Міністерство освіти і науки України

Національний університет “Львівська політехніка”

Кафедра ЕОМ

Звіт

З лабораторної роботи №2

Варіант – 9

З дисципліни: «Кросплатформні засоби програмування»

На тему: «Класи та пакети»

Виконав: ст. гр. КІ-305

Заставний Р.А.

Прийняв:

доцент кафедри ЕОМ

Іванов Ю. С.

Львів-2023

Мета роботи: Ознайомитися з базовими конструкціями мови Java та оволодіти навиками написання й автоматичного документування простих консольних програм 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.Lab3`;
   - 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:

9. Рослина

Код програми:

Main.java

```
package KI305.Zastavnyi.Lab2;

import java.io.IOException;
/**
 *
 * Main class implements main method for Ship class possibilities demonstration
 *
 * @author Roman Zastavnyi
 *
 */
public class Main {
	public static void main(String[] args) throws IOException {
		// Створення екземпляру класу Plant
		Plant plant = new Plant();


		plant.getStatus(); // Виклик методу інформації про рослину
		plant.Death(); //Виклик методу смерті рослини

		// Створення іншого екземпляру класу Plant з конкретними параметрами
		Plant tree = new Plant("Дуб", "Коричневий", 100);

		// Виклик методів та встановлення властивостей для екземпляру tree
		tree.getStatus();// Виклик методу інформації про рослину
		tree.photosynthesis(10);// Спроба фотосинтезу з кількістю світла 10
		int hum = 10;
		tree.setHumidity(hum); // Встановлення вологості в рослині, у кількості 10
		System.out.println("Встановлено " + hum + " вологості у рослині: " +
tree.getName());
```

```
            tree.photosynthesis(11);// Спроба фотосинтезу з кількістю світла 11
            System.out.println();
            tree.getStatus();
            tree.finish();

            // Створення екземпляру класу Plant з більшою кількістю параметрів
            Plant plant1 = new Plant("Яблуня", new Size(1, 2, 3), "Зелений",
                    new Fruit(true, "Зелений", "Стиглі"), new Leaf("Серцеподібна", true,
"Зелений"), 16, 10);


            plant1.getStatus();
            System.out.println("Назва рослини plant1: "+plant1.getName()); // Вивід назви
рослини plant1
            plant1.finish();
        }
}
```

