МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХКОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский технический университет связи и информатики»

Кафедра «Информатика»

Лабораторная работа №5

«Функции VС++ и консольные проекты

VisualStudio»

по дисциплине

«Разветвляющиеся алгоритмические структуры. Программная реализация базовых разветвляющихся структур и типовых алгоритмов»

Выполнил: студент гр. БЭИ2202 Тогузов А. А.

Вариант №26

Проверил: доц. Воробейчиков Л.А.

Москва, 2022 г.

Задание 1

Цель:

1) Изучить вопросы представления логического типа данных и логических выражений, формализации, алгоритмизации и программирования алгоритмов при решении задач, использующих разветвляющиеся структуры.

2) Выбрать вариант задания из таблицы 5.1.1.

3) Провести формализацию задачи:

• нарисовать рисунок, указанный в задании;

• выделить на рисунке заданную область;

• для выделенной области определить и записать логическое выражение с условием, зависящее от двух переменных x и y (где [x,y] координаты точки) и принимающее логическое значение false или true, в зависимости от попадания или непопадания точки с координатами x, y в выделенную область рисунка.

4) Разработать три функциональных алгоритма и соответствующие программные функции для решения поставленной задачи согласно индивидуальному заданию:

• использующие вложенные разветвляющие структуры и сложное логическое выражение;

• использующие вложенные разветвления только с помощью операций отношения, без применения логических операций и сложных логических выражений;

• использующие только сложное логическое выражение.

5) Разработать программные коды:

• функции ввода исходных данных;

• функции вывода результатов;

• главной функции main, которая вызывает описанные выше функции для решения поставленной задачи.

6) Создать консольный проект, содержащий 3 раздельно откомпилированных файла:

• файл, содержащий функцию ввода исходных данных и функцию

вывода результатов;

• файл с тремя разработанными функциями, соответствующими разработанным функциональным алгоритмам;

• файл с главной функцией, которая должна содержать только операторы вызова разработанных функций, причем обмен данными между функциями должен осуществляться через параметры, без использования глобальных переменных.

7) Подготовить тестовые исходные данные для решения задачи.

8) Выполнить проект и получить результаты.

9) Доказать правильность полученных результатов на разработанных тестовых данных.

1. Индивидуальное задание на разработку проекта

Определить, лежит ли заданная точка на одной из сторон треугольника, заданного своими вершинами.

1. Формализация и уточнение задачи

Построим «вручную» треугольник с заданными координатами вершин, Рисунок 1

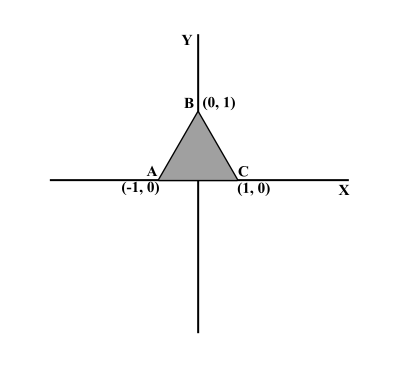


Рисунок 1 – Треугольник, заданный координатами своих вершин

1. Разработка схем алгоритмов

• схема алгоритма, использующего стандартное разветвление и сложное логическое выражение Рисунок 2

