Министерство науки и высшего образования РФ

ФГАОУ ВО Пермский национальный исследовательский

политехнический университет

Кафедра «Вычислительная математика, механика и биомеханика»

Отчет по лабораторной работе № 10

тема «Файлы»

по дисциплине «Информатика»

помогите…

Выполнил: студент группы ИСТ-22-1б Богданов Д.О.

Проверил: Нетбай Г.В.

Пермь, 2023

**Содержание**

[Задание 1 4](#_Toc136801676)

[1.1. Постановка задачи 4](#_Toc136801677)

[1.2. Решение задачи, код программы 4](#_Toc136801678)

[1.3. Тестирование работы программы с проверкой 5](#_Toc136801679)

[Задание 2 6](#_Toc136801680)

[2.1. Постановка задачи 6](#_Toc136801681)

[2.2. Решение задачи, код программы 6](#_Toc136801682)

[2.3. Тестирование работы программы с проверкой 6](#_Toc136801683)

[Задание 3 7](#_Toc136801684)

[3.1. Постановка задачи 7](#_Toc136801685)

[3.2. Решение задачи, код программы 7](#_Toc136801686)

[3.3. Тестирование работы программы с проверкой 7](#_Toc136801687)

[Задание 4 8](#_Toc136801688)

[4.1. Постановка задачи 8](#_Toc136801689)

[4.2. Решение задачи, код программы 8](#_Toc136801690)

[4.3. Тестирование работы программы 8](#_Toc136801691)

[Задание 5 10](#_Toc136801692)

[5.1. Постановка задачи 10](#_Toc136801693)

[5.2. Решение задачи, код программы 10](#_Toc136801694)

[5.3. Тестирование работы программы с проверкой 10](#_Toc136801695)

[Задание 6.2. 12](#_Toc136801696)

[6.1. Постановка задачи 12](#_Toc136801697)

[6.2. Решение задачи, код программы 12](#_Toc136801698)

[6.3. Тестирование работы программы с проверкой 12](#_Toc136801699)

[Задание 7 13](#_Toc136801700)

[7.1. Постановка задачи 13](#_Toc136801701)

[7.2. Решение задачи, код программы 13](#_Toc136801702)

[7.3. Тестирование работы программы с проверкой 13](#_Toc136801703)

[Задание 8 14](#_Toc136801704)

[8.1. Постановка задачи 14](#_Toc136801705)

[8.2. Решение задачи, код программы 14](#_Toc136801706)

[8.3. Тестирование работы программы с проверкой 14](#_Toc136801707)

[Задание 9 15](#_Toc136801708)

[9.1. Постановка задачи 15](#_Toc136801709)

[9.2. Решение задачи, код программы 15](#_Toc136801710)

[9.3. Тестирование работы программы с проверкой 16](#_Toc136801711)

[Задание 10 17](#_Toc136801712)

[10.1. Постановка задачи 17](#_Toc136801713)

[10.2. Решение задачи, код программы 17](#_Toc136801714)

[10.3. Тестирование работы программы с проверкой 17](#_Toc136801715)

[Задание 11 19](#_Toc136801716)

[11.1. Постановка задачи 19](#_Toc136801717)

[11.2. Решение задачи, код программы 19](#_Toc136801718)

[11.3. Тестирование работы программы с проверкой 19](#_Toc136801719)

[Задание 12 21](#_Toc136801720)

[12.1. Постановка задачи 21](#_Toc136801721)

[12.2. Решение задачи, код программы 21](#_Toc136801722)

[12.3. Тестирование работы программы с проверкой 21](#_Toc136801723)

[Задание 13 23](#_Toc136801724)

[13.1. Постановка задачи 23](#_Toc136801725)

[13.2. Решение задачи, код программы 23](#_Toc136801726)

[13.3. Тестирование работы программы с проверкой 24](#_Toc136801727)

[Задание 14 25](#_Toc136801728)

[14.1. Постановка задачи 25](#_Toc136801729)

[14.2. Решение задачи, код программы 25](#_Toc136801730)

[14.3. Тестирование работы программы с проверкой 26](#_Toc136801731)

[Задание 15 27](#_Toc136801732)

[15.1. Постановка задачи 27](#_Toc136801733)

[15.2. Решение задачи, код программы 27](#_Toc136801734)

[15.3. Тестирование работы программы с проверкой 30](#_Toc136801735)

# Задание 1

## 1.1. Постановка задачи

Задание № 1. Ввести массив, состоящий из N элементов целого типа. Массив является элементами целого дробной части числа в двоичной системе счисления. Проверить правильность введения элементов двоичного числа, т.е. в массиве должны быть только 0 и 1. Считается, что в массиве ведены только те цифры, что стоят после запятой. Если число введено верно, то перевести его в десятичную систему счисления и поэлементно записать число в массив, так же только цифры после запятой ограничиваясь 6 цифрами после запятой, если число при переводе не конечно). Пример:

|  |  |
| --- | --- |
| В 2-ой | В 10-ой |
| |  |  |  |  | | --- | --- | --- | --- | | 1 | 0 | 0 | 1 | | |  |  |  |  | | --- | --- | --- | --- | | 5 | 6 | 2 | 5 | |
| 0,1001 | 0,5625 |

