Міністерство освіти і науки України

Національний університет “Львівська політехніка ”

Кафедра ЕОМ



**ЗВІТ**

до лабораторної роботи №2

З дисципліни: «Кросплатформні засоби програмування»

На тему: «Класи та пакети»

Варіант - 7

Виконала:

ст. гр. КІ-305

Дзера А. Р.

Прийняв:

доц. каф. ЕОМ

Іванов Ю. С.

Львів – 2023

**Мета роботи:** 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.Lab2;
   * 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 – комп’ютер

**Код програми**

**Файл Computer.java**

*/\*\*  
 \* lab 2 package  
 \*/*package KI305.Dzera.Lab2;  
  
import java.io.\*;  
import java.util.ArrayList;  
import java.util.List;  
  
*/\*\*  
 \* Клас Computer представляє комп'ютер  
 \** ***@author*** *Amina  
 \** ***@version*** *1.0  
 \*/*public class Computer {  
 public static int *count*;  
 private final CPU cpu;  
 private final RAM ram;  
 private final HardDrive hardDrive;  
 private Monitor monitor;  
 private final Mouse mouse;  
 private final Keyboard keyboard;  
 private List<Software> installedSoftware; // Додати змінну для зберігання програмного забезпечення  
  
 private BufferedWriter logFile;  
  
 */\*\*  
 \* Конструктор класу Computer без параметрів. Створює комп'ютер і ініціалізує його компоненти за значеннями за замовчуванням.  
 \*/* public Computer() throws IOException {  
 this.cpu = new CPU("Intel Core i5");  
 this.ram = new RAM(8);  
 this.hardDrive = new HardDrive(256);  
 this.monitor = new Monitor("1920x1080");  
 this.mouse = new Mouse("Logitech");  
 this.keyboard = new Keyboard("QWERTY");  
 this.installedSoftware = new ArrayList<>(); // Ініціалізувати список програмного забезпечення  
  
 }  
  
 */\*\*  
 \* Конструктор класу Computer з параметрами. Створює комп'ютер і ініціалізує його компоненти заданими значеннями  
 \*  
 \** ***@param*** *cpu об'єкт процесора  
 \** ***@param*** *ram об'єкт оперативної пам'яті  
 \** ***@param*** *hardDrive об'єкт жорсткого диска  
 \** ***@param*** *monitor об'єкт монітора  
 \** ***@param*** *mouse об'єкт миші  
 \** ***@param*** *keyboard об'єкт клавіатури  
 \** ***@throws*** *IOException виняток, який може виникнути при створенні файлу логу  
 \*/* public Computer(CPU cpu, RAM ram, HardDrive hardDrive, Monitor monitor, Mouse mouse, Keyboard keyboard) throws IOException {  
 this.cpu = cpu;  
 this.ram = ram;  
 this.hardDrive = hardDrive;  
 this.monitor = monitor;  
 this.mouse = mouse;  
 this.keyboard = keyboard;  
 installedSoftware = new ArrayList<>();  
  
 logFile = new BufferedWriter(new FileWriter("computer\_log.txt"));  
 }  
  
  
 */\*\*  
 \* Метод для увімкнення комп'ютера  
 \* Цей метод вмикає комп'ютер та виводить повідомлення про його ввімкнення  
 \*  
 \** ***@throws*** *IOException виняток, який може виникнути при записі в файл логу  
 \*/* public void turnOn() throws IOException {  
 System.*out*.println("The computer is on.");  
 writeToLog("The computer is on.");  
 *count*++;  
 }  
  
  
 */\*\*  
 \* Метод для вимкнення комп'ютера  
 \* Цей метод вимикає комп'ютер та виводить повідомлення про його вимкнення  
 \*  
 \** ***@throws*** *IOException виняток, який може виникнути при записі в файл логу  
 \*/* public void turnOff() throws IOException {  
 System.*out*.println("The computer is turned off.");  
 writeToLog("The computer is turned off.");  
 }  
  
 */\*\*  
 \* Метод для виконання обчислень на комп'ютері  
 \* Цей метод виводить повідомлення про початок обчислень на комп'ютері  
 \*  
 \** ***@throws*** *IOException виняток, який може виникнути при записі в файл логу  
 \*/* public void performCalculations() throws IOException {  
 System.*out*.println("Calculations are being performed...");  
 writeToLog("Calculations are being performed...");  
 }  
  
 */\*\*  
 \* Метод для зберігання інформації на жорсткому диску  
 \*  
 \** ***@param*** *data дані, які потрібно зберегти  
 \** ***@throws*** *IOException виняток, який може виникнути при записі в файл логу  
 \*/* public void storeData(String data) throws IOException {  
 hardDrive.saveData(data);  
 System.*out*.println("The data is saved on the hard disk.");  
 writeToLog("The data is saved on the hard disk.");  
 }  
  
