Міністерство освіти і науки України

Національний університет „Львівська політехніка”

Кафедра “Електронних обчислювальних машин”



**з дисципліни**

**Кросплатформні засоби програмування**

**Звіт з Лабораторної роботи № 2**

**Класи та пакети**

**Виконав:**

КІ-305

Гунька А.А.

**Перевірив:**

Іванов Ю.С

**2023**

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

**Теоретичний вступ**

**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]

}

Приклад оголошення загальнодоступного класу:

public class StartClass

{

public StartClass()

{

str = "Hello";

}

public StartClass(String initString)

{

str = initString;

}

public void showMessage()

{

System.out.print(str);

}

private String str;

}

**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.

**Конструктори**

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

**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.

**Використання пакетів**

Клас може використовувати всі класи з власного пакету і всі загальнодоступні класи з інших пакетів. Доступ до класів з інших пакетів можна отримати двома шляхами:

1. вказуючи повне ім’я пакету перед іменем кожного класу.

2. використовуючи оператор import, що дозволяє підключати як один клас так і всі загальнодоступні класи пакету, позбавляючи необхідності записувати імена класів з вказуванням повної назви пакету перед ними.

**Варіант 6**



**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.

**Код**

/\*\*

\* lab 2 package **Plane.java**

\*/

package KI307.Hereha.Lab2;

import java.io.FileWriter;

import java.io.IOException;

import java.util.Date;

/\*\*

\* Клас що реалізує предметну область Літаку у вигляді класу.

\*/

public class Plane {

// Поля класу, що описують складові частини літака

private String model;

private Engine engine;

private Pilot pilot;

private AircraftFactory factory;

private int passengerCapacity;

// Конструктори

public Plane(String model, Engine engine, Pilot pilot, AircraftFactory factory, int passengerCapacity) {

this.model = model;

this.engine = engine;

this.pilot = pilot;

this.factory = factory;

this.passengerCapacity = passengerCapacity;

}

public Plane(String model, String engineType,

int enginePower, int passengerCapacity,

String firstName, String lastName, int flightCount,

String country, String state, String name

)

{

this.model = model;

this.engine = new Engine(engineType, enginePower);

this.pilot = new Pilot(firstName, lastName, flightCount);

this.factory = new AircraftFactory(country, state, name);

this.passengerCapacity = passengerCapacity;

}

// Методи для доступу до полів класу

public String getModel() {

return model;

}

// Метод класу, що відповідає за отримання типу двигуна

public Engine getEngine() {

return engine;

}

// Метод класу, що відповідає за отримання пілота

public Pilot getPilot() {

return pilot;

}

// Метод класу, що відповідає за отримання авіазаводу

public AircraftFactory getFactory() {

return factory;

}

// Метод класу, що відповідає за отримання кількості пасажирів

public int getPassengerCapacity() {

return passengerCapacity;

}

// Метод для запису протоколу в файл

public void logActivity(String message) {

try (FileWriter writer = new FileWriter("log.txt", true)) {

writer.write(new Date() + ": " + message + "\n");

} catch (IOException e) {

e.printStackTrace();

}

}

// Інші методи класу, що відповідають предметній області

public void takeOff() {

logActivity("Літак " + model + " вилетів.");

}

// Метод класу, що відповідає за логування посадки

public void land() {

logActivity("Літак " + model + " приземлився.");

}

// Метод класу, що відповідає за логування додавання кількості пасажирів

public void addPassengers(int count) {

passengerCapacity += count;

logActivity("До літака " + model + " додано " + count + " пасажирів.");

}

// Інші методи

// Методи, які демонструють роботу класу

public void displayInfo() {

System.out.println("Літак: " + model);

System.out.println("Двигун: " + engine.getType() + ", Потужність: " + engine.getPower());

System.out.println("Пасажиромісткість: " + passengerCapacity);

System.out.println("Пілот: " + pilot.getFullName());

System.out.println("Створено: " + factory.getDataAboutFactory());

}

// Внутрішній клас для представлення двигуна

class Engine {

private String type;

private int power;

public Engine(String type, int power) {

this.type = type;

this.power = power;

}

public String getType() {

return type;

}

public int getPower() {

return power;

}

}

// Внутрішній клас для представлення пілота

class Pilot {

private String firstName;

private String lastName;

private int flightCount;

public Pilot(String firstName, String lastName, int flightCount) {

this.firstName = firstName;

this.lastName = lastName;

this.flightCount = flightCount;

}

public String getFullName() {

return firstName + " " + lastName;

}

public int getExperience() {

return flightCount;

}

}

// Внутрішній клас для представлення авіазаводу

class AircraftFactory{

private String country;

private String state;

private String name;

public AircraftFactory(String country, String state, String name) {

this.country = country;

this.state = state;

this.name = name;

}

public String getDataAboutFactory() {

return country + ", " + state + ", " + name;

}

}

}

/\*\*

\* lab 2 package **PlaneDriver.java**

\*/

package KI307.Hereha.Lab2;

/\*\*

\* Клас драйвер для демонстрації роботи класу літака

\*/

public class PlaneDriver {

public static void main(String[] args) {

Plane plane1 = new Plane("Boeing 747", "Jet", 10000, 400, "Rostyslav", "Hereha", 100, "UA", "Kharkiv", "\r\n"

+ "Kharkiv Aviation Plant");

plane1.displayInfo();

plane1.takeOff();

plane1.land();

plane1.addPassengers(50);

// To create an instance of the inner class Engine, use the outer class instance.

Plane.Engine engine1 = plane1.new Engine("Turbofan", 7500);

Plane.Pilot pilot1 = plane1.new Pilot("Ostap", "Mahora", 0);

Plane.AircraftFactory factory1 = plane1.new AircraftFactory("UA", "Kyiv", "Antonov");

Plane plane2 = new Plane("Airbus A320", engine1, pilot1, factory1, 200);

plane2.displayInfo();

plane2.takeOff();

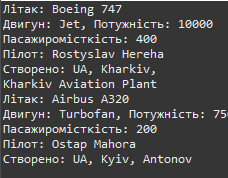
plane2.land();

plane2.addPassengers(30);

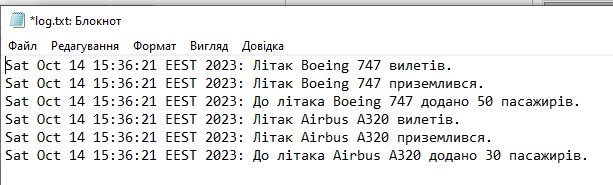
}

}

**Результат:**



**log.txt:**



**Короткий висновок**

Під час виконання даної лабораторної роботи я ознайомився з процесом розробки класів та пакетів мовою Java.