## 1.2. Решение задачи, код программы

import java.util.\*;  
import static java.lang.Math.\*;  
public class Lab8Tsk1 {  
 public static void main(String[] args) {  
 System.*out*.println("Pls, enter the length of the array");  
 Scanner scan = new Scanner(System.*in*);  
 int N = scan.nextInt();  
 double S = 0; // сумма из 2 в 10  
 int[] binary = new int[N];  
 int[] decimal = new int[N];  
 int M = N;  
 for (int i = 0; i < N; i++) { // заполенеие 2 массива  
 System.*out*.println("Pls, enter the binary digits to fill in the number-array\n" + M);  
 int m = scan.nextInt();  
 while (m != 0 && m != 1) { // проверка польз  
 System.*out*.println("Pls enter 0 or 1");  
 m = scan.nextInt();  
 if (m == 0 || m == 1) {  
 break;  
 }  
 }  
 binary[i] = m;  
 M--;  
 }  
 for (int k = N - 1; k >= 0; k--) {  
 S += *pow*(2, (-1 \* (k + 1))) \* binary[k]; // из 2 в 10  
 }  
 S = *round*(S \* 1000000d) / 1000000d; // округление до 6 цифр после  
 S \*= (*pow*(10, N)); // преобразование без ","  
 int Si = (int) S;  
 for (int j = N - 1; j >= 0; j--) {  
 decimal[j] = Si % 10; // массив 10 заполнение  
 Si /= 10;  
 }  
 for (int j = 0; j < N; j++) {  
 System.*out*.print(binary[j]); // 2 вывод  
 }  
 System.*out*.print(' ');  
 if (N <= 6) {  
 for (int j = 0; j < N; j++) {  
 System.*out*.print(decimal[j]);// вывод 10  
 }  
 } else {  
 for (int j = 0; j < 6; j++) {  
 System.*out*.print(decimal[j]);// вывод 10  
 }  
 }  
 }  
}

## 1.3. Тестирование работы программы с проверкой

Таблица 1

Тестирование работы программы и проверка результатов решения

|  |  |
| --- | --- |
| № п.п. | Решение Java |
| 1 |  |

# Задание 2

## 2.1. Постановка задачи

Задание № 2. Даны два массива действительных чисел по N и M элементов. Найти количество элементов, которые встречаются в 1 и 2 массиве одновременно (если есть повторения, то их не учитывать, так как один раз уже совпадение найдено). Подсчитать количество вхождений найденных одинаковых элементов в каждый массив и организовать вывод для каждого элемента в виде одномерного массива, где в 0 ячейке записано число, в 1 ячейке количество его вхождений в 1 массив, во 2 ячейке количество его вхождений во 2 массив.

## 2.2. Решение задачи, код программы

import java.util.\*;  
public class Lab8Tsk2 {  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.*in*);  
 int N = scan.nextInt();  
 int M = scan.nextInt();  
 int[] mas1 = new int[N];  
 int[] mas2 = new int[M];  
 ArrayList<Integer> com3 = new ArrayList<>(); // массив общих эл  
 for (int i = 0; i < N; i++) {  
 mas1[i] = scan.nextInt();  
 }  
 for (int j = 0; j < M; j++) {  
 mas2[j] = scan.nextInt();  
 }  
 for (int i = 0; i < N; i++) {  
 System.*out*.print(mas1[i] + " ");  
 }  
 System.*out*.format("%n");  
 for (int j = 0; j < M; j++) {  
 System.*out*.print(mas2[j] + " ");  
 }  
 System.*out*.format("%n");  
 for (int j = 0; j < N; j++) {  
 int ctmas1 = 0;  
 int ctmas2 = 0;  
 for (int t = 0; t < N; t++) {  
 if (mas1[j] == mas1[t]) {  
 ctmas1++;  
 }  
 }  
 if (!com3.contains(mas1[j])) { // проверка на повтор эл в 1 масе  
 for (int k = 0; k < M; k++) {  
 if (mas1[j] == mas2[k]) {  
 ++ctmas2;  
 if (!com3.contains(mas1[j])) {  
 com3.add(mas1[j]);  
 }  
 }  
 }  
 }  
 if (ctmas2 > 0) {  
 int[] resultForElement = {  
 mas1[j],  
 ctmas1,  
 ctmas2  
 };  
 System.*out*.println(Arrays.*toString*(resultForElement));  
 }  
  
 }  
 }  
 }

## 2.3. Тестирование работы программы с проверкой

Тестирование работы программы и проверка результатов решения

|  |  |
| --- | --- |
| № п.п. | Решение Java |
| 1 |  |

# Задание 3

## 3.1. Постановка задачи

Задание № 3. В одномерном массиве с четным количеством элементов (2N) находятся координаты N точек плоскости. Они располагаются в следующем порядке: x1, y1, х2, y2, x3, y3, и т.д. Пользователем введены координаты центра и радиус окружности 1 (внутренний радиус кольцевой области) , ,  и координаты центра и радиус окружности 2 (внешний радиус кольцевой области) , ,  Кольцевая область определена на координатной плоскости. Определить номера точек, которые лежат внутри кольца.

