|  |  |  |
| --- | --- | --- |
| Министерство науки и высшего образования РФ  РЫБИНСКИЙ ГОСУДАРСТВЕННЫЙ АВИАЦИОННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ имени П.А. СОЛОВЬЕВА | | |
| Институт информационные технологии и системы управления | | |
| Кафедра радиоэлектронных и телекоммуникационных систем (РТС)  Направление подготовки 13.03.02 Электроэнергетика и электротехника | | |
| КУРСОВАЯ РАБОТА | | |
| по дисциплине  ИНФОРМАтика и икт | | |
| на тему | | |
| ОСНОВЫ ПРоГРАММИРОВАНИЯ на языке С/С++ | | |
|  | | |
| Исполнитель, студент группы ЭЛБ-22 |  | Цикунов Н. В |
| “10” 06 2023 г. | | |
| Преподаватель |  | Кругликов С.Ю. |
| “10” 06 2023 г. | | |
|  | | |
| Рыбинск 2023 | | |

Содержание

[Задание. 3](#_Toc137044538)

[Условие задачи: 3](#_Toc137044539)

[Метод решения задачи. 3](#_Toc137044540)

[Описание переменных. 4](#_Toc137044541)

[Алгоритм решения задачи. 5](#_Toc137044542)

[Алгоритм функции bool IsPIn\_(Point a, Point b, Point c, Point p) 6](#_Toc137044543)

[Алгоритм функции bool trian(Point a, Point b, Point c) 6](#_Toc137044544)

[Алгоритм функции void writeP(Point\* mas,int n) 7](#_Toc137044545)

[Алгоритм функции pair<Point\*, int> Rfile() 7](#_Toc137044546)

[Листинг программы 8](#_Toc137044547)

[Результаты тестирования программы 13](#_Toc137044548)

[Список использованных источников 14](#_Toc137044549)

# Задание.

# Условие задачи:

Найти три треугольника с вершинами в заданном множестве из n точек на плоскости так, чтобы второй треугольник лежал строго внутри первого, а третий внутри второго.

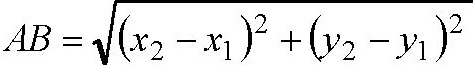
# Метод решения задачи.

Решение данной задачи можно разложить на следующие шаги:

1. Загрузка или ввод массива точек;

2. Перебор всех возможных комбинаций точек;

3. Расчет расстояний между точками. Расстояние между двумя точками рассчитывается по формуле (1):

 (1)

4. Проверка, являются ли три выбранные точки треугольником. Проверка осуществляется исходя из неравенства треугольника: «Из трех отрезков можно составить треугольник тогда, и только тогда, когда сумма длин меньших из них больше длинны большего.

5. Поиск точек которые находятся внутри треугольника. Метод относительности - выбирается ориентация движения по вершинам треугольника, например по часовой стрелке. По данной ориентации проходим все стороны треугольника, рассматривая их как прямые, и рассчитываем по какую сторону от текущей прямой лежит наша точка. Если точка для всех прямых, лежит с правой стороны, то значит точка принадлежит треугольнику, если хоть для какой-то прямой она лежит с левой стороны, то значит условие принадлежности не выполняется.

Реализация - считаются произведения (1,2,3 - вершины треугольника, 0 - точка) (2):

(x1-x0)\*(y2-y1)-(x2-x1)\*(y1-y0)

(x2-x0)\*(y3-y2)-(x3-x2)\*(y2-y0) (2)

(x3-x0)\*(y1-y3)-(x1-x3)\*(y3-y0)

Если они одинакового знака, то точка внутри треугольника, если что-то из этого - ноль, то точка лежит на стороне, иначе точка вне треугольника.

6. Для полученных точек повторяются шаги 3-5. В результате будет определен второй треугольник.

7. Для поиска третьего треугольника так же повторяются шаги 3-4.