 */\*\*  
 \* Метод для встановлення нового монітора для комп'ютера  
 \*  
 \** ***@param*** *monitor новий об'єкт монітора, який слід встановити  
 \** ***@throws*** *IOException виняток, який може виникнути при записі в файл логу  
 \*/* public void setMonitor(Monitor monitor) throws IOException {  
 this.monitor = monitor;  
 System.*out*.println("Monitor changed to " + monitor.getResolution());  
 writeToLog("Monitor changed to " + monitor.getResolution());  
 }  
  
 */\*\*  
 \* Метод для встановлення програмного забезпечення на комп'ютер  
 \*  
 \** ***@param*** *software об'єкт програмного забезпечення, яке слід встановити  
 \** ***@throws*** *IOException виняток, який може виникнути при записі в файл логу  
 \*/* public void installSoftware(Software software) throws IOException {  
 installedSoftware.add(software);  
 System.*out*.println("Installed software: " + software.getName());  
 writeToLog("Installed software: " + software.getName());  
 }  
  
 */\*\*  
 \* Метод для відображення списку встановленого програмного забезпечення на комп'ютері  
 \*  
 \** ***@throws*** *IOException виняток, який може виникнути при записі в файл логу  
 \*/* public void showInstalledSoftware() throws IOException {  
 System.*out*.println("Showing all installed software: ");  
 for (Software software : installedSoftware) {  
 System.*out*.println(software.getName());  
 writeToLog(software.getName());  
 }  
 }  
  
 */\*\*  
 \* Метод для виведення на екран інформації про доступну оперативну пам'ять комп'ютера  
 \*  
 \** ***@throws*** *IOException виняток, який може виникнути при записі в файл логу  
 \*/* public void checkAvailableMemory() throws IOException {  
 System.*out*.println("Available RAM: " + ram.getAvailableMemory() + " ГБ");  
 writeToLog("Available RAM: " + ram.getAvailableMemory() + " ГБ");  
 }  
  
 */\*\*  
 \* Метод для запуску програмного додатку на комп'ютері  
 \*  
 \** ***@param*** *appName назва програмного додатку, який потрібно запустити  
 \** ***@throws*** *IOException виняток, який може виникнути при записі в файл логу  
 \*/* public void launchApplication(String appName) throws IOException {  
 System.*out*.println("The program starts: " + appName);  
 writeToLog("The program starts: " + appName);  
 }  
  
 */\*\*  
 \* Метод для перевірки стану жорсткого диска комп'ютера  
 \*  
 \** ***@return*** *`true`, якщо стан жорсткого диска в порядку, інакше `false`  
 \*/* public boolean isHardDriveHealthy() {  
 // Логіка перевірки стану жорсткого диска  
 return true; // Повертаємо true, якщо все в порядку  
 }  
  
 */\*\*  
 \* Перевіряє стан жорсткого диска і виводить відповідне повідомлення  
 \*  
 \** ***@throws*** *IOException виняток, який може виникнути при записі в файл логу  
 \*/* public void checkHardDriveStatus() throws IOException {  
 boolean isHealthy = isHardDriveHealthy();  
 if (isHealthy) {  
 System.*out*.println("Hard drive is in good condition.");  
 writeToLog("Hard drive is in good condition.");  
 } else {  
 System.*out*.println("Hard drive requires maintenance.");  
 writeToLog("Hard drive requires maintenance.");  
 }  
 }  
  
 */\*\*  
 \* Метод для зміни налаштувань монітора комп'ютера, таких як яскравість та контраст  
 \*  
 \** ***@param*** *brightness яскравість монітора (значення від 0 до 100)  
 \** ***@param*** *contrast контраст монітора (значення від 0 до 100)  
 \** ***@throws*** *IOException виняток, який може виникнути при записі в файл логу  
 \*/* public void changeMonitorSettings(int brightness, int contrast) throws IOException {  
 monitor.setBrightness(brightness);  
 monitor.setContrast(contrast);  
 System.*out*.println("The monitor settings have been changed.");  
 writeToLog("The monitor settings have been changed.");  
 }  
  
 */\*\*  
 \* Метод для натискання лівої кнопки миші комп'ютера  
 \*  
 \** ***@throws*** *IOException виняток, який може виникнути при записі в файл логу  
 \*/* public void clickMouseLeftButton() throws IOException {  
 mouse.clickLeftButton();  
 writeToLog("Left mouse button clicked.");  
 }  
  
 */\*\*  
 \* Метод для натискання правої кнопки миші комп'ютера  
 \*  
 \** ***@throws*** *IOException виняток, який може виникнути при записі в файл логу  
 \*/* public void clickMouseRightButton() throws IOException {  
 mouse.clickRightButton();  
 writeToLog("Right mouse button clicked.");  
 }  
  
