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 **09.04.01/12 Интеллектуальный анализ больших данных в системах поддержки принятия решений.**

О Т Ч Е Т

по лабораторной работе № 6

Вариант 6

Название: Коллекции

Дисциплина: Языки программирования для работы с большими данными

Студент	ИУ6-23М		Г.Л. Кушнир
	(Группа)	(Подпись, дата)	(И.О. Фамилия)
Преподаватель			П.В. Степанов
		(Подпись, дата)	(И.О. Фамилия)

Москва, 2023

Цель: Изучить работу с коллекциями в java.

Вариант 1 задание 6: Не используя вспомогательных объектов, переставить отрицательные элементы данного списка в конец, а положительные – в начало этого списка.

Содержание текстового файла:

0

2

16

-45

0

35

-21

145

Вариант 1 задание 7: Ввести строки из файла, записать в список ArrayList. Выполнить сортировку строк, используя метод sort() из класса Collections.

Содержание текстового файла:

something

lego

yandex

char

integer

Karlos

memes

lol

Вариант 2 задание 6: На плоскости задано N точек. Вывести в файл описания всех прямых, которые проходят более чем через одну точку из заданных. Для каждой прямой указать, через сколько точек она проходит. Использовать класс HashMap.

Код класса Dot:

```
public class Dot {

    private double x, y;

    public Dot(double x, double y) {
        this.x = x;
        this.y = y;
    }

    public double getX() {
        return x;
    }

    public double getY() {
        return y;
    }
}
```

Код класса Line:

```
public class Line {

    private double a, b, c;

    public Line(Dot dot1, Dot dot2){
        if (dot1.getX() - dot2.getX() == 0){
            this.a = 1;
            this.b = 0;
            this.c = -dot1.getX();
        }
        else{
            this.a = - (dot1.getY() - dot2.getY()) / (dot1.getX() -
dot2.getX());
            this.b = 1;
            this.c = - this.a * dot2.getX() - this.b * dot2.getY();
        }
    }

    public double getA() {
        return a;
    }

    public double getB() {
        return b;
    }

    public double getC() {
        return c;
    }

    @Override
    public int hashCode(){
        int result = 17;
        long temp;

        temp = Double.doubleToLongBits(a);
        result = 31 * result + (int) (temp ^ (temp >>> 32));

        temp = Double.doubleToLongBits(b);
        result = 31 * result + (int) (temp ^ (temp >>> 32));

        temp = Double.doubleToLongBits(c);
        result = 31 * result + (int) (temp ^ (temp >>> 32));
```

```
        return result;
    }

    @Override
    public boolean equals(Object obj){
        if (this == obj)
            return true;
        if (obj == null || getClass() != obj.getClass())
            return false;
        Line helper = (Line) obj;
        return Double.compare(helper.a, a) == 0 &&
                Double.compare(helper.b, b) == 0 &&
                Double.compare(helper.c, c) == 0;
    }
}
```

Вариант 2 задание 7: На плоскости задано N отрезков. Найти точку пересечения двух отрезков, имеющую минимальную абсциссу. Использовать класс TreeMap.

Код класса Segment:

```
public class Segment {

    double x1, x2, y1, y2;

    public Segment(double x1, double x2, double y1, double y2) {
        this.x1 = x1;
        this.x2 = x2;
        this.y1 = y1;
        this.y2 = y2;
    }

    public List<double[]> getIntersectionPoints(Segment other) {
        List<double[]> intersections = new ArrayList<>();

        double x1 = this.x1, y1 = this.y1;
        double x2 = this.x2, y2 = this.y2;
        double x3 = other.x1, y3 = other.y1;
        double x4 = other.x2, y4 = other.y2;

        double den = (x1 - x2) * (y3 - y4) - (y1 - y2) * (x3 - x4);

        if (den != 0) {
            double t = ((x1 - x3) * (y3 - y4) - (y1 - y3) * (x3 - x4)) / den;
            double u = -((x1 - x2) * (y1 - y3) - (y1 - y2) * (x1 - x3)) /
den;

            if (t >= 0 && t <= 1 && u >= 0 && u <= 1) {
                double intersectX = x1 + t * (x2 - x1);
                double intersectY = y1 + t * (y2 - y1);
                intersections.add(new double[]{intersectX, intersectY});
            } else {
                System.out.println("The segments do not intersect");
            }
        } else {
            System.out.println("The segments are parallel");
        }

        return intersections;
```

```
    }
}
```