# Описание переменных.

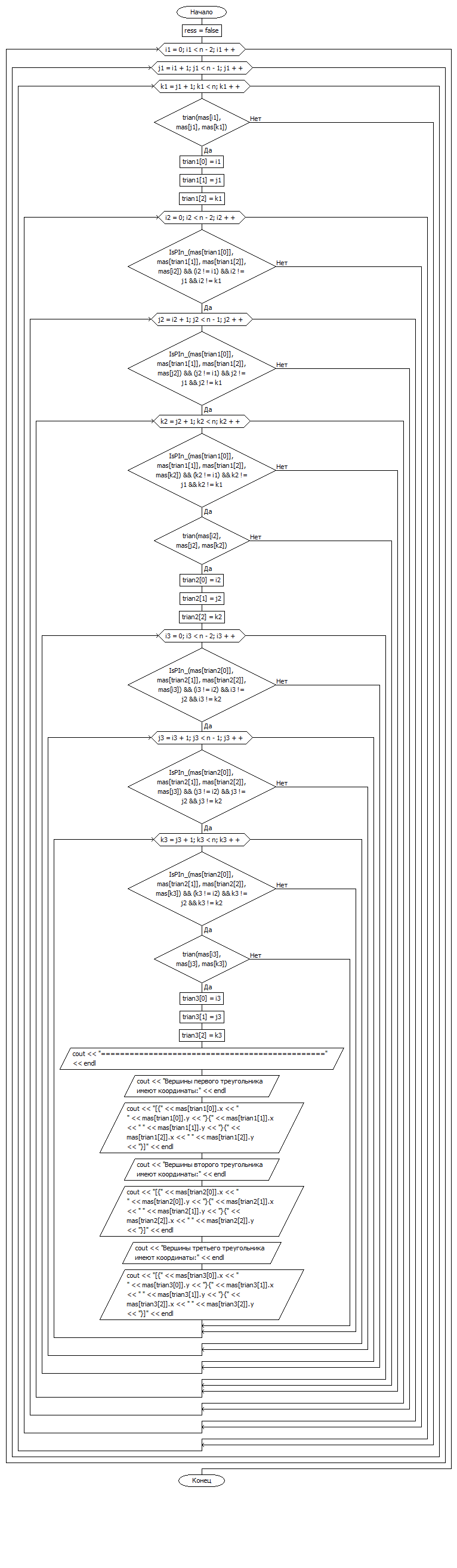
Описание переменных представлено в таблице 1.

Таблица 1. Описание переменных.

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор переменной | Тип данных | | Функция, выполняемая в программе |
| определение | ключевое слово |
| Point | структура | struct | Структура для задания координат точек |
| n | Целое | int | Количество заданных точек |
| masP, mas | Массив структур | Point | Массив точек, введенных в программу |
| I, i1, i2, i3,  j1, j2, j3,  k1, k2, k3 | Целое | int | Счетчик цикла |
| Lab, lac, lbc | вещественный с двойной точностью | double | Расстояние между вершинами |
| Af, bf, cf | вещественный с двойной точностью | double | Расположение точки относительно стороны |
| trian1[3], trian2[3], trian3[3] | Массив целых | Int [3] | Массив индексов точек вершин треугольников |
| bool IsPIn\_(Point a, Point b, Point c, Point p) | логическое значение | bool | Функция определения ли точка треугольнику |
| bool trian(Point a, Point b, Point c) | логическое значение | bool | Функция определения являются ли три точки треугольником |

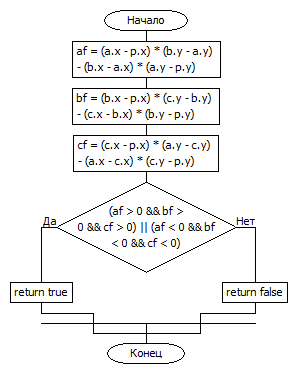
# Алгоритм решения задачи.

На рисунке 1 представлен алгоритм поиска вложенных треугольников.

  
Рисунок 1. Алгоритм поиска вложенных треугольников

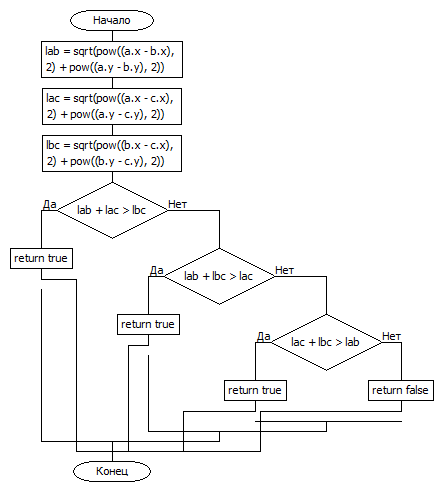
# Алгоритм функции bool IsPIn\_(Point a, Point b, Point c, Point p)

представлен на рисунке 2.

  
Рисунок 2. Алгоритм функции bool IsPIn

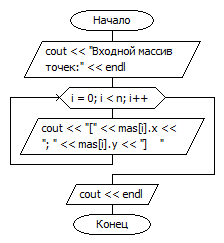
# Алгоритм функции bool trian(Point a, Point b, Point c)

представлен на рисунке 3.

