Министерство науки и высшего образования РФ

ФГАОУ ВО Пермский национальный исследовательский

политехнический университет

Кафедра «Вычислительная математика, механика и биомеханика»

Отчет по лабораторной работе № 6

тема «Одномерные массивы»

по дисциплине «Информатика»

Вариант 11

Выполнил: студент группу ИСТ-22-1б Золотарев И.О.

Пермь, 2022

**Содержание**

Оглавление

[Задание 1 4](#_Toc124785790)

[1.1. Постановка задачи 4](#_Toc124785791)

[1.2. Решение задачи, код программы 4](#_Toc124785792)

[1.3. Тестирование работы программы 5](#_Toc124785793)

[Задание 2 6](#_Toc124785794)

[2.1. Постановка задачи 6](#_Toc124785795)

[2.2. Решение задачи, код программы 6](#_Toc124785796)

[2.3. Тестирование работы программы 6](#_Toc124785797)

[Задание 3 7](#_Toc124785798)

[3.1. Постановка задачи 7](#_Toc124785799)

[3.2. Решение задачи, код программы 7](#_Toc124785800)

[3.3. Тестирование работы программы с проверкой 7](#_Toc124785801)

[Задание 4 9](#_Toc124785802)

[4.1. Постановка задачи 9](#_Toc124785803)

[4.2. Решение задачи, код программы 9](#_Toc124785804)

[4.3. Тестирование работы программы 10](#_Toc124785805)

[Задание 5 11](#_Toc124785806)

[5.1. Постановка задачи 11](#_Toc124785807)

[5.2. Решение задачи, код программы 11](#_Toc124785808)

[5.3. Тестирование работы программы 12](#_Toc124785809)

[Задание 6 14](#_Toc124785810)

[6.1. Постановка задачи 14](#_Toc124785811)

[6.2. Решение задачи, код программы 14](#_Toc124785812)

[6.3. Тестирование работы программы с проверкой 15](#_Toc124785813)

[Задание 7 16](#_Toc124785814)

[7.1. Постановка задачи 16](#_Toc124785815)

[7.2. Решение задачи, код программы 16](#_Toc124785816)

[7.3. Тестирование работы программы 17](#_Toc124785817)

[Задание 8 18](#_Toc124785818)

[8.1. Постановка задачи 18](#_Toc124785819)

[8.2. Решение задачи, код программы 18](#_Toc124785820)

[8.3. Тестирование работы программы 19](#_Toc124785821)

[9.1. Постановка задачи 20](#_Toc124785822)

[9.2. Решение задачи, код программы 20](#_Toc124785823)

[9.3. Тестирование работы программы с проверкой 21](#_Toc124785824)

[10.1. Постановка задачи 22](#_Toc124785825)

[10.2. Решение задачи, код программы 22](#_Toc124785826)

[10.3. Тестирование работы программы с проверкой 23](#_Toc124785827)

[Задание 11 24](#_Toc124785828)

[11.1. Постановка задачи 24](#_Toc124785829)

[11.2. Решение задачи, код программы 24](#_Toc124785830)

[11.3. Тестирование работы программы с проверкой 25](#_Toc124785831)

[Задание 12 26](#_Toc124785832)

[12.1. Постановка задачи 26](#_Toc124785833)

[12.2. Решение задачи, код программы 26](#_Toc124785834)

[12.3. Тестирование работы программы с проверкой 27](#_Toc124785835)

# Задание 1

## 1.1. Постановка задачи

Задание № 1. Ввести массив, состоящий из N элементов целого типа. Массив является элементами целого дробной части числа в шестеричной системе счисления. Проверить правильность введения элементов числа, т.е. в массиве должны быть только 0, 1, 2, 3, 4 и 5. 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 6-ой | В 10-ой |
| |  |  | | --- | --- | | 1 | 5 | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | 3 | 0 | 5 | 5 | 5 | 5 | |
| 0,15 | 0,305555….. |

