



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/07 Интеллектуальные системы анализа,
обработки и интерпретации больших данных**

О Т Ч Е Т

по лабораторной работе № 6

Вариант 12

Название: Коллекции в Java

Дисциплина: Языки программирования для работы с большими данными

Студент

ИУ6-23М

(Группа)

(Подпись, дата)

П.А. Мартынюк

(И.О. Фамилия)

Преподаватель

(Подпись, дата)

П.В. Степанов

(И.О. Фамилия)

Москва, 2022

Цель работы:

Получение навыков работы с коллекциями в Java.

Выполнение:

Задание 1:

1. С использованием множества выполнить попарное суммирование произвольного конечного ряда чисел по следующим правилам: на первом этапе суммируются попарно рядом стоящие числа, на втором этапе суммируются результаты первого этапа и т.д. до тех пор, пока не останется одно число.
2. Сложить два многочлена заданной степени, если коэффициенты многочленов хранятся в объекте HashMap.

Листинг выполнения подзадачи 1 (файл SumSetElements.java)

```
package task1;

import java.util.Iterator;
import java.util.LinkedHashSet;
import java.util.Random;

public class SumSetElements {

    public static void main(String[] args) {

        LinkedHashSet<Integer> set = new LinkedHashSet<>();

        Random random = new Random();
        int length = random.nextInt(5, 10);
        for (int i = 0; i < length; i++) {
            set.add(random.nextInt(5, 10));
        }

        System.out.println(set);
        Iterator<Integer> it = set.iterator();

        boolean iteration_done = false, global_done = false;
        int first, second;
        LinkedHashSet<Integer> new_set = new LinkedHashSet<>();

        while (!global_done) {
            new_set.clear();
            iteration_done = false;
            while (!iteration_done) {
                if (it.hasNext()) {
                    first = it.next();
                    if (it.hasNext()) {
                        second = it.next();
                        new_set.add(first + second);
                    } else {
                        new_set.add(first);
                        iteration_done = true;
                    }
                } else {
                    iteration_done = true;
                }
            }
        }
    }
}
```

```

    }
    set = new LinkedHashSet<>(new_set);
    it = set.iterator();
    System.out.println(set);
    if(set.size() == 1) global_done = true;
}
}
}

```

```

C:\Users\polin\.jdk\openjdk-17.0.2\bin\java.exe
[7, 5, 8, 6, 9]
[12, 14, 9]
[26, 9]
[35]
Process finished with exit code 0

```

Рисунок 1 - Результат выполнения кода решения подзадачи 1

Листинг выполнения подзадачи 2 (файл SumPolynomialElements.java)

```

package task1;

import java.util.HashMap;
import java.util.Random;
import java.util.Scanner;

public class SumPolynomialElements {

    public static void main(String[] args) {

        HashMap<Integer, Integer> polinom1 = new HashMap<>();
        HashMap<Integer, Integer> polinom2 = new HashMap<>();
        HashMap<Integer, Integer> polinom_result = new HashMap<>();

        Random random = new Random();
        Scanner in = new Scanner(System.in);
        System.out.print("Enter number n: ");
        int length = in.nextInt();

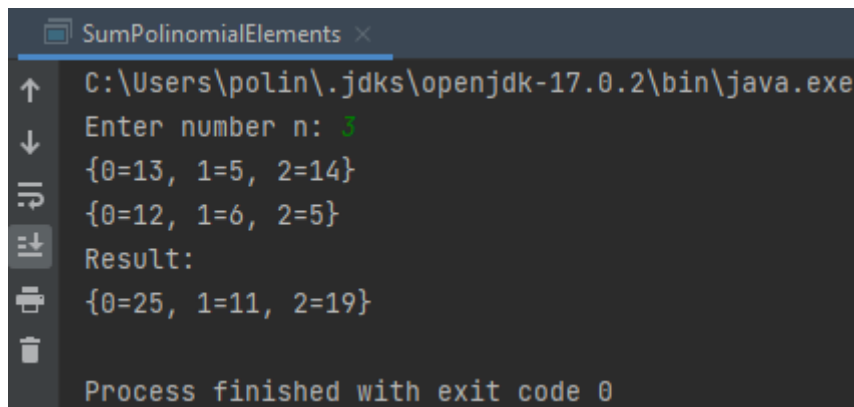
        for (int i = 0; i < length; i++) {
            polinom1.put(i, random.nextInt(15));
            polinom2.put(i, random.nextInt(15));
        }

        System.out.println(polinom1);
        System.out.println(polinom2);

        for (int key : polinom1.keySet()) {
            polinom_result.put(key, polinom1.get(key) + polinom2.get(key));
        }

        System.out.println("Result: ");
        System.out.println(polinom_result);
    }
}

