Министерство образования и науки Российской Федерации

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

федеральное государственное бюджетное образовательное учреждение

высшего образования

"Санкт-Петербургский государственный технологический институт

(технический университет)"

Кафедра САПР и У

Факультетинформатики и вычислительной техники

Курс 3

Группа 4291

Учебная дисциплина: «Разработка программных систем»

КОНТРОЛЬНАЯ РАБОТА №1

Вариант 3

Студент Борисова Д.С. Личная подпись Расшифровка подписи

Руководитель Личная подпись Расшифровка подписи должность

Г. Санкт-Петербург

2025 г.

**Контрольная работа №1**

Вариант 3

**Задание:**

Дана парабола с вершиной в точке (4,8), пересекающая ось Ох в точках (0,0) и (8,0), прямая линия, пересекающая ось Ох в точке (8,0) и ось Оу в точке (0,4), и прямая линия, проходящая параллельно оси Оу, пересекающая ось Ох точке (6,0). Необходимо разработать алгоритм, который определяет, в какую из имеющихся на плоскости областей попадает точка C заданными координатами. Программа должна выводить сообщение о номере области.

**Графическая модель:**

Построим фигуры и пронумеруем области:

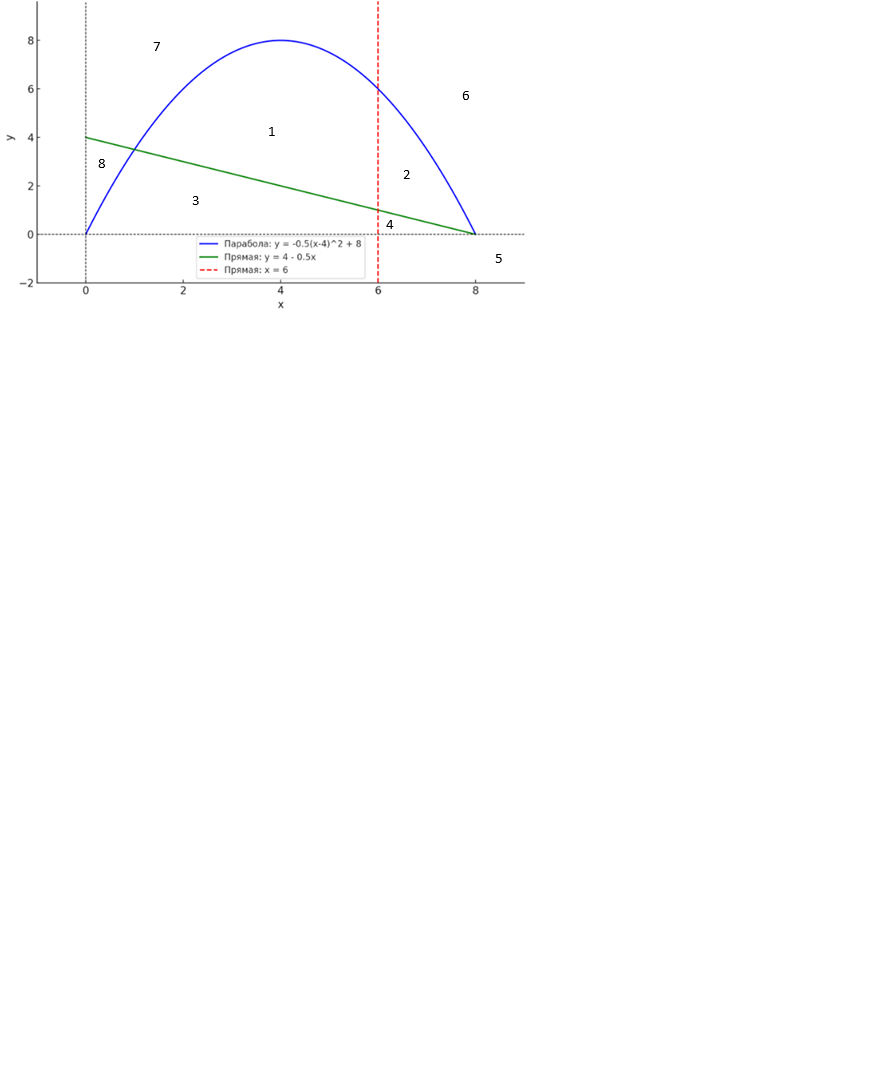


Рисунок 1 – Пример графической модели

**Математическая модель:**

Построим математические уравнения кривых и линий, образующих фигуры на плоскости:

Таблица 1 – Уравнения фигур

|  |  |  |
| --- | --- | --- |
| 1 | Парабола | Y= - 0.5(X-4)2 + 8 |
| 2 | Наклонная прямая | Y= -0.5X + 4 |
| 3 | Прямая | X= 6 |

Cоставим условия попадания точки с координатами (Xz, Yz) в каждую из фигур:

|  |  |  |
| --- | --- | --- |
| 1 | Внутри параболы | Y+ 0.5(X-4)2 – 8 ≤ 0 |
| 2 | Снаружи параболы | Y+ 0.5(X-4)2 – 8 ≥ 0 |
| 3 | Ниже наклонной прямой | Y+0.5X – 4 ≤ 0 |
| 4 | Выше наклонной прямой | Y+0.5X – 4 ≥ 0 |
| 5 | Слева от прямой | X < 6 |
| 6 | Справа от прямой | X > 6 |

Составим условия проверки попадания точки в каждую из областей:

|  |  |  |  |
| --- | --- | --- | --- |
| № | Парабола | Наклонная прямая | Прямая |
| 1 | внутри | выше | слева |
| 2 | внутри | выше | справа |
| 3 | внутри | ниже | слева |
| 4 | внутри | ниже | справа |
| 5 | снаружи | ниже | справа |
| 6 | снаружи | выше | справа |
| 7 | снаружи | выше | слева |
| 8 | снаружи | ниже | слева |

**Тест план:**

Для тестирования программы необходимо выбрать по точке из каждой области:

|  |  |  |  |
| --- | --- | --- | --- |
| Тест | Xz | Yz | Результат |
| 1 | 5 | 6 | область№1 |
| 2 | 7 | 1,5 | область№2 |
| 3 | 1 | 1 | область№3 |
| 4 | 6,5 | 0,5 | область№4 |
| 5 | 10 | -1 | область№5 |
| 6 | 10,5 | 11,1 | область№6 |
| 7 | 1 | 7 | область№7 |
| 8 | -1,5 | 3 | область№8 |
| 12 | ab | 1.0 | Некорректный ввод |
| 13 | 1.0 | ab | Некорректный ввод |

**Декомпозиция задач:**

Исходя из условий задания, в основной задаче Main можно выделить самостоятельную задачу CheckArea – проверка попадания точки в одну из областей на плоскости. Эту задачу можно разбить на три независимые подзадачи:

inParabola – проверка попадания точки внутрь параболы

inLine – проверка попадания точка выше или ниже прямой

inVerticalLine – проверка попадания точка слева или справа от прямой

**inVerticalLine**

**inLine**

**inParabola**

**CheckArea**

**Main**

Рисунок 2 – дерево декомпозиции задач

**Блок-схемы и описание алгоритмов:**

Алгоритм inParabola, представленный на рисунке 3, получает координаты точки, как исходные данные, проверяет условия 1 и 2, и возвращает True, если точка лежит внутри параболы, или False, если точка лежит снаружи параболы.

начало

x, y

Z= y+ 0.5(x-4)2 – 8

Да

Z0

Нет

True

False

конец

Рисунок 3 – Алгоритм inParabola

Алгоритм inLine, показанный на рисунке 4, получает координаты точки, как исходные данные, проверяет условия 3 и 4, и возвращает True, если точка лежит ниже прямой, или False, если точка лежит выше прямой.

начало

x, y

Z= y+0.5x – 4

Да

Z  0

Нет

True

False

конец

Рисунок 4 – Алгоритм inLine

Алгоритм inVerticalLine, показанный на рисунке 5, получает координаты точки, как исходные данные, проверяет условия 5 и 6, и возвращает True, если точка лежит слева от прямой, или False, если точка лежит справа от прямой.

начало

x

z= x - 6

Да

Z

Нет

True

False

конец

Рисунок 5 – Алгоритм inVerticalLine

Алгоритм CheckArea, показанный на рисунке 6, получает координаты точки, как исходные данные, и использует алгоритмы inParabola, inLine, inVerticalLine для проверки попадания точки в каждую из фигур по отдельности и, затем, осуществляет блок последовательных проверок в соответствии с таблицей 3 условий для проверки попадания точки в каждую из областей. Область 8 определяется как исключение, если ни одно из условий не выполнилось.

