

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



## **Звіт**

з лабораторної роботи № 6  
з дисципліни: «Кросплатформенні засоби програмування»

**На тему:** «Параметризоване програмування»

**Виконав:**

студент групи КІ-307  
Бажулін С.В.

**Прийняв:**

доцент кафедри ЕОМ  
Іванов Ю. С.

**Мета роботи:** оволодіти навиками параметризованого програмування мовою Java.

## **Завдання(варіант №2):**

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.

## **Індивідуальне завдання:**

### **2. Однозв'язний список**

#### **Файл MainBazhulin.java**

```
/*  
* Copyright (c) 2013-2023 Lviv Polytechnic National University. All Rights Reserved.  
*  
* This program and the accompanying materials are made available under the terms  
* of the Academic Free License v. 3.0 which accompanies this distribution, and is  
* available at https://opensource.org/license/afl-3-0-php/  
*  
* SPDX-License-Identifier: AFL-3.0  
*/  
package KI307.Bazhulin.Lab6;  
import java.util.*;  
import java.io.*;  
  
/**  
 * Class Main are testing a class MyList  
 * @author Bazhulin Serhiy KI-307  
 * @version 1.0  
 * @since version 1.0  
 */  
public class MainBazhulin {  
    public static void main(String[] args) throws IOException  
    {  
        MyList<String> list = new MyList();  
        list.add("abc");  
        list.add("qwert");  
        list.add("a");  
        list.display();  
        System.out.println(list.findMin());  
        list.remove("qwert");  
        list.display();  
    }  
}
```

## Файл MyList.java

```
package KI307.Bazhulin.Lab6;

/**
 * @author Bazhulin Serhiy KI-307
 * @version 1.0
 * @since version 1.0
 * @param <T>
 */
class Node<T> extends Comparable<T>>
{
    T data;
    Node<T> next;
    public Node(T data)
    {
        this.data = data;
        this.next = null;
    }
}

/**
 * Class MyList realisation
 * @author Bazhulin Serhiy KI-307
 * @version 1.0
 * @since version 1.0
 * @param <T>
 */
class MyList<T> extends Comparable<T>>
{
    Node<T> head;

    public MyList()
    {
        this.head = null;
    }

    //Adding an element
    public void add(T data)
    {
        Node<T> newNode = new Node<>(data);

        if (head == null) {
            head = newNode;
        }
        else {
            Node<T> current = head;
            while (current.next != null) {
                current = current.next;
            }
            current.next = newNode;
        }
    }

    //Removing element
    public void remove(T data)
    {
        if (head == null) {
            return;
        }

        if (head.data.equals(data)) {
            head = head.next;
            return;
        }
    }
}
```

```

        Node<T> current = head;
        while (current.next != null && !current.next.data.equals(data)) {
            current = current.next;
        }

        if (current.next != null) {
            current.next = current.next.next;
        }
    }

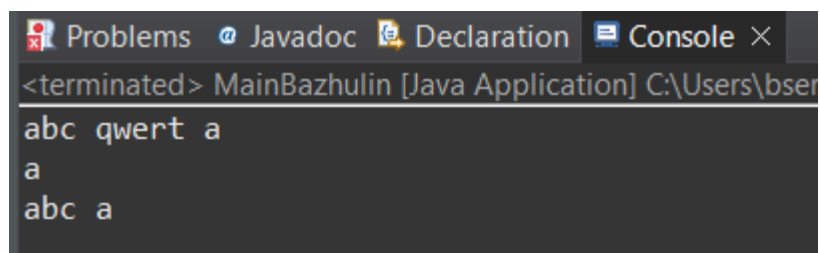
    //This method is finding a minimal value of list
    public T findMin()
    {
        if (head == null) {
            return null;
        }

        T min = head.data;
        Node<T> current = head.next;
        while (current != null) {
            if (current.data.compareTo(min) < 0) {
                min = current.data;
            }
            current = current.next;
        }
        return min;
    }

    //This method is displaying a list
    public void display()
    {
        Node<T> current = head;
        while (current != null) {
            System.out.print(current.data + " ");
            current = current.next;
        }
        System.out.println();
    }
}