Код основного класса Main:

```
public class Main {

    public static void main(String[] args) throws IOException {

        // Вариант 1 задание 6

        ArrayList<Integer> pos_neg = new ArrayList<Integer>();
        try {
            BufferedReader reader = new BufferedReader(new
FileReader("src/Lab6_1/Lab6_1_6.txt"));
            String line;
            while ((line = reader.readLine()) != null) {
                pos_neg.add(Integer.valueOf(line));
            }
            reader.close();
        } catch (IOException e) {
            System.out.println("Error: " + e.getMessage());
        }

        pos_neg.sort(Collections.reverseOrder());

        for (Integer sortedInt : pos_neg) {
            System.out.println(sortedInt);
        }

        System.out.println("----------------------------------------------------
------------------------------------------------------------");

        // Вариант 1 задание 7

        ArrayList<String> lines = new ArrayList<String>();
        try {
            BufferedReader reader = new BufferedReader(new
FileReader("src/Lab6_1/Lab6_1_7.txt"));
            String line;
            while ((line = reader.readLine()) != null) {
                lines.add(line);
            }
            reader.close();
        } catch (IOException e) {
            System.out.println("Error: " + e.getMessage());
        }

        Collections.sort(lines);

        for (String sortedLine : lines) {
            System.out.println(sortedLine);
        }

        System.out.println("----------------------------------------------------
------------------------------------------------------------");

        // Вариант 2 задание 6

        List<Dot> dots = new ArrayList<>();
        dots.add(new Dot(1, 2));
        dots.add(new Dot(2, 4));
        dots.add(new Dot(3, 6));
```

```
        dots.add(new Dot(1, 3));
        dots.add(new Dot(2, 2));

        HashMap<Line, Integer> intersectionLinePoint = new HashMap<>();

        for (int i = 0; i < dots.size(); i++){
            Dot dot1 = dots.get(i);
            for (int j = i + 1; j < dots.size(); j++){
                Dot dot2 = dots.get(j);
                Line lineCompare = new Line(dot1, dot2);
                intersectionLinePoint.put(lineCompare,
intersectionLinePoint.getOrDefault(lineCompare, 0) + 1);
            }
        }

        FileWriter writer = new FileWriter("lines.txt", true);

        for (Map.Entry<Line, Integer> entry :
intersectionLinePoint.entrySet()){
            Line lineOut = entry.getKey();
            int countDots  = (int) (1 + Math.sqrt(1 + 8 * entry.getValue()))
/ 2;
            intersectionLinePoint.put(lineOut, countDots);
            String writerLine = String.format("Line equation: ( %f ) * x + (
%f ) * y + ( %f ) = 0; Number of crossed points: %d%n",
            lineOut.getA(), lineOut.getB(), lineOut.getC(), countDots);
            writer.write(writerLine);
            System.out.printf(writerLine);
        }
        writer.close();

        System.out.println("----------------------------------------------
------------------------------------------------------------");

        // Вариант 2 задание 7

        List<Segment> segments = new ArrayList<>();
        segments.add(new Segment(2, 4, 3, 5));
        segments.add(new Segment(1, 4, 4, 1));
        segments.add(new Segment(4, 4, 1, 5));

        TreeMap<Double, double[]> intersectionPointsWithMinAbscissa = new
TreeMap<>();

        for (int i = 0; i < segments.size(); i++) {
            Segment s1 = segments.get(i);
            for (int j = i + 1; j < segments.size(); j++) {
                Segment s2 = segments.get(j);
                List<double[]> intersectionPoints =
s1.getIntersectionPoints(s2);
                for (double[] point : intersectionPoints) {
                    double x = point[0];
                    double y = point[1];
                    intersectionPointsWithMinAbscissa.put(x, new double[]{x,
y});
                    System.out.println("Intersection point: (" + x + ", " + y
+ ")");
                }
            }
        }

        if (!intersectionPointsWithMinAbscissa.isEmpty()) {
            double[] minPoint =
intersectionPointsWithMinAbscissa.firstEntry().getValue();
```

```
            System.out.println("Intersection point with minimum abscissa: ("
+ minPoint[0] + ", " + minPoint[1] + ")");
        } else {
            System.out.println("No intersection points");
        }

    }

}
```

Работа программы представлена на рисунке 1.

```
145
35
16
2
0
0
-1
-21
-45
-----------------------------------------------------------------------------------------------------
Karlos
char
integer
lego
lol
memes
something
yandex
-----------------------------------------------------------------------------------------------------
Line equation: ( -4,000000 ) * x + ( 1,000000 ) * y + ( 6,000000 ) = 0; Number of crossed points: 2
Line equation: ( -2,000000 ) * x + ( 1,000000 ) * y + ( 0,000000 ) = 0; Number of crossed points: 3
Line equation: ( 1,000000 ) * x + ( 0,000000 ) * y + ( -1,000000 ) = 0; Number of crossed points: 2
Line equation: ( 0,000000 ) * x + ( 1,000000 ) * y + ( -2,000000 ) = 0; Number of crossed points: 2
Line equation: ( -1,000000 ) * x + ( 1,000000 ) * y + ( -2,000000 ) = 0; Number of crossed points: 2
Line equation: ( 1,000000 ) * x + ( 0,000000 ) * y + ( -2,000000 ) = 0; Number of crossed points: 2
Line equation: ( -1,500000 ) * x + ( 1,000000 ) * y + ( -1,500000 ) = 0; Number of crossed points: 2
Line equation: ( 1,000000 ) * x + ( 1,000000 ) * y + ( -4,000000 ) = 0; Number of crossed points: 2
-----------------------------------------------------------------------------------------------------
Intersection point: (2.0, 3.0)
Intersection point: (4.0, 5.0)
Intersection point: (4.0, 1.0)
Intersection point with minimum abscissa: (2.0, 3.0)
```

Рисунок 1 – Работа программы

Вывод: Была изучена работа с коллекциями в java.