Plant.java

```
/**
 * Пакет, який містить класи для реалізації рослини.
 */
package KI305.Zastavnyi.Lab2;

import java.io.*;
import static java.lang.System.out;

/**
 * Клас, що представляє рослину. Рослина має ім'я, розмір, колір, плід, лист із вологістю,
кількість листків,
 * та можливість виводу інформації про стан рослини.
 * Реалізує різні конструктори для зручності створення об'єктів рослини.
 *
 * @author Roman Zastavnyi
 * @version 1.0
 */
public class Plant {
    private String name;         // Ім'я рослини
    private Size size;           // Розмір рослини (об'єкт класу Size)
    private String color;        // Колір рослини
    private Fruit fruit;         // Плід рослини (об'єкт класу Fruit)
    private Leaf leaf;           // Лист рослини (об'єкт класу Leaf)
    private double humidity;     // Вологість рослини
    private int countOfLeaves;   // Кількість листків рослини
    private PrintWriter fout;    // PrintWriter для запису в файл

    /**
     * Конструктор за замовчуванням. Ініціалізує всі властивості рослини значеннями за
замовчуванням
     * та створює об'єкт PrintWriter для запису в файл "Log.txt".
     *
     * @throws IOException Якщо виникає помилка при роботі з файлом
     */
    public Plant() throws IOException {
        this.name = "Немає";
        this.size = new Size();
        this.color = "Немає";
        this.fruit = new Fruit();
        this.leaf = new Leaf();
        this.humidity = 0;
        this.countOfLeaves = 0;
```

```
        fout = new PrintWriter(new FileWriter("Log.txt", true));
    }

    /**
     * Конструктор, який дозволяє встановити ім'я, колір та кількість листків рослини при створенні об'єкта.
     * Ініціалізує розмір за замовчуванням, а інші властивості значеннями за замовчуванням.
     * Створює об'єкт PrintWriter для запису в файл "Log.txt".
     *
     * @param _name         Ім'я рослини
     * @param _color        Колір рослини
     * @param countOfLeaves  Кількість листків рослини
     * @throws IOException   Якщо виникає помилка при роботі з файлом
     */
    public Plant(String _name, String _color, int countOfLeaves) throws IOException {
        this.name = _name;
        this.size = new Size();
        this.color = _color;
        this.fruit = new Fruit();
        this.leaf = new Leaf(true);
        this.humidity = 0;
        this.countOfLeaves = countOfLeaves;
        fout = new PrintWriter(new FileWriter("Log.txt", true));
    }

    /**
     * Конструктор, який дозволяє встановити всі властивості рослини при створенні об'єкта.
     * Створює об'єкт PrintWriter для запису в файл "Log.txt".
     *
     * @param name          Ім'я рослини
     * @param size          Розмір рослини (об'єкт класу Size)
     * @param color         Колір рослини
     * @param fruit         Плід рослини (об'єкт класу Fruit)
     * @param leaf          Лист рослини (об'єкт класу Leaf)
     * @param humidity      Вологість рослини
     * @param countOfLeaves  Кількість листків рослини
     * @throws IOException   Якщо виникає помилка при роботі з файлом
     */
    public Plant(String name, Size size, String color, Fruit fruit, Leaf leaf, double humidity, int countOfLeaves) throws IOException {
        this.name = name;
        this.size = size;
        this.color = color;
        this.fruit = fruit;
        this.leaf = leaf;
        this.humidity = humidity;
        this.countOfLeaves = countOfLeaves;
        fout = new PrintWriter(new FileWriter("Log.txt", true));
    }

    // Методи для отримання та встановлення властивостей рослини

    /**
     * Метод, що повертає розмір рослини.
     *
     * @return Розмір рослини (об'єкт класу Size)
     */
    public Size getSize() {
        return size;
    }

    /**
```

```java
     * Метод, що встановлює розмір рослини.
     *
     * @param size Розмір рослини (об'єкт класу Size)
     */
    public void setSize(Size size) {
        this.size = size;
    }

    /**
     * Метод, що повертає плід рослини.
     *
     * @return Плід рослини (об'єкт класу Fruit)
     */
    public Fruit getFruit() {
        return fruit;
    }

    /**
     * Метод, що встановлює плід рослини.
     *
     * @param fruit Плід рослини (об'єкт класу Fruit)
     */
    public void setFruit(Fruit fruit) {
        this.fruit = fruit;
    }

    /**
     * Метод, що повертає лист рослини.
     *
     * @return Лист рослини (об'єкт класу Leaf)
     */
    public Leaf getLeaf() {
        return leaf;
    }

    /**
     * Метод, що встановлює лист рослини.
     *