```

## Результат роботи програми:

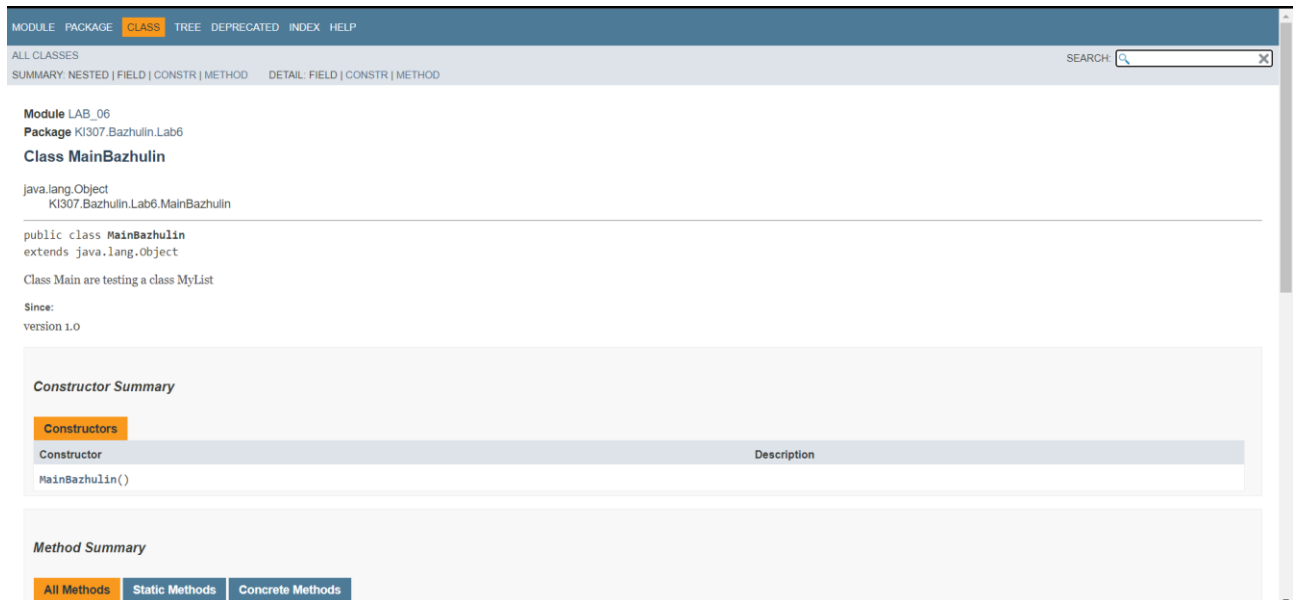


```

<terminated> MainBazhulin [Java Application] C:\Users\bsen
abc qwert a
a
abc a

```

## Фрагмент згенерованої документації:



## Відповідь на контрольні питання

### 1) Дайте визначення терміну «параметризоване програмування».

Параметризоване програмування - це підхід до програмування, коли код можна написати один раз для різних типів даних або об'єктів, використовуючи параметризовані (загальні) типи або методи.

### 2) Розкрийте синтаксис визначення простого параметризованого класу.

```
class MyClass<T> {
}
```

### 3) Розкрийте синтаксис створення об'єкту параметризованого класу.

```
MyClass<int> obj = new MyClass<int>();
```

### 4) Розкрийте синтаксис визначення параметризованого методу.

```
public void MyMethod<T>(T parameter) {
}
```

## **5) Розкрийте синтаксис виклику параметризованого методу.**

```
obj.MyMethod(5);
```

## **6) Яку роль відіграє встановлення обмежень для змінних типів?**

Встановлення обмежень для змінних типів дозволяє задати певні умови або вимоги для типів даних, які можуть бути використані в параметризованому коді.

## **7) Як встановити обмеження для змінних типів?**

Відповідь: Обмеження для змінних типів встановлюються за допомогою ключового слова ``where``. Приклад:

```
public class MyClass<T> where T : SomeBaseClass {  
    }  
}
```

## **8) Розкрийте правила спадкування параметризованих типів.**

Правила спадкування параметризованих типів спираються на ієрархію класів і обмеження, які визначені для типів.

## **9) Яке призначення підстановочних типів?**

Підстановочні типи дозволяють створювати загальні типи, які можуть працювати з різними типами даних, або вказувати обмеження для параметризованих типів.

## **10) Застосування підстановочних типів.**

Застосування підстановочних типів включає створення загальних колекцій, класів, методів і інших структур, які можуть працювати з різними типами даних без необхідності дублювати код.

**Висновок:** на цій лабораторній роботі, я оволодів навиками параметризованого програмування мовою Java.