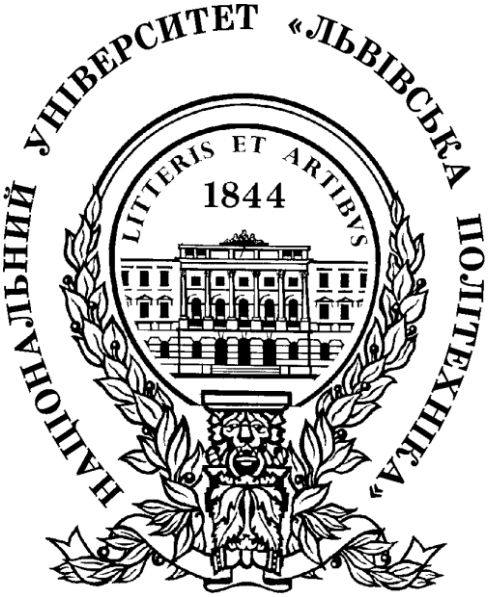
Міністерство освіти і науки України

Національний університет “Львівська політехніка”

Кафедра ЕОМ



ЗВІТ

до лабораторної роботи №6

З дисципліни: «Кросплатформні засоби програмування»

На тему: «ПАРАМЕТРИЗОВАНЕ ПРОГРАМУВАННЯ»

Варіант 16

Виконав:

ст. групи КІ-306

Мілян М.О.

Прийняв:

доцент кафедри ЕОМ

Олексів М.В.

Львів – 2024

Мета: оволодіти навиками параметризованого програмування мовою Java.

Завдання:
1. Створити параметризований клас, що реалізує предметну область задану варіантом. Клас має містити мінімум 4 методи опрацювання даних включаючи розміщення та виймання елементів. Парні варіанти реалізують пошук мінімального елементу, непарні – максимального. Написати на мові Java та налагодити програму-драйвер для розробленого класу, яка мстить мінімум 2 різні класи екземпляри яких розмішуються у екземплярі розробленого класу-контейнеру. Програма має розміщуватися в пакеті Група.Прізвище.Lab6 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.

Завдання згідно варіанту №16 – « Земельна ділянка »

Хід роботи

Код програми:
Driver.java

```
package KI.Milian.Lab6;

/**
 * Головний клас для демонстрації функціональності LandPlotContainer
 */
public class Driver {
    public static void main(String[] args) {
        // Створюємо контейнер для житлових ділянок
        LandPlotContainer<ResidentialPlot> residentialContainer = new
LandPlotContainer<>();

        // Додаємо житлові ділянки
        residentialContainer.addPlot(new ResidentialPlot("вул. Шевченка 1", 500,
100000));
        residentialContainer.addPlot(new ResidentialPlot("вул. Франка 5", 700,
150000));
        residentialContainer.addPlot(new ResidentialPlot("вул. Лесі Українки
10", 400, 80000));

        // Знаходимо та виводимо ділянку з мінімальною ціною
        System.out.println("Земельна ділянка з мінімальною ціною:");
        System.out.println(residentialContainer.findMinimum());

        // Створюємо контейнер для сільськогосподарських ділянок
        LandPlotContainer<AgriculturalPlot> agriculturalContainer = new
LandPlotContainer<>();

        // Додаємо сільськогосподарські ділянки
        agriculturalContainer.addPlot(new AgriculturalPlot("Київська область",
```

```
50.5, "Чорнозем"));
        agriculturalContainer.addPlot(new AgriculturalPlot("Полтавська область",
30.0, "Суглинок"));
        agriculturalContainer.addPlot(new AgriculturalPlot("Черкаська область",
45.5, "Чорнозем"));

        // Знаходимо та виводимо ділянку з мінімальною площею
        System.out.println("\nЗемельна ділянка з мінімальною площею:");
        System.out.println(agriculturalContainer.findMinimum());
    }
}
```