 */\*\*  
 \* Метод для прокручування миші вгору  
 \*  
 \** ***@throws*** *IOException виняток, який може виникнути при записі в файл логу  
 \*/* public void scrollMouseUp() throws IOException {  
 mouse.scrollUp();  
 writeToLog("Mouse scrolled up.");  
 }  
  
 */\*\*  
 \* Метод для прокручування миші вниз  
 \*  
 \** ***@throws*** *IOException виняток, який може виникнути при записі в файл логу  
 \*/* public void scrollMouseDown() throws IOException {  
 mouse.scrollDown();  
 writeToLog("Mouse scrolled down.");  
 }  
  
 */\*\*  
 \* Метод для натискання клавіші на клавіатурі  
 \*  
 \** ***@param*** *key символ, який представляє натиснуту клавішу  
 \*/* public void pressKeyOnKeyboard(char key) {  
 keyboard.pressKey(key);  
  
 }  
  
 */\*\*  
 \* Метод для введення тексту на клавіатурі  
 \*  
 \** ***@param*** *text рядок тексту, який буде введено на клавіатурі  
 \*/* public void typeTextOnKeyboard(String text) {  
 keyboard.typeText(text);  
  
 }  
  
 */\*\*  
 \* Метод для запису повідомлення в файл логу та додавання перехіду на новий рядок  
 \*  
 \** ***@param*** *message рядок повідомлення, який буде записаний в файл логу  
 \** ***@throws*** *IOException виняток, який може виникнути при записі в файл логу  
 \*/* public void writeToLog(String message) throws IOException{  
 logFile.write(message);  
 logFile.newLine(); //додати перехід на новий рядок  
 logFile.flush();  
 }  
  
 */\*\*  
 \* Метод для закриття файлу логу  
 \*  
 \** ***@throws*** *IOException виняток, який може виникнути при закритті файлу логу  
 \*/* public void closeLogFile() throws IOException{  
 logFile.close();  
 }  
  
 */\*\*  
 \* Метод для вимкнення комп'ютера та закриття файлу логу  
 \*  
 \** ***@throws*** *IOException виняток, який може виникнути при закритті файлу логу  
 \*/* public void shutdown() throws IOException{  
 writeToLog("Computer turned off.");  
 closeLogFile();  
 }  
}  
  
  
*/\*\*  
 \* Клас, що представляє центральний процесор (CPU) комп'ютера  
 \*/*class CPU {  
 private final String model;  
  
  
 */\*\*  
 \* Конструктор класу CPU ініціалізує модель процесора  
 \*  
 \** ***@param*** *model модель процесора  
 \*/* public CPU(String model) {  
 this.model = model;  
 }  
  
 */\*\*  
 \* Цей метод отримує модель процесора  
 \*  
 \** ***@return*** *модель процесора  
 \*/* public String getModel() {  
 return model;  
 }  
  
}

*/\*\*  
 \* Клас, що представляє оперативну пам'ять комп'ютера (RAM)  
 \*/*class RAM {  
 private final int capacity; // oбсяг оперативної пам'яті в ГБ  
 private static int *count* = 0;  
  
 */\*\*  
 \* Конструктор класу RAM. Ініціалізує об'єкт оперативної пам'яті з заданим обсягом  
 \*  
 \** ***@param*** *capacity обсяг оперативної пам'яті в ГБ  
 \*/* public RAM(int capacity) {  
 this.capacity = capacity;  
 if (capacity > 2) {  
 *count*++;  
 }  
 }  
  
 */\*\*  
 \* Статичний метод, що повертає кількість створених об'єктів RAM з обсягом пам'яті більше 2 ГБ  
 \*  
 \** ***@return*** *кількість об'єктів RAM з обсягом пам'яті більше 2 ГБ  
 \*/* public static int getCount() {  
 return *count*;  
 }  
  
 */\*\*  
 \* Метод, що повертає обсяг оперативної пам'яті поточного об'єкта RAM  
 \*  
 \** ***@return*** *обсяг оперативної пам'яті в ГБ  
 \*/* public int getCapacity() {  
 return capacity;  
 }  
  
  
 */\*\*  
 \* Отримує доступний обсяг оперативної пам'яті  
 \*  
 \** ***@return*** *обсяг доступної оперативної пам'яті в ГБ  
 \*/* public int getAvailableMemory() {  
 return capacity;  
 }  
}