## 1.2. Решение задачи, код программы

import java.util.\*;  
import static java.lang.Math.\*;  
import static java.lang.System.*exit*;  
public class oliver2 {  
  
 public static void main(String[] args) {  
 System.*out*.println("input n");  
 Scanner sc = new Scanner(System.*in*);  
 int n = sc.nextInt();  
 int[] arr = new int[n];  
 double result = 0;  
  
 for (int i = 0, j = 1; i < n; i++, j++)  
 {  
 arr [i] = sc.nextInt();  
 if (arr [i] > 6 || arr[i] < 0)  
 {  
 System.*out*.println("wrong number");  
 *exit*(0);  
 }  
 result += arr[i] \* *pow*(6,-j);  
 }  
 System.*out*.println(Arrays.*toString*(arr));  
 int len = Double.*toString*(result).length();  
 result \*= *pow*(10, len - 2);  
 int[] arr2 = new int[len - 1];  
  
 for (int k = len - 2; k > 0; k--)  
 {  
 arr2[k] = (int)(result % 10);  
 result /= 10;  
 }  
 for (int l = 0; l < len -5; l++)  
 {  
 if(arr2[l] == arr2[l + 1] & arr2[l] == arr2[l + 2] & arr2[l] == arr2[l + 3])  
 {  
 int[] newArr1 = Arrays.*copyOfRange*(arr2, 1, l + 4);  
 System.*out*.println(Arrays.*toString*(newArr1));  
 *exit*(0);  
 }  
 }  
 int[] newArr = Arrays.*copyOfRange*(arr2, 1, arr2.length);  
 System.*out*.println(Arrays.*toString*(newArr));  
 }  
}

## 1.3. Тестирование работы программы

Тестирование работы программы и проверка результатов решения

|  |  |  |
| --- | --- | --- |
| № п.п. | Решение Java | Решение онлайн-калькулятора |
| 1 |  |  |
| 2 |  |  |

Сравнение решения задачи с использованием двух прикладных пакетов показала, что решения задачи в Java и онлайн-калькулятора совпадает. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# Задание 2

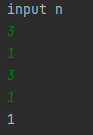
## 2.1. Постановка задачи

Задание № 2. Определить количество локальных максимумов в заданном числовом массиве. (Локальный максимум в числовом массиве – это последовательность трех рядом стоящих чисел, в которой среднее число больше стоящих слева и справа от него).

## 2.2. Решение задачи, код программы

import java.util.Scanner;  
  
public class oliver2 {  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
 System.*out*.println("input n");  
 int n = sc.nextInt();  
 int [] arr = new int [n];  
 for ( int j = 0; j < n; j++) {  
 arr[j] = sc.nextInt();  
 }  
 int k = 0;  
 for (int i = 1; i < n - 1; i++){  
 if (arr[i - 1] < arr[i] & arr[i] > arr[i + 1]) k++;  
 }  
 System.*out*.println(k);  
 }  
}

## 2.3. Тестирование работы программы



# 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 величины полуосей эллипса a и b, эллипс определен на координатной плоскости с центром в центре координат. Определить номера точек, которые лежат внутри эллипса.

## 3.2. Решение задачи, код программы

import java.util.Arrays;  
import java.util.Random;  
import java.util.Scanner;  
import static java.lang.Math.\*;  
  
public class oliver2 {  
 public static void main(String[] args) {  
 Scanner in = new Scanner(System.*in*);  
 int c = 0;  
 System.*out*.println("input n");  
 int n = in.nextInt();  
 double[] list = new double[2 \* n];  
 Random random = new Random();  
 for(int i = 0; i < 2 \* n; i++){  
 list[i] = random.nextInt(10);  
 }  
 System.*out*.println(Arrays.*toString*(list));  
 System.*out*.println("input a");  
 double a = in.nextDouble();  
 System.*out*.println("input b");  
 double b = in.nextDouble();  
 for (int i = 0; i < list.length - 1; i += 2){  
 if(*pow*(list[i], 2)/*pow*(a, 2) + *pow*(list[i + 1], 2)/*pow*(b, 2) <= 1){  
 c += 1;  
 System.*out*.println("dot №" + c + " is in ellipse");  
 }  
 else System.*out*.println("dot №" + c + " isn't in ellipse");  
 }  
 }  
}