```



```
SumPolinomialElements x
C:\Users\polin\.jdk\openjdk-17.0.2\bin\java.exe
Enter number n: 3
{0=13, 1=5, 2=14}
{0=12, 1=6, 2=5}
Result:
{0=25, 1=11, 2=19}
Process finished with exit code 0
```

Рисунок 2 - Результат выполнения кода решения подзадачи 2

Задание 2:

1. Реализовать класс, моделирующий работу N-местной автостоянки. Машина подъезжает к определенному месту и едет вправо, пока не встретится свободное место. Класс должен поддерживать методы, обслуживающие приезд и отъезд машины.
2. Во входном файле хранятся две разреженные матрицы A и B. Построить циклически связанные списки CA и CB, содержащие ненулевые элементы соответственно матриц A и B. Просматривая списки, вычислить: а) сумму $S = A + B$; б) произведение $P = A * B$.

Листинг выполнения подзадачи 1 (файл Parking.java)

```
package task2;

import java.util.ArrayList;
import java.util.Random;

public class Parking {

    private int N;
    private ArrayList<Boolean> parking_row;

    public Parking(int n) {
        N = n;
        parking_row = new ArrayList<>(this.N);
        Random random = new Random();
        for (int i = 0; i < n; i++) {
            parking_row.add(random.nextBoolean());
        }
        System.out.println("Parking created: ");
        System.out.println(this);
    }

    public boolean CarArrival(int start_point){
        System.out.println("---Car arrived to parking---");
        System.out.println("Car starts searching from place number " + start_point);
        for (int i = start_point-1; i < N; i++) {
            if(!parking_row.get(i)){
                System.out.println("Car took place number " + (i+1));
            }
        }
    }
}
```

```

        parking_row.set(i, true);
        return true;
    }
}
System.out.println("No free places");
return false;
}

public void CarDeparture(int parking_place){
    System.out.println("Car is leaving place number " + parking_place);
    parking_row.set(parking_place-1, false);
    System.out.println("---Car left parking---");
}

@Override
public String toString() {
    return "Parking{" +
        "parking_row=" + parking_row +
        '}';
}
}

```

Листинг выполнения подзадачи 1 (файл MainForParking.java)

```

package task2;

import java.util.Random;
import java.util.Scanner;

public class MainForParking {

    public static void main(String[] args) {

        Scanner in = new Scanner(System.in);
        System.out.print("Enter number n: ");
        int n = in.nextInt();
        Parking parking = new Parking(n);
        System.out.println();

        System.out.print("Enter start point for search: ");
        int search_start = in.nextInt();
        parking.CarArrival(search_start);
        System.out.println(parking);
        System.out.println();

        System.out.println("Enter place for leaving car: ");
        int leaving_car_place = in.nextInt();
        parking.CarDeparture(leaving_car_place);
        System.out.println(parking);
    }
}

