Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное учреждение высшего образования «Пермский национальный исследовательский политехнический университет»

ПНИПУ

**Лабораторная работа  
"Функции с переменным числом параметров"**

**Вариант 23**

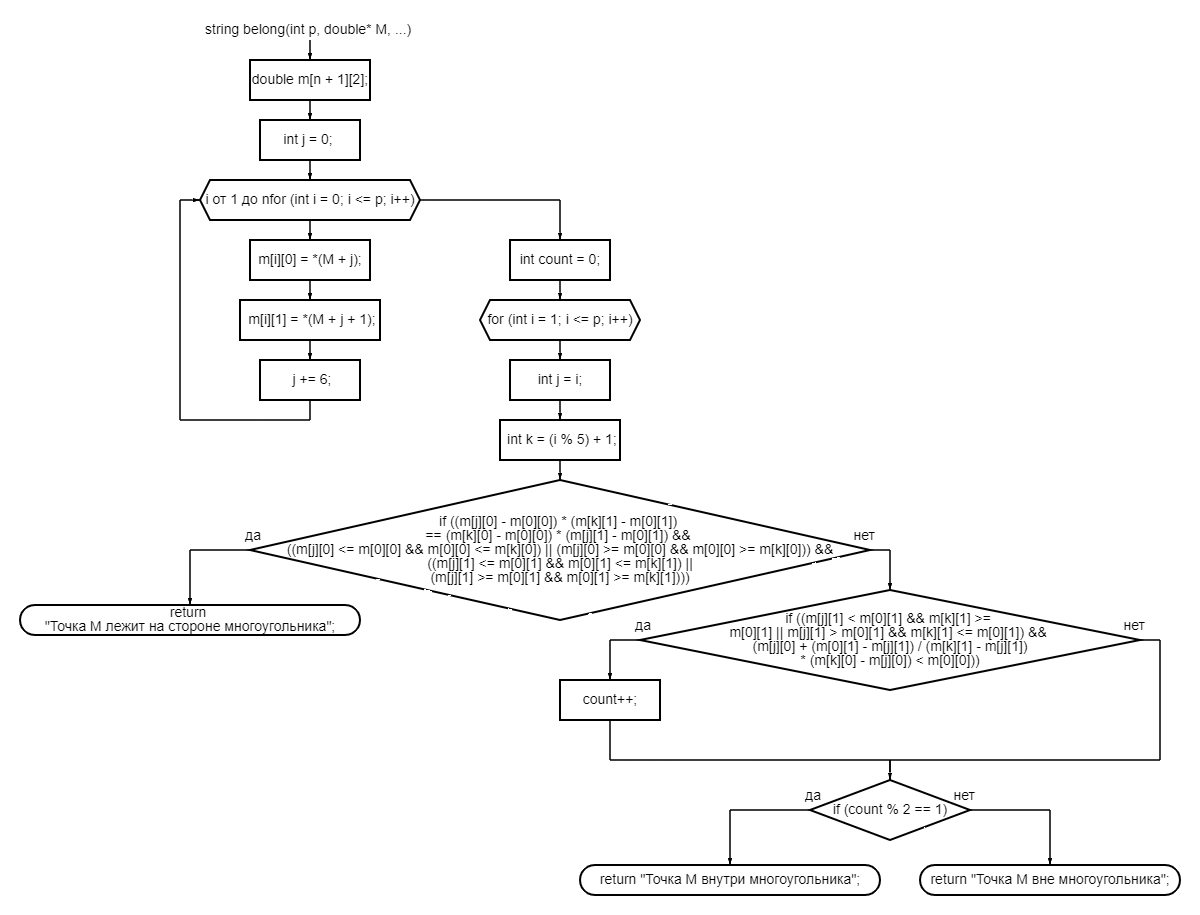
Выполнил:   
студент группы РИС-23-1б   
Карпов Иван Васильевич

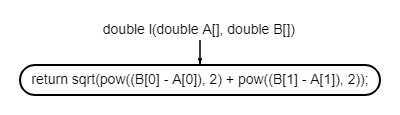
Проверила:   
доцент кафедры ИТАС   
О.А. Полякова

