Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Лабораторна робота №11

з курсу:

“ООП”

Виконав:
ст. гр. КН-110
Шаварський
Максим
Прийняв:
Гасько Р.Т.

Львів – 2018 р.

Лабораторна робота № 11

Мета

- **Вивчення принципів параметризації в Java**
- **Розробка параметризованих класів та методів.**
- **Розширення функціональності параметризованих класів.**

Вимоги 1. Створити власний клас-контейнер, що параметризується (Generic Type), (docs.oracle.com/javase/tutorial/java/generics/types.html) на основі зв'язних списків для реалізації колекції domain-об'єктів з лабораторної роботи №10 (Прикладні задачі. Список №2. 20 варіантів)

2. Для розроблених класів-контейнерів забезпечити можливість використання їх об'єктів у циклі foreach в якості джерела даних.

3. Забезпечити можливість збереження та відновлення колекції об'єктів: 1) за допомогою стандартної серіалізації; 2) не використовуючи протокол серіалізації.

4. Продемонструвати розроблену функціональність: створення контейнера, додавання елементів, видалення елементів, очищення контейнера, перетворення у масив, перетворення у рядок, перевірку на наявність елементів.

5. Забороняється використання контейнерів (колекцій) з Java Collections Framework - docs.oracle.com/javase/8/docs/technotes/guides/collections/

6. Розробити параметризовані методи (Generic Methods - docs.oracle.com/ javase/tutorial/java/generics/methods.html) для обробки колекцій об'єктів згідно (Прикладні задачі. Список №2. 20 варіантів).

7. Продемонструвати розроблену функціональність (створення, управління та обробку власних контейнерів) в діалоговому та автоматичному режимах.

а. Автоматичний режим виконання програми задається параметром командного рядка -auto . Наприклад, java ClassName -auto .

b. В автоматичному режимі діалог з користувачем відсутній, необхідні данні генеруються, або зчитуються з файлу.

Код програми

```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

class Node{
    private Node next;
    private Integer data;

    public Node() {
    }

    public Node getNext() {
        return next;
    }
    public void setNext(Node next) {
        this.next = next;
    }
    public Integer getData() {
        return data;
    }
    public void setData(Integer data) {
        this.data = data;
    }
}

 class LinkedList<T> {
    private Node golova;
    private Node hvist;
    private int count = 0;
    public LinkedList() {

    }

    public void add(Integer data) {
        Node Spysok = new Node();
        Spysok.setData((Integer) data);
        if (golova == null) {
            golova = Spysok;
            hvist = Spysok;
        }
        else {
            hvist.setNext(Spysok);
            hvist = Spysok;
        }
        count++;
    }
    public Integer get(int index) {
        Node Spysok = golova;
        if (index > count) {
            return null;
        }
        else {
            for (int i = 0; i <= index; i++) {
                if (i == index) {
                    return Spysok.getData();
                }
                Spysok = Spysok.getNext();
```

```
            }
        }
        return null;
    }
    public int size() {
        return count;
    }
    public boolean delete(int index) {
        Node mysor = null;
        Node mysor2 = golova;
        if (index>count) {
            return false;
        }
        else if (index == 0) {
            golova = golova.getNext();
            if (golova == null) {
                hvist = null;
            }
            count--;
            return true;
        }
        else if (index == count - 1) {
            for (int i = 0; i <= index; i++) {
                if (i == index) {
                    hvist = mysor;
                    hvist.setNext(null);
                    count--;
                    return true;
                }
                mysor = mysor2;
                mysor2 = mysor2.getNext();
            }
        }
        else {
            for (int i = 0; i <= index; i++) {
                if (i == index) {
                    mysor.setNext(mysor2.getNext());
                    count--;
                    return true;
                }
                mysor = mysor2;
                mysor2 = mysor2.getNext();
            }
        }
        return false;
    }
 }

public class lab
{
    static void printList(ArrayList<?> list) {
        for (Object l : list)
            System.out.print("{" + l + "}"+ " ");
    }
    public static void main(String[] args) {
        if (args.length != 0 && args[0].equalsIgnoreCase("-auto")) {
            List<Integer> list = new ArrayList<>();
            LinkedList<Character> List = new LinkedList<>();
            for (int I = 0; I < 10; I++) {
                list.add(I);
            }
            System.out.print("Foreach for LinkedList:");
            printList((ArrayList<?>) list);
            System.out.print("\nRemove for LinkedList:");
            list.remove(5);
```

```
            printList((ArrayList<?>) list);
            System.out.print("\n");
            System.out.print("Contains for LinkedList:");
            if (list.contains(5)) {
                System.out.println(" Container contains" + 5);
            } else {
                System.out.println(" Container doesn't " + 5);
            }
            System.out.print("ToArray for LinkedList:");
            Object[] arr = list.toArray();
            for (int i = 0; i < arr.length; i++) {
                System.out.print("{" + arr[i] + "}" + " ");
            }
            System.out.print("\n");
            System.out.print("ToString for LinkedList:");
            String str = list.toString();
            System.out.println(str);
        }else {


            List<Integer> l = new ArrayList<>();
            while (true) {
                System.out.println("\n");
                System.out.println("1 - Add\n2 - Foreach\n3 - Remove\n4 -
Contains\n5 - ToArray\n6 - ToString\n7 - Exit");
            Scanner sc = new Scanner(System.in);
            int choose = sc.nextInt();
            switch (choose) {
                case 1:
                    System.out.println("Input:");
                    while (choose != 0) {
                        choose = sc.nextInt();
                        l.add(choose);
                    }
                    l.remove(l.size()-1);
                    break;
                case 2:
                    printList((ArrayList<?>) l);
                    break;
                case 3:
                    System.out.println("Input index of element you wanting
to remove:");
                    choose = sc.nextInt();
                    l.remove(choose);
                    break;
                case 4:
                    System.out.println("Input element to check if container
contains this:");
                    choose = sc.nextInt();
                    if (l.contains(choose)) {
                        System.out.println(" Container contains" + 5);
                    } else {
                        System.out.println(" Container doesn't " + 5);
                    }
                    break;
                case 5:
                    Object[] arr = l.toArray();
                    for (int i = 0; i < arr.length; i++) {
                        System.out.print("{" + arr[i] + "}" + " ");
                    }
                    break;
                case 6:
                    String str = l.toString();
                    System.out.println(str);
```

```
                    break;
                case 7: System.exit(0);
                    break;
                default:
                    System.out.println("Error404");
                    break;
            }
        }
    }
}
```