Министерство науки и высшего образования 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

по дисциплине «Платформонезависимое

программирование»

Тема: «Фреймворк коллекций»

Выполнил:

студент гр. ПРИ-123

А.Ц. Нямаа

Приняла:

Ст. пр. кафедры ИСПИ

О.Н. Шамышева

Владимир, 2025 г.

ЦЕЛЬ РАБОТЫ:

Познакомиться с библиотеками коллекций и научиться создавать компараторы.

ЗАДАНИЕ 1 (вариант 12).

Необходимо расположить в правильно порядке точки. Каждая точка определяется двумя координатами x и y (обе координаты целочисленные). Точки должны сортироваться сначала по оси абсцисс, затем по оси ординат. Дан набор из N точек, который необходимо отсортировать и вывести в упорядоченном порядке. Число N запрашивается у пользователя, сортировка осуществляется при помощи компаратора.

ВЫПОЛНЕНИЕ РАБОТЫ:

В начале выполнения данного задания я определил физическую структуру приложения.

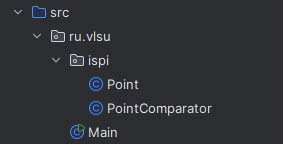


Рисунок 1. Физическая структура программы

Далее реализовал созданные классы таким образом, что исполняющим классом является Main, класс Point представляет собой объект точки, содержащий в себе поля координат, и класс PointComparator, являющийся компаратором. Код программы представлен ниже:

Листинг Main.java:

package ru.vlsu;  
  
import java.util.ArrayList;  
import java.util.Collections;  
import java.util.List;  
import java.util.Scanner;  
  
import ru.vlsu.ispi.Point;  
import ru.vlsu.ispi.PointComparator;  
  
public class Main {  
 public static void main(String[] args) {  
 Scanner in = new Scanner(System.in);  
 List<Point> pointList = new ArrayList<>();  
  
 System.out.println("Введите количество точек:");  
 int countPoints = in.nextInt();  
  
 System.out.println("Введите координату x и y через пробел:");  
 for (int i = 0; i < countPoints; i++){  
 System.out.print("Координата " + (i + 1) + ": ");  
 pointList.add(new Point(in.nextInt(), in.nextInt()));  
 }  
  
 Collections.sort(pointList, new PointComparator());  
 for (Point pointUnit : pointList){  
 System.out.println(pointUnit);  
 }  
 }  
}

Листинг Point.java:

package ru.vlsu.ispi;  
  
public class Point {  
 private final int xOrd;  
 private final int yOrd;  
  
 public Point(int xOrd, int yOrd){  
 this.xOrd = xOrd;  
 this.yOrd = yOrd;  
 }  
  
 public Integer getXOrd(){ return xOrd; }  
 public Integer getYOrd(){ return yOrd; }  
  
 public String toString() { return xOrd + " " + yOrd; }  
}

Листинг PointComparator.java:

package ru.vlsu.ispi;  
  
import java.util.Comparator;  
  
public class PointComparator implements Comparator {  
 @Override  
 public int compare(Object point1, Object point2){  
 Point pointFirst = (Point) point1;  
 Point pointSecond = (Point) point2;  
  
 int result = pointFirst.getXOrd().compareTo(pointSecond.getXOrd());  
 if (result != 0){  
 return result;  
 }  
 return pointFirst.getYOrd().compareTo(pointSecond.getYOrd());  
 }  
}

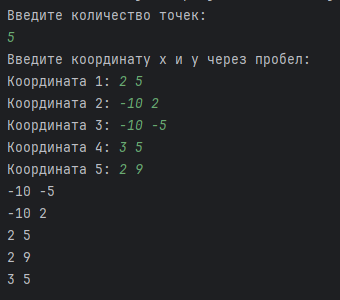


Рисунок 2. Выполнение программы.

ЗАДАНИЕ 2 (вариант 12).

Пример: дан список людей, который необходимо отсортировать в лексикографическом порядке. У людей есть фамилия и имя. Если у людей разные фамилии, то первым в списке должен идти тот человек, у которого фамилия идет раньше в алфавитном порядке. Для двух людей с одинаковыми фамилиями все разрешается по имени. Необходимо отсортировать имеющийся список и вывести его на экран.

ВЫПОЛНЕНИЕ РАБОТЫ:

В начале выполнения данного задания я определил физическую структуру приложения-примера.

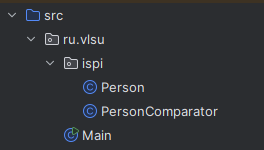


Рисунок 3. Физическая структура программы-примера.

В ходе реализации программы-примеры были созданы следующие классы: исполняющий класс main, класс Person, соответствующий человеку, и класс компаратор PersonComparator.

Листинг Main.java:

package ru.vlsu;  
  
import java.util.Collections;  
import java.util.LinkedList;  
import java.util.List;  
import java.util.Scanner;  
  
import ru.vlsu.ispi.PersonComparator;  
import ru.vlsu.ispi.Person;  
  
public class Main {  
 private Person read(Scanner in) {  
 String lastname = in.next();  
 String firstname = in.next();  
 return new Person(lastname, firstname);  
 }  
  
 private void run() {  
 Scanner in = new Scanner(System.in);  
 List<Person> peopleList = new LinkedList<>();  
  
 System.out.println("Введите количество человек:");  
 int countPeople = in.nextInt();  
  
 System.out.println("Введите имя и фамилию через пробел:");  
 for (int i = 0; i < countPeople; i++) {  
 System.out.print(i + 1 + ": ");  
 peopleList.add(read(in));  
 }  
  
 Collections.sort(peopleList, new PersonComparator());  
 for (Person personUnit : peopleList) {  
 System.out.println(personUnit);  
 }  
 }  
  
 public static void main(String[] args) {  
 new Main().run();  
 }  
}

Листинг Person.java:

package ru.vlsu.ispi;  
  
public class Person {  
 private final String lastname;  
 private final String firstname;  
  
 public Person(String lastname, String firstname) {  
 this.lastname = lastname;  
 this.firstname = firstname;  
 }  
  
 public String getLastname() { return lastname; }  
 public String getFirstname() { return firstname; }  
  
 public String toString() { return lastname + " " + firstname; }  
}

Листинг PointComparator.java:

package ru.vlsu.ispi;  
  
import java.util.Comparator;  
  
public class PersonComparator implements Comparator {  
 @Override  
 public int compare(Object o1, Object o2) {  
 Person p1 = (Person) o1;  
 Person p2 = (Person) o2;  
  
 int res = p1.getLastname().compareTo(p2.getLastname());  
 if (res != 0){  
 return res;  
 }  
 return p1.getFirstname().compareTo(p2.getFirstname());  
 }  
}

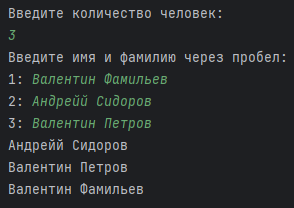


Рисунок 4. Выполнение программы-примера.

ВЫВОД К РАБОТЕ:

Я познакомился с библиотеками коллекций и научился создавать компараторы.