Національний університет «Львівська Політехніка»

Інститут комп’ютерних технологій, автоматики та метрології

Кафедра електронних обчислювальних машин



Звіт

Про виконання лабораторної роботи №6

### З дисципліни «Кросплатформлені засоби програмування»

**Виконав:**

студент групи КІ-305

Рудий В.В.

**Перевірив:**

Доцент кафедри ЕОМ

Іванов Ю. С.

Львів – 2023

**Тема:** Параметризоване програмування.

**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:**

***Варіант: 23 Завдання: Хлів***

**Виконання:**

**Код програми:**

import java.util.\*;

/\*\*

 \* Class Lab6RudyiKI305 realize example program to lab work 6

 \*

 \* @author Rudyi Volodymyr

 \* @version 1.1

 \* @since version 1.0

 \*/

public class Lab6RudyiKI305 {

    public static void main(String[] args) {

        Village<? super Barn> Village = new Village<Barn>();

        Village.AddBarn(new Cow("Milka", 20, 540));

        Village.AddBarn(new Horse("Lucky", 880));

        Village.AddBarn(new Horse("Fox", 700));

        Village.AddBarn(new Cow("Lada", 45, 650));

        Barn res1 = Village.findMin();

        System.out.print("The smallest barn in the Village has: \n");

        res1.print();

        Barn res2 = Village.findMax();

        System.out.print("The biggest barn in the Village has: \n");

        res2.print();

    }

}

/\*\*

 \* Interface <code>Barn</code> implements barn

 \*/

interface Barn extends Comparable<Barn> {

    public int getWeight();

    public void print();

}

/\*\*

 \* Parameterized class <code>Village</code> implements Village

 \*/

class Village<T extends Barn> { // село

    private ArrayList<T> arr;

    /\*\*

     \* Constructor

     \*/

    public Village() {

        arr = new ArrayList<T>();

    }

    /\*\*

     \* Method find land plot with the smallest size

     \*/

    public T findMin() {

        if (!arr.isEmpty()) {

            T min = arr.get(0);

            for (int i = 1; i < arr.size(); i++) {

                if (arr.get(i).compareTo(min) < 0)

                    min = arr.get(i);

            }

            return min;

        }

        return null;

    }

    /\*\*

     \* Method find land plot with the biggest size

     \*/

    public T findMax() {

        if (!arr.isEmpty()) {

            T max = arr.get(0);

            for (int i = 1; i < arr.size(); i++) {

                if (arr.get(i).compareTo(max) > 0)

                    max = arr.get(i);

            }

            return max;

        }

        return null;

    }

    /\*\*

     \* Method add new land plot to Village

     \*

     \* @param <code>Barn</code> The class value

     \*/

    public void AddBarn(T Barn) {

        arr.add(Barn);

        System.out.print("Element added: ");

        Barn.print();

    }

    /\*\*

     \* Method delete land plot from Village

     \*

     \* @param <code>i</code> The index of element of array in Village

     \*/

    public void DeleteBarn(int i) {

        arr.remove(i);

    }

}

class Cow implements Barn { // корова

    private String CowName;

    private int milkYield;

    private int weight;

    /\*\*

     \* Constructor

     \*

     \* @param <code>hFlours</code> The Cow flours type

     \* @param <code>pLive</code>   The count of people who live in the Cow

     \* @param <code>pSize</code>   The size of Cow

     \*/

    public Cow(String hName, int pYield, int pWeight) {

        CowName = hName;

        milkYield = pYield;

        weight = pWeight;

    }

    /\*\*

     \* Method return Cow name

     \*

     \* @return CowFloors

     \*/

    public String getCowName() {

        return CowName;

    }

    /\*\*

     \* Method set Cow name for Cow

     \*

     \* @param <code>name</code> The Cow name

     \*/

    public void setCowName(String name) {

        CowName = name;

    }

    /\*\*

     \* Method return count of milk yield in the Cow

     \*

     \* @return peopleLive

     \*/

    public int getMilkYield() {

        return milkYield;

    }

    /\*\*

     \* Method set count of milk yield in the Cow

     \*

     \* @param <code>n</code> The milk yield

     \*/

    public void setMilkYield(int n) {

        milkYield = n;

    }

    /\*\*

     \* Method return weight of Cow

     \*

     \* @return weight

     \*/

    public int getWeight() {

        return weight;

    }

    /\*\*

     \* Method compare objects type Barn

     \*

     \* @param <code>p</code> The weight of object

     \* @return num (0, 1, -1)