     * @param leaf Лист рослини (об'єкт класу Leaf)
     */
    public void setLeaf(Leaf leaf) {
        this.leaf = leaf;
    }

    /**
     * Метод, що встановлює ім'я рослини.
     *
     * @param name Ім'я рослини
     */
    public void setName(String name) {
        this.name = name;
    }

    /**
     * Метод, що встановлює колір рослини.
     *
     * @param color Колір рослини
     */
    public void setColor(String color) {
        this.color = color;
    }

    /**
     * Метод, що встановлює вологість рослини.
```

```
     *
     * @param humidity Вологість рослини
     */
    public void setHumidity(double humidity) {
        this.humidity = humidity;
    }

    /**
     * Метод, що встановлює кількість листків рослини.
     * Якщо передане значення більше 0, то встановлює його. Якщо дорівнює 0, то встановлює його
     * та стан листя на наявний. В іншому випадку виводить повідомлення про неправильне значення.
     *
     * @param countOfLeaves Кількість листків рослини
     */
    public void setCountOfLeaves(int countOfLeaves) {
        if (countOfLeaves > 0)
            this.countOfLeaves = countOfLeaves;
        else if (countOfLeaves == 0) {
            this.countOfLeaves = countOfLeaves;
            this.leaf.setState(true);
        } else {
            out.println("Вказано неправильне значення кількості листків");
        }
    }

    /**
     * Метод, що повертає ім'я рослини.
     *
     * @return Ім'я рослини
     */
    public String getName() {
        return name;
    }

    /**
     * Метод, що повертає колір рослини.
     *
     * @return Колір рослини
     */
    public String getColor() {
        return color;
    }

    /**
     * Метод, що повертає вологість рослини.
     *
     * @return Вологість рослини
     */
    public double getHumidity() {
        return humidity;
    }

    /**
     * Метод, що повертає кількість листків рослини.
     *
     * @return Кількість листків рослини
     */
    public int getCountOfLeaves() {
        return countOfLeaves;
    }

    /**
```

```java
    * Метод, що сприяє росту рослини. Змінює розмір рослини на більший та збільшує кі-
лькість листків на одиницю.
    * Для росту рослини необхідна наявність вологи.
    */
    public void GrowUp() {
        if (this.humidity > 0) {
            size.changeSizeToGrow();
            setCountOfLeaves(this.countOfLeaves + 1);
        } else {
            out.println("Рослина не може рости: недостатньо вологості.");
        }
    }

    /**
    * Метод, що виводить інформацію про стан рослини на консоль та у файл за допомо-
гою PrintWriter.
    */
    public void getStatus() {
        out.println("Рослина: " + this.name);
        out.println("Колір: " + this.color);
        out.println("Вологість: " + this.humidity);

        fout.println("Рослина: " + this.name);
        fout.println("Колір: " + this.color);
        fout.println("Вологість: " + this.humidity);

        size.getStatus(fout);

        out.println("Кількість листків: " + this.countOfLeaves);

        fout.println("Кількість листків: " + this.countOfLeaves);

        leaf.getStatus(fout);
        fruit.getStatus(fout);
        out.println();
        fout.append("\n");
    }

    /**
    * Метод, що проводить процес фотосинтезу в рослині. Результатом є виділення кисню.
    *
    * @param countLight Кількість світла для фотосинтезу
    */
    public void photosynthesis(double countLight) {
        if (countLight > 0 && this.humidity >= countLight * 0.1 && this.humidity > 0 &&
this.countOfLeaves > 0) {
            this.humidity -= countLight * 0.1;
            out.println("Сфотосинтезовано " + countLight * 1.2 + " кисню.");
        } else if (this.humidity <= 0) {
            out.println("Недостатньо вологості для фотосинтезу.");
        } else if (this.countOfLeaves <= 0) {
            out.println("Недостатньо листя для фотосинтезу.");
        } else {
            out.println("Недостатньо світла для фотосинтезу.");
        }
    }


    /**
    * Метод, що викликається при смерті рослини. Змінює розмір рослини на "Смерть",
встановлює колір на "Немає",
    * кількість листків на 0, вологість на 0 та завершує запис у файл журналу.
    */
    public void Death() {
```

```
        size.changeSizeToDeath();
        setColor("Немає");
        setCountOfLeaves(0);
        setHumidity(0);
        finish();
    }

    /**
     * Метод, що завершує запис у файл журналу. Закриває PrintWriter, який використовується для запису в файл.
     */
    public void finish() {
        fout.close();
    }
}
```

