Міністерство освіти і науки України
Національний університет «Львівська політехніка»
Кафедра «Електронних обчислювальних машин»

Звіт
з лабораторної роботи № 2
з дисципліни: «Кросплатформенні засоби програмування»
на тему: «Класи та пакети»

Виконав:
студент групи КІ-306
Довганюк О.С.
Прийняв:
доцент кафедри ЕОМ
Іванов Ю. С.

Львів – 2023

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

Завдання:

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.

Автоматично згенерувати документацію до розробленої програми.

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

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

Дати відповідь на контрольні запитання.

Варіант 6:

6. Літак

Вихідний код програми

Airplane.java

```
package KI306.Dovganiuk.Lab2;

import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.time.LocalTime;
import java.time.temporal.ChronoUnit;

/**
 * Клас, що представляє літак
 */
public class Airplane {
    private String model;
    private Engine engine;
    private Avionics avionics;
    private Body body;
    private double fuelCapacity;
    private boolean isFlying;
    private int currentAltitude;
    private boolean landingGearDeployed;
    private int passengersOnBoard;
    private boolean lightsOn;

    /**
     * Конструктор, що створює літак з заданими параметрами
     * @param model модель літака
     * @param engine двигун літака
     * @param avionics авіоніка літака
     * @param body кабіна літака
     */
    public Airplane(String model, Engine engine, Avionics avionics, Body body) {
        this.model = model;
        this.engine = engine;
        this.avionics = avionics;
        this.body = body;
    }

    /**
     * Конструктор, що створює літак з заданою моделлю
     * @param model модель літака
     */
    public Airplane(String model) {
        this.model = model;
        this.engine = null;
        this.avionics = null;
        this.body = null;
    }

    /**
     * Метод, що викликає взлет літака
     */
    public void takeOff() {
        logAction("Взлетів");
        isFlying = true;
    }

    /**
     * Метод, що викликає приземлення літака
```

```
 */
public void land() {
    logAction("Приземлився");
    logAction("Приземлення успішне!!!! \n==========================================");
    isFlying = false;
}

/**
 * Метод, що змінює висоту польоту літака
 * @param altitude висота польоту
 */
public void fly(int altitude) {
    if (isFlying) {
        currentAltitude = altitude;
        logAction("Летить на висоті " + altitude + " метрів");
    } else {
        logAction("Літак не в повітрі. Неможливо змінити висоту.");
    }
}

/**
 * Метод, що встановлює двигун літака
 * @param engine двигун літака
 */
public void setEngine(Engine engine) {
    this.engine = engine;
    logAction("Двигун встановлено");
}

/**
 * Метод, що видаляє двигун з літака
 */
public void removeEngine() {
    this.engine = null;
    logAction("Двигун видалено");
}

/**
 * Метод, що висунути шасі
 */
public void deployLandingGear() {
    if (isFlying) {
        landingGearDeployed = true;
        logAction("Висування шасі");
    } else {
        logAction("Літак не в повітрі. Неможливо висунути шасі.");
    }
}

/**
 * Метод, що закрити шасі
 */
public void retractLandingGear() {
    if (isFlying) {
        landingGearDeployed = false;
        logAction("Закриття шасі");
    } else {
        logAction("Літак не в повітрі. Неможливо закрити шасі.");
```

```
    }
}

/**
 * Метод, що додає пасажирів на борт
 * @param count кількість пасажирів
 */
public void addPassengers(int count) {
    if (!isFlying || body.seats >= count) {
        passengersOnBoard += count;
        logAction("Пасажирів на борту: " + passengersOnBoard);
    } else if (isFlying) {
        logAction("Літак у повітрі. Неможливо додати пасажирів.");
    } else {
        logAction("Літак вміщує не більше " + body.seats + " пасажирів");
    }
}

/**
 * Метод, що увімкне освітлення
 */
public void turnOnLights() {
    lightsOn = true;
    logAction("Увімкнено освітлення");
}

/**
 * Метод, що вимкне освітлення
 */
public void turnOffLights() {
    lightsOn = false;
    logAction("Вимкнено освітлення");
}

/**
 * Метод, що поповнює пальне
 * @param liters кількість літрів пального
 */
public void refuel(double liters) {
    fuelCapacity += liters;
    logAction("Поповнено пальне. Поточний об'єм пального: " + fuelCapacity + " л");
}

/**
 * Метод, що записує дію в файл логу
 * @param action дія
 */
public void logAction(String action) {
    try {
        BufferedWriter writer = new BufferedWriter(new FileWriter("log_lab2.txt", true));
        writer.write(LocalTime.now().truncatedTo(ChronoUnit.SECONDS)+"   "+model + ": " + action);
        writer.newLine();
        writer.close();
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

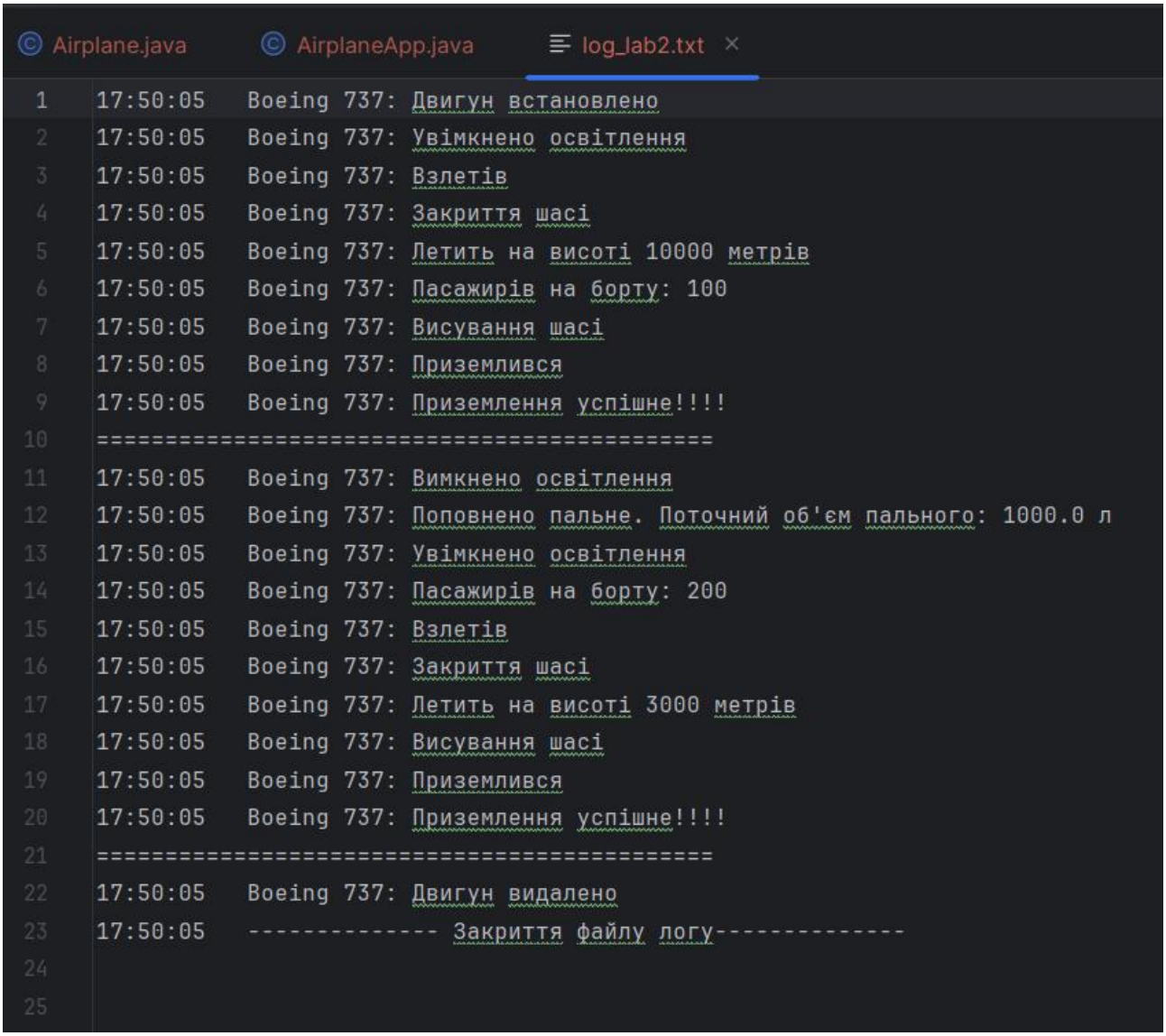
```
    /**
     * Клас, що представляє двигун
     */
    public static class Engine {
        private String type;