## 3.2. Решение задачи, код программы

import java.util.Scanner;  
import java.util.Random;  
import java.util.Arrays;  
public class Lab8Tsk3 {  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.*in*);  
 Random rand = new Random();  
  
 // Ввод количества точек N  
 System.*out*.print("Введите N: ");  
 int n = scan.nextInt();  
  
 // Массивы координат x и y размера 2N  
 int[] x = new int[2 \* n];  
 int[] y = new int[2 \* n];  
  
 // Заполняем координаты x и y числами от 1 до 10  
 for (int i = 0; i < 2 \* n; i++) {  
 if (i % 2 == 0)  
 x[i] = rand.nextInt(10) + 1;  
 else  
 y[i / 2] = rand.nextInt(10) + 1;  
 }  
  
 // Выводим массив координат точек  
 System.*out*.println(Arrays.*toString*(x));  
 System.*out*.println(Arrays.*toString*(y));  
  
 // Ввод центров и радиусов кругов  
 System.*out*.print("Введите x1: ");  
 int x1 = scan.nextInt();  
 System.*out*.print("Введите y1: ");  
 int y1 = scan.nextInt();  
 System.*out*.print("Введите радиус R1: ");  
 int R1 = scan.nextInt();  
  
 System.*out*.print("Введите x2: ");  
 int x2 = scan.nextInt();  
 System.*out*.print("Введите y2: ");  
 int y2 = scan.nextInt();  
 System.*out*.print("Введите радиус R2: ");  
 int R2 = scan.nextInt();  
  
 // Метод для подсчета точек в кольцевой области  
  
  
 System.*out*.println("Точек в кольцевой области: " + *countPointsInRing*(x,y,n,x1,y1,R1,x2,y2,R2));  
 }  
 public static int countPointsInRing(int[] x, int[] y, int n,  
 int x1, int y1, int R1,  
 int x2, int y2, int R2) {  
  
 int pointsInRing = 0;  
  
 for (int i = 0; i < n; i++) {  
  
 int xPoint = x[2 \* i];  
 int yPoint = y[i];  
  
 // Проверяем попала ли точка в кольцевую область  
 if (((xPoint - x1) \* (xPoint - x1) + (yPoint - y1) \* (yPoint - y1)) >= R1 \* R1 &&  
 ((xPoint - x2) \* (xPoint - x2) + (yPoint - y2) \* (yPoint - y2)) <= R2 \* R2) {  
  
 pointsInRing++;  
 }  
 }  
  
 return pointsInRing;  
 }  
}

## 3.3. Тестирование работы программы с проверкой

Таблица 3

Тестирование работы программы и проверка результатов решения

|  |  |
| --- | --- |
| № п.п. | Решение Java |
| 1 |  |

# 

# Задание 4

## 4.1. Постановка задачи

Задание № 4. Дан массив n действительных чисел. Требуется упорядочить его по возрастанию. Выполнить сортировку элементов массива с использованием «гномьей» сортировки: сравниваются два соседних элемента ai и аi+1. Если ai ≤ ai+1, то продвигаются на один элемент вперед. Если ai > ai+1, то производится перестановка и сдвигаются на один элемент назад. Выполнить сравнение «гномьей» сортировки с сортировкой «пузырек», рассмотрев количество итераций, а так же промежуточные результаты сортировки.

## 4.2. Решение задачи, код программы

import java.util.Arrays;  
import java.util.Scanner;  
  
public class Lab8Tsk4 {  
 public static void gnomeSort(int[] arr) {  
 int i = 0;  
 while (i < arr.length) {  
 if (i == 0 || arr[i - 1] <= arr[i]) {  
 i++;  
 }  
 else {  
 int temp = arr[i];  
 arr[i] = arr[i - 1];  
 arr[i - 1] = temp;  
 i--;  
 }  
 }  
 }  
  
 public static void bubbleSort(int[] arr) {  
 for (int i = 0; i < arr.length - 1; i++) {  
 for (int j = 0; j < arr.length - i - 1; j++) {  
 if (arr[j] > arr[j + 1]) {  
 int temp = arr[j];  
 arr[j] = arr[j + 1];  
 arr[j + 1] = temp;  
 }  
 }  
 }  
 }  
  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.*in*);  
 System.*out*.println("Введите размер массива: ");  
 int n = scan.nextInt();  
  
 int[] arr = new int[n];  
  
 for (int i = 0; i < n; i++) {  
 System.*out*.print("arr[" + i + "] = ");  
 int value = scan.nextInt();  
 arr[i] = value;  
 }  
  
 int[] arrCopy = Arrays.*copyOf*(arr, arr.length);  
  