LandPlotContainer.java

```
package KI.Milian.Lab6;

import java.util.ArrayList;
import java.util.List;

/**
 * Узагальнений клас-контейнер для управління земельними ділянками
 * @param <T> Тип елементів для зберігання, що реалізує інтерфейс Comparable
 */
public class LandPlotContainer<T extends Comparable<T>> {
    private List<T> plots;

    /**
     * Конструктор ініціалізує порожній контейнер
     */
    public LandPlotContainer() {
        this.plots = new ArrayList<>();
    }

    /**
     * Додає новий елемент до контейнера
     * @param plot Елемент для додавання
     */
    public void addPlot(T plot) {
        plots.add(plot);
    }

    /**
     * Видаляє елемент з контейнера
     * @param plot Елемент для видалення
     * @return true якщо елемент був видалений, false в іншому випадку
     */
    public boolean removePlot(T plot) {
        return plots.remove(plot);
    }

    /**
     * Знаходить мінімальний елемент у контейнері
     * @return Мінімальний елемент або null якщо контейнер порожній
     */
    public T findMinimum() {
        if (plots.isEmpty()) {
            return null;
        }
        T min = plots.get(0);
        for (T plot : plots) {
            if (plot.compareTo(min) < 0) {
                min = plot;
            }
        }
        return min;
```

```
    }

    /**
     * Отримує всі елементи в контейнері
     * @return Список всіх елементів
     */
    public List<T> getAllPlots() {
        return new ArrayList<>(plots);
    }
}
```

ResidentialPlot.java

```
package KI.Milian.Lab6;

/**
 * Представляє житлову земельну ділянку
 */
public class ResidentialPlot implements Comparable<ResidentialPlot>{
    private String address;
    private double area;
    private double price;

    /**
     * Конструктор для житлової ділянки
     *
     * @param address Адреса ділянки
     * @param area    Площа ділянки в квадратних метрах
     * @param price   Ціна ділянки
     */
    public ResidentialPlot(String address, double area, double price) {
        this.address = address;
        this.area = area;
        this.price = price;
    }

    public double getArea() {
        return area;
    }

    public double getPrice() {
        return price;
    }

    @Override
    public String toString() {
        return "Житлова ділянка{адреса='" + address + "', площа=" + area + ",
ціна=" + price + "}";
    }

    @Override
    public int compareTo(ResidentialPlot other) {
        // Порівнюємо за ціною
        return Double.compare(this.price, other.price);
    }
}
```

AgriculturalPlot.java

```
package KI.Milian.Lab6;

/**
 * Представляє сільськогосподарську земельну ділянку
```

```java
 */
public class AgriculturalPlot implements Comparable<AgriculturalPlot>{
    private String location;
    private double area;
    private String soilType;

    /**
     * Конструктор для сільськогосподарської ділянки
     * @param location Розташування ділянки
     * @param area Площа ділянки в гектарах
     * @param soilType Тип ґрунту
     */
    public AgriculturalPlot(String location, double area, String soilType) {
        this.location = location;
        this.area = area;
        this.soilType = soilType;
    }

    public double getArea() {
        return area;
    }

    @Override
    public String toString() {
        return "Сільськогосподарська ділянка{розташування='" + location + "',
площа=" + area + ", тип ґрунту='" + soilType + "'}";
    }

    @Override
    public int compareTo(AgriculturalPlot other) {
        // Порівнюємо за площею
        return Double.compare(this.area, other.area);
    }
}
```

```
Земельна ділянка з мінімальною ціною:
Житлова ділянка{адреса='вул. Лесі Українки 10', площа=400.0, ціна=80000.0}

Земельна ділянка з мінімальною площею:
Сільськогосподарська ділянка{розташування='Полтавська область', площа=30.0, тип ґрунту='Суглинок'}

Process finished with exit code 0
```