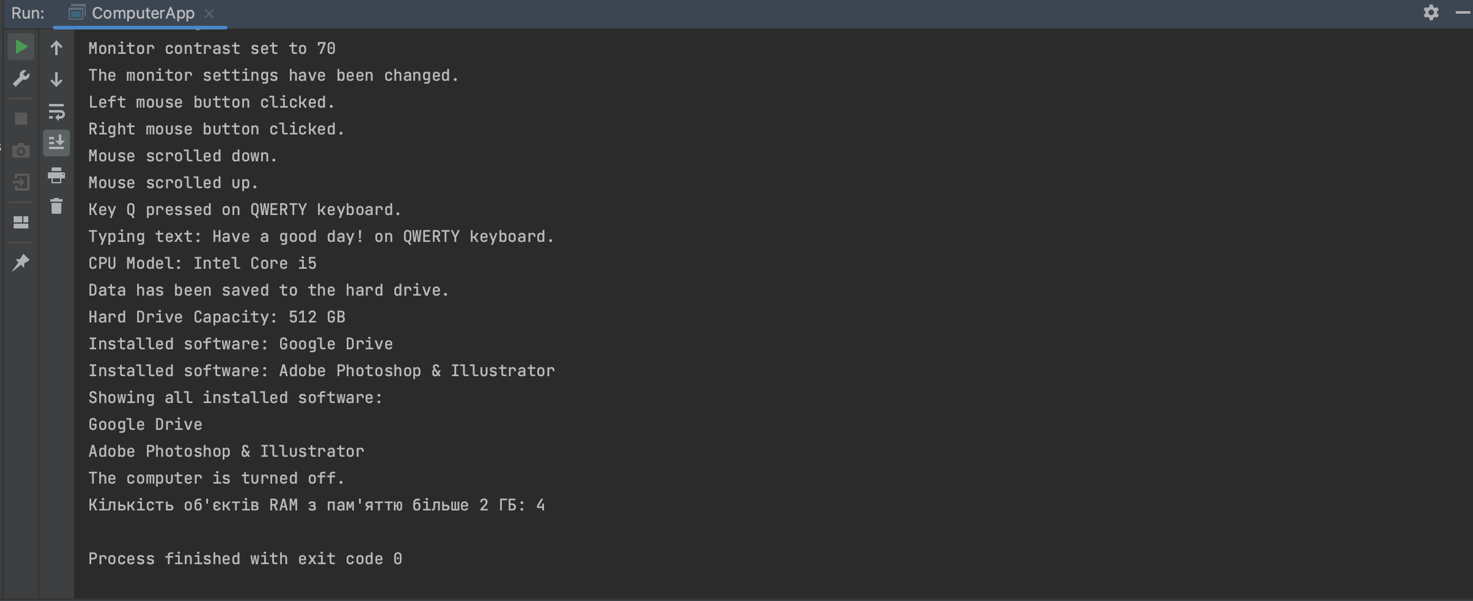
*/\*\*  
 \* Клас, що представляє жорсткий диск комп'ютера  
 \*/*class HardDrive {  
 private final int storageCapacity; // Обсяг жорсткого диска в ГБ  
  
 */\*\*  
 \* Конструктор класу HardDrive ініціалізує обсяг жорсткого диска  
 \*  
 \** ***@param*** *storageCapacity обсяг жорсткого диска в ГБ  
 \*/* public HardDrive(int storageCapacity) {  
 this.storageCapacity = storageCapacity;  
 }  
  
 */\*\*  
 \* Отримує обсяг жорсткого диска  
 \*  
 \** ***@return*** *обсяг жорсткого диска в ГБ  
 \*/* public int getStorageCapacity() {  
 return storageCapacity;  
 }  
  
 */\*\*  
 \* Зберігає дані на жорсткому диску  
 \*  
 \** ***@param*** *data дані, які потрібно зберегти  
 \*/* public void saveData(String data) {  
 try{  
 File file = new File("data.txt");  
 FileWriter fileWriter = new FileWriter(file);  
 BufferedWriter bufferedWriter = new BufferedWriter(fileWriter);  
 bufferedWriter.write(data);  
 bufferedWriter.close();  
 System.*out*.println("Data has been saved to the hard drive.");  
 } catch (IOException e){  
 e.printStackTrace();  
 System.*err*.println("Error occurred while saving data to the hard drive.");  
 }  
 }  
}  
  
  
*/\*\*  
 \* Клас, що представляє монітор комп'ютера  
 \*/*class Monitor {  
 private final String resolution;  
  
 */\*\*  
 \* Конструктор класу Monitor ініціалізує роздільну здатність монітора  
 \*  
 \** ***@param*** *resolution роздільна здатність монітора  
 \*/* public Monitor(String resolution) {  
 this.resolution = resolution;  
 }  
  
 */\*\*  
 \* Отримує роздільну здатність монітора  
 \*  
 \** ***@return*** *роздільна здатність монітора  
 \*/* public String getResolution() {  
 return resolution;  
 }  
  