```

```
MainForParking x
C:\Users\polin\.jdk\openjdk-17.0.2\bin\java.exe "-javaagent:C:
Enter number n: 5
Parking created:
Parking{parking_row=[true, true, true, true, false]}

Enter start point for search: 2
---Car arrived to parking---
Car starts searching from place number 2
Car took place number 5
Parking{parking_row=[true, true, true, true, true]}

Enter place for leaving car:
1
Car is leaving place number 1
---Car left parking---
Parking{parking_row=[false, true, true, true, true]}

Process finished with exit code 0
```

Рисунок 3 - Результат выполнения кода решения подзадачи 1

Исходные матрицы для выполнения подзадачи 2 (файл MatrixAB.txt)

```
0 1 0
0 2 0
1 0 0

1 0 0
0 0 1
3 0 1
```

Листинг выполнения подзадачи 2 (файл Element.java)

```
package task2;

public class Element {
    private int row;
    private int col;
    private int meaning;

    public Element(int row, int col, int meaning) {
        this.row = row;
        this.col = col;
        this.meaning = meaning;
    }

    public int getRow() {
        return row;
    }

    public void setRow(int row) {
        this.row = row;
    }

    public int getCol() {
        return col;
    }

    public void setCol(int col) {
```

```

        this.col = col;
    }

    public int getMeaning() {
        return meaning;
    }

    public void setMeaning(int meaning) {
        this.meaning = meaning;
    }

    @Override
    public String toString() {
        return "Element{" +
            "row=" + row +
            ", col=" + col +
            ", meaning=" + meaning +
            '}';
    }
}

```

Листинг выполнения подзадачи 2 (файл MainForMatrix.java)

```

package task2;

import java.io.File;
import java.io.FileNotFoundException;
import java.util.LinkedList;
import java.util.Scanner;

public class MainForMatrix {

    public static void main(String[] args) {

        File inp_file = new
File("C:\\Users\\polin\\IdeaProjects\\JavaMagisterCourse\\Lab6\\src\\task2\\matrixAB.txt");

        Scanner scanner = null;
        try {
            scanner = new Scanner(inp_file);
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
        boolean next_matrix = false;
        int row_num = 1;
        int col_num = 1;
        LinkedList<Element> CA = new LinkedList<>();
        LinkedList<Element> CB = new LinkedList<>();

        while(scanner.hasNextLine()) {
            String line = scanner.nextLine();
            String[] nums = line.split(" ");
            if (nums.length < 2) {
                next_matrix = true;
                row_num = 1;
            } else {
                col_num = 1;
                for (String num : nums) {
                    if(Integer.parseInt(num) != 0) {
                        if (!next_matrix) {
                            CA.addLast(new Element(row_num, col_num, Integer.parseInt(num)));
                        } else {
                            CB.addLast(new Element(row_num, col_num, Integer.parseInt(num)));
                        }
                    }
                    col_num++;
                }
                row_num++;
            }
        }

        System.out.println(CA);
    }
}