Рис.1 Вивід у консоль

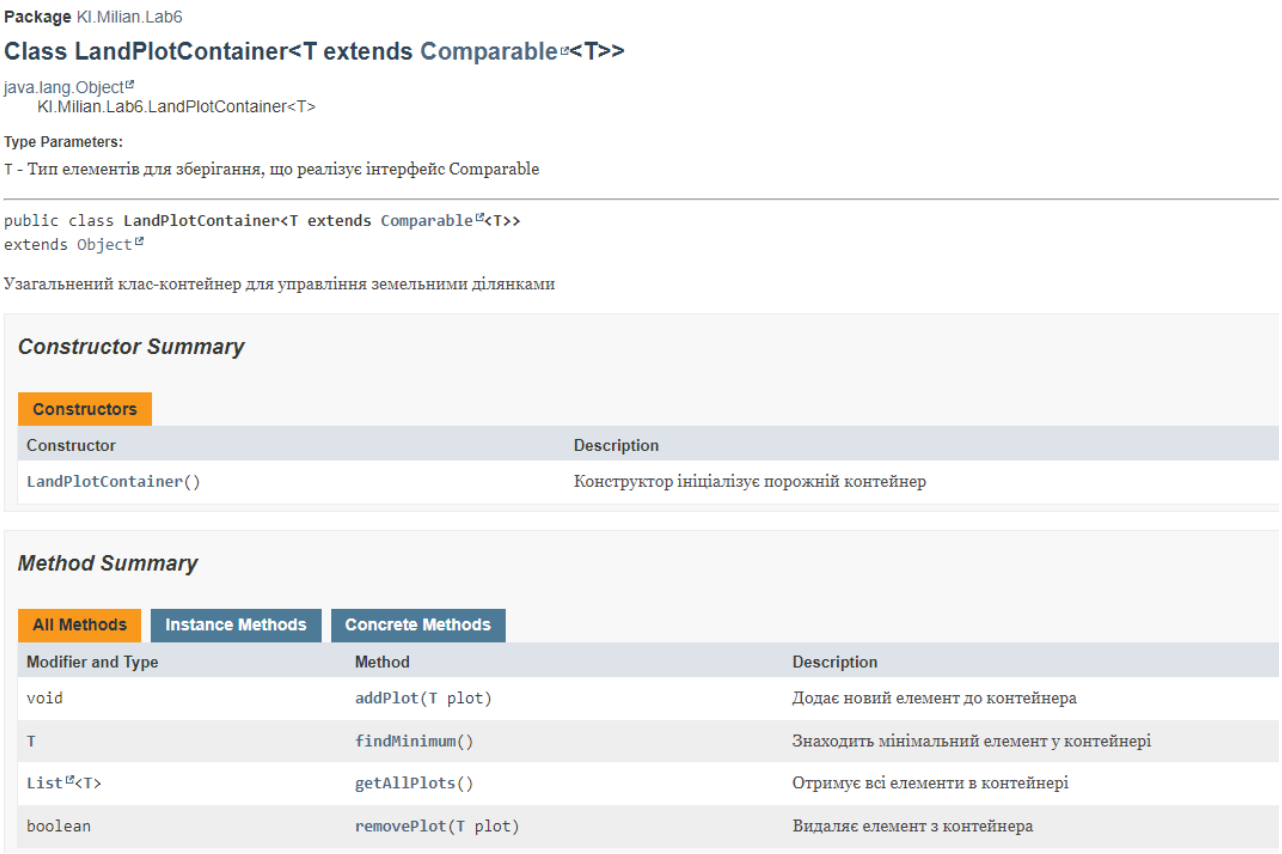
Package KI.Milian.Lab6

package KI.Milian.Lab6

Classes

Class	Description
AgriculturalPlot	Представляє сільськогосподарську земельну ділянку
Driver	Головний клас для демонстрації функціональності LandPlotContainer
LandPlotContainer<T extends Comparable<T>>	Узагальнений клас-контейнер для управління земельними ділянками
ResidentialPlot	Представляє житлову земельну ділянку

Рис.2.1 Фрагмент згенерованої документації



Package KI.Milian.Lab6

Class LandPlotContainer<T extends Comparable<T>>

java.lang.Object
KI.Milian.Lab6.LandPlotContainer<T>

Type Parameters:

T - Тип елементів для зберігання, що реалізує інтерфейс Comparable

```
public class LandPlotContainer<T extends Comparable<T>>
extends Object
```

Узагальнений клас-контейнер для управління земельними ділянками

Constructor Summary

Constructors

Constructor	Description
LandPlotContainer()	Конструктор ініціалізує порожній контейнер

Method Summary

All Methods | Instance Methods | Concrete Methods

Modifier and Type	Method	Description
void	addPlot(T plot)	Додає новий елемент до контейнера
T	findMinimum()	Знаходить мінімальний елемент у контейнері
List<T>	getAllPlots()	Отримує всі елементи в контейнері
boolean	removePlot(T plot)	Видаляє елемент з контейнера

Рис.2.2 Фрагмент згенерованої документації

Висновок: На лабораторній роботі я оволодів навиками параметризованого програмування мовою Java.