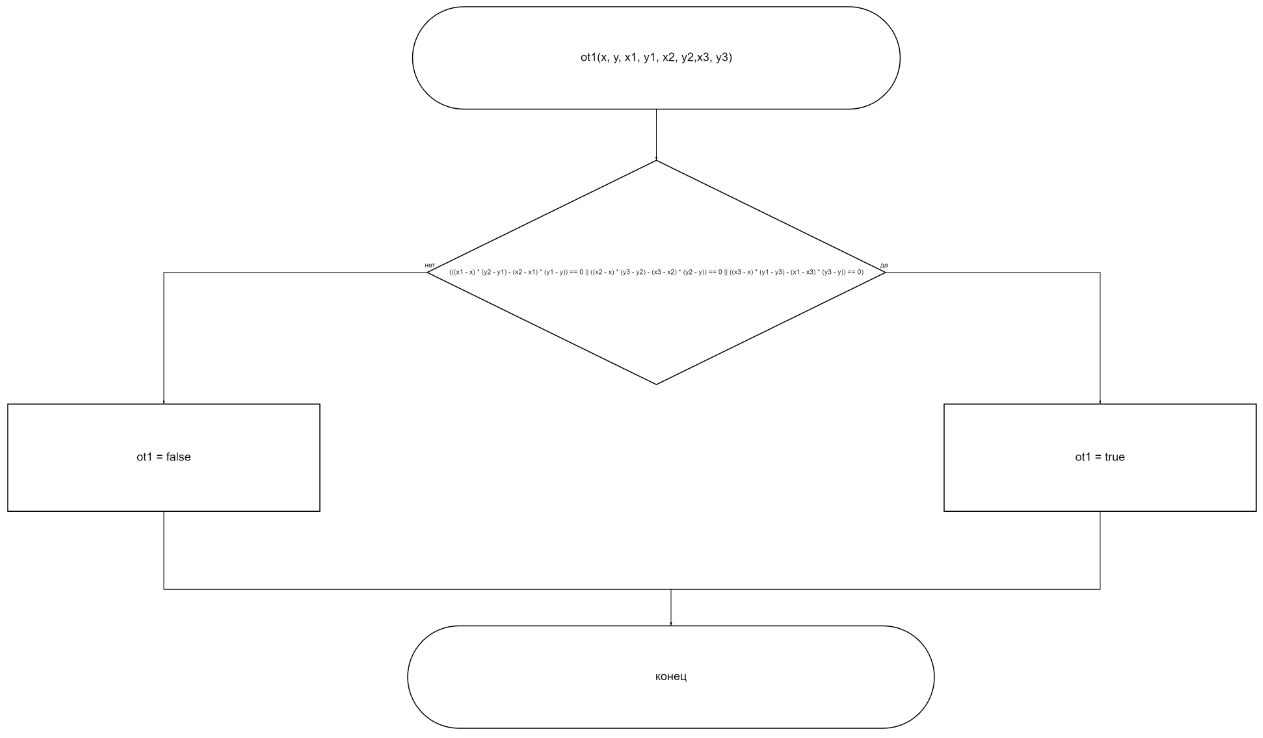


Рисунок 2 – Схема

* схема алгоритма, использующего вложенные разветвления только с помощью операций отношения, без использования логических операций и сложных логических выражений, Рисунок 3

