Министерство образования и науки Российской Федерации

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

«Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

Кафедра информационных систем

и программной инженерии

Лабораторная работа № 4

“ Фреймворк коллекций.”

по дисциплине

«Платформонезависимое программирование»

Выполнил:

ст. гр. ПРИ-118

Баранов В. В.

Принял:

Казаев А. Т.

Владимир 2020

**Цель работы**

Познакомиться с библиотеками коллекций и научиться создавать компараторы.

**Выполнение работы**

**Задание.** Во всех вариантах необходимо запросить у пользователя N различных объектов аналогично тому, как это было сделано в программе в приложении, и заполнить на их основе коллекцию. Необходимо реализовать класс, хранящий в себе эти объекты, и переопределить в нем функцию toString, чтобы она выводила понятную для пользователя информацию. Необходимо реализовать компаратор аналогично тому, как это было сделано в примере. Необходимо отсортировать коллекцию, используя этот компаратор, после чего вывести ее пользователю в отсортированном порядке. Число N надо запросить у пользователя.

Необходимо расположить в правильном порядке точки. Каждая точка определяется двумя координатами x и y (обе координаты целочисленные). Точки должны сортироваться сначала по оси абсцисс, затем по оси ординат. Т.е. точка, которая находится левее, должна идти первой. В случае равенства по x первой должна идти та точка, которая располагается ниже по оси ординат. Вам дан набор из N точек на поверхности. Необходимо отсортировать их, после чего вывести в упорядоченном порядке в следующем формате:

Point: <Координата X> <Координата Y>

**Решение:**

Исходный код:

com.coordsystem.primitives.Point.java

1. package com.coordsystem.primitives;
2. public class Point {
3. private double x;
4. private double y;
6. public Point(double x, double y) {
7. this.x = x;
8. this.y = y;
9. }
11. @Override
12. public String toString() {
13. return "Point: " + x + " " + y;
14. }
16. public double getX() {
17. return x;
18. }
20. public double getY() {
21. return y;
22. }
23. }

com.coordsystem.comparators.PointComparator.java

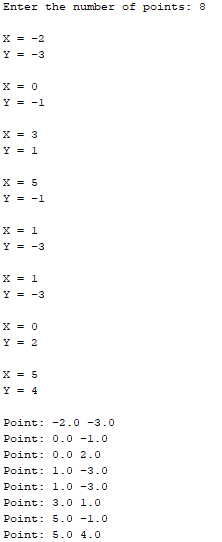
1. package com.coordsystem.comparators;
2. import com.coordsystem.primitives.Point;
3. import java.util.Comparator;
4. public class PointComparator implements Comparator<Object> {
5. @Override
6. public int compare(Object o1, Object o2) {
7. Point p1 = (Point) o1;
8. Point p2 = (Point) o2;
9. return (p1.getX() < p2.getX()) ? -1 : ((p1.getX() == p2.getX()) ? 0 : 1);
10. }
11. }

com.coordsystem.MainClass.java

1. package com.coordsystem;
2. import com.coordsystem.comparators.PointComparator;
3. import com.coordsystem.primitives.Point;
4. import java.util.ArrayList;
5. import java.util.Collections;
6. import java.util.Scanner;
7. public class MainClass {
9. public static void main(String[] args) {
10. Scanner console = new Scanner(System.in);
11. ArrayList pointList = new ArrayList();
12. int N;

15. System.out.print("Enter the number of points: ");
16. N = console.nextInt();
17. System.out.println();
19. for(int i = 0; i < N; i++) {
20. System.out.print("X = ");
21. double x = console.nextDouble();
22. System.out.print("Y = ");
23. double y = console.nextDouble();
24. System.out.println();
26. pointList.add(new Point(x,y));
27. }
29. sort(pointList);
30. }
32. private static void sort(ArrayList list) {
33. Collections.sort(list, new PointComparator());
34. for(Object obj : list) {
35. Point point = (Point)obj;
36. System.out.println(obj.toString());
37. }
38. }
39. }

Тестирование и отладка:



*Рисунок 1. Рейзльтат работы программы*

**Вывод**

В результа выполнения лабораторной раоты было получено знакомство с библиотеками коллекций и получены навыки создания компараторов.