Міністерство освіти і науки України

Національний університет “Львівська політехніка”

Кафедра ЕОМ



ЗВІТ

до лабораторної роботи №3

З дисципліни: «Кросплатформні засоби програмування»

На тему: «СПАДКУВАННЯ ТА ІНТЕРФЕЙСИ»

Варіант 20

**Виконала:**

ст. групи КІ-306

Петрук О.С.

**Прийняв:**

доцент кафедри ЕОМ

Олексів М.В.

Львів – 2024

**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 №2, для реалізації предметної області заданої варіантом. Суперклас, що реалізований у лабораторній роботі №2, зробити абстрактним. Розроблений підклас має забезпечувати механізми свого коректного функціонування та реалізовувати мінімум один інтерфейс. Програма має розміщуватися в пакеті Група.Прізвище.Lab3 та володіти коментарями, які дозволять автоматично згенерувати документацію до розробленого пакету.

2. Автоматично згенерувати документацію до розробленого пакету.

3. Завантажити код на GitHub згідно методичних вказівок по роботі з GitHub.

4. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації та завантажити його у ВНС.

5. Дати відповідь на контрольні запитання.

**Тема згідно варіанту №20 – «Чоботи»**

**GitHub Repository:**

**Хід роботи**

Код програми:

***Shoes.java***

// File: Ki\_306/Petruk/Lab3/Shoes.java  
package Ki\_306.Petruk.Lab3;  
  
*/\*\*  
 \* Абстрактний клас Shoes, що описує основні характеристики та функціональність взуття.  
 \*/*public abstract class Shoes {  
 private String brand;  
 private double size;  
 private String color;  
 private boolean laced;  
 private boolean worn;  
 private double distanceRun;  
  
 */\*\*  
 \* Конструктор для створення взуття.  
 \*  
 \* @param brand Бренд взуття.  
 \* @param size Розмір взуття.  
 \* @param color Колір взуття.  
 \*/* public Shoes(String brand, double size, String color) {  
 this.brand = brand;  
 this.size = size;  
 this.color = color;  
 this.laced = false;  
 this.worn = false;  
 this.distanceRun = 0.0;  
 }  
  
 */\*\*  
 \* Перевіряє, чи потрібно ремонтувати взуття.  
 \*  
 \* @return true, якщо взуття пробігло 100 км або більше, і потребує ремонту.  
 \*/* public boolean needsRepair() {  
 return distanceRun >= 100;  
 }  
  
 */\*\*  
 \* Додає дистанцію пробігу до загального пробігу взуття.  
 \*  
 \* @param distance Дистанція, що була пробігла.  
 \*/* protected void addDistance(double distance) {  
 this.distanceRun += distance;  
 }  
  