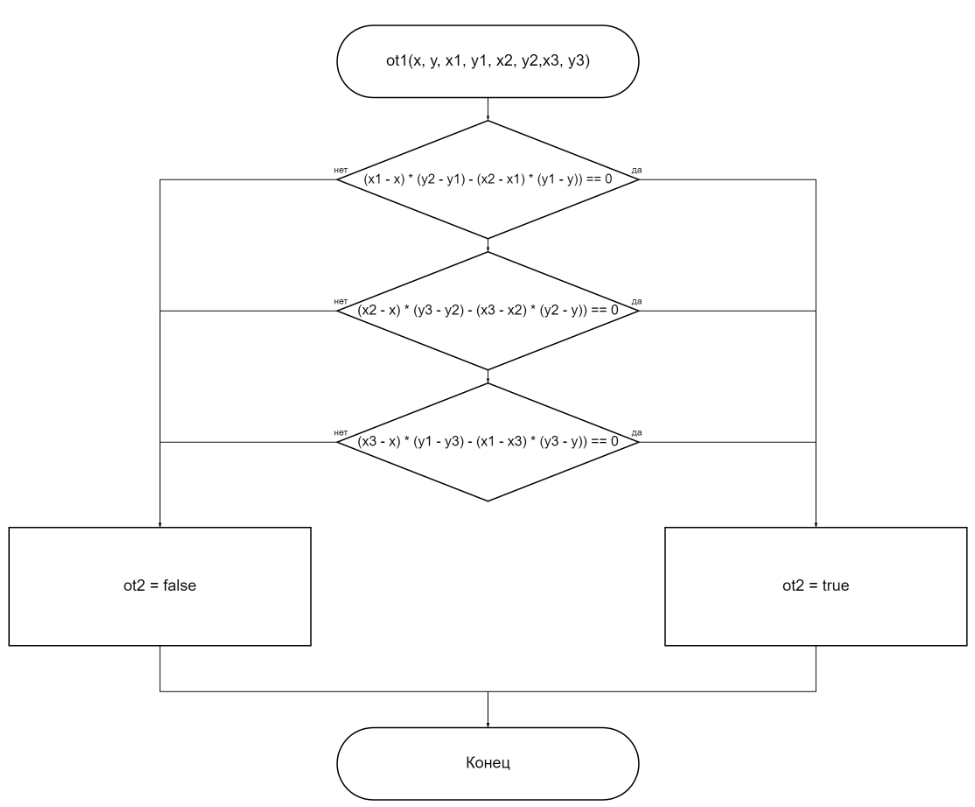


Рисунок 3 – Схема

* схема алгоритма, использующего только сложное логические выражения, Рисунок 4

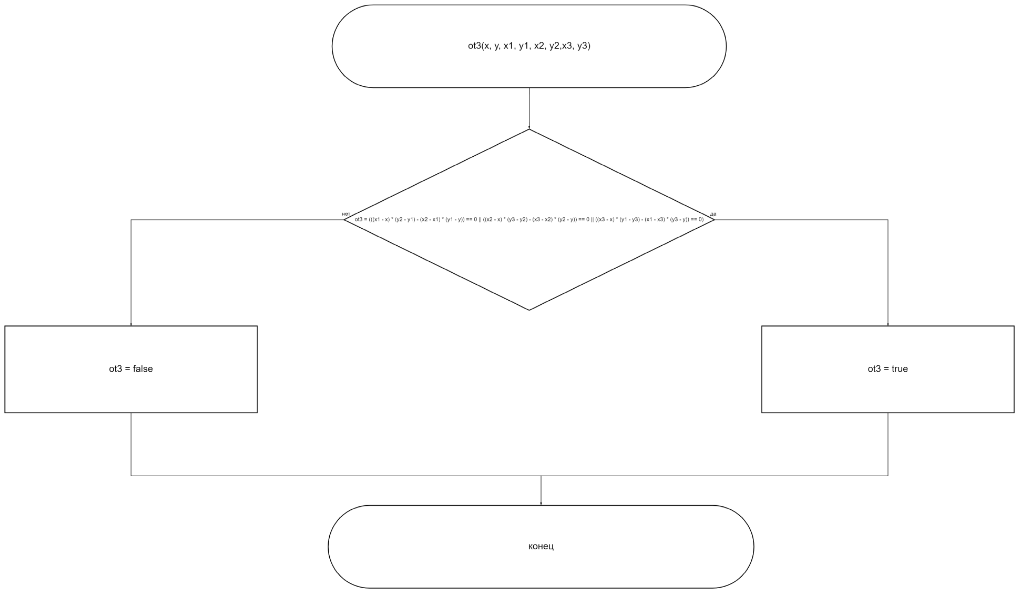


Рисунок 4 – Схема

1. Программный код проекта

#include <iostream>

#include <cmath>

using namespace std;

void GetXY(float&, float&);

void GetABC(float&, float&, float&, float&, float&, float&);

void Put(bool, float, float);

bool ot1(float, float, float, float, float, float, float, float);

bool ot2(float, float, float, float, float, float, float, float);

bool ot3(float, float, float, float, float, float, float, float);

int main()

{

float x, y;

float x1, y1, x2, y2, x3, y3;

bool p1, p2, p3;

GetXY(x, y);

GetABC(x1, y1, x2, y2, x3, y3);

cout << " Решение 1-й функции:" << endl;

p1 = ot1(x, y, x1, y1, x2, y2, x3, y3);

Put(p1, x, y);

cout << " \nРешение 2-й функции:" << endl;

p2 = ot2(x, y, x1, y1, x2, y2, x3, y3);

Put(p2, x, y);

cout << "\n Решение 3-й функции:" << endl;

p3 = ot3(x, y, x1, y1, x2, y2, x3, y3);

Put(p1, x, y);

system("PAUSE");

return 0;

}

bool ot1(float x, float y, float x1, float x2, float x3, float y1, float y2,float y3)

{

if (((x1 - x) \* (y2 - y1) - (x2 - x1) \* (y1 - y)) == 0 || ((x2 - x) \* (y3 - y2) - (x3 - x2) \* (y2 - y)) == 0 || ((x3 - x) \* (y1 - y3) - (x1 - x3) \* (y3 - y)) == 0)

return true;

else

return false;

}

bool ot2(float x, float y, float x1, float x2, float x3, float y1, float y2, float y3)

{

if (((x1 - x) \*(y2 - y1) - (x2 - x1) \* (y1 - y)) != 0)

if (((x2 - x) \*(y3 - y2) - (x3 - x2) \* (y2 - y)) != 0)

if (((x3 - x) \* (y1 - y3) - (x1 - x3) \* (y3 - y)) != 0)

return false;

return true;