Начало

X , Y

bParapola = inParabola (x, y)

bLine = inLine (x, y)

bLine2 = inVerticalLine (x, y)

Да

bParapola = true

bLine = true Area 1

bLine2 =true

Нет

Да

bParapola = true Area 2

bLine = true

bLine2 =false

Нет

Да

bParapola = true

bLine = false Area 3

bLine2 = true

Нет

\

Да

bParapola = true

bLine = false Area 4

bLine2 =false

Нет

Да

bParapola = false

bLine = false Area 5

bLine2 =false

Нет

Нет

Да

bParapola = false

bLine = true Area 6

bLine2 =false

Нет

Да

bParapola = false

bLine = true

bLine2 = true

Area 8

Area 7

конец

Рисунок 6 – Алгоритм CheckArea

**Листинг программы:**

MyForm.cpp

#include "MyForm.h"

#include <iostream>

#include <locale.h>

#include <string>

#include < cmath >

using namespace System;

using namespace System::Windows::Forms; //правильный функцианал для главный функции

//[STAThreadAttribute] //создание отдельного потока данный для мэйн

int main(array<String^>^ args) {//стринг, потому что больше функций с отдельной строкой

// ^-очистка памяти массива автоматом

Application::SetCompatibleTextRenderingDefault(false); //корректная обработка текста

Application::EnableVisualStyles(); //подключаем различ визуал стили

Project1::MyForm form; //создаем наш объект

Application::Run(% form); //передача по ссылке на объект форм

}

// Уравнение параболы: y = -0.5 \* (x - 4) ^ 2 + 8

bool inParabola(double x, double y) {

return ((y + 0.5 \* pow((x - 4), 2)-8) <= 0) ? true : false;

}

// Уравнение прямой 1: y = -0.5 \* x + 4

bool inLine(double x, double y) {

return ((y+0.5\*x-4) >= 0) ? true : false;

}

// Уравнение вертикальной прямой x = 6

bool inVerticalLine(double x) {

return (x < 6) ? true : false;

}

// Функция для определения, в какую область попала точка

int CheckArea(double x, double y) {

bool bParabola = inParabola (x,y);

bool bLine = inLine(x, y);

bool bLine2 = inVerticalLine(x);

if (bParabola == true && bLine == true && bLine2 == true) {

return 1;

}

else if (bParabola == true && bLine == true && bLine2 == false) {

return 2;

}

else if (bParabola == true && bLine == false && bLine2 == true) {

return 3;

}

else if (bParabola == true && bLine == false && bLine2 == false) {

return 4;

}

else if (bParabola == false && bLine == false && bLine2 == false) {

return 5;

}

else if (bParabola == false && bLine == true && bLine2 == false) {

return 6;

}

else if (bParabola == false && bLine == true && bLine2 == true) {

return 7;

}

return 8;

}

//функция реализации работы кнопки

System::Void Project1::MyForm::button1\_Click(System::Object^ sender, System::EventArgs^ e) {

if (textBox1->Text == "" || textBox2->Text == "") {

MessageBox::Show("Внимание! Вы укажите все координаты точки");

}

//преобразовываем введеные данные в дабл

double x = Convert::ToDouble(textBox1->Text);

double y = Convert::ToDouble(textBox2->Text);

int result = CheckArea(x, y);

this->label5->Text = "Область " + result.ToString();

}

System::Void Project1::MyForm::textBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e)

{

return System::Void();

}

System::Void Project1::MyForm::textBox2\_TextChanged(System::Object^ sender, System::EventArgs^ e)

{

return System::Void();

}

System::Void Project1::MyForm::label5\_Click(System::Object^ sender, System::EventArgs^ e)

{

return System::Void();

}

MyForm.h

#pragma once

namespace Project1 {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

/// <summary>

/// Сводка для MyForm

/// </summary>

public ref class MyForm : public System::Windows::Forms::Form

{

public:

MyForm(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~MyForm()

{

if (components)

{

delete components;

}

}

private: System::Windows::Forms::Panel^ panel1;

protected:

private: System::Windows::Forms::TextBox^ textBox1;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::TextBox^ textBox2;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::Button^ button1;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::Label^ label5;

private: System::Windows::Forms::Panel^ panel2;

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

this->panel1 = (gcnew System::Windows::Forms::Panel());

this->label5 = (gcnew System::Windows::Forms::Label());

this->button1 = (gcnew System::Windows::Forms::Button());

this->label4 = (gcnew System::Windows::Forms::Label());

this->label3 = (gcnew System::Windows::Forms::Label());

this->label2 = (gcnew System::Windows::Forms::Label());

this->label1 = (gcnew System::Windows::Forms::Label());

this->textBox2 = (gcnew System::Windows::Forms::TextBox());

this->textBox1 = (gcnew System::Windows::Forms::TextBox());

this->panel2 = (gcnew System::Windows::Forms::Panel());

this->panel1->SuspendLayout();

this->panel2->SuspendLayout();

this->SuspendLayout();