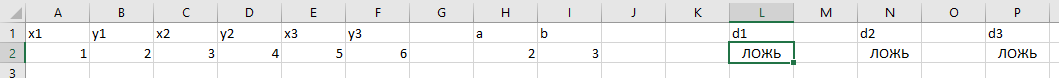
## 3.3. Тестирование работы программы с проверкой

Для проверки задачи в MS Excel создана таблица данных, в которой в ячейку в диапазоне А2:F2 записаны точки, которые необходимо проверить на условие, заданное в задании. В диапазоне H2:I2 записаны величины полуосей. Далее записана проверка: лежит ли точка внутри эллипса. Формулы:

L2 =A2^2/$H$2+B2^2/$I$2<=1

N2 =C2^2/$H$2+D2^2/$I$2<=1

P2 =E2^2/$H$2+F2^2/$I$2<=1



Далее в таблице 1 представлено тестирование работы программы с проверкой решения задачи на языке Java с решением задачи в MS Excel.

Тестирование работы программы и проверка результатов решения

|  |  |  |
| --- | --- | --- |
| № п.п. | Решение Java | Решение MS Excel |
| 1 |  |  |
| 2 |  |  |

Сравнение решения задачи с использованием двух прикладных пакетов показала, что решения задачи в Java и MS Excel совпадает. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# Задание 4

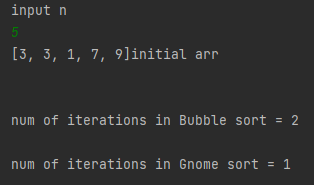
## 4.1. Постановка задачи

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 также промежуточные результаты сортировки.

## 4.2. Решение задачи, код программы

import java.util.Arrays;  
import java.util.Random;  
import java.util.Scanner;  
  
public class oliver2 {  
 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];  
 *getArr*(arr);  
 int [] arr1 = Arrays.*copyOf*(arr,n);  
 System.*out*.println(Arrays.*toString*(arr) + " initial arr");  
 System.*out*.println();  
 int temp;  
 int i = 1;  
 int counter = 0; int counter1 = 0;  
  
 while (i < arr.length){ //Gnomesort  
 if (arr[i - 1] <= arr[i]){  
 i++;  
 }  
 else {  
 temp = arr[i - 1];  
 arr[i - 1] = arr[i];  
 arr[i] = temp;  
 if(i > 1){  
 i--;  
 counter++;  
 if (counter == 10) System.*out*.println(Arrays.*toString*(arr) + "comparison of the Gnome sorting process on the 10th cycle");  
 }  
 }  
 }  
  
 boolean isSorted = false; // Bubblesort  
 while (!isSorted) {  
 isSorted = true;  
 for (int j = 0; j < arr1.length - 1; j++) {  
 if (arr1[j] > arr1[j + 1]) {  
 temp = arr1[j];  
 arr1[j] = arr1[j + 1];  
 arr1[j + 1] = temp;  
 isSorted = false;  
 counter1++;  
 if (counter1 == 10) System.*out*.println(Arrays.*toString*(arr1) + "comparison of the Bubble sorting process on the 10th cycle");  
 }  
 }  
 }  
 System.*out*.println();  
 System.*out*.println("num of iterations in Bubble sort = " + counter1);  
 System.*out*.println();  
 System.*out*.println("num of iterations in Gnome sort = " + counter);  
 }  
 public static void getArr(int [] arr){  
 Random r = new Random();  
 for (int i = 0; i < arr.length; i++) {  
 arr[i] = r.nextInt(10);  
 }  
 }  
}

## 4.3. Тестирование работы программы



# Задание 5

## 5.1. Постановка задачи

Задание № 5. Дана квадратная матрица. Найти в каждой строке наибольший элемент и поменять его местами с элементом главной диагонали.