}

bool ot3(float x, float y, float x1, float x2, float x3, float y1, float y2, float y3)

{

return (((x1 - x) \* (y2 - y1) - (x2 - x1) \* (y1 - y)) == 0 || ((x2 - x) \* (y3 - y2) - (x3 - x2) \* (y2 - y)) == 0 || ((x3 - x) \* (y1 - y3) - (x1 - x3) \* (y3 - y)) == 0);

}

void GetXY(float& x, float& y)

{

setlocale(LC\_ALL, "rus");

cout << " Введите координаты точки x, y\n ";

cin >> x >> y;

}

void GetABC(float& x1, float& y1, float& x2, float& y2, float& x3, float& y3)

{

setlocale(LC\_ALL, "rus");

cout << " Введите координаты вершин треугольника A B C\n ";

cin >> x1 >> y1 >> x2 >> y2 >> x3 >> y3;

}

void Put(bool b, float x, float y)

{

setlocale(LC\_ALL, "rus");

cout << " Точка с координатами ( " << x << " ," << y << " )" << endl;

if (b)

cout << "попала на сторону треугольника" << endl;

else

cout << "не попала на сторону треугольника" << endl;

}

1. Результаты выполнения проекта

Получены результаты выполнения проекта, приведенные на рисунках 5 - 6, для двух тестовых точек

