|  |  |
| --- | --- |
| **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**

**ОТЧЕТ**

**по лабораторной работе № 6**

|  |  |
| --- | --- |
| **Название:** | Коллекции |
| **Дисциплина:** | Языки программирования для работы с большими данными |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-22М |  | 28.04.2023 | И.Б. Нуриддинов |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | П.В. Степанов |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2023

Цель работы: Освоение и применение концепций коллекций в языке программирования Java.

*Вариант 1. Задание 1.* Сложить два многочлена заданной степени, если коэффициенты многочленов хранятся в объекте HashMap.

public class Main {

public static void main(String[] args) {

// Создание и заполнение первого многочлена

Map<Integer, Integer> polynomial1 = new HashMap<>();

polynomial1.put(0, 2);

polynomial1.put(1, 4);

polynomial1.put(2, 1);

// Создание и заполнение второго многочлена

Map<Integer, Integer> polynomial2 = new HashMap<>();

polynomial2.put(0, 1);

polynomial2.put(1, 3);

polynomial2.put(2, 5);

// Вычисление суммы многочленов

Map<Integer, Integer> sum = addPolynomials(polynomial1, polynomial2);

// Вывод результата

System.out.println("Сумма многочленов: " + sum);

}

public static Map<Integer, Integer> addPolynomials(Map<Integer, Integer> poly1, Map<Integer, Integer> poly2) {

// Создание нового многочлена для хранения суммы

Map<Integer, Integer> sum = new HashMap<>();

// Обработка коэффициентов первого многочлена

for (Map.Entry<Integer, Integer> term : poly1.entrySet()) {

int exponent = term.getKey();

int coefficient = term.getValue();

sum.put(exponent, coefficient);

}

// Обработка коэффициентов второго многочлена

for (Map.Entry<Integer, Integer> term : poly2.entrySet()) {

int exponent = term.getKey();

int coefficient = term.getValue();

if (sum.containsKey(exponent)) {

// Если степень уже присутствует, складываем коэффициенты

int existingCoefficient = sum.get(exponent);

sum.put(exponent, existingCoefficient + coefficient);

} else {

// Если степени нет, просто добавляем новую пару степень-коэффициент

sum.put(exponent, coefficient);

}

}

return sum;

}

}

*Вариант 2. Задание 1.* 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.

public class Main {

public static void main(String[] args) {

String inputFileName = "input.txt";

List<ObjectInfo> c1 = buildC1(inputFileName);

System.out.println("Список C1 до сжатия:");

printObjectList(c1);

compressObjectList(c1);

System.out.println("Список C1 после сжатия:");

printObjectList(c1);

}

private static List<ObjectInfo> buildC1(String fileName) {

List<ObjectInfo> c1 = new ArrayList<>();

try (BufferedReader reader = new BufferedReader(new FileReader(fileName))) {

String line;

while ((line = reader.readLine()) != null) {

// Парсинг строки наименования и шифра объекта

String[] parts = line.split(";");

if (parts.length == 2) {

String name = parts[0].trim();

int code = Integer.parseInt(parts[1].trim());

ObjectInfo objectInfo = new ObjectInfo(name, code);

c1.add(objectInfo);

}

}

} catch (IOException e) {

e.printStackTrace();

}

// Сортировка списка по возрастанию шифров

Collections.sort(c1);

return c1;

}

private static void compressObjectList(List<ObjectInfo> objectList) {

Set<String> namesSet = new HashSet<>();

List<ObjectInfo> compressedList = new ArrayList<>();

for (ObjectInfo objectInfo : objectList) {

if (!namesSet.contains(objectInfo.getName())) {

compressedList.add(objectInfo);

namesSet.add(objectInfo.getName());

}

}

objectList.clear();

objectList.addAll(compressedList);

}

private static void printObjectList(List<ObjectInfo> objectList) {

for (ObjectInfo objectInfo : objectList) {

System.out.println(objectInfo.getName() + " - " + objectInfo.getCode());

}

}

private static class ObjectInfo implements Comparable<ObjectInfo> {

private String name;

private int code;

public ObjectInfo(String name, int code) {

this.name = name;

this.code = code;

}

public String getName() {

return name;

}

public int getCode() {

return code;

}

@Override

public int compareTo(ObjectInfo other) {

return Integer.compare(this.code, other.code);

}

}

}

Вывод: Выполнение лабораторной работы позволило овладеть важными навыками работы с коллекциями и их различными реализациями. Эти навыки могут быть успешно применены в различных областях разработки программного обеспечения, улучшая эффективность и надежность программ.