|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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.**

**Отчет**

**по лабораторной работе № 6**

**Вариант № 6**

**Название:** Коллекции

**Дисциплина:** Языки программирования для работы с большими данными

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-22М |  |  | Д.А. Залимханов |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | П.В. Степанов |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2024

**Цель:** изучить и освоить работу с коллекциями в Kotlin.

**Задание 1:** не используя вспомогательных объектов, переставить отрицательные элементы данного списка в конец, а положительные – в начало этого списка.

Код решения задачи:

fun rearrangeList(nums: MutableList<Int>) {

val (positive, negative) = nums.partition { it > 0 }

nums.clear()

nums.addAll(positive)

nums.addAll(negative)

}

fun main(){

val nums: MutableList<Int> = mutableListOf(1,-5,2,-4, -18, 12, 2)

rearrangeList(nums)

// Либо просто отсортировать по убыванию

// nums.sortDescending()

println(nums)

}

Результат работы программы представлен на рисунке 1.

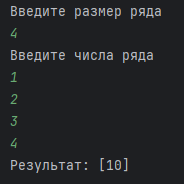


Рисунок 1 – Результат работы программы

**Задание 2:** ввести строки из файла, записать в список ArrayList. Выполнить сортировку строк, используя метод sort() из класса Collections.

Код решения задачи:

fun sortFileLines(fileName: String): ArrayList<String> {

val lines = File(fileName).readLines().toCollection(ArrayList())

lines.sort()

return lines

}

fun main() {

val sortedLines = sortFileLines("D:\\kotlinBigData\\kotlin\_labs\\src\\main\\kotlin\\lab6\\var1\\lab6.txt")

println("Sorted lines:")

sortedLines.forEach { println(it) }

}

Результат работы программы показан на рисунке 2.

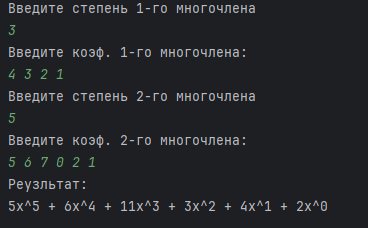


Рисунок 2 – Результат работы программы

**Задание 3:** 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 решения задачи:

data class Point(val x: Int, val y: Int)

fun findLines(points: List<Point>): Map<String, MutableList<Point>> {

val lines = mutableMapOf<String, MutableList<Point>>()

for (i in points.indices) {

for (j in i + 1 until points.size) {

val p1 = points[i]

val p2 = points[j]

val key = if (p1.x == p2.x) {

"x=${p1.x}" // вертикальная линия

} else {

val slope = (p2.y - p1.y).toDouble() / (p2.x - p1.x)

val intercept = p1.y - slope \* p1.x

"y=${slope}x+$intercept"

}

lines.getOrPut(key) { mutableListOf() }.apply {

add(p1)

add(p2)

}

}

}

return lines

}

fun main() {

val points = listOf(Point(0, 0), Point(1, 1), Point(2, 2), Point(3, 3), Point(1, 2))

val lines = findLines(points)

lines.forEach { (line, pts) ->

println("Прямая $line проходит через точки: ${pts.distinct()}")

}

}

Результат работы программы представлен на рисунке 3.

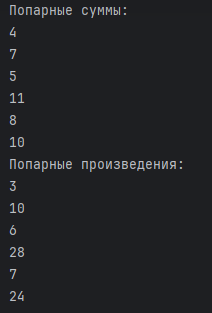
****

Рисунок 3 – Результат работы программы

**Задание 4:** на плоскости задано N отрезков. Найти точку пересечения двх отрезков, имеющую минимальную абсциссу. Использовать класс TreeMap.

Код решения задачи:

data class MyLine(val a: Double, val b: Double) // Представление отрезка как y = ax + b

data class MyPoint(val x: Double, val y: Double)

fun findIntersection(l1: MyLine, l2: MyLine): MyPoint? {

if (l1.a == l2.a) {

return null // Линии параллельны, если угловые коэффициенты равны

}

val x = (l2.b - l1.b) / (l1.a - l2.a)

val y = l1.a \* x + l1.b

return MyPoint(x, y)

}

fun findMinimumXIntersection(lines: List<MyLine>): MyPoint? {

val intersections = TreeMap<Double, MyPoint>()

for (i in lines.indices) {

for (j in i + 1 until lines.size) {

val l1 = lines[i]

val l2 = lines[j]

val intersection = findIntersection(l1, l2)

intersection?.let {

intersections[it.x] = it

}

}

}

return intersections.firstEntry()?.value

}

fun main() {

val lines = listOf(

MyLine(1.0, 0.0),

MyLine(-1.0, 4.0),

MyLine(0.5, 1.0),

MyLine(2.0, -1.0)

)

val intersection = findMinimumXIntersection(lines)

if (intersection != null) {

println("Точка пересечения с минимальной абсциссой это: (${intersection.x}, ${intersection.y})")

} else {

println("Нет пересечений")

}

}

Результат работы программы показан на рисунке 4.

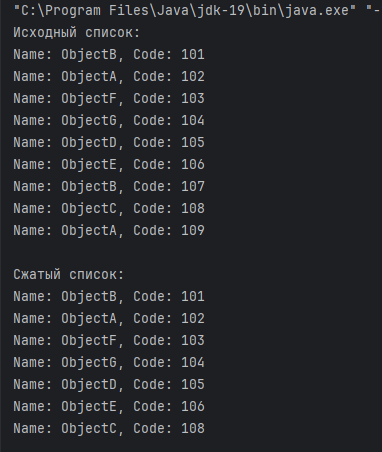


Рисунок 4 – Результат работы программы

**Вывод:** была освоена работа с коллекциями в Kotlin.