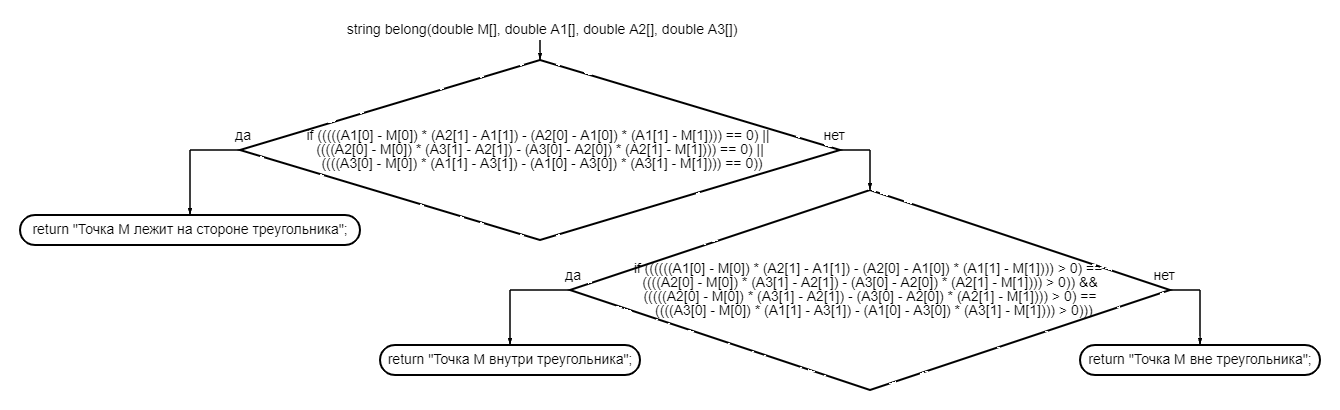
Пермь 2024 г.

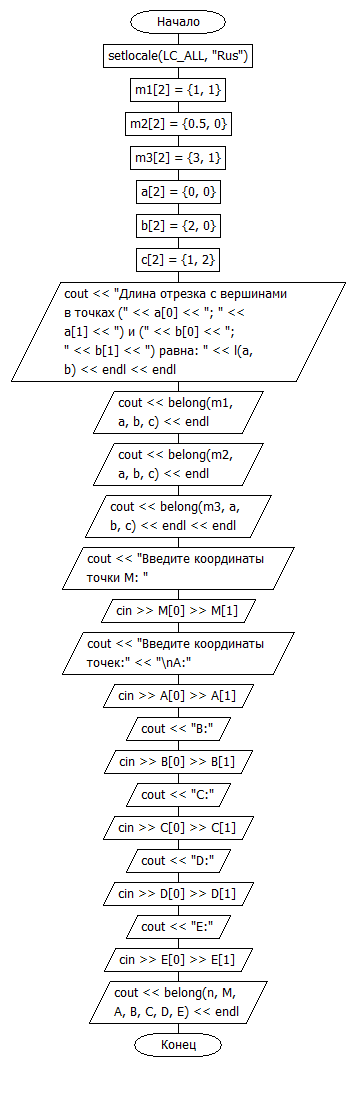
**Условие:**

Написать функцию (или макроопределение), которая находит длину стороны по координатам его точек. Написать функцию belong, которая определяет принадлежит ли точка М с координатами (х, у) треугольнику, заданному координатами вершин. Написать функцию c переменным числом параметров, которая определяет принадлежит ли точка М выпуклому многоугольнику, заданному координатами своих вершин.