  
Рисунок 3. Алгоритм функции bool trian

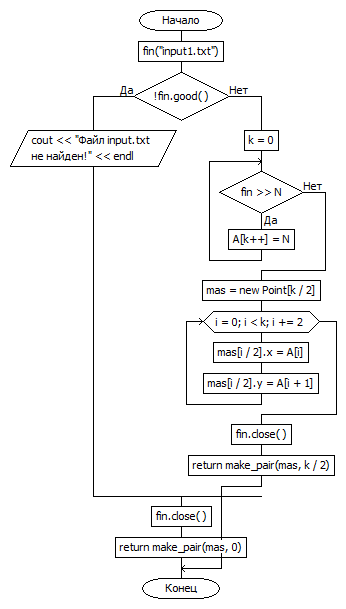
# Алгоритм функции void writeP(Point\* mas,int n)

представлен на рисунке 4.

  
Рисунок 4. Алгоритм функции void writeP

# Алгоритм функции pair<Point\*, int> Rfile()

представлен на рисунке 5.

  
Рисунок 5. Алгоритм функции pair<Point\*, int> Rfile()

# Листинг программы

// Найти три треугольника с вершинами в заданном множестве из n точек на

// плоскости так, чтобы второй треугольник лежал строго внутри первого, а

// третий внутри второго.

#include <iostream>

#include <Windows.h>

#include <fstream>

#include<array>

using namespace std;

//структура точка

struct Point

{

double x, y;

};

//ввод массива с клавиатуры

pair<Point\*, int> Rkey()

{

int n;

cout << endl << "Введите количество точек в массиве: ";

cin >> n;

Point\* mas = new Point[n];

cout << endl;

for (int i = 0; i < n; i++)

{

cout << "Введите x и y точки ["<<i+1<<"]: ";

cin >> mas[i].x;

cin >> mas[i].y;

}

return make\_pair(mas, n);

}

//ввод массива с файла

pair<Point\*, int> Rfile()

{

ifstream fin("input1.txt");

int N, k;

int A[1000];

Point\* mas;

if (!fin.good())

{

cout << "Файл input.txt не найден!" << endl;

}

else

{

k = 0;

while (fin >> N)

{

A[k++] = N;

}

mas = new Point[k / 2];

for (int i = 0; i < k; i += 2)

{

mas[i / 2].x = A[i];

mas[i / 2].y = A[i + 1];

}

fin.close();

return make\_pair(mas, k / 2);

}

fin.close();

return make\_pair(mas, 0);

}

//вывод массива точек

void writeP(Point\* mas,int n)

{

cout << "Входной массив точек:" << endl;

for (int i = 0; i <n; i++)

{

cout << "[" << mas[i].x << "; " << mas[i].y << "] ";

}

cout << endl;

}

//являются ли три точки треугольником

bool trian(Point a, Point b, Point c)

{

double lab, lac, lbc;

lab = sqrt(pow((a.x - b.x), 2) + pow((a.y - b.y), 2));

lac = sqrt(pow((a.x - c.x), 2) + pow((a.y - c.y), 2));

lbc = sqrt(pow((b.x - c.x), 2) + pow((b.y - c.y), 2));

if (lab + lac > lbc)

return true;

else

if (lab + lbc > lac)

return true;

else

if (lac + lbc > lab)

return true;

else

return false;

}

bool IsPIn\_(Point a, Point b, Point c, Point p)

{

int af = (a.x - p.x) \* (b.y - a.y) - (b.x - a.x) \* (a.y - p.y);

int bf = (b.x - p.x) \* (c.y - b.y) - (c.x - b.x) \* (b.y - p.y);

int cf = (c.x - p.x) \* (a.y - c.y) - (a.x - c.x) \* (c.y - p.y);

if ((af > 0 && bf > 0 && cf > 0) || (af < 0 && bf < 0 && cf < 0))

{

return true;

}

else

{

return false;

}

}

//поиск трех вложенных треугольников

void serh(Point\* mas, int n)

{

int trian1[3];

int trian2[3];

int trian3[3];

bool ress = false;

//первый треугольник

for (int i1 = 0; i1 < n - 2; i1++)

{

for (int j1 = i1+1; j1 < n - 1; j1++)

{

for (int k1 = j1+1; k1 < n; k1++)