## 5.2. Решение задачи, код программы

import java.util.Scanner;  
import java.util.Random;  
  
public class oliver2 {  
 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];  
 *getArr*(arr);  
 *PrintOut*(arr);  
 System.*out*.println();  
 *getNewArr*(arr);  
 *PrintOut*(arr);  
 }  
 public static int getMax(int[] ints){  
 int maxIndex = 0;  
 int max = 0;  
 for (int index = 0; index < ints.length; index++) {  
 if (ints[index] > max) {  
 max = ints[index];  
 maxIndex = index;  
 }  
 }  
 return maxIndex;  
 }  
 public static void getArr(int [][] arr){  
 Random r = new Random();  
 for (int i = 0; i < arr.length; i++) {  
 for (int j = 0; j < arr[i].length; j++) {  
 arr[i][j] = r.nextInt(10);  
 }  
 }  
 }  
 public static void PrintOut(int[][] arr){  
 for (int[] ints : arr) {  
 for (int anInt : ints) {  
 System.*out*.print(anInt + "\t");  
 }  
 System.*out*.println();  
 }  
 }  
 public static void getNewArr(int[][] arr){  
 int temp;  
 int maxId;  
 int c = 0;  
 for (int[] ints : arr){  
 maxId = *getMax*(ints);  
 if (arr[c][maxId] > arr[c][c]){  
 temp = arr[c][c];  
 arr[c][c] = arr[c][maxId];  
 arr[c][maxId] = temp;  
 }  
 c++;  
 }  
 }  
}

## 5.3. Тестирование работы программы

Тестирование работы программы и проверка результатов решения

|  |  |
| --- | --- |
| № п.п. | Решение Java |
| 1 |  |
| 2 |  |

# Задание 6

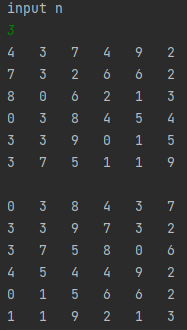
## 6.1. Постановка задачи

Задание № 6. Дана действительная квадратная матрица порядка 2n. Получить новую матрицу, переставляя ее блоки размера n × n по часовой стрелке, начиная с блока в левом верхнем углу.

## 6.2. Решение задачи, код программы

import java.util.Random;  
import java.util.Scanner;  
  
public class oliver2 {  
 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[2 \* n][2 \* n];  
 int i, j;  
 *getArr*(arr);  
 *PrintOut*(arr);  
 System.*out*.println();  
 for (i = 0; i < n; i++)  
 for (j = 0; j < n; j++){  
 *Swap*(arr, i, j, i, n + j);  
 *Swap*(arr, i, j, n + i, n + j);  
 *Swap*(arr, i , j, n + i, j);  
 }  
 *PrintOut*(arr);  
 }  
 public static void getArr(int [][] arr){  
 Random r = new Random();  
 for (int i = 0; i < arr.length; i++) {  
 for (int j = 0; j < arr[i].length; j++) {  
 arr[i][j] = r.nextInt(10);  
 }  
 }  
 }  
 public static void PrintOut(int[][] arr){  
 for (int[] ints : arr) {  
 for (int anInt : ints) {  
 System.*out*.print(anInt + "\t");  
 }  
 System.*out*.println();  
 }  
 }  
 public static void Swap(int [][] arr, int i, int j, int i2, int j2){  
 int temp;  
 temp = arr[i][j];  
 arr[i][j] = arr[i2][j2];  
 arr[i2][j2] = temp;  
 }  
}

## 6.3. Тестирование работы программы с проверкой



# Задание 7

## 7.1. Постановка задачи

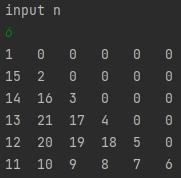
Задание № 7. Заполнить двумерный массив n×n по образцу. Образец заполнения и пример:

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

## 7.2. Решение задачи, код программы

import java.util.Scanner;  
  
public class oliver2 {  
 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 = 1;  
 for(int i = 0; i < n/2; i ++){  
 for (int j = i \* 2, b = i; j < n - i; j++, b++){  
 arr[j][b] = k;  
 k++;  
 }  
 for (int j = n - 2 - i \* 2; j > i; j--){  
 arr[n - 1 - i][j] = k;  
 k++;  
 }  
 for (int j = n - i - 1; j > i \* 2; j--){  
 arr[j][i] = k;  
 k++;  
 }  
 }  
 *PrintOut*(arr);  
 }  
 public static void PrintOut(int[][] arr){  
 for (int[] ints : arr) {  
 for (int anInt : ints) {  
 System.*out*.print(anInt + "\t");  
 }  
 System.*out*.println();  
 }  
 }  
}

