ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»  
(РУТ (МИИТ))

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

ОТЧЁТ  
О ПРАКТИЧЕСКОЙ РАБОТЕ №2-1

По дисциплине «Процедурное программирование»

Выполнил: ст. гр. ТКИ – 112

Потапов А.К.

Проверил: к.т.н., доц.

Васильева М.А.

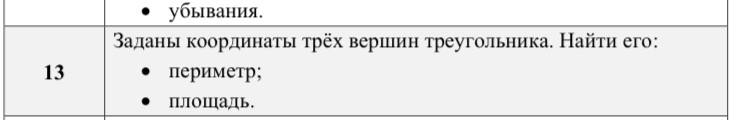
Москва 2021

**Содержание**

Задание 2-1……………………………………. 2

**Задание 2-1**

1. **Формулировка задачи**

Создать консольное приложение для решения задачи, представленной в таблице, с использованием перечислимого типа. Выбор формулы вычисления зависит от пользователя. Данные для решения задачи так же вводит пользователь. Ввод необходимо проверять на правильность. Все результаты вывести на экран. Дополнить свой отчёт блок-схемой алгоритма.  


1. **Блок-схема алгоритма**

Блок-схемы алгоритмов функций представлены на рисунках (Рисунок 1, Рисунок 2, Рисунок 3).

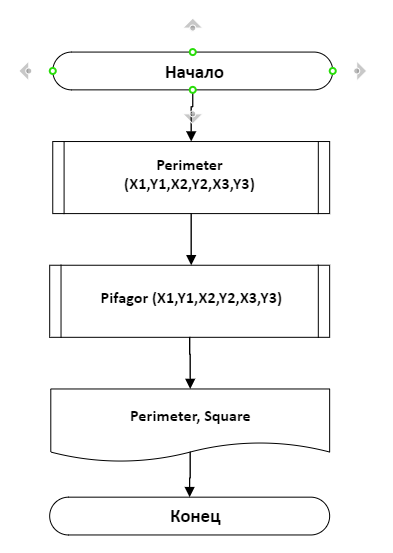
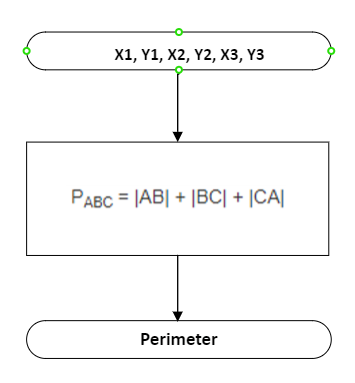
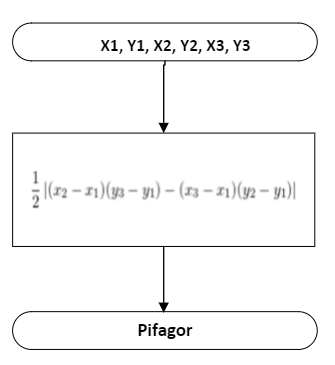
****

Рисунок 1 – Блок-схема функции main()



**Рисунок 2 – Блок-схема функции Perimeter**



**Рисунок 3 – Блок-схема функции Pifagor**

1. **Решение задачи на языке программирования C++**

#define \_USE\_MATH\_DEFINES // for C++

#include <cmath>

#include <iomanip>

#include <iostream>

using namespace std;

/\*\*

\* \brief Рассчитывает длину стороны по формуле Пифагора

\* \param x1 первая координата точки

\* \param y1 вторая координата точки

\* \param x2 первая координата точки

\* \param y2 вторая координата точки

\* \param x3 первая координата точки

\* \param y3 вторая координата точки

\* \return Длину стороны в случае успеха

\*/

double Pifagor(double x1, double x2, double y1, double y2);

/\*\*

\* \brief Рассчитывает периметр треугольника

\* \param x1 первая координата точки

\* \param y1 вторая координата точки

\* \param x2 первая координата точки

\* \param y2 вторая координата точки

\* \param x3 первая координата точки

\* \param y3 вторая координата точки

\* \param Perimeter периметр треугольника

\* \return периметр

\*/

double Perimeter(double x1, double x2, double x3, double y1, double y2, double y3);

/\*\*

\* \brief Вычисляет площадь треугольника

\* \param x1 первая координата точки

\* \param y1 вторая координата точки

\* \param x2 первая координата точки

\* \param y2 вторая координата точки

\* \param x3 первая координата точки

\* \param y3 вторая координата точки

\* \param Square площадь треугольника

\* \return Square площадь треугольника

\*/

double Square(double x1, double x2, double x3, double y1, double y2, double y3);

/\*\*

\* \brief Пользовательский ввод

\* \param Perimeter Расчет периметра прямоугольника.

\* \param Расчет Square площади треугольника.

\*/

enum class userInput

{

Perimeter = 1,

Square

};

/\*\*

\* \brief Точка входа в программу

\* \return 0 в случае успеха

\*/

int main()

{

setlocale(LC\_ALL, "Russian");

cout << "\nВведите координаты вершин треугольника a(x1, y1), b(x2, y2), c(x3,y3)";

double x1, y1;

cout << "\nВведите координаты вершины a, x1="; cin >> x1; cout << " y1="; cin >> y1;

double x2, y2;

cout << "\nВведите координаты вершины b, x2="; cin >> x2; cout << " y2="; cin >> y2;

double x3, y3;

cout << "\nВведите координаты вершины c, x3="; cin >> x3; cout << " y3="; cin >> y3;

cout << "\n Периметр треуголника с заданными координатам=" << Perimeter(x1, x2, x3, y1, y2, y3) << endl;

cout << "\n Площадь треуголника с заданными координатам=" << Square(x1, x2, x3, y1, y2, y3) << endl;

return 0;

}

double Perimeter(double x1, double x2, double x3, double y1, double y2, double y3) {

double a = Pifagor(x1, x2, y1, y2);

double b = Pifagor(x2, x3, y2, y3);

double c = Pifagor(x1, x3, y1, y3);

return (a + b + c);

};

double Square(double x1, double x2, double x3, double y1, double y2, double y3) {

double a = Pifagor(x1, x2, y1, y2);

double b = Pifagor(x2, x3, y2, y3);

double c = Pifagor(x1, x3, y1, y3);

double p = 0.5 \* Perimeter(x1, x2, x3, y1, y2, y3);

return (sqrt(p \* (p - a) \* (p - b) \* (p - c)));

};

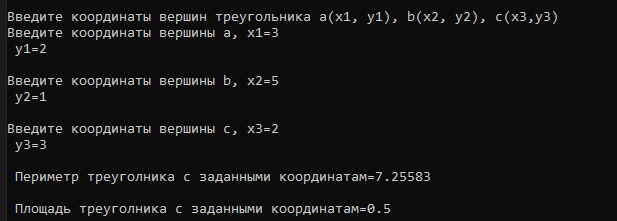
double Pifagor(double x1, double x2, double y1, double y2) {

return (sqrt(pow(x1 - x2, 2) + pow(y1 - y2, 2)));

}

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

1. **Решение тестовых примеров**



1. **Зачет задания в GitHub**