     \*/

    public int compareTo(Barn p) {

        Integer s = weight;

        return s.compareTo(p.getWeight());

    }

    /\*\*

     \* Method print information about Cow

     \*/

    public void print() {

        System.out.print("Cow: " + CowName + ", Milk yield: " + milkYield +

                ", Cow Weight: " + weight + ";\n");

    }

}

class Horse implements Barn { // кінь

    private String HorseName;

    private int HorseWeight;

    /\*\*

     \* Constructor

     \*

     \* @param <code>sName</code> The name of Horse

     \* @param <code>sWeight</code> The weight of Horse

     \*/

    public Horse(String sName, int sWeight) {

        HorseName = sName;

        HorseWeight = sWeight;

    }

    /\*\*

     \* Method return name of Horse

     \*

     \* @return HorseName

     \*/

    public String getName() {

        return HorseName;

    }

    /\*\*

     \* Method set name of Horse

     \*

     \* @param <code>n</code> The HorseName

     \*/

    public void setName(String name) {

        HorseName = name;

    }

    /\*\*

     \* Method return weight of Horse

     \*

     \* @return weight

     \*/

    public int getWeight() {

        return HorseWeight;

    }

    /\*\*

     \* Method set weight of Horse

     \*

     \* @param <code>n</code> The HorseWeight

     \*/

    public void setWeight(int n) {

        HorseWeight = n;

    }

    /\*\*

     \* Method compare objects type Barn

     \*

     \* @param <code>p</code> The weight of object

     \* @return num (0, 1, -1)

     \*/

    public int compareTo(Barn p) {

        Integer s = HorseWeight;

        return s.compareTo(p.getWeight());

    }

    /\*\*

     \* Method print information about Horse

     \*/

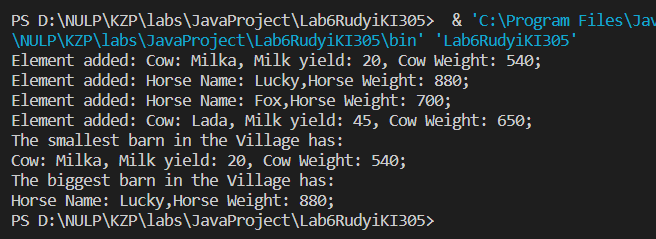
    public void print() {

        System.out.print("Horse Name: " + HorseName + ",Horse Weight: " + HorseWeight + ";\n");