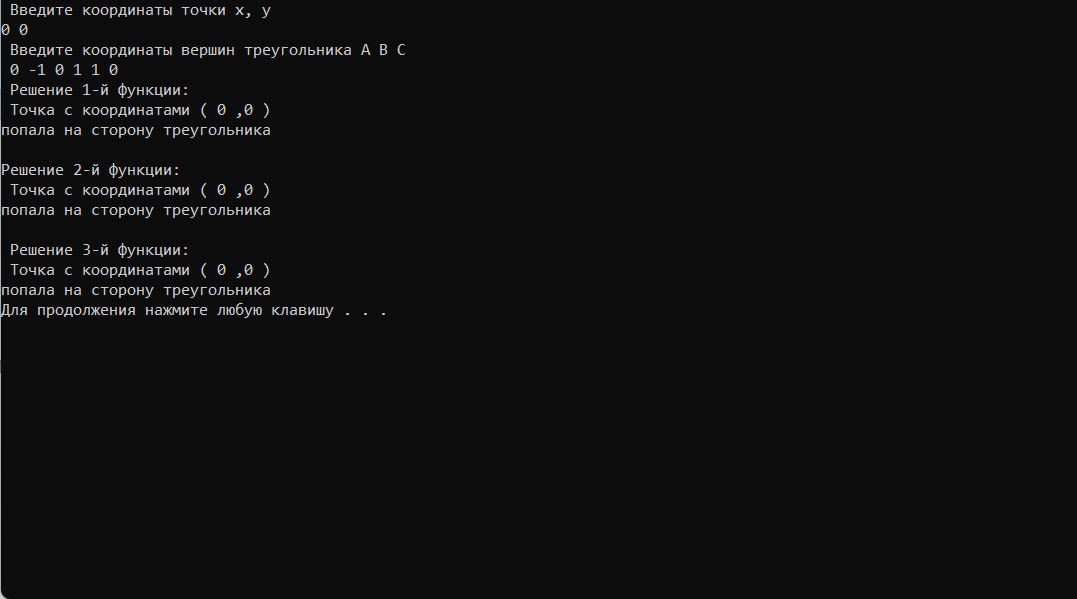


Рисунок 5 – Полученный результат



Рисунок 6 – Полученный результат

1. Доказательство, Рисунок 7

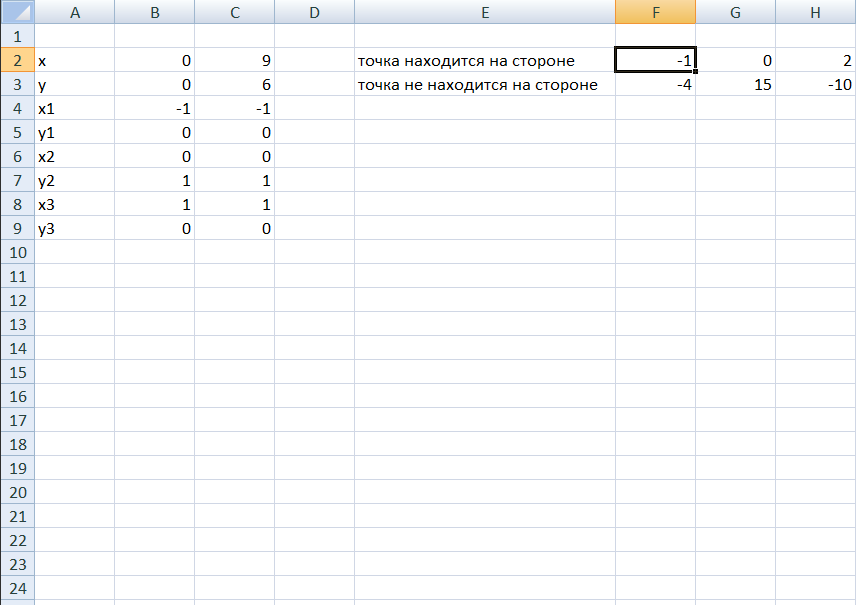


Рисунок 7 – Доказательство

Задание 2

Цель:

1) Изучить вопросы алгоритмизации и программирования алгоритмов при решении задач вычисления сложных выражений с условием и нахождения наименьшего (наибольшего) из нескольких значений.

2) Выбрать вариант задания из таблицы 5.2.1.

3) Провести формализацию задачи.

4) Разработать два функциональных алгоритма и соответствующие программные функции для решения поставленной задачи согласно индивидуальному заданию:

• использующие базовые алгоритмы нахождения наименьшего (наибольшего) из нескольких значений без вспомогательных функций min и max;

• использующие собственные алгоритмы нахождения наименьшего (наибольшего) из двух значений и соответствующие вспомогательные функции min и max.

5) Разработать программные коды:

• функции ввода исходных данных;

• функции вывода результатов;

• главной функции main, которая вызывает описанные выше функции для решения поставленной задачи с выбором способа решения задачи с помощью оператора switch.

6) Создать консольный проект, содержащий 3 раздельно откомпилированных файла:

• файл, содержащий функцию ввода исходных данных и функцию

вывода результатов;

• файл с четырьмя разработанными функциями, соответствующими

разработанным функциональным алгоритмам;

• файл с главной функцией, которая должна содержать только опера-

торы вызова разработанных функций, причем обмен данными между функциями должен осуществляться через параметры, без использования глобальных переменных.

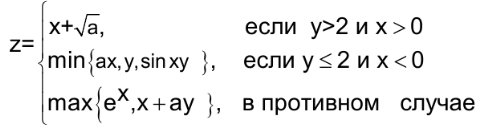
7) Подготовить тестовые исходные данные для решения задачи с проверкой всех ветвей при вычислении сложного выражения с условием.

8) Выполнить проект и получить результаты.

9) Доказать правильность полученных результатов на разработанных тестовых данных.

1. Индивидуальное задание на разработку проекта

Создать программный проект в соответствии с общим заданием для вычисления условного выражения r = f(x, y, a):



1. Формализация и уточнение задачи

Алгоритм решения данной задачи представляет собой комбинацию вычисления сложного выражения с условием и выбора наименьшего (наибольшего) из нескольких значений, используя все виды разветвлений. Можно решить эту задачу двумя способами:

• создать функциональный алгоритм и соответствующую программную функцию, используя вложенные разветвления с базовыми алгоритмами нахождения наибольшего и наименьшего значений, без использования библиотечных или собственных функций max и min;

• создать функциональный алгоритм и соответствующую программную функцию, используя вложенные разветвления и собственные алгоритмы и программные функции нахождения наибольшего и наименьшего из двух значений.

3) Разработка схем алгоритмов

Разработаны два функциональных алгоритма Razv и Razm:

• функциональный алгоритм Razv – вычисление условного выражения, используя вложенные разветвления с базовыми алгоритмами нахождения наибольшего и наименьшего значений без использования библиотечных функций или собственных вспомогательных функций Рисунок 8;

