Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра ЕОМ

Звіт

до лабораторної роботи № 3

з дисципліни: «Кросплатформні засоби програмування»

«Класи та пакети»

Варіант - 7

Виконав:

студент групи КІ-34

Карплюк Ю.Б.

Прийняв:

Іванов Ю.С.

Львів 2022

Мета: ознайомитися з процесом розробки класів та пакетів мовою Java.

ЗАВДАННЯ

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.Lab3`;
   - 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 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. Комп'ютер

Код програми:

File Computer.java

```
package KI34.Karpliuk.Lab3;

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

/**
 * Class <code>Computer</code> implements computer
 *
 * @author Yurii Karpliuk
 * @version 1.0
 */

public class Computer {
    private PrintWriter fout;
    private RAM ram;
    private Processor processor;
    private PowerSupply powerSupply;

    private OS os;

    //Computer brand name value
    private String name;

    static int amount;
```

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

    public Computer(String nameV,String processorName,int amountOfCores,int
ramSize,String ramName) throws FileNotFoundException {
        ram = new RAM(ramSize,ramName);
        powerSupply = new PowerSupply();
        processor = new Processor(amountOfCores,processorName);
        os = new OS();
        fout = new PrintWriter("Log.txt");
        name=nameV;
    }

    /**
     * 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");
        if(powerValue>100 && powerValue<500){
            amount++;
        }
    }

    /**
     * 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();
    }

    /**
     * Method simulates switching off computer
     */
    public void switchOff() {
        powerSupply.turnOff();
        processor.finishProcessorWorking();
        os.finishOS();
```

```
        System.out.println("Computer is switched off!");
        fout.print("Computer is switched off! " + "\n");
        fout.print("Power supply is turned off! Status: " +
powerSupply.isTurnedOn() + "\n");
        fout.print("Processor finishes working : " +
processor.getUsagePercentage() + "\n");
        fout.print("OS finishes working! Status:" + os.isStarted() + "\n");
        fout.flush();
    }

    /**
     * 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 info about Computer brand name
     *
     * @return Computer brand name
     */
    public String getName() {
        return name;
    }
    /**
     * Method sets Computer brand name
     */
    public void setName(String name) {
        this.name = name;
    }

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


}
```

File ComputerApplication.java

```
package KI34.Karpliuk.Lab3;

import java.io.FileNotFoundException;

/**
 * Computer Application class implements main method for Computer
 * class possibilities demonstration
 *
```

```
 * @author Yurii Karpliuk
 * @version 1.0
 */
public class ComputerApplication {
    /**
     * @param args
     * @throws FileNotFoundException
     */
    public static void main(String[] args) throws FileNotFoundException {
        Computer computer = new Computer(850);
        Computer computer1 = new Computer(250);
//        Computer computer3 = new Computer(320);
//        Computer computer4 = new Computer(400);
//        Computer computer5 = new Computer(400);
//        Computer computer6 = new Computer(350);
//
//        System.out.println(Computer.amount);
        computer1.switchOn();
        System.out.println("RAM size: "+ computer1.getRAMSize()+" GB");
        System.out.println("RAM name: "+computer1.getRAMName());
        System.out.println("Processor cores value:
"+computer1.getProcessorCoresValue());
        System.out.println("Processor name: "+computer.getProcessorNameValue());
        System.out.println("Power Supply value:
"+computer.getPowerSupplyPowerValue()+" W");
        System.out.println("Processor usage percentage:
"+computer.getProcessorUsagePercentage()+ "%");
        System.out.println("Operating System name: "+computer.getOSTypeName());
        computer.changeOSType(OSType.LINUX);
        computer.switchOff();
        computer.increaseRAM(8);
        computer.dispose();
    }
}
```

File OS.java

```
package KI34.Karpliuk.Lab3;
/**
 * 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
     * <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.Lab3;

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

File PowerSupply.java

```
package KI34.Karpliuk.Lab3;
/**
 * 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.Lab3;
/**
 * Class <code>Processor</code> 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) {
```

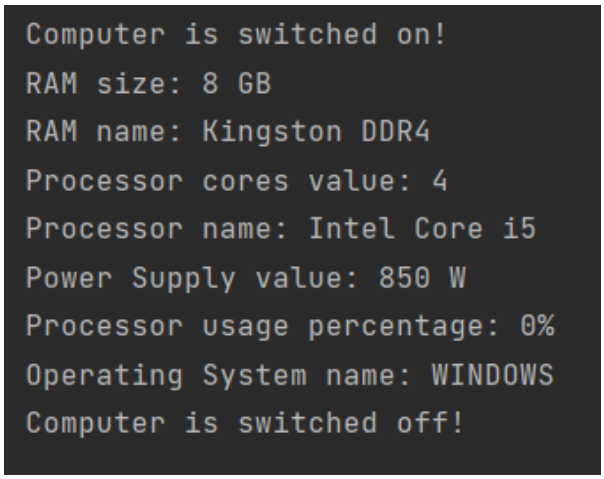
```
        amountOfCores = pAmountOfCores;
        nameOfProcessor = pNameOfProcessor;
        usagePercentage = 0;
    }

    /**
     * Constructor
     */
    public Processor() {
        nameOfProcessor = "Intel Core i5";
        amountOfCores = 4;
    }

    /**
     * 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;
    }
}
```

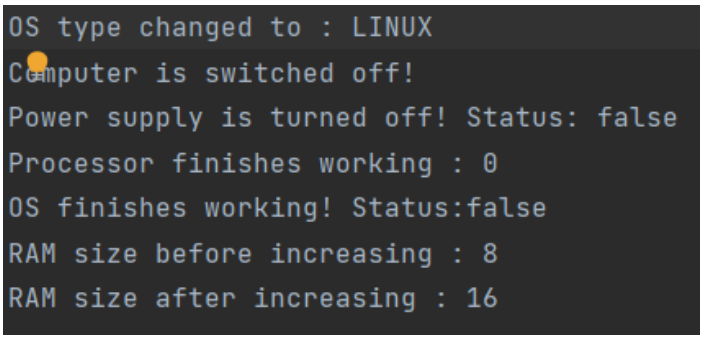
File RAM.java

```
package KI34.Karpliuk.Lab3;
/**
 * Class <code>RAM</code> implements random-access memory
 *
 * @author Yurii Karpliuk
 * @version 1.0
 */
public class RAM {
    // RAM size value
    private int size;
```

```java
    // 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 прогами:





Висновок: на лабораторній роботі я ознайомився з процесом розробки класів та пакетів мовою Java.