## 7.3. Тестирование работы программы



# Задание 8

## 8.1. Постановка задачи

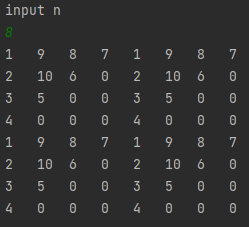
Задание № 8. Заполнить двумерный массив n×n по образцу. Матрица заполняется автоматически для любого n. Создать метод заполнения блока матрицы и отпараметризировав заполнение заполнить при помощи метода всю матрицу. Образец заполнения:



## 8.2. Решение задачи, код программы

import java.util.Scanner;  
  
public class laba8\_8 {  
 public static void main(String[] args) {  
 Scanner in = new Scanner(System.*in*);  
 System.*out*.println("Введите n");  
 int n = in.nextInt();  
 int [][] arr = new int[n][n];  
 int k\_max = (n \* n - n)/2 + n;  
 *Shtuka*(arr,0,0,n/2 - 1, n);  
 *Shtuka*(arr,0,n/2,n/2 - 1, n);  
 *Shtuka*(arr,n/2,0,n/2 - 1, n);  
 *Shtuka*(arr,n/2,n/2,n/2 - 1, n);  
 *PrintArr*(arr);  
 }  
 public static void Shtuka(int [][] arr,int x, int y, int m, int n){  
 int k = 1;  
 for (int i = 0; i < n/2; i++){  
 for (int j = x + 2 \* i, b = y + i;j <= x + m - i;j++){  
 arr[b][j] = k++;  
 }  
 for (int j =x + m - i, b = y + i + 1; b <=y + m - 2 \* i;b++){  
 arr[b][j] = k++;  
 }  
 for (int j = x + m - 1 - i, b = y + m - 1 - 2 \* i; j > x + 2 \* i; j--, b--){  
 arr[b][j] = k++;  
 }  
 }  
 }  
 public static void PrintArr(int[][] arr){  
 for (int i = 0; i<= arr.length-1; i++) {  
 for (int j = 0; j <= arr[0].length-1; j++){  
 System.*out*.print(arr[i][j] + "\t");  
 }  
 System.*out*.println();  
 }  
 }  
}

## 8.3. Тестирование работы программы

Задание 9

## 9.1. Постановка задачи

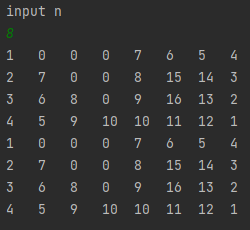
Задание № 9. Заполнить двумерный массив 2n×2n по образцу. Матрица заполняется автоматически для любого n. Создать метод заполнения блоков матрицы и отпараметризировав выполнить заполнение всей матрицы при помощи методов. Образец заполнения:



## 9.2. Решение задачи, код программы

import java.util.Scanner;  
  
public class oliver2 {  
 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];  
 *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

Задание 10

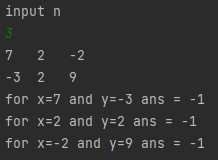
## 10.1. Постановка задачи

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.Random;  
import java.util.Scanner;  
  