 */\*\*  
 \* Налаштовує яскравість монітора  
 \*  
 \** ***@param*** *brightness значення яскравості  
 \*/* public void setBrightness(int brightness) {  
 System.*out*.println("Monitor brightness set to " + brightness);  
 }  
  
 */\*\*  
 \* Налаштовує контрастність монітора  
 \*  
 \** ***@param*** *contrast значення контрастності  
 \*/* public void setContrast(int contrast) {  
 System.*out*.println("Monitor contrast set to " + contrast);  
 }  
  
}  
  
*/\*\*  
 \* Клас, що представляє комп'ютерну мишу  
 \*/*class Mouse {  
  
 public final String brand;  
  
 */\*\*  
 \* Конструктор класу Mouse ініціалізує бренд миші та виводить інформацію про бренд  
 \*  
 \** ***@param*** *brand бренд миші  
 \*/* public Mouse(String brand) {  
 this.brand = brand;  
 }  
  
 */\*\*  
 \* Метод виконує клік лівою кнопкою миші  
 \*/* public void clickLeftButton() {  
 System.*out*.println("Left mouse button clicked.");  
 }  
  
 */\*\*  
 \* Метод виконує клік правою кнопкою миші  
 \*/* public void clickRightButton() {  
 System.*out*.println("Right mouse button clicked.");  
 }  
  
 */\*\*  
 \* Метод прокручує мишу вгору  
 \*/* public void scrollUp() {  
 System.*out*.println("Mouse scrolled up.");  
 }  
  
 */\*\*  
 \* Метод прокручує мишу вниз  
 \*/* public void scrollDown() {  
 System.*out*.println("Mouse scrolled down.");  
 }  
  
}  
  
*/\*\*  
 \* Клас, що представляє комп'ютерну клавіатуру з певним розкладом  
 \*/*class Keyboard {  
 private final String layout;  
  
 */\*\*  
 \* Конструктор класу Keyboard ініціалізує розклад клавіатури  
 \*  
 \** ***@param*** *layout розклад клавіатури  
 \*/* public Keyboard(String layout) {  
 this.layout = layout;  
 }  
  
 */\*\*  
 \* Метод симулює натискання клавіші на клавіатурі та виводить повідомлення  
 \*  
 \** ***@param*** *key натискана клавіша  
 \*/* public void pressKey(char key) {  
 System.*out*.println("Key " + key + " pressed on " + layout + " keyboard.");  
 }  
  
 */\*\*  
 \* Метод симулює введення тексту на клавіатурі та виводить повідомлення  
 \*  
 \** ***@param*** *text введений текст  
 \*/* public void typeText(String text) {  
 System.*out*.println("Typing text: " + text + " on " + layout + " keyboard.");  
 }  
}  
  
  
*/\*\*  
 \* Клас, що представляє програмне забезпечення на комп'ютері  
 \*/*class Software {  
 private final String name;  
  
 */\*\*  
 \* Конструктор класу Software ініціалізує назву програмного забезпечення  
 \*  
 \** ***@param*** *name назва програмного забезпечення  
 \*/* public Software(String name) {  
 this.name = name;  
 }  
  
 */\*\*  
 \* Метод отримує назву програмного забезпечення  
 \*  
 \** ***@return*** *назва програмного забезпечення  
 \*/* public String getName() {  
 return name;  
 }  
}

