start charging battery! Capacity is:82
Cursor moved to coordinate: (22;5)
Information displayed on screen successfully:
Application starts running!
Laptop is switched off!

```

```

Laptop is switched on!
Power supply is turned on! Status: true
Processor start's working : 51
OS start's working! Status: true
OS type changed to : LINUX
Laptop is switched off!
Power supply is turned off! Status: false
Processor finishes working : 0
OS finishes working! Status:false
RAM size before increasing : 8
RAM size after increasing : 16

```

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