 // Геттери та сеттери для атрибутів класу  
  
 public String getBrand() {  
 return brand;  
 }  
  
 public double getSize() {  
 return size;  
 }  
  
 public String getColor() {  
 return color;  
 }  
  
 public boolean isLaced() {  
 return laced;  
 }  
  
 protected void setLaced(boolean laced) {  
 this.laced = laced;  
 }  
  
 public boolean isWorn() {  
 return worn;  
 }  
  
 protected void setWorn(boolean worn) {  
 this.worn = worn;  
 }  
  
 public double getDistanceRun() {  
 return distanceRun;  
 }  
}

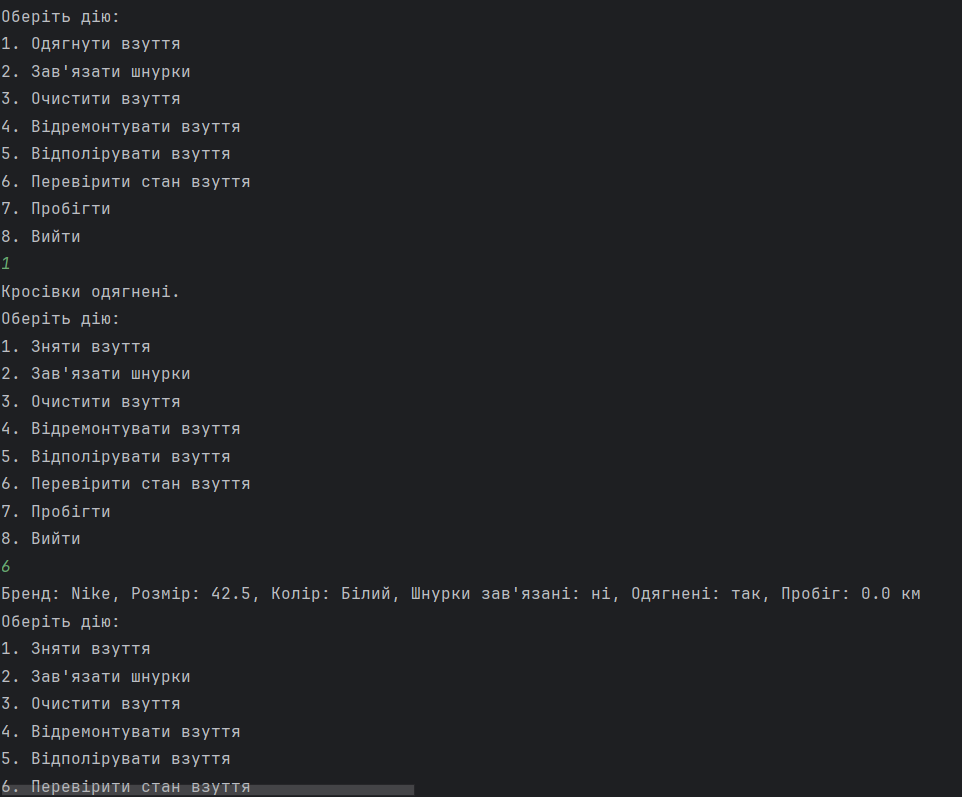
***package Ki\_306.Petruk.Lab3;****/\*\*  
 \* Клас Sneakers, що представляє кросівки. Наслідує клас Shoes та реалізує інтерфейс ShoeOperations.  
 \*/  
public class Sneakers extends Shoes implements ShoeOperations {  
  
 /\*\*  
 \* Конструктор для створення кросівок.  
 \*  
 \* @param brand Бренд кросівок.  
 \* @param size Розмір кросівок.  
 \* @param color Колір кросівок.  
 \*/  
 public Sneakers(String brand, double size, String color) {  
 super(brand, size, color);  
 }  
  
 /\*\* Одягає кросівки, якщо вони не одягнуті. \*/  
 @Override  
 public void wear() {  
 if (isWorn()) {  
 System.out.println("Кросівки вже одягнені.");  
 } else {  
 setWorn(true);  
 System.out.println("Кросівки одягнені.");  
 logOperation("Кросівки одягнені.");  
 }  
 }  
  
 /\*\* Знімає кросівки, якщо вони надягнуті. \*/  
 @Override  
 public void remove() {  
 if (!isWorn()) {  
 System.out.println("Кросівки вже зняті.");  
 } else {  
 setWorn(false);  
 System.out.println("Кросівки зняті.");  
 logOperation("Кросівки зняті.");  
 }  
 }  
  
 /\*\* Очищає кросівки. \*/  
 @Override  
 public void clean() {  
 System.out.println("Кросівки очищені.");  
 logOperation("Кросівки очищені.");  
 }  
  
 /\*\* Відремонтувати кросівки, якщо вони зняті. \*/  
 @Override  
 public void repair() {  
 if (isWorn()) {  
 System.out.println("Спочатку зніміть кросівки, щоб відремонтувати їх.");  
 } else {  
 System.out.println("Кросівки відремонтовані.");  
 logOperation("Кросівки відремонтовані.");  
 addDistance(-100); // Скидаємо пробіг на 100 км після ремонту  
 }  
 }  
  
 /\*\* Відполірувати кросівки, якщо вони зняті. \*/  
 @Override  
 public void polish() {  
 if (isWorn()) {  
 System.out.println("Спочатку зніміть кросівки, щоб відполірувати їх.");  
 } else {  
 System.out.println("Кросівки відполіровані.");  
 logOperation("Кросівки відполіровані.");  
 }  
 }  
  
 /\*\* Зав'язує шнурки, якщо кросівки надягнуті. \*/  
 @Override  
 public void laceUp() {  
 if (!isWorn()) {  
 System.out.println("Спочатку одягніть кросівки, щоб зав'язати шнурки.");  
 } else if (isLaced()) {  
 System.out.println("Шнурки вже зав'язані.");  
 } else {  
 setLaced(true);  
 System.out.println("Шнурки зав'язані.");  
 logOperation("Шнурки зав'язані.");  
 }  
 }  
  
 /\*\* Розв'язує шнурки, якщо вони зав'язані. \*/  
 @Override  
 public void unlace() {  
 if (!isLaced()) {  
 System.out.println("Шнурки вже розв'язані.");  
 } else {  
 setLaced(false);  
 System.out.println("Шнурки розв'язані.");  
 logOperation("Шнурки розв'язані.");  
 }  
 }  
  