**Файл ComputerApp.java**

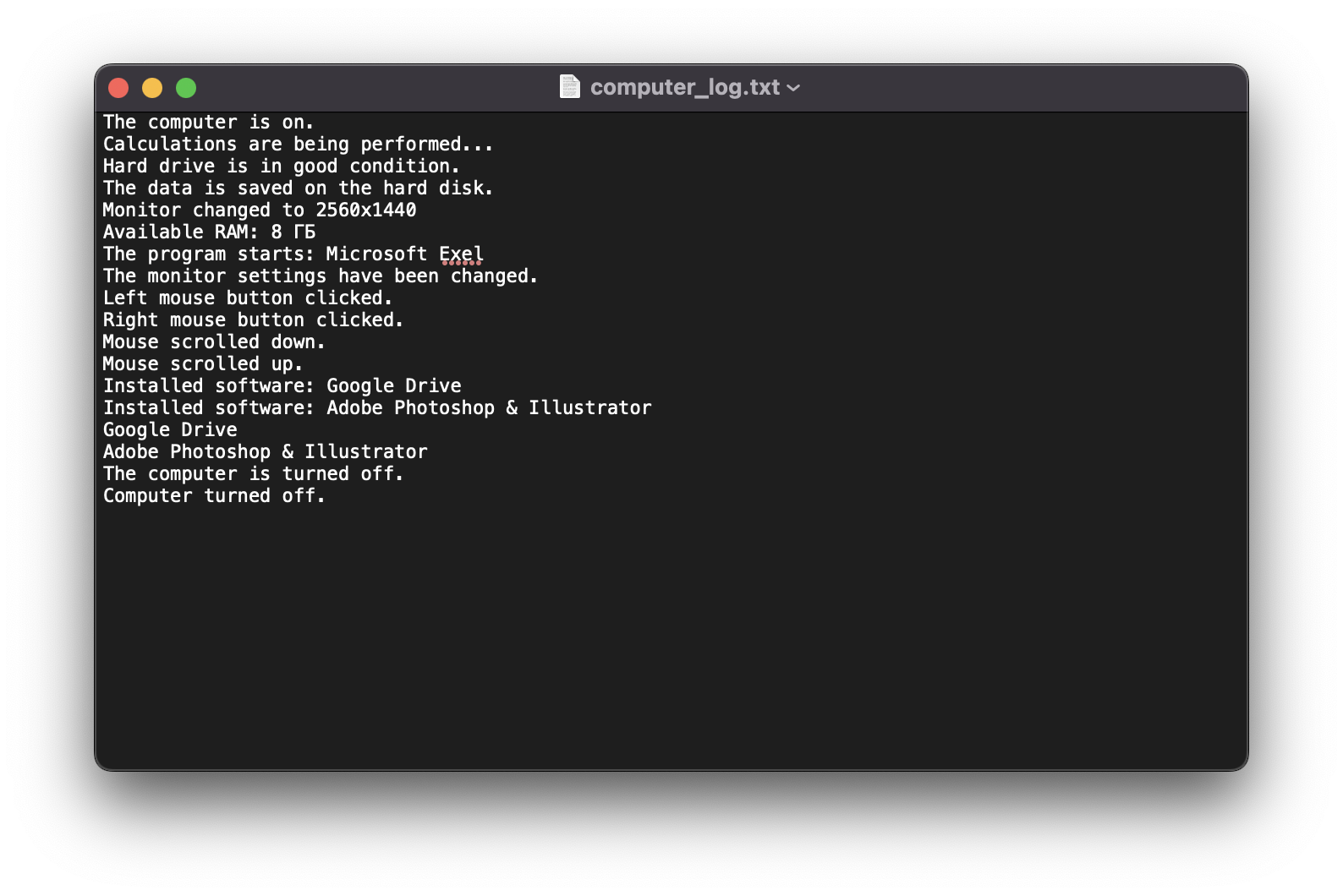
*/\*\*  
 \* lab 2 package  
 \*/*package KI305.Dzera.Lab2;  
  
import java.io.\*;  
  
*/\*\*  
 \* Клас, що містить метод `main`, який представляє собою додаток для використання комп'ютера  
 \*/*public class ComputerApp {  
  
 */\*\*  
 \* Основний метод програми, який використовує комп'ютер та його компоненти  
 \*  
 \** ***@param*** *args параметри командного рядка (не використовуються у цьому додатку)  
 \** ***@throws*** *IOException викидається, якщо виникають проблеми з обробкою введення/виведення  
 \*/* public static void main (String[] args) throws IOException {  
 //cтворення об'єктів для компонентів комп'ютера  
 CPU cpu = new CPU("Intel Core i5");  
 RAM ram = new RAM(8);  
 HardDrive hardDrive = new HardDrive(512);  
 Monitor monitor = new Monitor("1920x1080");  
 Mouse mouse = new Mouse("Logitech");  
 Keyboard keyboard = new Keyboard("QWERTY");  
  
 Computer computer = new Computer(cpu, ram, hardDrive, monitor, mouse, keyboard);  
 computer.turnOn();  
  
 //це для завдання для захисту лаб, підрахунок кількості включених комп'ютерів  
 Computer computer1 = new Computer(cpu, ram, hardDrive, monitor, mouse, keyboard);  
 computer1.turnOn();  
  
 Computer computer2 = new Computer(cpu, ram, hardDrive, monitor, mouse, keyboard);  
 computer2.turnOn();  
  
 Computer computer3 = new Computer(cpu, ram, hardDrive, monitor, mouse, keyboard);  
 computer3.turnOn();  
 System.*out*.println("Кількість включених комп'ютерів: " + Computer.*count*);  
  
  
 computer.performCalculations();  
 computer.isHardDriveHealthy();  
 computer.checkHardDriveStatus();  
 computer.storeData("Some important data");  
 computer.setMonitor(new Monitor("2560x1440"));  
 computer.checkAvailableMemory();  
 computer.launchApplication("Microsoft Exel");  
 computer.changeMonitorSettings(80,70);  
 computer.clickMouseLeftButton();  
 computer.clickMouseRightButton();  
 computer.scrollMouseDown();  
 computer.scrollMouseUp();  
 computer.pressKeyOnKeyboard('Q');  
 computer.typeTextOnKeyboard("Have a good day!");  
  
 //отримання та виведення інформації про модель процесора  
 String cpuModel = cpu.getModel();  
 System.*out*.println("CPU Model: " + cpuModel);  
  
  
 //збереження даних на жорсткий диск та отримання інформації про його обсяг  
 hardDrive.saveData("This is data about the computer's hard drive.");  
 int storageCapacity = hardDrive.getStorageCapacity();  
 System.*out*.println("Hard Drive Capacity: " + storageCapacity + " GB");  
  
 //встановлення та виведення інформації про встановлене програмне забезпечення  
 Software software1 = new Software("Google Drive");  
 Software software2 = new Software("Adobe Photoshop & Illustrator");  
 computer.installSoftware(software1);  
 computer.installSoftware(software2);  
 computer.showInstalledSoftware();  
  
 //вимкнення комп'ютера та завершення роботи  
 computer.turnOff();  
 computer.shutdown();  
  
  
 //це для завдання для захисту лаб, вивід кількості об'єктів RAM з пам'яттю більше 2 ГБ  
 RAM ram1 = new RAM(4);  
 RAM ram2 = new RAM(8);  
 RAM ram3 = new RAM(2);  
 RAM ram4 = new RAM(16);  
 RAM ram5 = new RAM(1);  
 RAM ram6 = new RAM(1);  
  
 System.*out*.println("Кількість об'єктів RAM з пам'яттю більше 2 ГБ: " + RAM.*getCount*());  
  