ad

Fruit.java

```
package KI305.Zastavnyi.Lab2;

import java.io.PrintWriter;

import static java.lang.System.out;

/**
 * Клас, що представляє об'єкт "Плід". Кожен плід може мати стан (наявний або відсутній), колір та ступінь стиглості.
 * Для спрощення можливості створення об'єктів за замовчуванням та з заданими параметрами є конструктори.
 */
public class Fruit {
    private boolean existence;  // Стан плоду (наявний - true, відсутній - false)
    private String color;       // Колір плоду
    private String ripeness;    // Стиглість плоду

    /**
     * Конструктор за замовчуванням. Встановлює всі властивості плоду на значення "Немає".
     */
    public Fruit() {
        this.existence = false;
        this.color = "Немає";
        this.ripeness = "Немає";
    }

    /**
     * Конструктор, який дозволяє встановити стан, колір та ступінь стиглості плоду при створенні об'єкта.
     *
     * @param existence Стан плоду (наявний - true, відсутній - false)
     * @param color     Колір плоду
     * @param ripeness  Стиглість плоду
     */
    public Fruit(boolean existence, String color, String ripeness) {
        this.existence = existence;
        this.color = color;
        this.ripeness = ripeness;
    }

    /**
     * Метод, що повертає стан плоду.
     *
```

```java
     * @return Стан плоду (наявний - true, відсутній - false)
     */
    public boolean isExistence() {
        return existence;
    }

    /**
     * Метод, що встановлює стан плоду.
     *
     * @param existence Стан плоду (наявний - true, відсутній - false)
     */
    public void setExistence(boolean existence) {
        this.existence = existence;
    }

    /**
     * Метод, що повертає колір плоду.
     *
     * @return Колір плоду
     */
    public String getColor() {
        return color;
    }

    /**
     * Метод, що встановлює колір плоду.
     *
     * @param color Колір плоду
     */
    public void setColor(String color) {
        this.color = color;
    }

    /**
     * Метод, що повертає ступінь стиглості плоду.
     *
     * @return Стиглість плоду
     */
    public String getRipeness() {
        return ripeness;
    }

    /**
     * Метод, що встановлює ступінь стиглості плоду.
     *
     * @param ripeness Стиглість плоду
     */
    public void setRipeness(String ripeness) {
        this.ripeness = ripeness;
    }

    /**
     * Метод, що виводить статус плоду на консоль та у файл за допомогою PrintWriter.
     *
     * @param fout PrintWriter об'єкт для запису у файл
     */
    public void getStatus(PrintWriter fout) {
        if (this.existence) {
            out.println("Плоди: Є");
            out.println("Колір: " + this.color);
            out.println("Стиглість: " + this.ripeness);

            fout.println("Плоди: Є");
            fout.println("Колір: " + this.color);
```

```
            fout.println("Стиглість: " + this.ripeness);
        } else {
            out.println("У рослини немає плодів.");
            fout.println("У рослини немає плодів.");
        }
    }
}
```

Leaf.java

Leaf.java

```
package KI305.Zastavnyi.Lab2;

import java.io.PrintWriter;

import static java.lang.System.out;

/**
 * Клас, що представляє об'єкт "Лист". Кожен лист може мати форму, стан (наявний або відсутній) та колір.
 * Для спрощення можливості створення об'єктів за замовчуванням та з заданими параметрами є конструктори.
 */
public class Leaf {
    private String shape;  // Форма листя
    private boolean state; // Наявність листя (наявний - true, відсутній - false)
    private String color;  // Колір листя

    /**
     * Конструктор за замовчуванням. Встановлює всі властивості листя на значення "Немає".
     */
    public Leaf() {
        this.shape = "Немає";
        this.state = false;
        this.color = "Немає";
    }

    /**
     * Конструктор, який дозволяє встановити стан листя при створенні об'єкта.
     *
     * @param state Стан листя (наявний - true, відсутній - false)
     */
    public Leaf(boolean state) {
        this.shape = "Немає";
        this.state = state;
        this.color = "Немає";
    }

    /**
     * Конструктор, який дозволяє встановити форму, стан та колір листя при створенні об'єкта.
     *
     * @param shape Форма листя
     * @param state Стан листя (наявний - true, відсутній - false)
     * @param color Колір листя
     */
    public Leaf(String shape, boolean state, String color) {
        this.shape = shape;
        this.state = state;
        this.color = color;
    }
```

```java
/**
 * Метод, що повертає форму листя.
 *
 * @return Форма листя
 */
public String getShape() {
    return shape;
}

/**
 * Метод, що встановлює форму листя.
 *
 * @param shape Форма листя
 */
public void setShape(String shape) {
    this.shape = shape;
}

/**
 * Метод, що повертає стан листя.
 *
 * @return Стан листя (наявний - true, відсутній - false)
 */
public boolean getState() {
    return state;
}

/**
 * Метод, що встановлює стан листя.
 *
 * @param state Стан листя (наявний - true, відсутній - false)
 */
public void setState(boolean state) {
    this.state = state;
}

/**
 * Метод, що повертає колір листя.
 *
 * @return Колір листя
 */
public String getColor() {
    return color;
}

/**
 * Метод, що встановлює колір листя.
 *
 * @param color Колір листя
 */
public void setColor(String color) {
    this.color = color;
}

/**
 * Метод, що виводить статус листя на консоль та у файл за допомогою PrintWriter.
 *
 * @param fout PrintWriter об'єкт для запису у файл
 */
public void getStatus(PrintWriter fout) {
    if (state) {
        out.println("Листя: Є");
        fout.println("Листя: Є");
    } else {
```

```
            out.println("Листя: Немає");
            fout.println("Листя: Немає");
        }
        out.println("Форма: " + this.shape);
        out.println("Колір: " + this.color);

        fout.println("Форма: " + this.shape);
        fout.println("Колір: " + this.color);
    }
}
```