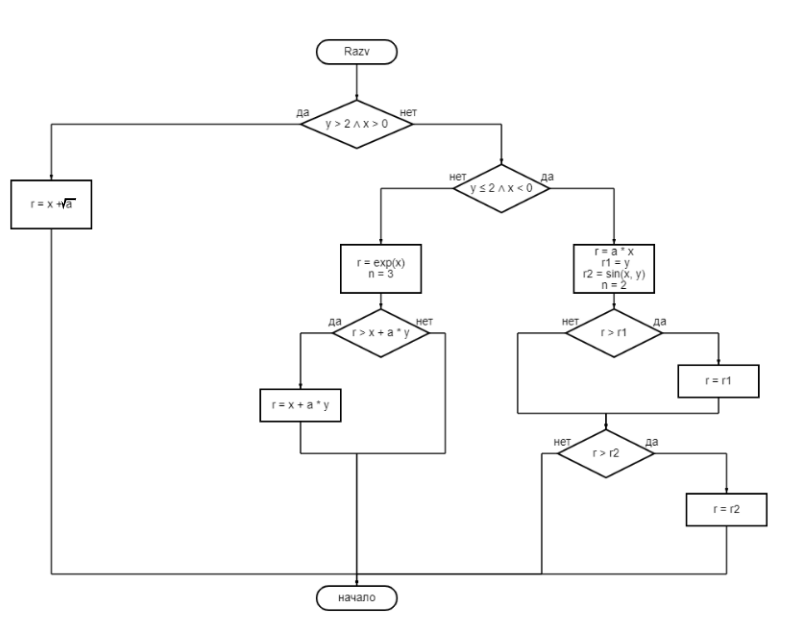


Рисунок 8 – Схема

* функциональный алгоритм Razm – вычисление условного выражения, используя предопределенные алгоритмы вычисления наибольшего и наименьшего значения двух величин – MaxMy и MinMy, Рисунок 9;

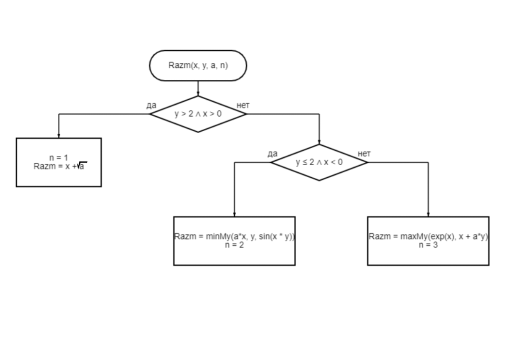


Рисунок 9 – Схема

• схема алгоритма главной функции main представлена на Рисунок 10.

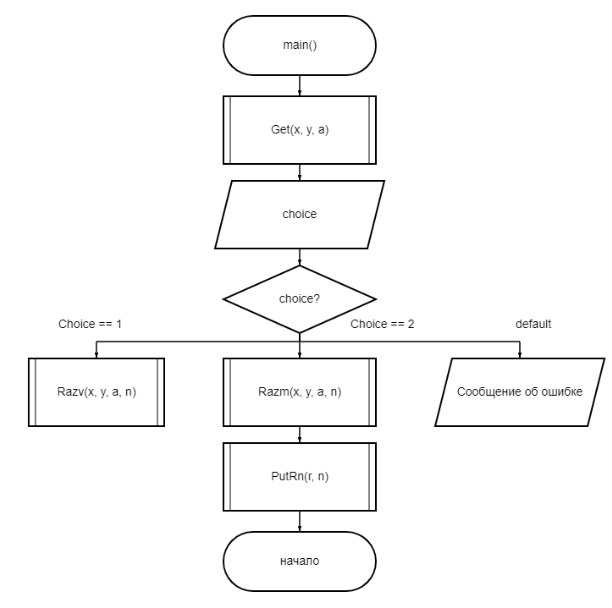


Рисунок 10 – Схема

4) Программный код проекта

Разработан программный код проекта, состоящий из трех файлов исходного кода:

• файла с функциями Razv, Razm, maxMy и minMy, которые решают задачу вычисления условного выражения первым и вторым способами;

• файла с функциями ввода исходных данных и вывода результатов;

• файла с главной функцией main.