 /\*\*  
 \* Перевіряє та виводить поточний стан кросівок.  
 \*/  
 @Override  
 public void checkStatus() {  
 String status = "Бренд: " + getBrand() + ", Розмір: " + getSize() + ", Колір: " + getColor() +  
 ", Шнурки зав'язані: " + (isLaced() ? "так" : "ні") +  
 ", Одягнені: " + (isWorn() ? "так" : "ні") +  
 ", Пробіг: " + getDistanceRun() + " км";  
 if (needsRepair()) {  
 status += " (Потребує ремонту)";  
 }  
 System.out.println(status);  
 logOperation("Перевірено стан: " + status);  
 }  
  
 /\*\*  
 \* Виконує пробіг на задану відстань, якщо кросівки в належному стані.  
 \*  
 \* @param distance Відстань для пробігу в км.  
 \*/  
 public void run(double distance) {  
 if (needsRepair()) {  
 System.out.println("Кросівки потребують ремонту. Спочатку відремонтуйте їх, щоб знову бігати.");  
 return;  
 }  
  
 if (!isLaced()) {  
 System.out.println("Потрібно зав'язати шнурки перед пробігом.");  
 return;  
 }  
  
 addDistance(distance);  
 System.out.println("Ви пробігли " + distance + " км.");  
 logOperation("Пробігли " + distance + " км.");  
 if (needsRepair()) {  
 System.out.println("Кросівки потребують ремонту після пробігу 100 км.");  
 }  
 }  
  
 /\*\*  
 \* Записує проведену операцію в log-файл.  
 \*  
 \* @param operation Опис операції для запису в log-файл.  
 \*/  
 private void logOperation(String operation) {  
 // Тут буде код для запису в log-файл.  
 }  
}*

***ShoesApp.java***

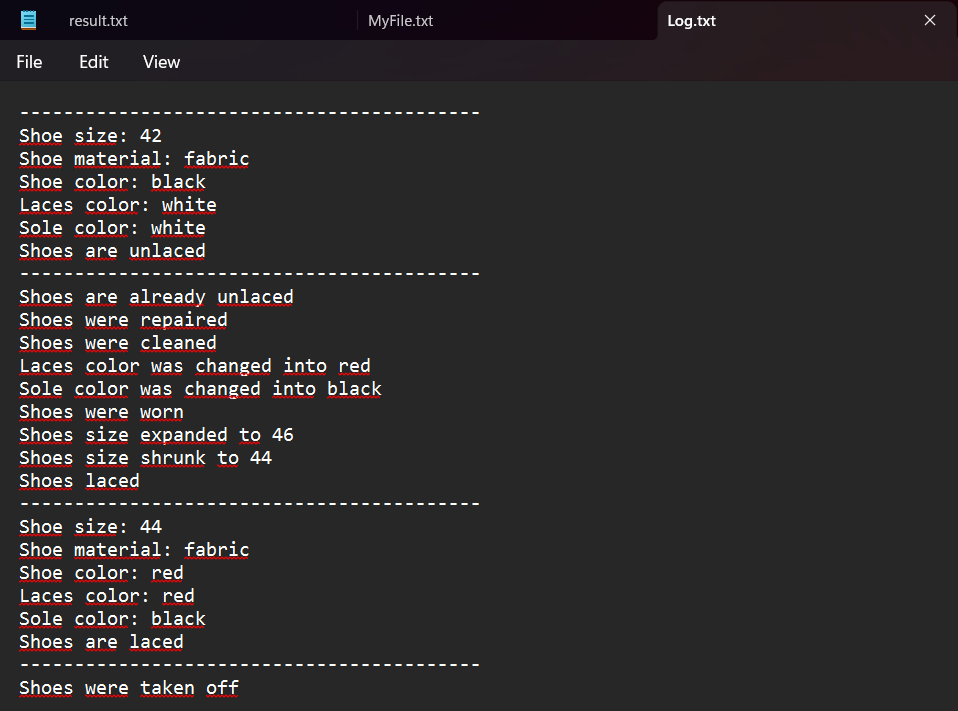
*package Ki\_306.Petruk.Lab3;  
  
import java.util.Scanner;  
  