 System.*out*.println("Исходный массив: ");  
 System.*out*.println(Arrays.*toString*(arr));  
  
 System.*out*.println("Сортировка гномья: ");  
 *gnomeSort*(arr);  
 System.*out*.println(Arrays.*toString*(arr));  
  
 System.*out*.println("Сортировка пузырьком: ");  
 *bubbleSort*(arrCopy);  
 System.*out*.println(Arrays.*toString*(arrCopy));  
 }  
}

## 4.3. Тестирование работы программы

Таблица 4

Тестирование работы программы и проверка результатов решения

|  |  |
| --- | --- |
| № п.п. | Решение Java |
| 1 |  |

# Задание 5

## 5.1. Постановка задачи

В заданной матрице поменять местами главную и побочную диагонали.

## 5.2. Решение задачи, код программы

public class Lab8Tsk5 {  
  
 public static void swapDiagonals(int[][] matrix) {  
  
 int size = matrix.length;  
  
 for (int i = 0; i < size; i++) {  
 for (int j = 0; j < size; j++) {  
  
 if (i == j) { // Элемент на главной диагонали  
  
 int temp = matrix[i][j];  
 matrix[i][j] = matrix[j][size - 1 - j];  
 matrix[j][size - 1 - j] = temp;  
 }  
  
 if (i + j == size - 1) { // Элемент на побочной диагонали  
  
 int temp = matrix[i][j];  
 matrix[i][j] = matrix[size - 1 - i][size - 1 - j];  
 matrix[size - 1 - i][size - 1 - j] = temp;  
 }  
 }  
 }  
 }  
  
  
  
 public static void main(String[] args) {  
  
 int[][] matrix = {  
 {1, 2, 3},  
 {4, 5, 6},  
 {7, 8, 9}  
 };  
  
 System.*out*.println("Исходная матрица:");  
 *printMatrix*(matrix);  
  
 *swapDiagonals*(matrix);  
  
 System.*out*.println("Матрица после обмена диагоналей:");  
 *printMatrix*(matrix);  
 }  
  
 public static void printMatrix(int[][] matrix) {  
 for (int[] ints : matrix) {  
  
 for (int anInt : ints) {  
  
 System.*out*.print(anInt + " ");  
  
 }  
  
 System.*out*.println();  
  
  
 }  
 }  
}

## 5.3. Тестирование работы программы с проверкой

Таблица 5

Тестирование работы программы и проверка результатов решения

|  |  |
| --- | --- |
| № п.п. | Решение Java |
| 1 |  |

# Задание 6.

## 6.1. Постановка задачи

Задание № 6. Пусть целочисленная матрица размером n × m содержит информацию об учениках некоторого класса из n человек. В 0-м столбце проставлена масса (кг), во 1-ом – рост (см), во 2-ом – успеваемость (средний балл), в 3-ем – численность семьи. Ученик называется  *среднестатистическим* по k-му параметру (*уникальным* по k-му параметру), если на нем достигается минимум (максимум) модуля разности среднего арифметического чисел из k-го столбца и значения k-го параметра этого ученика. Определить количество среднестатистических учеников в классе по массе (минимум (максимум) модуля разности среднего арифметического чисел из k-го столбца и значения k-го параметра этого ученика ±δ (параметр веденный пользователем, бесконечно малое)) и записать их в отдельный двумерный массив и самого уникального ученика по массе и записать его данные в одномерный массив. Создать уникальный стилизованный вывод результатов.

## 6.2. Решение задачи, код программы

import java.util.Scanner;  
public class Lab8Tsk6 {  
 // n - строки  
 static int *n*;  
  
 // m - столбцы  
 static int *m*;  
  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
 System.*out*.println("Введите кол-во строк");  
 int n = sc.nextInt();  
 System.*out*.println("Введите кол-во столбцов");  
 int m = sc.nextInt();  
  
 int[][] matrix = new int[n][m];  
 for (int i = 0; i < n; i++)  
 for (int j = 0; j < m; j++)  
 matrix[i][j] = (int)(Math.*random*() \* 100);  
 System.*out*.println("Введите бескон малое");  
 double delta = sc.nextDouble();  
  
  
 if (delta >= 1) {  
 System.*out*.println("Погрешность должна быть бесконечно малой!");  
 return;  
 }  
  
 int[][] averageStudents = *getAverageStudents*(matrix, 0, delta);  
 int[] uniqueStudent = *getUniqueStudent*(matrix, 0, delta);  
  