        /**
         * Конструктор, що створює двигун з заданим типом
         * @param type тип двигуна
         */
        public Engine(String type) {
            this.type = type;
        }
    }

    /**
     * Клас, що представляє авіоніку
     */
    public static class Avionics {
        private String equipment;

        /**
         * Конструктор, що створює авіоніку з заданим обладнанням
         * @param equipment обладнання авіоніки
         */
        public Avionics(String equipment) {
            this.equipment = equipment;
        }
    }

    /**
     * Клас, що представляє кабіну
     */
    public static class Body {
        private int seats;

        /**
         * Конструктор, що створює кабіну з заданою кількістю місць
         * @param seats кількість місць
         */
        public Body(int seats) {
            this.seats = seats;
        }
    }

/**
 * Метод, що закриває файл
 */
    public static void closeLogFile() {
        //logAction("--------------File Close----------------");
        try {
            BufferedWriter writer = new BufferedWriter(new FileWriter("log_lab2.txt", true));
            writer.write(LocalTime.now().truncatedTo(ChronoUnit.SECONDS) + "    " + "---------- Закриття
файлу логу----------\n");
            writer.newLine();
            writer.close();
        } catch (IOException e) {
            e.printStackTrace();
```

```
    }
  }
}
```

AirplaneApp.java

```
package KI306.Dovganiuk.Lab2;

/**
 * lab 2 package
 */
public class AirplaneApp {
  public static void main(String[] args) {
    // Приклад використання класу Airplane

    Airplane.Engine engine = new Airplane.Engine("Jet Engine");
    Airplane.Avionics avionics = new Airplane.Avionics("Advanced Avionics");
    Airplane.Body body = new Airplane.Body(250);

    Airplane airplane1 = new Airplane("Boeing 737", engine, avionics, body);
    airplane1.setEngine(engine);
    airplane1.turnOnLights();
    airplane1.takeOff();
    airplane1.retractLandingGear();
    airplane1.fly(10000);
    airplane1.addPassengers(100);
    airplane1.deployLandingGear();
    airplane1.land();
    airplane1.turnOffLights();

    airplane1.refuel(1000);
    airplane1.turnOnLights();
    airplane1.addPassengers(100);

    airplane1.takeOff();
    airplane1.retractLandingGear();
    airplane1.fly(3000);
    airplane1.deployLandingGear();
    airplane1.land();

    airplane1.removeEngine();
    airplane1.closeLogFile();
  }
}
```

Текстовий файл з результатом виконання програми



Висновок

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