**Блок схема:**







**Код:**

#include <iostream>

#include <string>

using namespace std;

// 1 zadanie

double l(double A[], double B[])

{

return sqrt(pow((B[0] - A[0]), 2) + pow((B[1] - A[1]), 2));

}

// 2 zadanie

string belong(double M[], double A1[], double A2[], double A3[])

{

if (((((A1[0] - M[0]) \* (A2[1] - A1[1]) - (A2[0] - A1[0]) \* (A1[1] - M[1]))) == 0) ||

((((A2[0] - M[0]) \* (A3[1] - A2[1]) - (A3[0] - A2[0]) \* (A2[1] - M[1]))) == 0) ||

((((A3[0] - M[0]) \* (A1[1] - A3[1]) - (A1[0] - A3[0]) \* (A3[1] - M[1]))) == 0))

{

return "Точка M лежит на стороне треугольника";

}

else

{

if ((((((A1[0] - M[0]) \* (A2[1] - A1[1]) - (A2[0] - A1[0]) \* (A1[1] - M[1]))) > 0) ==

((((A2[0] - M[0]) \* (A3[1] - A2[1]) - (A3[0] - A2[0]) \* (A2[1] - M[1]))) > 0)) &&

(((((A2[0] - M[0]) \* (A3[1] - A2[1]) - (A3[0] - A2[0]) \* (A2[1] - M[1]))) > 0) ==

((((A3[0] - M[0]) \* (A1[1] - A3[1]) - (A1[0] - A3[0]) \* (A3[1] - M[1]))) > 0)))

{

return "Точка M внутри треугольника";

}

else

{

return "Точка M вне треугольника";

}

}

}

// 3 zadanie

string belong(int p, double\* M, ...);

const int n = 5; // Kolichestvo uglov

void main()

{

setlocale(LC\_ALL, "Rus");

double m1[2] = { 1, 1 };

double m2[2] = { 0.5, 0 };

double m3[2] = { 3, 1 };

double a[2] = { 0, 0 };

double b[2] = { 2, 0 };

double c[2] = { 1, 2 };

cout << "Длина отрезка с вершинами в точках (" << a[0] << "; " << a[1] << ") и (" << b[0] << "; " << b[1] << ") равна: " << l(a, b) << endl << endl;

cout << belong(m1, a, b, c) << endl; // точка внутри треугольника

cout << belong(m2, a, b, c) << endl; // точка на ребре треугольника

cout << belong(m3, a, b, c) << endl << endl; // точка вне треугольника

double M[2];

double A[2];

double B[2];

double C[2];

double D[2];

double E[2];

cout << "Введите координаты точки M: ";

cin >> M[0] >> M[1];

cout << "Введите координаты точек:" << "\nA:";

cin >> A[0] >> A[1];

cout << "B:";

cin >> B[0] >> B[1];

cout << "C:";

cin >> C[0] >> C[1];

cout << "D:";

cin >> D[0] >> D[1];

cout << "E:";

cin >> E[0] >> E[1];

cout << belong(n, M, A, B, C, D, E) << endl;

// Примеры для 3 задания:

//double M1[2] = { 1, 1 };

//double M2[2] = { 4, 1 };

//double M3[2] = { 1.5, 0 };

//double A[2] = { 0, 0 };

//double B[2] = { 2, 0 };

//double C[2] = { 3, 2 };

//double D[2] = { 1, 3 };

//double E[2] = { 0, 2 };

//cout << belong(n, M1, A, B, C, D, E) << endl; // точка внутри многоугольника

//cout << belong(n, M2, A, B, C, D, E) << endl; // точка вне многоугольника

//cout << belong(n, M3, A, B, C, D, E) << endl; // точка на ребре многоугольника

}

// 3 zadanie

string belong(int p, double\* M, ...)

{

double m[n + 1][2];

int j = 0;

// Zanosim vershini tochki i mnogougolnika v massiv

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

{

m[i][0] = \*(M + j);

m[i][1] = \*(M + j + 1);

j += 6;

}

int count = 0;

// Perebiraem storoni mnogougolnika

for (int i = 1; i <= p; i++)

{

int j = i;

int k = (i % 5) + 1;

// Proverka, ne lezit li tocha na storone mnogougolnika

if ((m[j][0] - m[0][0]) \* (m[k][1] - m[0][1]) == (m[k][0] - m[0][0]) \* (m[j][1] - m[0][1]) &&