/\*\*  
 \* Головний клас Main для запуску програми, що керує кросівками.  
 \*/  
public class ShoesApp {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.in);  
 Sneakers sneakers = new Sneakers("Nike", 42.5, "Білий");  
  
 while (true) {  
 System.out.println("Оберіть дію:");  
  
 if (!sneakers.isWorn()) {  
 System.out.println("1. Одягнути взуття");  
 } else {  
 System.out.println("1. Зняти взуття");  
 }  
 if (!sneakers.isLaced()) {  
 System.out.println("2. Зав'язати шнурки");  
 } else {  
 System.out.println("2. Розв'язати шнурки");  
 }  
 System.out.println("3. Очистити взуття");  
 System.out.println("4. Відремонтувати взуття");  
 System.out.println("5. Відполірувати взуття");  
 System.out.println("6. Перевірити стан взуття");  
 System.out.println("7. Пробігти");  
 System.out.println("8. Вийти");  
  
 int choice = scanner.nextInt();  
  
 switch (choice) {  
 case 1 -> {  
 if (!sneakers.isWorn()) {  
 sneakers.wear();  
 } else {  
 sneakers.remove();  
 }  
 }  
 case 2 -> {  
 if (!sneakers.isLaced()) {  
 sneakers.laceUp();  
 } else {  
 sneakers.unlace();  
 }  
 }  
 case 3 -> sneakers.clean();  
 case 4 -> sneakers.repair();  
 case 5 -> sneakers.polish();  
 case 6 -> sneakers.checkStatus();  
 case 7 -> {  
 if (sneakers.isWorn()) {  
 System.out.print("Введіть дистанцію для пробігу (в км): ");  
 double distance = scanner.nextDouble();  
 sneakers.run(distance);  
 } else {  
 System.out.println("Спочатку одягніть кросівки, щоб пробігти.");  
 }  
 }  
 case 8 -> {  
 System.out.println("Вихід з програми.");  
 scanner.close();  
 return;  
 }  
 default -> System.out.println("Невірний вибір, спробуйте ще раз.");  
 }  
 }  
 }  
}*

***ShoesOperations.java***

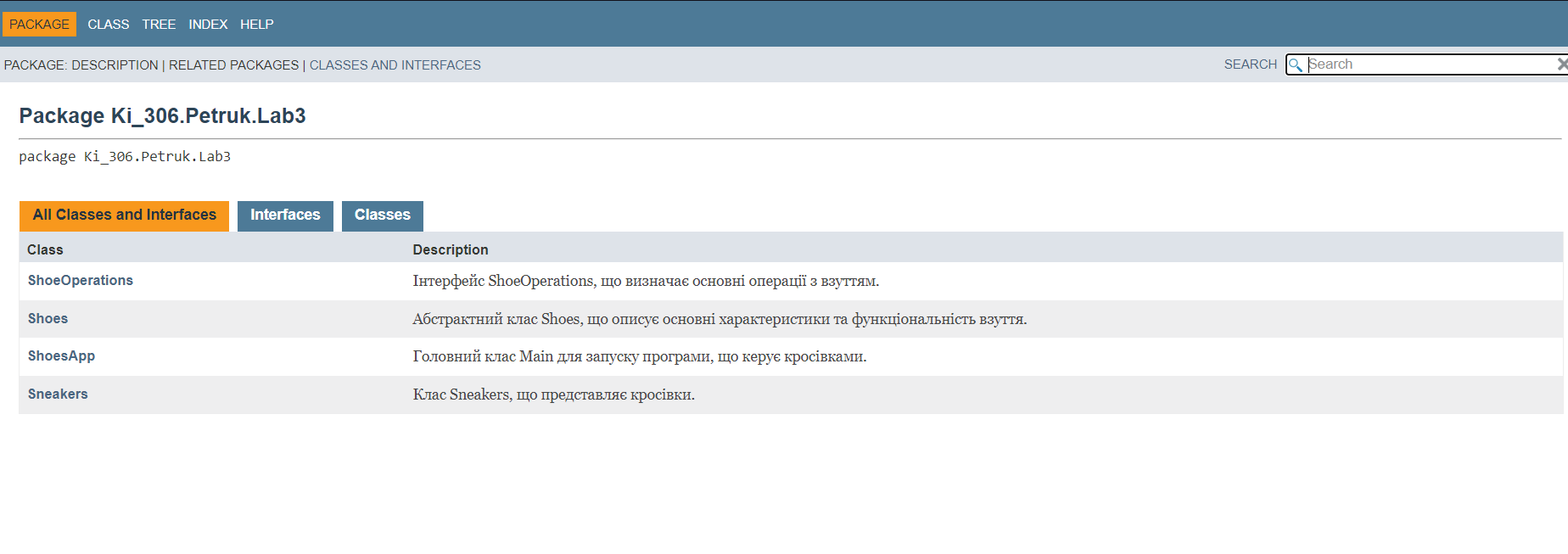
*package Ki\_306.Petruk.Lab3;  
  
