|  |  |
| --- | --- |
| 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 |

**Название:** Коллекции

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

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

Москва, 2024

**Цель работы:** освоить базовые принципы работы с коллекциями в Java.

**Вариант: 2.**

**Задание 1:** Списки (стеки, очереди) I(1..n) и U(1..n) содержат результаты n измерений тока и напряжения на неизвестном сопротивлении R. Найти приближенное число R методом наименьших квадратов.

Код решения приведен в листинге 1.

Листинг 1 — Задание 1

import java.util.ArrayList;  
import java.util.Random;  
import java.util.stream.IntStream;  
  
public class Main {  
 public static void main(String[] args) {  
 var I = new ArrayList<Double>();  
 var U = new ArrayList<Double>();  
  
 var n = 5;  
  
 var random = new Random();  
  
 for (int i = 0; i < n; i++) {  
 I.add(random.nextInt(1, 10) + (double) random.nextInt(0, 10) / 10);  
 U.add((double) random.nextInt(100, 220));  
 }  
  
 System.*out*.print("I: ");  
  
 for (Double entry : I) {  
 System.*out*.printf("%s ", entry);  
 }  
  
 System.*out*.println();  
 System.*out*.print("U: ");  
  
 for (Double entry : U) {  
 System.*out*.printf("%s ", entry);  
 }  
  
 System.*out*.println();  
 System.*out*.printf("R: %s", *getR*(I, U));  
 }  
  
 private static double getR(ArrayList<Double> I, ArrayList<Double> U) {  
 var top = IntStream  
 .*range*(0, I.size())  
 .mapToObj(i -> I.get(i) \* U.get(i))  
 .mapToDouble(x -> x)  
 .sum();  
  
 var bottom = I.stream()  
 .mapToDouble(x -> x \* x)  
 .sum();  
  
 return top / bottom;  
 }  
}

**Задание 2:** 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 решения приведен в листинге 2.

Листинг 2 — Задание 2

import java.util.HashSet;  
import java.util.Random;  
import java.util.SortedSet;  
import java.util.TreeSet;  
  
public class Main {  
 public static void main(String[] args) {  
 var n = 10;  
 var random = new Random();  
  
 var start = random.nextInt(0, 100);  
  
 SortedSet<Integer> set = new TreeSet<>();  
  
 for (int i = 0; i < n; i++) {  
 set.add(start + i);  
 }  
  
 System.out.println(set);  
  
 while (set.size() > 1) {  
 SortedSet<Integer> subSet = new TreeSet<>();  
 var sum = 0;  
 var isSecondElement = false;  
  
 for (var x : set) {  
 sum += x;  
  
 if (isSecondElement) {  
 subSet.add(sum);  
  
 sum = 0;  
 isSecondElement = false;  
  
 continue;  
 }  
  
 isSecondElement = true;  
 }  
  
 if (isSecondElement) {  
 subSet.add(sum);  
 }  
  
 System.*out*.println(subSet);  
 set = subSet;  
 }  
 }  
}

**Задание 3:** Реализовать класс, моделирующий работу N-местной автостоянки. Машина подъезжает к определенному месту и едет вправо, пока не встретится свободное место. Класс должен поддерживать методы, обслуживающие приезд и отъезд машины.

Код решения приведен в листингах 3 и 4.

Листинг 3 – Задание 3 (1)

public class Main {  
 public static void main(String[] args) throws Exception {  
 var parking = new Parking(10);  
  
 parking.placeCar("U234");  
 parking.placeCar("P032");  
  
 parking.accountCarDeparture("P032");  
  
 parking.placeCar("X909");  
 }  
}

Листинг 4 – Задание 3 (2)

import java.util.HashMap;  
import java.util.HashSet;  
  
public class Parking {  
 private final int \_size;  
 private final HashMap<String, Integer> \_plateToParkingLotMap;  
 private final HashSet<Integer> \_occupiedParkingLots;  
  
 public Parking(int size) {  
 \_size = size;  
 \_plateToParkingLotMap = new HashMap<>(size);  
 \_occupiedParkingLots = new HashSet<>(size);  
 }  
  
 public void placeCar(String plateNumber) throws Exception {  
 if (\_plateToParkingLotMap.containsKey(plateNumber)) {  
 throw new Exception(String.*format*("Car with plate number %s already parked", plateNumber));  
 }  
  