Size.java

Size.java

```
package KI305.Zastavnyi.Lab2;

import java.io.PrintWriter;

import static java.lang.System.out;
/**
 * Клас, що представляє розміри об'єкта. Кожен розмір може мати ширину, висоту та довжину.
 * Для спрощення можливості створення об'єктів за замовчуванням та з заданими параметрами є конструктори.
 */
public class Size {
    private double width;   // Ширина
    private double height;  // Висота
    private double length;  // Довжина

    /**
     * Конструктор за замовчуванням. Встановлює всі властивості розміру на значення 0.
     */
    public Size() {
        this.width = 0;
        this.height = 0;
        this.length = 0;
    }

    /**
     * Конструктор, який дозволяє встановити ширину, висоту та довжину розміру при створенні об'єкта.
     *
     * @param width  Ширина
     * @param height Висота
     * @param length Довжина
     */
    public Size(double width, double height, double length) {
        this.width = width;
        this.height = height;
        this.length = length;
    }

    /**
     * Метод, що повертає ширину розміру.
     *
     * @return Ширина
     */
    public double getWidth() {
        return width;
    }

    /**
     * Метод, що встановлює ширину розміру.
```

```java
     *
     * @param width Ширина
     */
    public void setWidth(double width) {
        this.width = width;
    }

    /**
     * Метод, що повертає висоту розміру.
     *
     * @return Висота
     */
    public double getHeight() {
        return height;
    }

    /**
     * Метод, що встановлює висоту розміру.
     *
     * @param height Висота
     */
    public void setHeight(double height) {
        this.height = height;
    }

    /**
     * Метод, що повертає довжину розміру.
     *
     * @return Довжина
     */
    public double getLength() {
        return length;
    }

    /**
     * Метод, що встановлює довжину розміру.
     *
     * @param length Довжина
     */
    public void setLength(double length) {
        this.length = length;
    }

    /**
     * Метод, що виводить статус розміру на консоль та у файл за допомогою PrintWriter.
     *
     * @param fout PrintWriter об'єкт для запису у файл
     */
    public void getStatus(PrintWriter fout) {
        out.println("Ширина: " + this.width);
        out.println("Довжина: " + this.length);
        out.println("Висота: " + this.height);

        fout.println("Ширина: " + this.width);
        fout.println("Довжина: " + this.length);
        fout.println("Висота: " + this.height);
    }

    /**
     * Метод, що змінює розмір на зростання.
     */
    public void changeSizeToGrow() {
        setWidth(this.width + 1.1);
        setHeight(this.height + 1.1);
```

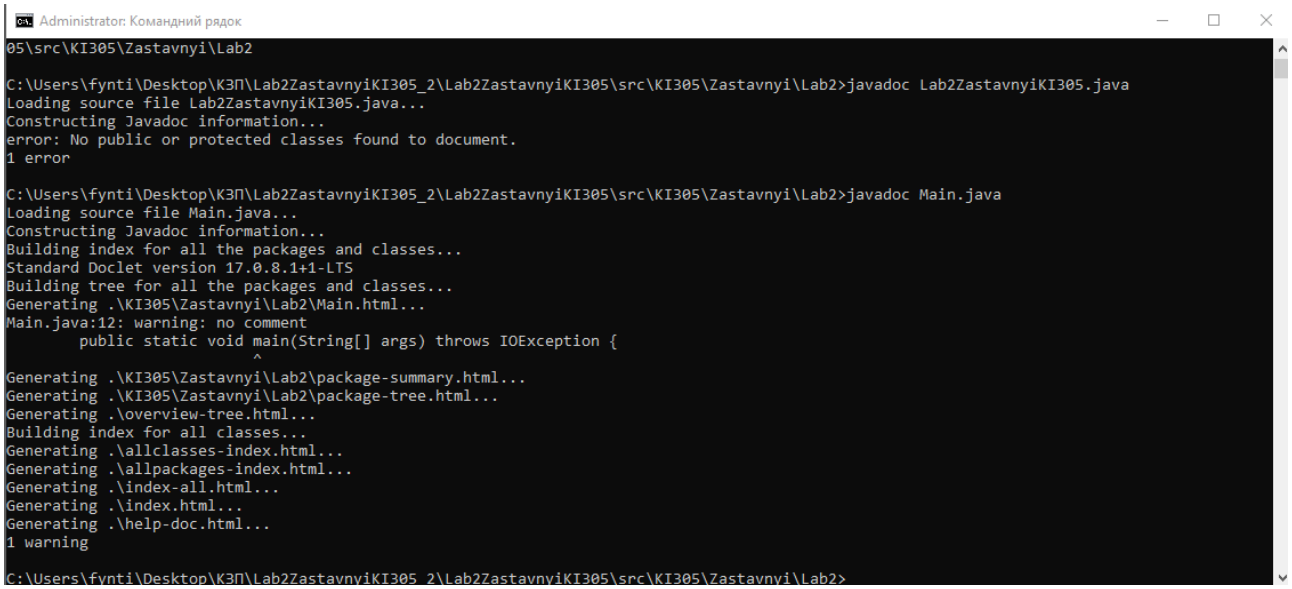
```
        setLength(this.length + 1.1);
    }

    /**
     * Метод, що змінює розмір на "Смерть", встановлюючи всі властивості розміру на значення 0.
     */
    public void changeSizeToDeath() {
        setWidth(0);
        setHeight(0);
        setLength(0);
    }
}
```