{

if (trian(mas[i1], mas[j1], mas[k1]))

{

//Первы треугольник найден

trian1[0] = i1; trian1[1] = j1; trian1[2] = k1;

//поиск второго треугольника

for (int i2 = 0; i2 < n-2 ; i2++)

{

if (IsPIn\_(mas[trian1[0]], mas[trian1[1]], mas[trian1[2]], mas[i2])

&& (i2 !=i1 ) && i2!=j1 &&i2!=k1)

{

for (int j2 = i2 + 1; j2 < n - 1; j2++)

{

if (IsPIn\_(mas[trian1[0]], mas[trian1[1]], mas[trian1[2]], mas[j2])

&& (j2 != i1) && j2 != j1 && j2 != k1)

{

for (int k2 = j2 + 1; k2 < n; k2++)

{

if (IsPIn\_(mas[trian1[0]], mas[trian1[1]], mas[trian1[2]], mas[k2])

&& (k2 != i1) && k2 != j1 && k2 != k1)

{

if (trian(mas[i2], mas[j2], mas[k2]))

{

//Второй треугольник найден

trian2[0] = i2; trian2[1] = j2; trian2[2] = k2;

//Поиск третьего треугольника

for (int i3 = 0; i3 < n - 2; i3++)

{

if (IsPIn\_(mas[trian2[0]], mas[trian2[1]], mas[trian2[2]], mas[i3])

&& (i3 != i2) && i3 != j2 && i3 != k2)

{

for (int j3 = i3 + 1; j3 < n - 1; j3++)

{

if (IsPIn\_(mas[trian2[0]], mas[trian2[1]], mas[trian2[2]], mas[j3])

&& (j3 != i2) && j3 != j2 && j3 != k2)

{

for (int k3 = j3 + 1; k3 < n; k3++)

{

if (IsPIn\_(mas[trian2[0]], mas[trian2[1]], mas[trian2[2]], mas[k3])

&& (k3 != i2) && k3 != j2 && k3 != k2)

{

if (trian(mas[i3], mas[j3], mas[k3]))

{

//Второй треугольник найден

trian3[0] = i3; trian3[1] = j3; trian3[2] = k3;

cout << "===============================================" << endl;

cout << "Вершины первого треугольника имеют координаты:" << endl;

cout << "[{" << mas[trian1[0]].x << " " << mas[trian1[0]].y <<

"}{" << mas[trian1[1]].x << " " << mas[trian1[1]].y <<

"}{" << mas[trian1[2]].x << " " << mas[trian1[2]].y << "}]" << endl;

cout << "Вершины второго треугольника имеют координаты:" << endl;

cout << "[{" << mas[trian2[0]].x << " " << mas[trian2[0]].y <<

"}{" << mas[trian2[1]].x << " " << mas[trian2[1]].y <<

"}{" << mas[trian2[2]].x << " " << mas[trian2[2]].y << "}]" << endl;

cout << "Вершины третьего треугольника имеют координаты:" << endl;

cout << "[{" << mas[trian3[0]].x << " " << mas[trian3[0]].y <<

"}{" << mas[trian3[1]].x << " " << mas[trian3[1]].y <<

"}{" << mas[trian3[2]].x << " " << mas[trian3[2]].y << "}]" << endl;

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

Point\* masP;

int n;

cout << "Задание:" << endl;

cout << "Найти три треугольника с вершинами в заданном множестве из n точек на" << endl;

cout << "плоскости так, чтобы второй треугольник лежал строго внутри первого, а" << endl;

cout << "третий внутри второго." << endl << endl;

cout << "Выберете способ загрузки масива точек:" << endl;

cout << " 1. Ввод из файла." << endl;

cout << " 2. Ввод с клавиатуры." << endl;

int choise;

cin >> choise;

switch (choise)

{

case 1:

{

pair< Point\*, int > p = Rfile();

masP = p.first;

n = p.second;

break;

}

case 2:

{

pair< Point\*, int > p = Rkey();

masP = p.first;

n = p.second;

break;

}

default:

masP = NULL;

break;

}

system("cls");

cout << "Задание:" << endl;

cout << "Найти три треугольника с вершинами в заданном множестве из n точек на" << endl;

cout << "плоскости так, чтобы второй треугольник лежал строго внутри первого, а" << endl;

cout << "третий внутри второго." << endl << endl;

writeP(masP,n);

serh(masP, n);