//

// panel1

//

this->panel1->Controls->Add(this->panel2);

this->panel1->Controls->Add(this->button1);

this->panel1->Controls->Add(this->label4);

this->panel1->Controls->Add(this->label3);

this->panel1->Controls->Add(this->label2);

this->panel1->Controls->Add(this->label1);

this->panel1->Controls->Add(this->textBox2);

this->panel1->Controls->Add(this->textBox1);

this->panel1->Location = System::Drawing::Point(12, 29);

this->panel1->Name = L"panel1";

this->panel1->Size = System::Drawing::Size(409, 232);

this->panel1->TabIndex = 0;

//

// label5

//

this->label5->AutoSize = true;

this->label5->Font = (gcnew System::Drawing::Font(L"XO Caliburn", 9, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label5->Location = System::Drawing::Point(49, 9);

this->label5->Name = L"label5";

this->label5->Size = System::Drawing::Size(11, 14);

this->label5->TabIndex = 7;

this->label5->Text = L"-";

this->label5->TextAlign = System::Drawing::ContentAlignment::MiddleCenter;

this->label5->Click += gcnew System::EventHandler(this, &MyForm::label5\_Click);

//

// button1

//

this->button1->Location = System::Drawing::Point(124, 97);

this->button1->Name = L"button1";

this->button1->Size = System::Drawing::Size(168, 32);

this->button1->TabIndex = 6;

this->button1->Text = L"Определить область";

this->button1->UseVisualStyleBackColor = true;

this->button1->Click += gcnew System::EventHandler(this, &MyForm::button1\_Click);

//

// label4

//

this->label4->AutoSize = true;

this->label4->Location = System::Drawing::Point(15, 155);

this->label4->Name = L"label4";

this->label4->Size = System::Drawing::Size(181, 13);

this->label4->TabIndex = 5;

this->label4->Text = L"Область, соответствующая точке:";

//

// label3

//

this->label3->AutoSize = true;

this->label3->Location = System::Drawing::Point(289, 52);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(21, 13);

this->label3->TabIndex = 4;

this->label3->Text = L"y - ";

this->label3->Click += gcnew System::EventHandler(this, &MyForm::label3\_Click);

//

// label2

//

this->label2->AutoSize = true;

this->label2->Location = System::Drawing::Point(198, 52);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(18, 13);

this->label2->TabIndex = 3;

this->label2->Text = L"x -";

//

// label1

//

this->label1->AutoSize = true;

this->label1->Location = System::Drawing::Point(15, 52);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(147, 13);

this->label1->TabIndex = 2;

this->label1->Text = L"Введите координаты точки:";

this->label1->Click += gcnew System::EventHandler(this, &MyForm::label1\_Click);

//

// textBox2

//

this->textBox2->Location = System::Drawing::Point(307, 44);

this->textBox2->Name = L"textBox2";

this->textBox2->Size = System::Drawing::Size(54, 20);

this->textBox2->TabIndex = 1;

this->textBox2->TextChanged += gcnew System::EventHandler(this, &MyForm::textBox2\_TextChanged);

//

// textBox1

//

this->textBox1->Location = System::Drawing::Point(216, 45);

this->textBox1->Name = L"textBox1";

this->textBox1->Size = System::Drawing::Size(54, 20);

this->textBox1->TabIndex = 0;

this->textBox1->TextChanged += gcnew System::EventHandler(this, &MyForm::textBox1\_TextChanged);

//

// panel2

//

this->panel2->BackColor = System::Drawing::SystemColors::Window;

this->panel2->Controls->Add(this->label5);

this->panel2->Location = System::Drawing::Point(124, 186);

this->panel2->Name = L"panel2";

this->panel2->Size = System::Drawing::Size(168, 32);

this->panel2->TabIndex = 8;

//

// MyForm

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(433, 273);

this->Controls->Add(this->panel1);

this->Name = L"MyForm";

this->Text = L"Контрольная работа №1";

this->Load += gcnew System::EventHandler(this, &MyForm::MyForm\_Load);

this->panel1->ResumeLayout(false);

this->panel1->PerformLayout();

this->panel2->ResumeLayout(false);

this->panel2->PerformLayout();

this->ResumeLayout(false);

}

#pragma endregion

private: System::Void MyForm\_Load(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void label1\_Click(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void label3\_Click(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void textBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e);

private: System::Void textBox2\_TextChanged(System::Object^ sender, System::EventArgs^ e);

private: System::Void button1\_Click(System::Object^ sender, System::EventArgs^ e);

private: System::Void label5\_Click(System::Object^ sender, System::EventArgs^ e);

};

}