 var lotFound = false;  
  
 for (int i = 0; i < \_size; i++) {  
 if (!\_occupiedParkingLots.contains(i)) {  
 \_plateToParkingLotMap.put(plateNumber, i);  
 \_occupiedParkingLots.add(i);  
  
 lotFound = true;  
  
 break;  
 }  
 }  
  
 if (!lotFound) {  
 throw new Exception("No free parking lots at the moment");  
 }  
 }  
  
 public void accountCarDeparture(String plateNumber) throws Exception {  
 if (!\_plateToParkingLotMap.containsKey(plateNumber)) {  
 throw new Exception(String.format("Car with plate number %s is not parked", plateNumber));  
 }  
  
 var parkingLot = \_plateToParkingLotMap.get(plateNumber);  
  
 \_plateToParkingLotMap.remove(plateNumber);  
 \_occupiedParkingLots.remove(parkingLot);  
 }  
}

**Задание 4:** 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 решения приведен в листингах 5 и 6.

Листинг 5 — Задание 4 (1)

import java.io.BufferedReader;  
import java.io.FileReader;  
import java.io.IOException;  
import java.util.LinkedList;  
  
public class Main {  
 public static void main(String[] args) throws IOException {  
 var reader = new BufferedReader(new FileReader(args[0] + '/' + args[1]));  
  
 var firstList = new LinkedList<MatrixElement>();  
  
 String currentLine = reader.readLine();  
 var y = 0;  
  
 while (currentLine != null) {  
 if (readMatrixLine(currentLine, y, firstList)) break;  
  
 currentLine = reader.readLine();  
 y++;  
 }  
  
 var secondList = new LinkedList<MatrixElement>();  
  
 currentLine = reader.readLine();  
 y = 0;  
  
 while (currentLine != null) {  
 if (readMatrixLine(currentLine, y, secondList)) break;  
  
 currentLine = reader.readLine();  
 y++;  
 }  
  
 var sum = new int[5][5];  
  
 MatrixElement lastCheckedElement = null;  
  
 for (var matrixElement : firstList) {  
 int i = 0;  
 int j = 0;  
  
 if (lastCheckedElement != null) {  
 i = lastCheckedElement.Y;  
 j = lastCheckedElement.X + 1;  
 }  
  
 while (i <= matrixElement.Y) {  
 var jUpperBound = i == matrixElement.Y ? matrixElement.X : 5;  
  
 while (j < jUpperBound) {  
 sum[i][j] = 0;  
 j++;  
 }  
  
 i++;  
 }  
  
 sum[matrixElement.Y][matrixElement.X] = matrixElement.Value;  
  
 lastCheckedElement = matrixElement;  
 }  
  
 for (var matrixElement : secondList) {  
 sum[matrixElement.Y][matrixElement.X] += matrixElement.Value;  
 }  
  
 var product = new int[5][5];  
  
 for (int i = 0; i < 5; i++) {  
 for (int j = 0; j < 5; j++) {  
 product[i][j] = 0;  
 }  
 }  
  
 for (var firstElement : firstList) {  
 for (var secondElement : secondList) {  
 if (secondElement.Y == firstElement.X) {  
 product[firstElement.Y][secondElement.X] += firstElement.Value \* secondElement.Value;  
 }  
 }  
 }  
 }  
  
 private static boolean readMatrixLine(String currentLine, int y, LinkedList<MatrixElement> list) {  
 if (currentLine.isEmpty()) {  
 return true;  
 }  
  
 var x = 0;  
  
 var elements = currentLine.split(" ");  
  
 for (var element : elements) {  
 var value = Integer.parseInt(element);  
  
 if (value != 0) {  
 var matrixElement = new MatrixElement(value, x, y);  
  
 list.add(matrixElement);  
 }  
  
 x++;  
 }  
  
 return false;  
 }  
}

Листинг 6 – Задание 4 (2)

public class MatrixElement {  
 public int Value;  
 public int X;  
 public int Y;  
  
 public MatrixElement(int value, int x, int y) {  
 Value = value;  
 X = x;  
 Y = y;  
 }  
}

**Вывод:** в результате выполнения лабораторной работы были разработаны несколько программ в соответствие с требованиями задания.