 *printResults*(averageStudents, uniqueStudent);  
 }  
  
 public static double getAverage(int[][] matrix, int col) {  
  
 int sum = 0;  
 for (int[] ints : matrix) {  
 sum += ints[col];  
 }  
  
 return (double) sum / matrix.length;  
 }  
  
 public static int[][] getAverageStudents(int[][] matrix, int col, double delta) {  
  
 double average = *getAverage*(matrix, col);  
  
 int[][] result = new int[*n*][*m*];  
 int count = 0;  
 for (int i = 0; i < *n*; i++) {  
 if (Math.*abs*(matrix[i][col] - average) <= delta+ 0.00001) {  
 result[count++] = matrix[i];  
 }  
 if (count == result.length) break;  
 }  
 return result;  
 }  
  
 public static int[] getUniqueStudent(int[][] matrix, int col, double delta) {  
  
 double minDiff = Double.*MAX\_VALUE*;  
 int index = 0;  
  
 for (int i = 0; i < *n*; i++) {  
 double diff = Math.*abs*(matrix[i][col] - *getAverage*(matrix, col));  
 if (diff < minDiff && diff <= delta+ 0.00001){  
 minDiff = diff;  
 index = i;  
 }  
 }  
 return matrix[index];  
 }  
 public static void printResults(int[][] average, int[] unique) {  
 System.*out*.println("Среднестатистики:");  
 for (int[] row : average) {  
  
 // Вывод данных одного студента  
 System.*out*.println(row[0] + " " +  
 row[1] + " " +  
 row[2] + " " +  
 row[3]);  
 }  
  
 System.*out*.println("Уникальный студент:");  
 System.*out*.println(unique[0] + " " +  
 unique[1] + " " +  
 unique[2] + " " +  
 unique[3]);  
 }  
}

## 6.3. Тестирование работы программы с проверкой

Таблица 6

Тестирование работы программы и проверка результатов решения

|  |  |
| --- | --- |
| № п.п. | Решение Java |
| 1 |  |

# Задание 7

## 7.1. Постановка задачи

Задание № 7. Заполнить двумерный массив n×n по образцу. Образец заполнения и пример:

|  |  |
| --- | --- |
| Образец заполнения массива | Пример |
|  |  |

Задание № 8.Заполнить двумерный массив n×n по образцу. Матрица заполняется автоматически для любого n. Создать метод заполнения блока матрицы и отпараметризировав заполнение заполнить при помощи метода всю матрицу. Образец заполнения:

## 7.2. Решение задачи, код программы

import java.util.Scanner;  
public class Lab8Tsk7 {  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.*in*);  
 System.*out*.println("Введите размер мтарицы");  
 int n = scan.nextInt();  
 int[][] spiral = new int[n][n];  
  
 int num = n \* n; // Начальное значение  
 int minRow = 0;  
 int minCol = 0;  
 int maxRow = n - 1;  
 int maxCol = n - 1;  
 if (n % 2 == 0) *even*(minRow, minCol, maxCol, maxRow, num, spiral);  
 else *odd*(minRow, minCol, maxCol, maxRow, num, spiral);  
 *out*(n, spiral);  
  
  
 }  
  
 private static void even(int minRow, int minCol, int maxCol, int maxRow, int num, int[][] spiral) {  
  
 while (num >= 1) {  
 // Левая граница  
 for (int i = minRow; i <= maxRow; i++) {  
 spiral[i][minCol] = num--;  
 }  
 minCol++;///????  
  
 // Нижняя граница  
 for (int i = minCol; i <= maxCol; i++) {  
 spiral[maxRow][i] = num--;  
 }  
 maxRow--;  
  
 // Правая граница  
 for (int i = maxRow; i >= minRow; i--) {  
 spiral[i][maxCol] = num--;  
 }  
 maxCol--;  
  
  
 // Верхняя граница  
 for (int i = maxCol; i >= minCol; i--) {  
 spiral[minRow][i] = num--;  
 }  
 minRow++;  
 }  
  
 }  
 private static void odd(int minRow, int minCol, int maxCol, int maxRow, int num, int[][] spiral) {  
  
 while (num >= 1) {  
 // Правая граница  
 for (int i = maxRow; i >= minRow; i--) {  
 spiral[i][maxCol] = num--;  
 }  
 maxCol--;  
  
 // Верхняя граница  
 for (int i = maxCol; i >= minCol; i--) {  
 spiral[minRow][i] = num--;  
 }  
 minRow++;  
  
 // Левая граница  
 for (int i = minRow; i <= maxRow; i++) {  
 spiral[i][minCol] = num--;  
 }  
 minCol++;  
  
 // Нижняя граница  
 for (int i = minCol; i <= maxCol; i++) {  
 spiral[maxRow][i] = num--;  
 }  
 maxRow--;  
  
 }  
  
 }  
  
 public static void out(int n, int[][] spiral) {  
 for (int i = 0; i < n; i++) {  
  
 for (int j = 0; j < n; j++) {  
  
 if (spiral[i][j] < 10) {  
 System.*out*.print(" " + spiral[i][j] + " ");  
 } else {  
 System.*out*.print(spiral[i][j] + " ");  
 }  
 }  
 System.*out*.println();  
 }  
 }  
 }