public class oliver2 {  
 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[2][n];  
 *getArr*(arr);  
 *PrintArr*(arr);  
 *PrintAns*(n,arr);  
 }  
 public static void getArr( int[][] arr){  
 Random r = new Random();  
 for (int i = 0; i < arr.length; i++) {  
 for (int j = 0; j < arr[i].length; j++) {  
 arr[i][j] = r.nextInt(-10, 10);  
 }  
 }  
 }  
 public static void PrintArr( int[][] arr){  
 for (int[] ints : arr) {  
 for (int anInt : ints) {  
 System.*out*.print(anInt + "\t");  
 }  
 System.*out*.println();  
 }  
 }  
 public static boolean f1(double x, double y){  
 return ((y > -6 \* x - 54) & (y > 0.67 \* x - 0.67) & (y > 3.67 \* x + 14.33)  
 & (y < 2 \* x + 18) & (y < 0.25 \* x + 7.5))&(!((y > 1.4 \* x + 9.8) & (y < 6 \* x + 42 & y >= 0 & x <= 6)));  
 }  
 public static boolean f2(double x, double y){  
 return ((y > -1.67 \* x - 5.33) & (y > 2.5 \* x - 9.5) &  
 (y < 6 \* x + 10) & (y < 0.4 \* x + 4.4) || (y > - 0.33 \* x - 1 & x >= 3 & x <= 6))&(!((y < x + 2) & (y > - 0.83 \* x + 2)));  
 }  
 public static boolean f1edge (double x, double y){  
 return ((y == -6 \* x - 54) || (y == 0.67 \* x - 0.67) || (y == 3.67 \* x + 14.33)  
 || (y == 2 \* x + 18) || (y == 0.25 \* x + 7.5));  
 }  
 public static boolean f2edge(double x, double y){  
 return (y == -1.67 \* x - 5.33) || (y > 2.5 \* x - 9.5) ||  
 (y == 6 \* x + 10) || (y == 0.4 \* x + 4.4) || (y == -0.33 \* x - 1 & x >= 3 & x <= 6);  
 }  
 public static void PrintAns (int n, int[][] arr){  
 for (int i = 0; i < n; i++){  
 if ((*f1*(arr[0][i], arr[1][i])&*f1edge*(arr[0][i], arr[1][i])) || *f2*(arr[0][i], arr[1][i])&*f2edge*(arr[0][i], arr[1][i]))  
 System.*out*.println("for x=" + arr[0][i] + " and y=" + arr[1][i] + " ans = 0 \t");  
 else if (*f1*(arr[0][i], arr[1][i]) || *f2*(arr[0][i], arr[1][i])) {  
 System.*out*.println("for x=" + arr[0][i] + " and y=" + arr[1][i] + " ans = 1 \t");  
 }  
 else System.*out*.println("for x=" + arr[0][i] + " and y=" + arr[1][i] + " ans = -1 \t");  
 }  
 }  
}

## 10.3. Тестирование работы программы с проверкой



Задание 11

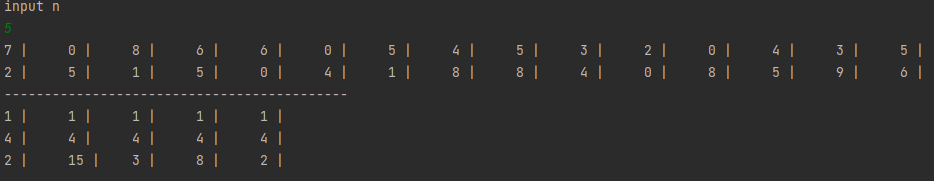
## 11.1. Постановка задачи

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.\*;  
  
import static java.lang.Math.\*;  
  
public class oliver2 {  
 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[2][3 \* n];  
 *getArr*(arr);  
 *PrintOutInt*(arr);  
 System.*out*.println("-------------------------------------------");  
 double s = 0; double ab; double bc; double ca;  
 double [][] ans = new double[3][n];  
 for (int j = 0, i = 0; j < 3 \* n; j += 3, i++){  
 ab = *sqrt*(*pow*(arr[0][j] - arr[0][j + 1], 2) + *pow*(arr[1][j] - arr[1][j + 1], 2));  
 bc = *sqrt*(*pow*(arr[0][j + 1] - arr[0][j + 2], 2) + *pow*(arr[1][j + 1] - arr[1][j + 2], 2));  
 ca = *sqrt*(*pow*(arr[0][j + 2] - arr[0][j], 2) + *pow*(arr[1][j + 2] - arr[1][j], 2));  
 s = 0.5 \* *abs*((arr[0][j] - arr[0][j + 2]) \* (arr[1][j + 1] - arr[1][j + 2]) - (arr[0][j + 1] - arr[0][j + 2]) \* (arr[1][j] - arr[1][j + 2]));  
 if (s == 0){  
 ans[0][i] = 0;  
 ans[1][i] = 0;  
 }  
 else {  
 ans[0][i] = 1;  
 ans[2][i] = s;  
 }  
 if ((ab == bc & bc == ca) & s != 0) ans[1][i] = 1;  
 else if ((ab == bc || bc == ca || ca == ab) & s != 0) ans[1][i] = 3;  
 else if ((*pow*(ab,2) == *pow*(bc,2) + *pow*(ca, 2) || *pow*(bc,2) == *pow*(ab,2) + *pow*(ca, 2) || *pow*(ca,2) == *pow*(bc,2) + *pow*(ab, 2)) & s!= 0) ans[1][i] = 2;  
 else if (s != 0) ans[1][i] = 4;  
  