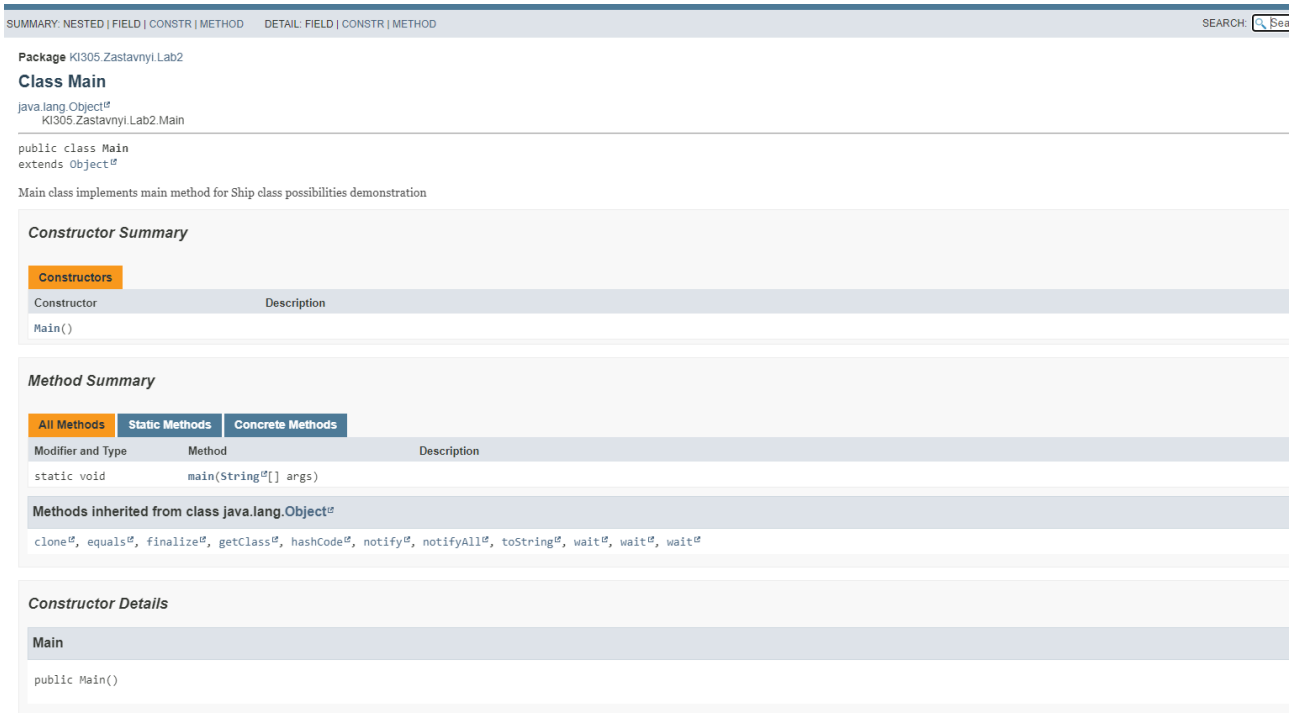
Результата роботи програми:

```
"C:\Program Files\Microsoft\jdk-17.0.8.1
Рослина: Немає
Колір: Немає
Вологість: 0.0
Ширина: 0.0
Довжина: 0.0
Висота: 0.0
Кількість листків: 0
Листя: Немає
Форма: Немає
Колір: Немає
У рослини немає плодів.

Рослина: Дуб
Колір: Коричневий
Вологість: 0.0
Ширина: 0.0
Довжина: 0.0
Висота: 0.0
Кількість листків: 100
Листя: Є
Форма: Немає
Колір: Немає
У рослини немає плодів.

Недостатньо вологості для фотосинтезу.
Встановлено 10 вологості у рослині: Дуб
Сфотосинтезовано 13.2 кисню.
```

```
Рослина: Дуб
Колір: Коричневий
Вологість: 8.9
Ширина: 0.0
Довжина: 0.0
Висота: 0.0
Кількість листків: 100
Листя: Є
Форма: Немає
Колір: Немає
У рослини немає плодів.

Рослина: Яблуня
Колір: Зелений
Вологість: 16.0
Ширина: 1.0
Довжина: 3.0
Висота: 2.0
Кількість листків: 10
Листя: Є
Форма: Серцеподібна
Колір: Зелений
Плоди: Є
Колір: Зелений
Стиглість: Стиглі

Назва рослини plant1: Яблуня

Process finished with exit code 0
```

Фрагмент згенерованої документації





SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD SEARCH:

Package KI305.Zastavnyi.Lab2

Class Main

java.lang.Object
KI305.Zastavnyi.Lab2.Main

```
public class Main
extends Object
```

Main class implements main method for Ship class possibilities demonstration

Constructor Summary

Constructors

Constructor	Description
Main()	

Method Summary

All Methods | Static Methods | Concrete Methods

Modifier and Type	Method	Description
static void	main(String[] args)	

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Details

Main

```
public Main()
```

Висновок: Я ознайомився з базовими конструкціями мови Java та оволодів навиками написання й автоматичного документування простих консольних програм мовою Java.