Таблица 7

## 7.3. Тестирование работы программы с проверкой

|  |  |
| --- | --- |
| № п.п. | Решение Java |
| 1 |  |

# Задание 8

## 8.1. Постановка задачи

Задание № 8.Заполнить двумерный массив n×n по образцу. Матрица заполняется автоматически для любого n. Создать метод заполнения блока матрицы и отпараметризировав заполнение заполнить при помощи метода всю матрицу. Образец заполнения:



## 8.2. Решение задачи, код программы

import java.util.Scanner;  
public class Lab8Tsk8 {  
 public static void main(String[] args) {  
 Scanner in = new Scanner(System.*in*);  
 System.*out*.println("input n");  
 int n = in.nextInt();  
 int [][] arr = new int[n][n];  
 int k\_max = (n \* n - n)/2 + n;  
 *FillTheBlock*(arr,n/2 - 1,n/2 - 1,0,0,k\_max);  
 *FillTheBlock*(arr,n/2 - 1,n-1,0,n/2,k\_max);  
 *FillTheBlock*(arr,n/2,0,n/2 - 1,n - 1,k\_max);  
 *FillTheBlock*(arr,n/2,n/2,n - 1,n - 1,k\_max);  
 *PrintArr*(arr);  
 }  
 public static void FillTheBlock(int [][] arr,int x, int y, int m, int end\_x,int k\_max){  
 int k = 1;  
 for (int lp = 0,c = x; lp < m; lp++,c++){  
 if (k >= k\_max) break;  
 for (int i = y + lp; i <= m - lp - 1; i++){  
 arr[i][c] = k++;  
 }  
 for (int l = m - 3 \* lp, b = x + 2 \* lp; l >= y + lp; l--, b++){  
 arr[l][b] = k++;  
 }  
 for (int h = end\_x - 2 \* lp - 1,h1 = y + lp; h >= x + 1 + lp ; h--){  
 arr[h1][h] = k++;  
 }  
 }  
 }  
 public static void PrintArr(int[][] arr){  
 for (int[] ints : arr) {  
 for (int anInt : ints) {  
 System.*out*.print(anInt + "\t");  
 }  
 System.*out*.println();  
 }  
 }  
  
}

Таблица 8

## 8.3. Тестирование работы программы с проверкой

|  |  |
| --- | --- |
| № п.п. | Решение Java |
| 1 |  |

# Задание 9

## 9.1. Постановка задачи

Задание № 9. Заполнить двумерный массив 2n×2n по образцу. Матрица заполняется автоматически для любого n. Создать метод заполнения блоков матрицы и отпараметризировав выполнить заполнение всей матрицы при помощи методов. Образец заполнения:



9.2. Решение задачи, код программы

import java.util.Scanner;  
 class Lab8Tsk9 {  
 public static void main(String[] args) {  
 Scanner in = new Scanner(System.*in*);  
 System.*out*.println("input n");  
 int n = in.nextInt();  
 int [][] arr = new int[n][n];  
 int k\_max = (n \* n - n)/2 + n;  
 *FillTheBlock\_1*(arr,0,0,n/2 - 1);  
 *FillTheBlock\_1*(arr,0,n/2,n - 1);  
 *FillTheBlock\_2*(arr,n/2, 0,n - 1, n/2 \* n/2 ,n/2, n);  
 *FillTheBlock\_2*(arr,n/2, n/2,n - 1, n/2 \* n/2 ,n, n);  
 *PrintArr*(arr);  
 }  
 public static void FillTheBlock\_1(int [][] arr, int x, int y, int m){  
 int k = 1;  
 for (int lp = 0,c = x; lp <= m/2; lp++,c += 2){  
 for (int i = y + 2 \* lp; i <= m; i ++){  
 arr[i][c] = k++;  
 }  
 for (int j = m, j\_y = c + 1; j >= y + 2 \* lp + 1; j--){  
 arr[j][j\_y] =k++;  
 }  
 }  
 }  
 public static void FillTheBlock\_2(int [][] arr,int x, int y, int m, int max, int end\_y, int size){ // x = 4 y = 0 m = 7  
 int k = 1;int lp = 0;  
 while (k < max){  
 for (int i = end\_y - lp - 1, j = m - lp; i >= y + lp; i--){  
 arr[i][j] = k++;  
 }  
 for (int i = m - lp - 1, j = y + lp; i >= x + lp; i--){  
 arr[j][i] = k++;  
 }  
 for (int i = y + lp + 1, j = x + lp; i < end\_y - lp; i++){  
 arr[i][j] = k++;  
 }  
 for (int i = x + lp + 1, j = end\_y - lp - 1; i < m - lp; i++){  
 arr[j][i] = k++;  
 }  
 lp++;  
 if ((size/2) % 2 != 0){  
 arr[y + lp][x + lp] = max;  
 }  
 }  
 }  
 public static void PrintArr(int[][] arr){  
 for (int[] ints : arr) {  
 for (int anInt : ints) {  
 System.*out*.print(anInt + "\t");  
 }  
 System.*out*.println();  
 }  
 }  
}

## 9.3. Тестирование работы программы с проверкой

|  |  |
| --- | --- |
| № п.п. | Решение Java |
| 1 |  |

# Задание 10

## 10.1. Постановка задачи

Задание № 10. Дана матрица с N точками в пространстве, 0-я строчка координаты х, 1-я строчка координаты y. Определить для каждой точки попала ли она в закрашенную область, с использованием массива N элементов (1 – точка попала внутрь области, 0 – точка на границе области, -1 – точка вне области). Области взять из лабораторной работы № 2 задание 5 (условия проверки реализовать как метод). Создать метод для стилизованного вывода ответов по попаданию точки в область в соответствии с массивом данных о попадании точки.