((m[j][0] <= m[0][0] && m[0][0] <= m[k][0]) || (m[j][0] >= m[0][0] && m[0][0] >= m[k][0])) &&

((m[j][1] <= m[0][1] && m[0][1] <= m[k][1]) || (m[j][1] >= m[0][1] && m[0][1] >= m[k][1])))

{

return "Точка M лежит на стороне многоугольника";

}

// Proverka, peresekaet li luch iz tochki tekushuu storonu po X

if ((m[j][1] < m[0][1] && m[k][1] >= m[0][1] || m[j][1] > m[0][1] && m[k][1] <= m[0][1]) &&

(m[j][0] + (m[0][1] - m[j][1]) / (m[k][1] - m[j][1]) \* (m[k][0] - m[j][0]) < m[0][0]))

{

count++;

}

}

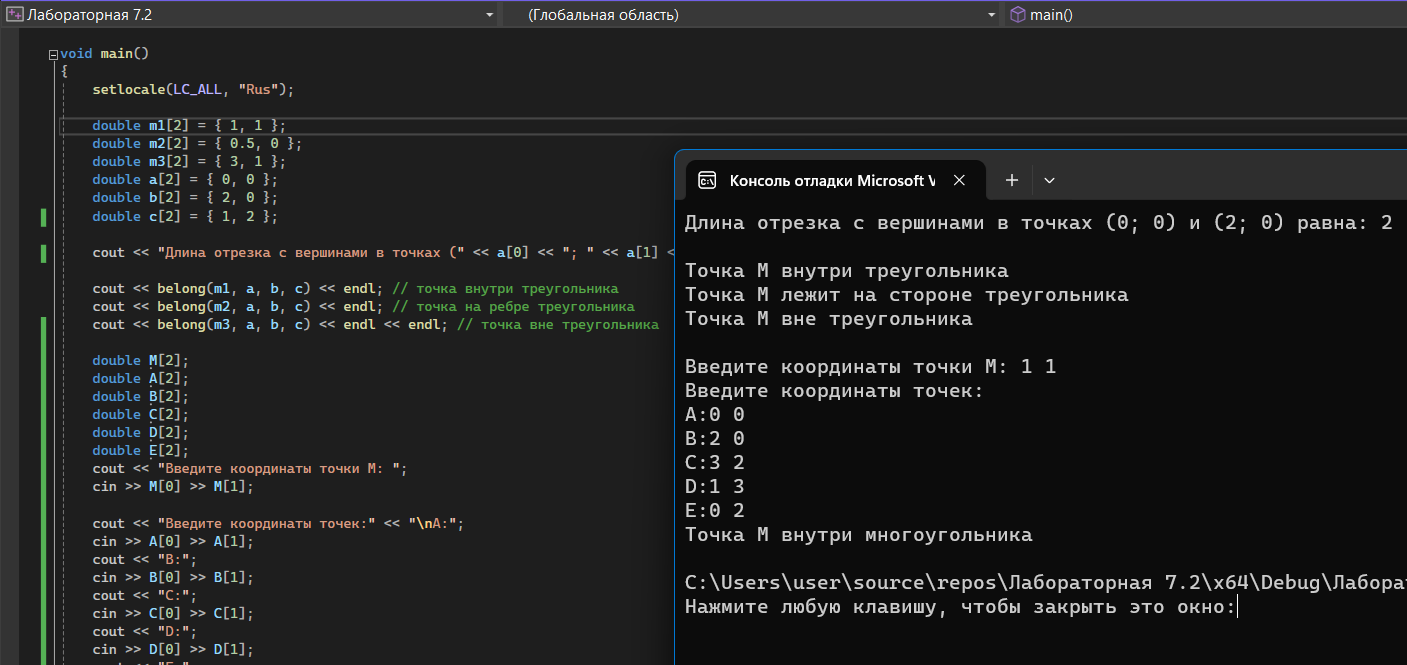
if (count % 2 == 1)

return "Точка M внутри многоугольника";

else

return "Точка M вне многоугольника";

}

**Работа программы:**