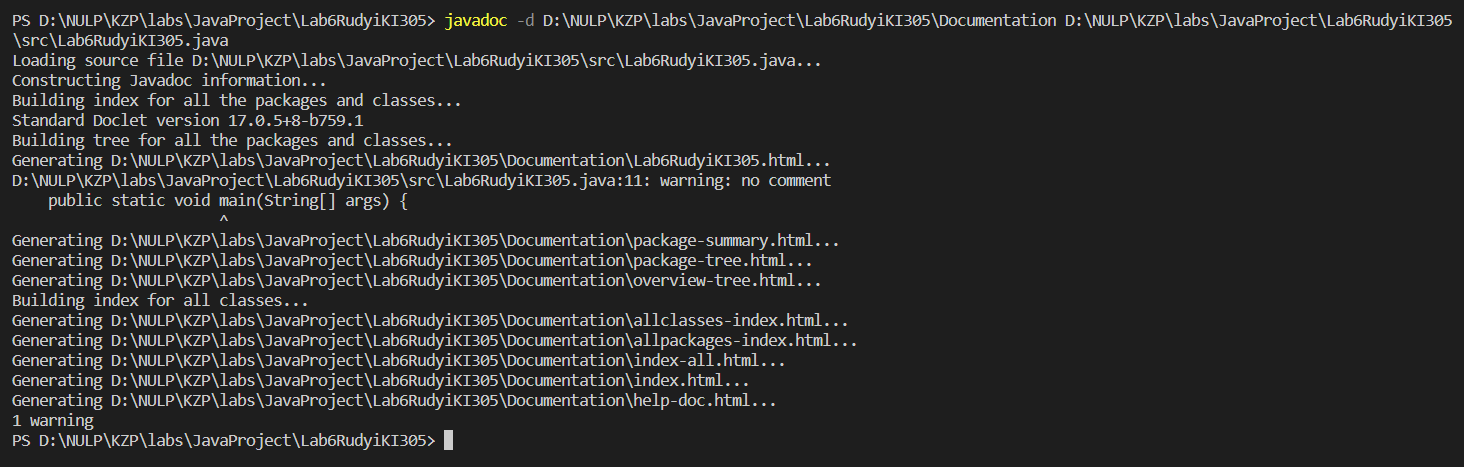
    }

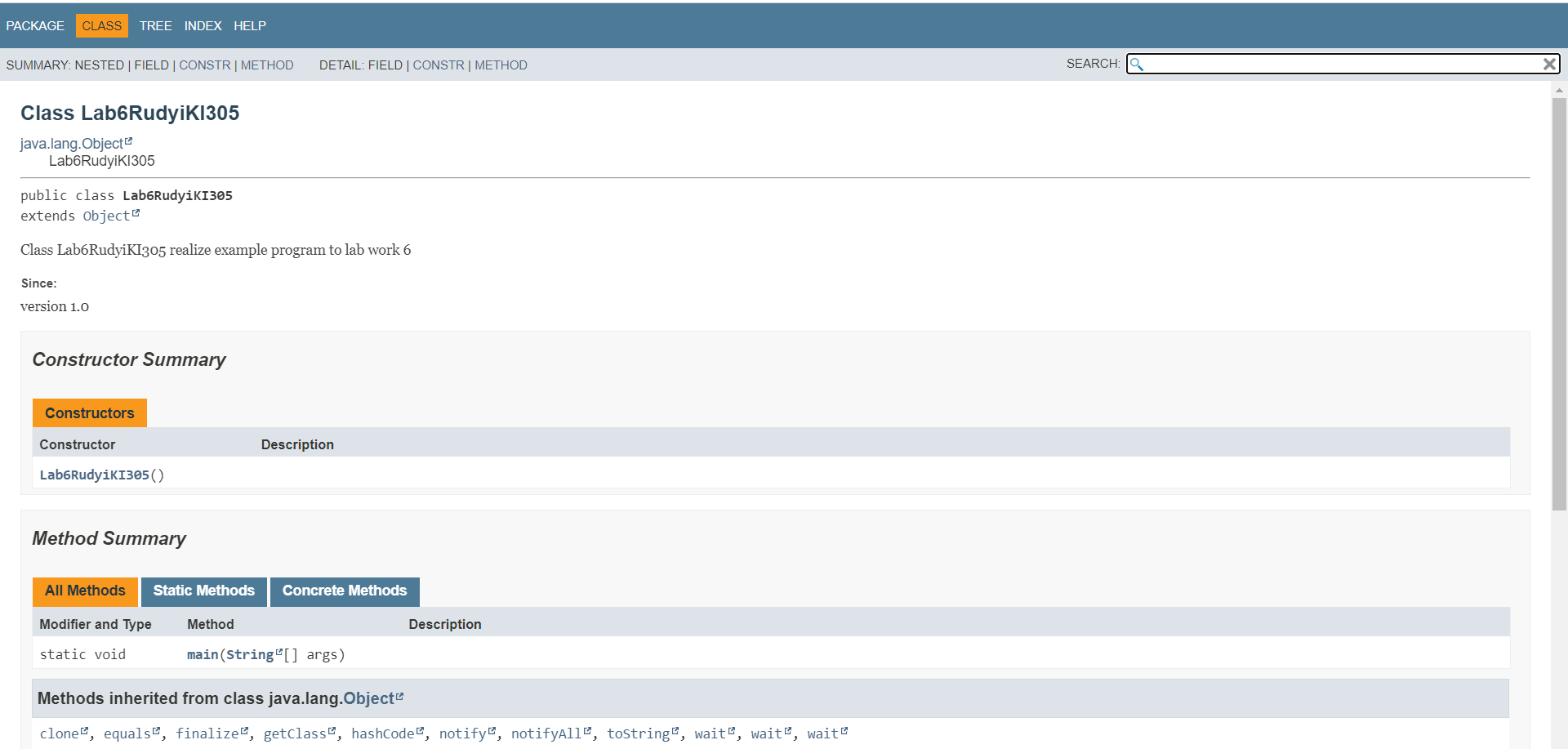
}

**Результат роботи програми у консолі та файлі:**



**Фрагмент генерації документації:**

****

****

**Висновок:** на цій лабораторній роботі я оволодів навиками параметризованого програмування мовою Java. Створив параметризований клас, який працює з двома класами, екземпляри якого розміщуються у списку цього класу в програмі-драйвері.