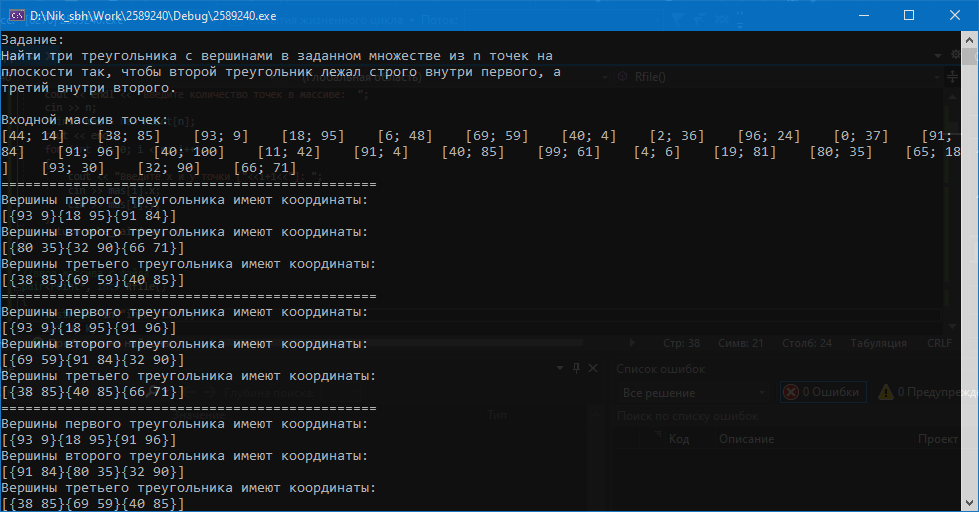
system("pause");

return 0;

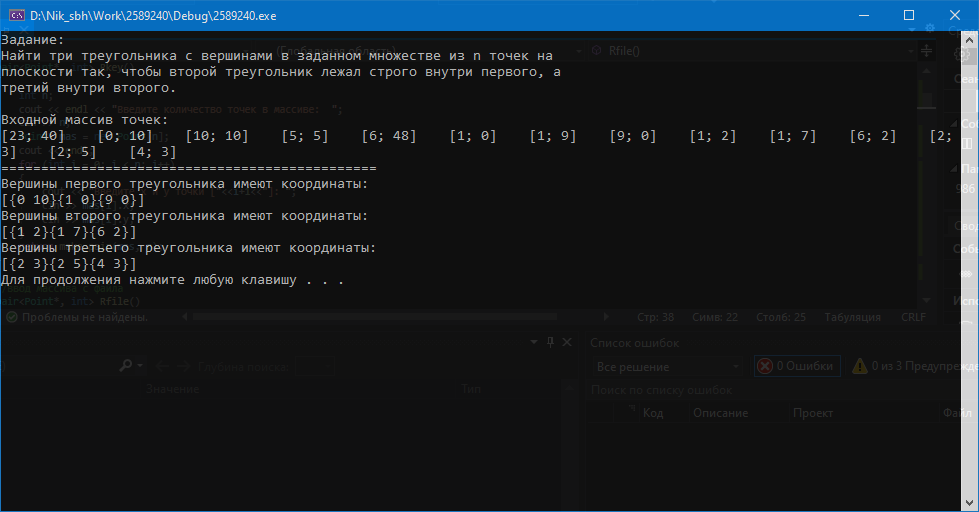
}

# Результаты тестирования программы

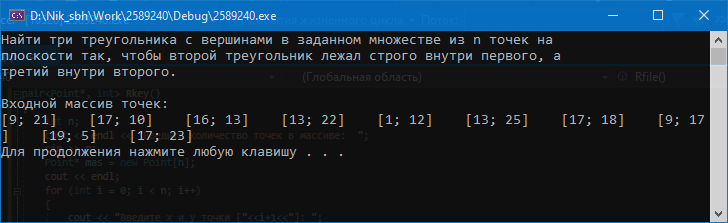
Тест 1



Тест 2



Тест 3



# Список использованных источников

1.Информатика: Лабораторный практикум/ Сост. С.Ю. Кругликов, А.Н. Лебедев. – Рыбинск: РГАТА, 2008. – Ч.1., Ч2 .

2. ГОСТ 19.002 – 80. ГОСТ 19.003 – 80. Единая система программной документации (ЕСПД). Схемы алгоритмов и программ. Правила выполнения. Обозначения условные графические.

3. С/С++. Программирование на языке высокого уровня / Т. А. Павловская. – СПб.: Питер, 2002. – 464 с.: ил.