/\*\*  
 \* Інтерфейс ShoeOperations, що визначає основні операції з взуттям.  
 \*/  
public interface ShoeOperations {  
 void wear();  
 void remove();  
 void clean();  
 void repair();  
 void polish();  
 void laceUp();  
 void unlace();  
 void checkStatus();  
}*



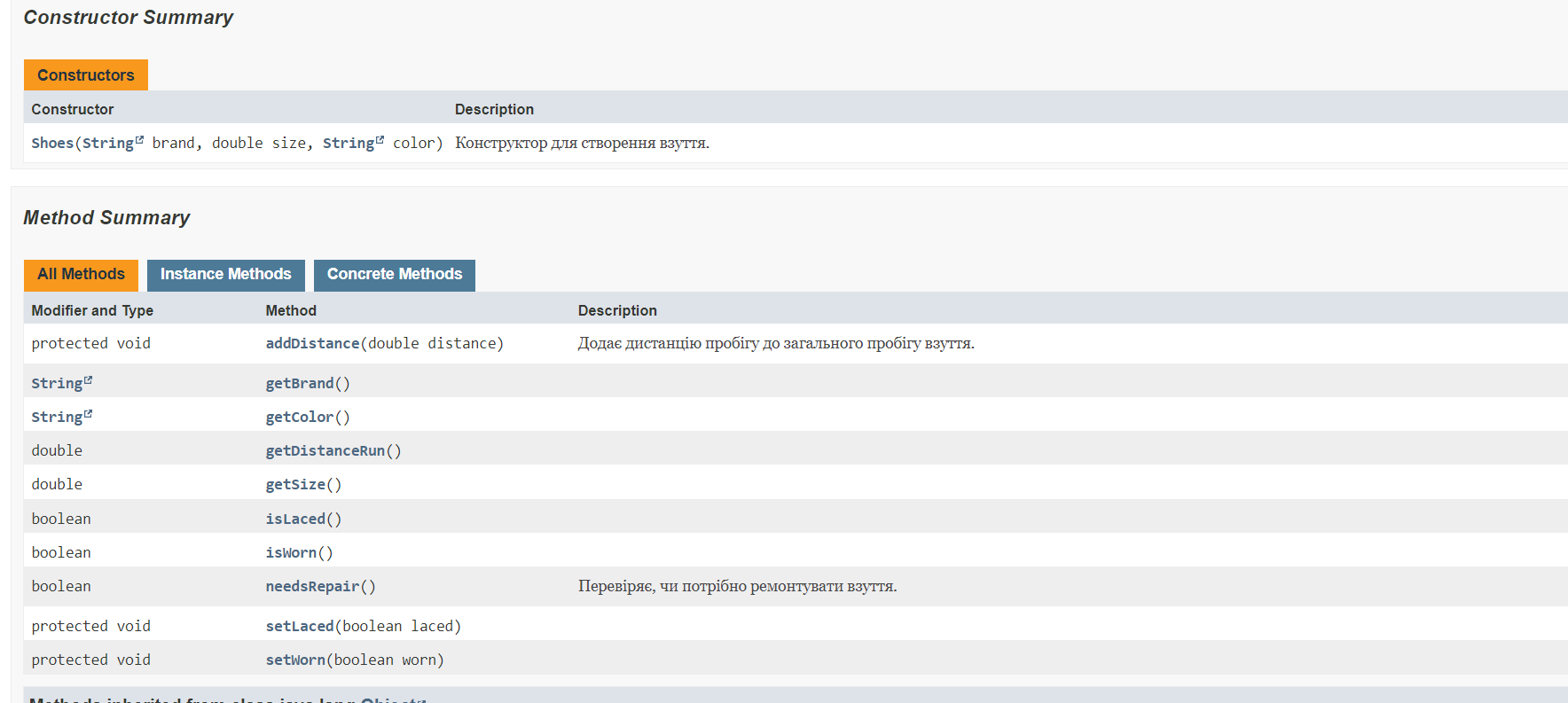
**Рис.1 Вивід логу у консоль**



**Рис.2 Вивід логу у текстовий файл**



**Рис.3.1 Фрагмент згенерованої документації**



**Рис.3.2 Фрагмент згенерованої документації**

**Висновок:** На лабораторній роботі я ознайомився зі спадкуванням та інтерфейсами у мові Java.