## 10.2. Решение задачи, код программы

import java.util.Scanner;  
  
public class Lab8Tsk10 {  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
  
 // Считываем координаты точек  
 int N = sc.nextInt(); // Количество точек  
 int[][] points = new int[N][2]; // Матрица координат Nx2  
 for (int i = 0; i < N; i++) {  
 points[i][0] = sc.nextInt(); // x  
 points[i][1] = sc.nextInt(); // y  
 }  
  
 // Определяем попадание в область  
 int[] result = *checkPoints*(points);  
  
 // Вывод результатов  
 *printResults*(result);  
 }  
  
 public static int[] checkPoints(int[][] points) {  
 int[] result = new int[points.length];  
  
 // Проверка попадания каждой точки в область  
 for (int i = 0; i < points.length; i++) {  
  
 // Условие, какая область  
 if (points[i][0] >= 0 && points[i][1] >= 0) {  
 result[i] = 1; // Попала в область  
 } else {  
 result[i] = -1; // Вне области  
 }  
 }  
  
 return result;  
 }  
  
 public static void printResults(int[] result) {  
 for (int i = 0; i < result.length; i++) {  
  
 if (result[i] == 1) {  
 System.*out*.println("Точка " + (i + 1) + " попала внутрь области");  
 } else {  
 System.*out*.println("Точка " + (i + 1) + " вне области");  
 }  
 }  
 }  
}

Таблица 10

## 10.3. Тестирование работы программы с проверкой

|  |  |
| --- | --- |
| № п.п. | Решение Java |
| 1 |  |

# Задание 11

## 11.1. Постановка задачи

Задание № 11. Дан двумерный массив 2×3N, где N – количество предполагаемых треугольников. Пользователь вводит данные по координатам вершин треугольников в двумерный массив. Создать двумерный массив ответов для N предполагаемых треугольников: 0-я строчка результаты проверки о существовании треугольника с введенными вершинами (1 – существует; 0 – не существует); 1-я строчка классификация треугольников (1 – равносторонний; 2 – прямоугольный; 3 – равнобедренный; 4 – произвольный; 0 – если такого треугольника не существует); 2-я строчка площадь треугольника, если он существует, иначе 0. Создать метод для стилизованного вывода ответов. Проверить есть ли треугольники одинаковой площади, вывести ответ, если такие треугольники есть, то указать их номера.

## 11.2. Решение задачи, код программы

import java.util.\*;  
public class Lab8Tsk11 {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
 System.*out*.println("Введите кол-во треугольников:");  
 int n = scanner.nextInt();  
 double[][] vertices = new double[2][3 \* n];  
 for (int i = 0; i < 3 \* n; i++) {  
 vertices[i % 2][i] = scanner.nextDouble();  
 }  
 for (int i = 0; i < n; i++) {  
 double[] v1 = {vertices[0][3 \* i], vertices[1][3 \* i]};  
 double[] v2 = {vertices[0][3 \* i + 1], vertices[1][3 \* i + 1]};  
 double[] v3 = {vertices[0][3 \* i + 2], vertices[1][3 \* i + 2]};  
 double a = *dist*(v1, v2);  
 double b = *dist*(v2, v3);  
 double c = *dist*(v3, v1);  
 double s = *area*(a, b, c);  
 if (Double.*isNaN*(s)) {  
 System.*out*.printf("Треугольник %d: не существует.\n", i + 1);  
 } else {  
 int type = *classify*(a, b, c);  
 String typeStr = *getTypeString*(type);  
 int sides = *getSides*(a, b, c);  
 String sidesStr = *getSidesString*(sides);  
 System.*out*.printf("Треугольник %d: %s, %s, площадь = %.2f.\n", i + 1, typeStr, sidesStr, s);  
 }  
 }  
 Set<Integer> sameAreaTriangles = new HashSet<>();  
 for (int i = 0; i < n; i++) {  
 for (int j = i + 1; j < n; j++) {  
 if (vertices[0][3 \* i] == vertices[0][3 \* j] && vertices[1][3 \* i] == vertices[1][3 \* j] && vertices[0][3 \* i + 1] == vertices[0][3 \* j + 1] && vertices[1][3 \* i + 1] == vertices[1][3 \* j + 1] && vertices[0][3 \* i + 2] == vertices[0][3 \* j + 2] && vertices[1][3 \* i + 2] == vertices[1][3 \* j + 2]) {  
 sameAreaTriangles.add(i + 1);  
 sameAreaTriangles.add(j + 1);  
 }  
 }  
 }  
 if (!sameAreaTriangles.isEmpty()) {  
 System.*out*.printf("Треугольники с одинаковой площадью: %s\n", sameAreaTriangles);  
 }  
 }  
  