 }  
}

**Результати роботи програми**

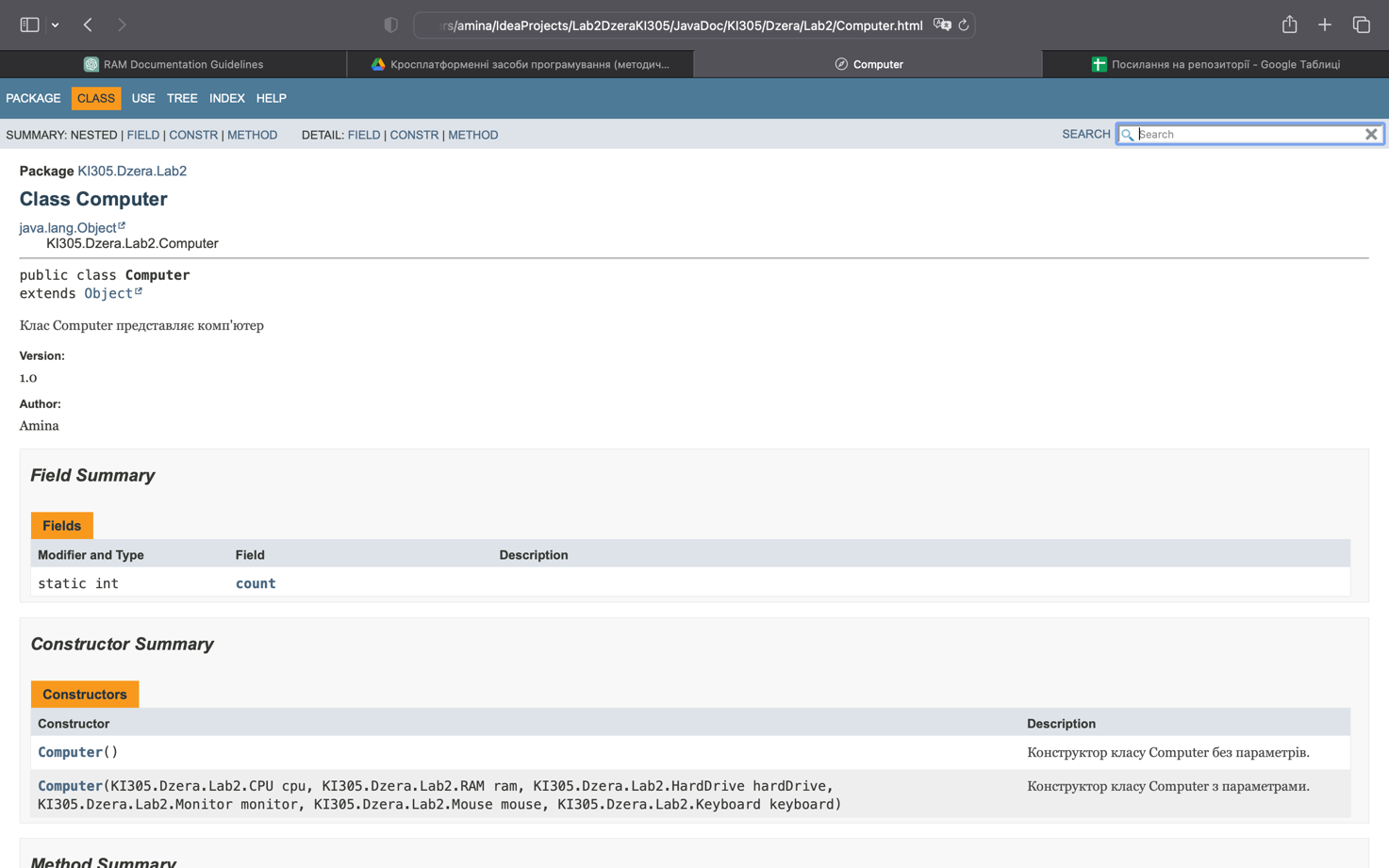


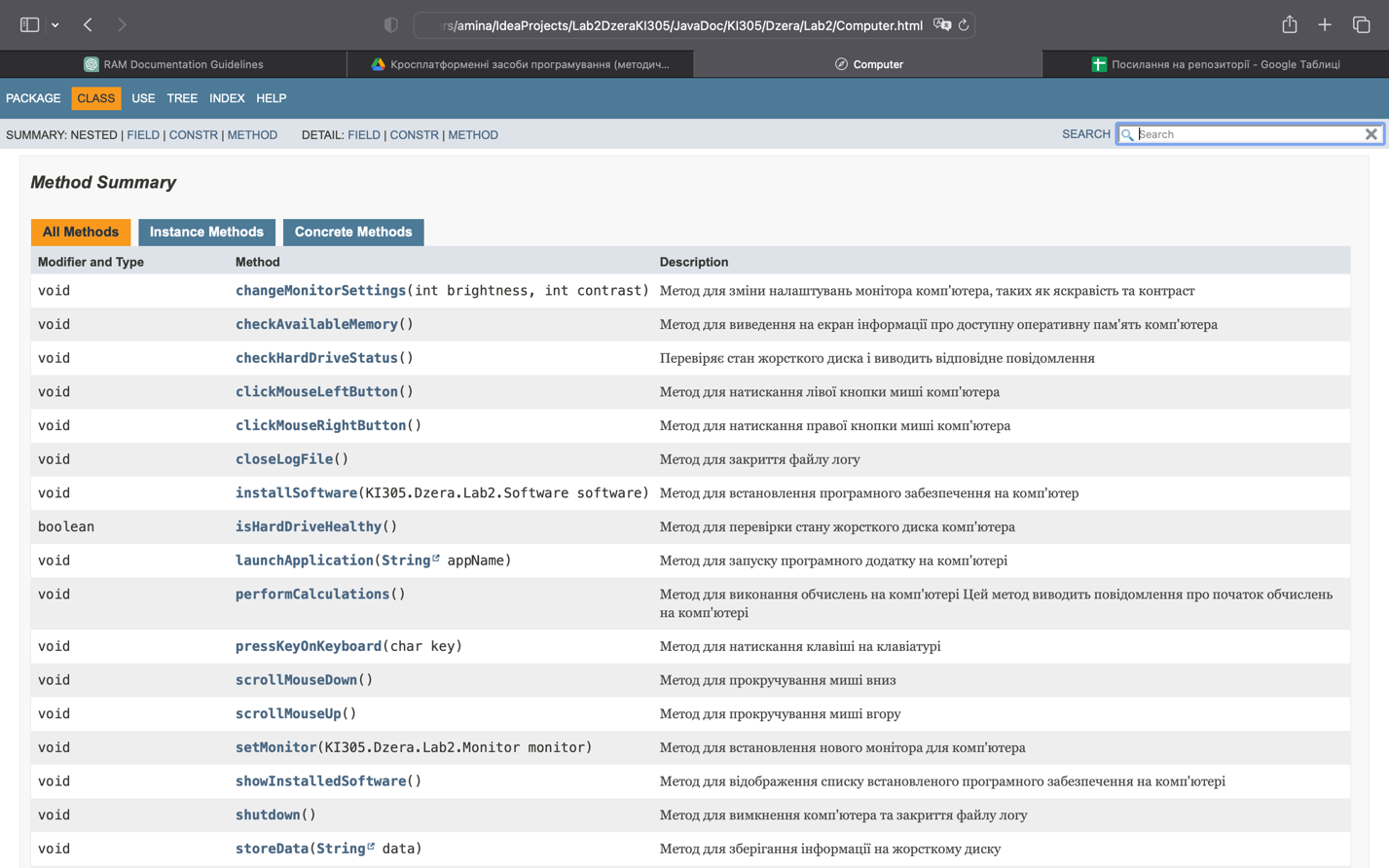


**Вміст файлу computer\_log.txt**



**Фрагмент згенерованої документації**





**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**

Під час виконання даної лабораторної роботи я ознайомилася з процесом розробки класів та пакетів мовою Java, а також з базовими конструкціями цієї мови, такими як оголошення класів, методів та полів.