#include <iostream>

#include <cmath>

using namespace std;

void GetXYZ(double&, double&, double&);

void PutRN(double, int);

double Razv(double, double, double, int&);

double Razm(double, double, double, int&);

int main()

{

double x, y, a, r;

int n;

GetXYZ(x, y, a);

int choice;

cout << " Каким способом решать задачу?\n";

cout << " 1 - с вложенными разветвлениями ";

cout << " без дополнительных функций max и min \n ";

cout << "2 - со своими функциями minMy и maxMy \n ";

cout << " Что выбираете 1 или 2 ?\n ";

cin >> choice;

switch (choice)

{

case 1:

r = Razv(x, y, a, n);

break;

case 2:

r = Razm(x, y, a, n);

break;

default:

cout << " Вы ввели что-то не то! ";

cout << endl;

system("PAUSE");

return 0;

}

PutRN(r, n);

system("PAUSE");

return 0;

}

void GetXYZ(double& x, double& y, double& a)

{

setlocale(LC\_ALL, "rus");

cout << " Введите x, y, a\n ";

cin >> x >> y >> a;

}

void PutRN(double R, int N)

{

setlocale(LC\_ALL, "rus");

cout << " Ответ R= " << R << endl;

cout << " Номер ветки разветвления " << N << endl;

}

double Razv(double x, double y, double a, int& n)

{

double r;

if (y > 2 && x > 0)

{

r = x + sqrt(a);

n = 1;

}

else

if (y <= 2 && x < 0)

{

r = a \* x;

double r1 = y, r2 = sin(x\*y);

if (r > r2)

r = r2;

if (r > r1)

r = r1;

n = 2;

}

else

{

r = exp(x);

if (r < (x + a \* y))

r = x + a \* y;

n = 3;

}

return r;

}

double maxMy(double, double);

double minMy(double, double, double);

double Razm(double x, double y, double a, int& n)

{

if (y > 2 && x > 0)

{

n = 1;

return x + sqrt(a);

}

else if (y <= 2 && x < 0)

{

n = 2;

return minMy(a\*x, y, sin(x \* y));

}

else

{

n = 3;

return maxMy(exp(x), x + a\*y);

}

}

double maxMy(double x, double y)

{

double f;

if (x > y)

f = x;

else

f = y;

return f;

}

double minMy(double x, double y, double a)

{

double f;

f = x;

if (f > y)

f = y;

if (f > a)

f = y;

return f;

}

1. Результаты выполнения проекта

Результаты выполнения функций Razv и Razm при заданных значениях исходных данных должны быть одинаковы. На Рисунке 11 приведены результаты работы проекта для тестовых данных первой ветви при решении задачи вторым способом.

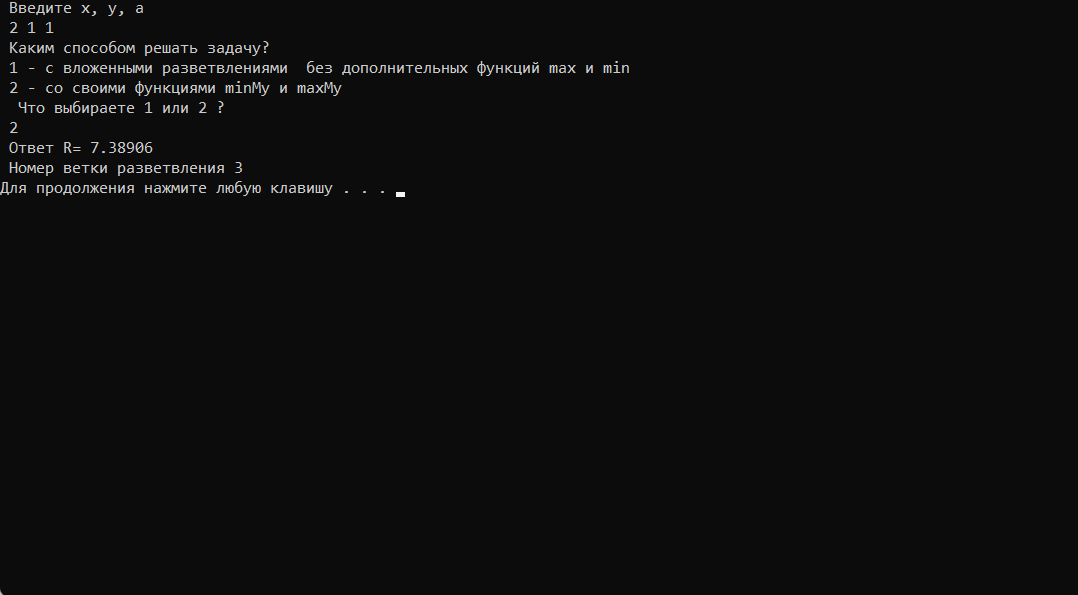


Рисунок 11 – Полученный результат

1. Доказательство правильности результата

Для тестовых исходных данных, проверяющих все ветви разветвления, были получены следующие результаты выполнения проекта:

