|  |  |
| --- | --- |
| 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 **09.04.01/12 Интеллектуальный анализ больших**

**данных в системах поддержки принятия решений.**

**Отчет**

**по лабораторной работе № 6**

**Вариант № 3**

**Название:** Коллекции

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-23М |  |  | В.Н. Зыкин |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | П.В. Степанов |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2023

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

**Задание 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.

Код класса Main:

import java.util.\*;  
  
public class Main {  
 public static void main(String[] args) {  
 Scanner in = new Scanner(System.in);  
 Set<Integer> set = new LinkedHashSet<>();  
 System.out.println("Введите размер ряда");  
 int n = in.nextInt();  
 System.out.println("Введите числа ряда");  
 for (int i = 0; i < n; i++) {  
 set.add(in.nextInt());  
 }  
 System.out.println("Результат: " + sumPairs(set));  
  
  
 }  
  
 public static Set<Integer> sumPairs(Set<Integer> set) {  
 if (set.size() < 2) {  
 return set;  
 }  
 Set<Integer> pairs = new LinkedHashSet<>();  
 for (Iterator<Integer> it = set.iterator(); it.hasNext();) {  
 Integer p1 = it.next();  
 if (it.hasNext()) {  
 p1 += it.next();  
 }  
 pairs.add(p1);  
 }  
 return sumPairs(pairs);  
 }  
}

Результат работы программы представлен на рисунке 1.

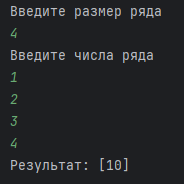


Рисунок 1 – Результат работы программы

**Задание 2:** сложить два многочлена заданной степени, если коэффициенты многочленов хранятся в объекте HashMap.

Код класса Main:

import java.util.HashMap;  
import java.util.Map;  
import java.util.Scanner;  
  
public class Main {  
 public static void main(String[] args) {  
 Scanner in = new Scanner(System.in);  
 Map<Integer,Integer> ch1 = new HashMap<>();  
 Map<Integer,Integer> ch2 = new HashMap<>();  
  
 System.out.println("Введите степень 1-го многочлена");  
 int n1 = in.nextInt();  
 System.out.println("Введите коэф. 1-го многочлена:");  
 for (int i = 0; i <= n1; i++) {  
 ch1.put(n1-i, in.nextInt());  
 }  
  
 System.out.println("Введите степень 2-го многочлена");  
 int n2 = in.nextInt();  
 System.out.println("Введите коэф. 2-го многочлена:");  
 for (int i = 0; i <= n2; i++) {  
 ch2.put(n2-i, in.nextInt());  
 }  
  
 Map<Integer, Integer> res = new HashMap<>();  
 for(int i = 0; i <= Integer.max(n1,n2); i++) {  
 res.put(i,ch1.getOrDefault(i,0) + ch2.getOrDefault(i,0));  
 }  
 System.out.println("Реузльтат:");  
 StringBuilder sb = new StringBuilder();  
 for (int i = Integer.max(n1,n2); i >= 0 ; i--) {  
 sb.append(res.get(i)).append("x^").append(i);  
 if(i > 0) {  
 sb.append(" + ");  
 }  
 }  
  
 System.out.println(sb);  
  
 }  
}

Результат работы программы показан на рисунке 2.

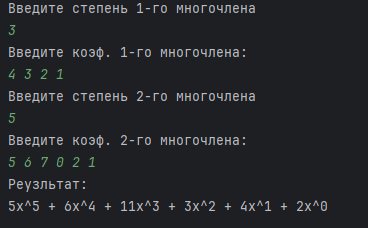


Рисунок 2 – Результат работы программы

**Задание 3:** во входном файле хранятся две разреженные матрицы А и В. Построить циклически связанные списки СА и СВ, содержащие ненулевые элементы соответственно матриц А и В. Просматривая списки, вычислить: а) сумму S = A + B; б) произведение P = A \* B.

Код класса Main:

import java.io.File;  
import java.io.FileNotFoundException;  
import java.util.ArrayList;  
import java.util.List;  
import java.util.Scanner;  
  
class ListNode {  
 int value;  
 ListNode next;  
  
 ListNode(int value) {  
 this.value = value;  
 this.next = this;  
 }  
}  
  
public class Main {  
  
 private static List<ListNode> readMatrixAndCreateList(String filePath) throws FileNotFoundException {  
 File file = new File(filePath);  
 Scanner scanner = new Scanner(file);  
 List<ListNode> nodes = new ArrayList<>();  
  
 while (scanner.hasNextLine()) {  
 String[] currentLine = scanner.nextLine().trim().split("\\s+");  
 for (String valueStr : currentLine) {  
 int value = Integer.parseInt(valueStr);  
 if (value != 0) { // Добавляем только ненулевые элементы  
 nodes.add(new ListNode(value));  
 }  
 }  
 }  
 scanner.close();  
  