 private static double dist(double[] v1, double[] v2) {  
 return Math.*sqrt*((v1[0] - v2[0]) \* (v1[0] - v2[0]) + (v1[1] - v2[1]) \* (v1[1] - v2[1]));  
 }  
  
 private static double area(double a, double b, double c) {  
 double s = (a + b + c) / 2.0;  
 return Math.*sqrt*(s \* (s - a) \* (s - b) \* (s - c));  
 }  
  
 private static int classify(double a, double b, double c) {  
 if (a == b && b == c) {  
 return 1; // Равносторонний  
 } else if (a == b || b == c || c == a) {  
 return 2; // Равнобедренный  
 } else {  
 return 3; // Разносторонний  
 }  
 }  
  
 private static int getSides(double a, double b, double c) {  
 if (a == b && b == c) {  
 return 3; // Три одинаковых стороны  
 } else if (a == b || b == c || c == a) {  
 return 2; // Две одинаковых стороны  
 } else {  
 return 0; // Нет одинаковых сторон  
 }  
 }  
  
 private static String getTypeString(int type) {  
 switch (type) {  
 case 1:  
 return "равносторонний";  
 case 2:  
 return "равнобедренный";  
 case 3:  
 return "разносторонний";  
 default:  
 return "некорректный тип треугольника";  
 }  
 }  
  
 private static String getSidesString(int sides) {  
 switch (sides) {  
 case 0:  
 return "нет одинаковых сторон";  
 case 2:  
 return "две одинаковых стороны";  
 case 3:  
 return "три одинаковых стороны";  
 default:  
 return "некорректное число одинаковых сторон";  
 }  
 }  
}

Таблица 11

## 11.3. Тестирование работы программы с проверкой

|  |  |
| --- | --- |
| № п.п. | Решение Java |
| 1 |  |

# Задание 12

## 12.1. Постановка задачи

Задание № 12. Дан двумерный квадратный массив A и двумерный квадратный массив обратный к первому A-1. При этом пользователь вводит размерность массива и данные сам и может допустить ошибку при вычислении обратной матрицы или при вводе. Поэтому выполнить проверку соответствуют ли массивы свойству A\* A-1=E, где E – это единичная матрица.

## 12.2. Решение задачи, код программы

import java.util.Scanner;  
public class Lab8Tsk12 {  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
  
 System.*out*.print("Введите размер квадратной матрицы: ");  
 int n = sc.nextInt();  
 int[][] a = new int[n][n];  
  
 System.*out*.println("Введите элементы матрицы A:");  
 for (int i = 0; i < n; i++) {  
 for (int j = 0; j < n; j++) {  
 a[i][j] = sc.nextInt();  
 }  
 }  
  
 int[][] a\_inverse = new int[n][n];  
  
 System.*out*.println("Введите элементы обратной матрицы A-1:");  
 for (int i = 0; i < n; i++) {  
 for (int j = 0; j < n; j++) {  
 a\_inverse[i][j] = sc.nextInt();  
 }  
 }  
  
 // Проверяем выполняется ли условие A \* A-1 = E  
 boolean check = *multiply*(a, a\_inverse);  
  
 if (check) {  
 System.*out*.println("Матрицы A и A-1 корректны");  
 } else {  
 System.*out*.println("Ошибка в обратной матрице или исходных данных");  
 }  
 }  
  
 public static boolean multiply(int[][] a, int[][] inverse) {  
  
 int[][] result = *multiplyMatrix*(a, inverse);  
 int[][] e = *createUnitMatrix*(a.length);  
  
 for (int i = 0; i < a.length; i++) {  
 for (int j = 0; j < a[0].length; j++) {  
 if (result[i][j] != e[i][j]) return false;  
 }  
 }  
 return true;  
 }  
 public static int[][] multiplyMatrix(int[][] a, int[][] b) {  
 int[][] result = new int[a.length][b[0].length];  
  
 for (int i = 0; i < a.length; i++) {  
 for (int j = 0; j < b[0].length; j++) {  
 for (int k = 0; k < a[0].length; k++) {  
 result[i][j] += a[i][k] \* b[k][j];  
 }  
 }  
 }  
 return result;  
 }  
 public static int[][] createUnitMatrix(int n) {  
 int[][] e = new int[n][n];  
  
 for (int i = 0; i < n; i++) {  
 for (int j = 0; j < n; j++) {  
 if(i == j) e[i][j] = 1;  
 else e[i][j] = 0;  
 }  
 }  
 return e;  
 }  
  
}

Таблица 12

## 12.3. Тестирование работы программы с проверкой

|  |  |
| --- | --- |
| № п.п. | Решение Java |
| 1 |  |