Міністерство освіти і науки України

Національний університет “Львівська політехніка”

Кафедра ЕОМ



**Звіт**

З лабораторної роботи №2

Варіант – 24

З дисципліни: «Кросплатформні засоби програмування»

На тему: «Класи та пакети»

Виконав: ст. гр. КІ-301

Чоланюк О.Ю.

Прийняв:

Майдан М.В.

Львів 2023

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

**Теоретичні відомості**

**Класи**

Мова Java є повністю об’єктно-орієнтованою мовою програмування, тому вона дозволяє писати програми лише з використанням об’єктно-орієнтованих парадигм програмування, що базуються на понятті класів.

Синтаксис оголошення простого класу в мові Java має наступний вигляд:

[public] class НазваКласу

{

[конструктори] [методи] [поля]

}

Необов’язковий специфікатор доступу public робить клас загальнодоступним. У кожному файлі з кодом програми може бути лише один загальнодоступний клас, ім’я якого співпадає з назвою файлу, та безліч класів без специфікатора public.

## Методи

*Метод* – функція-член класу, яка призначена маніпулювати станом об’єкту класу. Методи можуть бути перевантаженими. Перевантаження методів відбувається шляхом вказування різної кількості параметрів та їх типів методам з однаковими назвами. Синтаксис оголошення методу наступний:

[СпецифікаторДоступу] [static] [final] Тип назваМетоду([параметри]) [throws класи]

{

[Тіло методу] [return [значення]];

}

Конструктори, методи, та поля класу можуть бути відкритими (public), закритими (private) та захищеними (protected), що визначається специфікатором доступу.

## Конструктори

*Конструктор* – спеціальний метод класу, який не повертає значення, має ім’я класу та призначений для початкової ініціалізації об’єктів класу. Синтаксис оголошення конструктора:

[СпецифікаторДоступу] НазваКласу([параметри])

{

Тіло конструктора

}

## Поля

*Поле* (властивість) – це дані-члени класу, що призначені для зберігання стану об’єкту. Поле може бути статичним (в цьому випадку воно називається *полем класу*), незмінним (*константне поле*), простим типом чи об’єктом та мати різні рівні доступу, що визначаються специфікатором доступу. Допускається ініціалізація поля в місці оголошення. Синтаксис оголошення поля наступний:

[СпецифікаторДоступу] [static] [final] Тип НазваПоля [= ПочатковеЗначення];

**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 розробленої програми.
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.

**Варіант завдання:** 24. Спорядження альпініста.

Файл: Main

package KZP.Lab2.Cholaniuk;

import java.io.FileWriter;

import java.io.IOException;

import java.text.SimpleDateFormat;

import java.util.Date;

// Press Shift twice to open the Search Everywhere dialog and type `show whitespaces`,

// then press Enter. You can now see whitespace characters in your code.

public class Main {

//Fields

private String item;

private double weight;

private String description;

private double price;

private static final String LOG\_File\_Name="gearLog.txt";

public Main(String item, double weight, String description, double price) {

this.item = item;

this.weight = weight;

this.description = description;

this.price = price;

}

public Main(String item, double weight, String description) {

this.item = item;

this.weight = weight;

this.description = description;

}

public Main(String item, double weight) {

this.item = item;

this.weight = weight;

}

public String getItemName() {

return item;

}

public void setWeight(double weight) {

this.weight = weight;

}

public double getWeight() {

return weight;

}

public void setDescription(String description) {

this.description = description;

}

public String getDescription() {

return description;

}

public void setPrice(double price) {

this.price = price;

}

public double getPrice() {

return price;

}

private void logToFile(String message) {

SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

String timestamp = dateFormat.format(new Date());

String logMessage = "[" + timestamp + "] " + message;

try (FileWriter writer = new FileWriter(LOG\_File\_Name, true)) {

writer.write(logMessage + "\n");

} catch (IOException e) {

e.printStackTrace();

}

}

public void logItemToFile(String filename) {

FileWriter writer = null;

try {

writer = new FileWriter(filename, true);

writer.write("Назва: " + item + "\n");

writer.write("Вага: " + weight + " g\n");

writer.write("Ціна: $" + price + "\n");

writer.write("Опис предмету: " + description + "\n\n");

} catch (IOException e) {

e.printStackTrace();

} finally {

try {

if (writer != null) {

writer.close();

}

} catch (IOException e) {

e.printStackTrace();

}

}

logToFile("\n" + "Зареєстрований елемент обладнання: " + item);

}

public void use() {

System.out.println("\n\n" + "Ви використовуєте предмет обладнання: " + item);

System.out.println(item + " ціни " + weight + " грам(и)");

System.out.println(item + " \n" + "витрати $" + price);

System.out.println(item + " має такий опис: " + description);

logToFile("Елемент обладнання, що використовується: " + item);

}

public void inspect() {

System.out.println("\nОгляд обладнання:");

System.out.println("Назва: " + item);

System.out.println("Вага: " + weight + " g");

System.out.println("Ціна: $" + price);

System.out.println("Опис предмету: " + description);

if (weight >= 2000) {

System.out.println("Це спорядження досить важке.");

} else if (weight < 2000 && weight > 500) {

System.out.println("Це обладнання не надто важке.");

} else {

System.out.println("Це легке спорядження, ідеальне для тривалих походів.");

}

if(price == 0){

System.out.println("Це обладнання ви отримали безкоштовно.");

}

else if (price > 600) {

System.out.println("Це обладнання досить дороге.");

} else if (price <= 600 && price > 200) {

System.out.println("Коштує таке обладнання не надто дорого.");

} else {

System.out.println("Це обладнання доступне за нижчою ціною.");

}

if (description.contains("захист від дощу")) {

System.out.println("Це обладнання чудово захищає від дощу.");

}

if (description.contains("для безпеки")) {

System.out.println("Це обладнання ідеально підходить для безпеки.");

}

if (description.contains("легко переносити")) {

System.out.println("Це компактне обладнання, яке зручно носити з собою.");

}

if (description.contains("\n" + "високоякісні матеріали")) {

System.out.println("Дане обладнання має високу якість і надійність.");

}

logToFile("Оглянуте обладнання: " + item);

}

}

Файл: App

package KZP.Lab2.Cholaniuk;

import KZP.Lab2.Cholaniuk.Main;

import java.io.\*;

public class App {

public static void main(String[] args) throws FileNotFoundException {

// Create instances of AlpinistsEquipment to test different constructors

Main gear1 = new Main("Канат для скелелазіння", 2500, "\n" + "Високоякісна альпіністська мотузка", 150);

Main gear2 = new Main("Карабін", 100, "Міцний карабін для безпеки", 20);

Main gear3 = new Main("Скоби", 1800, "Черевики з шипами для льодолазіння");

// Inspect the equipment

gear1.inspect();

gear2.inspect();

gear3.inspect();

// Use the equipment

gear1.use();

gear2.use();

// Log equipment details to a file

String path = "equipment\_log.txt";

gear1.logItemToFile(path);

gear2.logItemToFile(path);

gear3.logItemToFile(path);

}

} 