 // Создаем циклический связный список  
 for (int i = 0; i < nodes.size() - 1; i++) {  
 nodes.get(i).next = nodes.get(i + 1);  
 }  
 if (!nodes.isEmpty()) {  
 nodes.get(nodes.size() - 1).next = nodes.get(0); // Замыкаем цикл  
 }  
  
 return nodes;  
 }  
  
 private static void processLists(List<ListNode> listA, List<ListNode> listB) {  
 List<Integer> sums = new ArrayList<>();  
 List<Integer> products = new ArrayList<>();  
  
 for (int i = 0; i < Math.min(listA.size(), listB.size()); i++) {  
 int sum = listA.get(i).value + listB.get(i).value;  
 int product = listA.get(i).value \* listB.get(i).value;  
 sums.add(sum);  
 products.add(product);  
 }  
  
 System.out.println("Попарные суммы:");  
 sums.forEach(System.out::println);  
  
 System.out.println("Попарные произведения:");  
 products.forEach(System.out::println);  
 }  
  
 public static void main(String[] args) {  
 try {  
 List<ListNode> listA = readMatrixAndCreateList("matrixA.txt");  
 List<ListNode> listB = readMatrixAndCreateList("matrixB.txt");  
 processLists(listA, listB);  
 } catch (FileNotFoundException e) {  
 System.err.println("Файл не найден: " + e.getMessage());  
 }  
 }  
}

Результат работы программы представлен на рисунке 3.

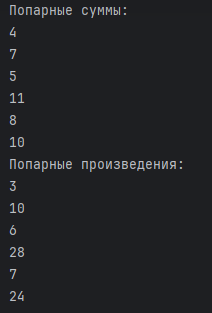
****

Рисунок 3 – Результат работы программы

**Задание 4:** во входном файле хранятся наименования некоторых объектов. Построить список C1, элементы которого содержат наименования и шифры данных объектов, причем элементы списка должны быть упорядочены по возрастанию шифров. Затем “сжать” список C1, удаляя дублирующие наименования объектов.

Код класса Main:

import java.io.File;  
import java.io.FileNotFoundException;  
import java.util.ArrayList;  
import java.util.List;  
import java.util.Scanner;  
import java.util.Comparator;  
  
class Item {  
 String name;  
 int code;  
  
 Item(String name, int code) {  
 this.name = name;  
 this.code = code;  
 }  
}  
  
public class Main {  
  
 public static void main(String[] args) {  
 String filePath = "input.txt"; // Замените на ваш путь к файлу  
 List<Item> items = new ArrayList<>();  
  
 // Чтение и создание списка объектов  
 try (Scanner scanner = new Scanner(new File(filePath))) {  
 while (scanner.hasNextLine()) {  
 String line = scanner.nextLine();  
 String[] parts = line.split("\\s+");  
 items.add(new Item(parts[0], Integer.parseInt(parts[1])));  
 }  
 } catch (FileNotFoundException e) {  
 System.err.println("Файл не найден: " + e.getMessage());  
 return;  
 }  
  
 items.sort(Comparator.comparingInt(item -> item.code));  
  
 // Вывод исходного списка  
 System.out.println("Исходный список:");  
 items.forEach(item -> System.out.println("Name: " + item.name + ", Code: " + item.code));  
  
 // Удаление дубликатов по наименованию, сохраняя первое вхождение  
 List<Item> compressedList = new ArrayList<>();  
 items.stream()  
 .filter(item -> compressedList.stream().noneMatch(it -> it.name.equals(item.name)))  
 .forEach(compressedList::add);  
  
 // Вывод сжатого списка  
 System.out.println("\nСжатый список:");  
 compressedList.forEach(item -> System.out.println("Name: " + item.name + ", Code: " + item.code));  
 }  
}

Результат работы программы показан на рисунке 4.

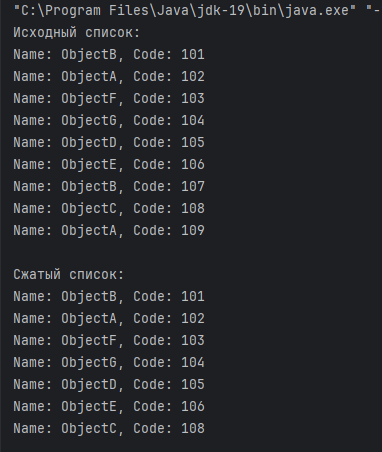


Рисунок 4 – Результат работы программы

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