 }  
 *PrintOut*(ans);  
  
 for (int i = 0; i < n; i++){  
 for (int j = 0; j < n; j++){  
 if (ans[2][i] == ans[2][j] & i != j) System.*out*.println("area of triangle " + (i + 1) + " equals to triangle " + (j + 1));  
 }  
 }  
 }  
 public static void getArr(int [][] arr){  
 Random r = new Random();  
 for (int i = 0; i < arr.length; i++) {  
 for (int j = 0; j < arr[i].length; j++) {  
 arr[i][j] = r.nextInt(10);  
 }  
 }  
 }  
 public static void PrintOut(double[][] arr){  
 for (double[] ints : arr) {  
 for (double anInt : ints) {  
 System.*out*.print((int)anInt + " | \t");  
 }  
 System.*out*.println();  
 }  
 }  
 public static void PrintOutInt(int[][] arr){  
 for (int[] ints : arr) {  
 for (int anInt : ints) {  
 System.*out*.print((int)anInt + " | \t");  
 }  
 System.*out*.println();  
 }  
 }  
}

## 11.3. Тестирование работы программы с проверкой



# 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

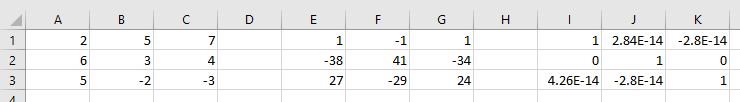
import java.util.Scanner;  
import static java.lang.System.*exit*;  
  
public class oliver2 {  
 public static void main(String[] args){  
 Scanner in = new Scanner(System.*in*);  
 System.*out*.println("input n");  
 int n = in.nextInt();  
 double [][] arr = new double[n][n];  
 double [][] arr\_1 = new double[n][n];  
 System.*out*.println("input elements of array");  
 for (int i = 0; i < arr.length; i++) {  
 for (int j = 0; j < arr[i].length; j++) {  
 arr[i][j] = in.nextDouble();  
 }  
 }  
 System.*out*.println("input elements of array\_1");  
 for (int i = 0; i < arr\_1.length; i++) {  
 for (int j = 0; j < arr\_1[i].length; j++) {  
 arr\_1[i][j] = in.nextDouble();  
 }  
 }  
 int [][] e = new int[n][n];  
 for (int i = 0; i < n; i++){  
 for (int j = 0; j < n; j++){  
 for (int k = 0; k < n; k++){  
 e[i][j] += arr[i][k] \* arr\_1[k][j];  
 }  
 }  
 }  
 for (int i = 0; i < n; i++){  
 for (int j = 0; j < i; j++){  
 if (e[i][i] != 1 || e[i][j] != 0 || e[n - i - 1][n - 1 - j] != 0) {  
 System.*out*.println("Wrong input");  
 *exit*(0);  
 }  
 }  
 }  
 System.*out*.println("Everything is good!");  
 }  
}

## 12.3. Тестирование работы программы с проверкой

Для проверки задачи в MS Excel создана таблица данных в которой в диапазоне А1:C3 изначальная матрица, в диапазоне E1:G3 обратная матрица, в диапазоне I1:K3 проверка умножением матриц результатом должна быть единичная матрица. Формулы:

E1:G3=МОБР(A1:C3)

I1:K3=МУМНОЖ(A1:C3,E1:G3)



Далее в таблице 1 представлено тестирование работы программы с проверкой решения задачи на языке Java с решением задачи в MS Excel.

Тестирование работы программы и проверка результатов решения

|  |  |  |
| --- | --- | --- |
| № п.п. | Решение Java | Решение MS Excel |
| 1 |  |  |
| 2 |  |  |

Сравнение решения задачи с использованием двух прикладных пакетов показала, что решения задачи в Java и MS Excel совпадает. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.