```

```

System.out.println(CB);

//-----Summation

LinkedList<Element> sum_matrix = new LinkedList<>();
int ind_a = 0;
int ind_b = 0;
boolean sum_flag = true;

while(sum_flag) {
    if((CA.get(ind_a).getRow() == CB.get(ind_b).getRow()) &&
        (CA.get(ind_a).getCol() == CB.get(ind_b).getCol())) {
        sum_matrix.addLast(new Element(CA.get(ind_a).getRow(), CA.get(ind_a).getCol(),
CA.get(ind_a).getMeaning() + CB.get(ind_b).getMeaning()));
        ind_a++;
        ind_b++;
    } else {
        if (CA.get(ind_a).getRow() == CB.get(ind_b).getRow()) {
            if (CA.get(ind_a).getCol() > CB.get(ind_b).getCol()) {
                sum_matrix.addLast(new Element(CB.get(ind_b).getRow(),
CB.get(ind_b).getCol(), CB.get(ind_b).getMeaning()));
                ind_b++;
            } else {
                sum_matrix.addLast(new Element(CA.get(ind_a).getRow(),
CA.get(ind_a).getCol(), CA.get(ind_a).getMeaning()));
                ind_a++;
            }
        } else {
            if (CA.get(ind_a).getRow() > CB.get(ind_b).getRow()) {
                sum_matrix.addLast(new Element(CB.get(ind_b).getRow(),
CB.get(ind_b).getCol(), CB.get(ind_b).getMeaning()));
                ind_b++;
            } else {
                sum_matrix.addLast(new Element(CA.get(ind_a).getRow(),
CA.get(ind_a).getCol(), CA.get(ind_a).getMeaning()));
                ind_a++;
            }
        }
    }
    if(ind_a == CA.size()) {
        sum_flag = false;
        for (int i = ind_b; i < CB.size(); i++) {
            sum_matrix.addLast(new Element(CB.get(i).getRow(), CB.get(i).getCol(),
CB.get(i).getMeaning()));
        }
    } else if(ind_b == CB.size()){
        sum_flag = false;
        for (int i = ind_a; i < CA.size(); i++) {
            sum_matrix.addLast(new Element(CA.get(i).getRow(), CA.get(i).getCol(),
CA.get(i).getMeaning()));
        }
    }
}

System.out.println(sum_matrix);

//-----Multiplication
// A[i, k] * B[k, j]

LinkedList<Element> mul_matrix = new LinkedList<>();
int answer = 0;

for (int i = 1; i < 4; i++) {
    for (int j = 1; j < 4; j++) {
        answer = 0;
        for (int k = 1; k < 4; k++) {

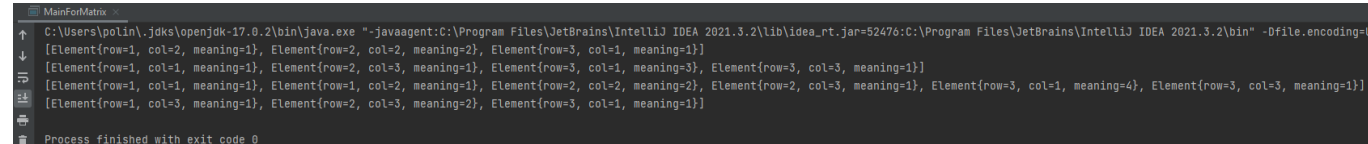
            for (int l = 0; l < CA.size(); l++) {
                if((CA.get(l).getRow() == i) && (CA.get(l).getCol() == k)){

                    for (int m = 0; m < CB.size(); m++) {
                        if(CB.get(m).getRow() == k && CB.get(m).getCol() == j){
                            answer += CA.get(l).getMeaning() * CB.get(m).getMeaning();

```



```
        break;
    }
    }
    break;
}
}
}
if(answer != 0) {
    mul_matrix.addLast(new Element(i, j, answer));
}
}
}
System.out.println(mul_matrix);
}
}
```



```
Process finished with exit code 0
[Element{row=1, col=2, meaning=1}, Element{row=2, col=2, meaning=2}, Element{row=3, col=1, meaning=1}]
[Element{row=1, col=1, meaning=1}, Element{row=2, col=3, meaning=1}, Element{row=3, col=1, meaning=3}, Element{row=3, col=3, meaning=1}]
[Element{row=1, col=1, meaning=1}, Element{row=1, col=2, meaning=1}, Element{row=2, col=2, meaning=2}, Element{row=2, col=3, meaning=1}, Element{row=3, col=1, meaning=4}, Element{row=3, col=3, meaning=1}]
[Element{row=1, col=3, meaning=1}, Element{row=2, col=3, meaning=2}, Element{row=3, col=1, meaning=1}]
```

Рисунок 4 - Результат выполнения кода решения подзадачи 2

Ссылка на программное решение:

Программное решение представлено в репозитории распределённой системы управления версиями Git:

<https://github.com/Owlfeather/JavaMagisterCourse/tree/main/Lab6/src>

Вывод:

При выполнении лабораторной работы были получены навыки работы с коллекциями в Java.