1. Задание

Целью работы является освоение практических навыков реализации графических приложений с двумерным интерфейсом с использованием стандартных вызовов библиотек и реализации простых геометрических преобразований, используемых в компьютерной графике.

№ вар.	Геометрическая фигура	Виды преобразований
1		1 – поворот по часовой стрелке 2 – поворот против часовой стрелки 3 – отражение по горизонтали 4 – отражение по вертикали

2. Текст программы

MainForm.h

```
#pragma once
#include "Figure.h"
#include "iostream"

namespace CGkontrol {

	using namespace System;
	using namespace System::ComponentModel;
	using namespace System::Collections;
	using namespace System::Windows::Forms;
	using namespace System::Data;
	using namespace System::Drawing;


	public ref class MainForm : public System::Windows::Forms::Form
	{
	public:
		MainForm(void)
		{
			InitializeComponent();

			int points[] = { 0, 0, 0, 80, -160, 80, -160, 0, 0, 0};
			rect = new rectangle(points);

			rectangleDraw(rect);
		}

	protected:

		~MainForm()
		{
			if (components)
			{
				delete components;
			}
		}
	private: Graphics^ g;
	private: rectangle* rect;
	private: System::Windows::Forms::Button^ clockwise;
```

```
        protected:
        private: System::Windows::Forms::Button^ reflectHorizont;
        private: System::Windows::Forms::Button^ reflectVertic;
        private: System::Windows::Forms::PictureBox^ picture;
        private: System::Windows::Forms::TextBox^ setAngle;
        private:

               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->clockwise = (gcnew System::Windows::Forms::Button());
                      this->reflectHorizont = (gcnew System::Windows::Forms::Button());
                      this->reflectVertic = (gcnew System::Windows::Forms::Button());
                      this->picture = (gcnew System::Windows::Forms::PictureBox());
                      this->setAngle = (gcnew System::Windows::Forms::TextBox());
                      (cli::safe_cast<System::ComponentModel::ISupportInitialize^>(this-
>picture))->BeginInit();
                      this->SuspendLayout();
                      //
                      // clockwise
                      //
                      this->clockwise->Location = System::Drawing::Point(95, 378);
                      this->clockwise->Name = L"clockwise";
                      this->clockwise->Size = System::Drawing::Size(75, 23);
                      this->clockwise->TabIndex = 0;
                      this->clockwise->Text = L"Поворот";
                      this->clockwise->UseVisualStyleBackColor = true;
                      this->clockwise->Click += gcnew System::EventHandler(this,
&MainForm::clockwise_Click);
                      //
                      // reflectHorizont
                      //
                      this->reflectHorizont->Location = System::Drawing::Point(192, 378);
                      this->reflectHorizont->Name = L"reflectHorizont";
                      this->reflectHorizont->Size = System::Drawing::Size(168, 23);
                      this->reflectHorizont->TabIndex = 2;
                      this->reflectHorizont->Text = L"Отражение по горизонтали";
                      this->reflectHorizont->UseVisualStyleBackColor = true;
                      this->reflectHorizont->Click += gcnew System::EventHandler(this,
&MainForm::reflectHorizont_Click);
                      //
                      // reflectVertic
                      //
                      this->reflectVertic->Location = System::Drawing::Point(366, 378);
                      this->reflectVertic->Name = L"reflectVertic";
                      this->reflectVertic->Size = System::Drawing::Size(161, 23);
                      this->reflectVertic->TabIndex = 3;
                      this->reflectVertic->Text = L"Отражение по вертикали";
                      this->reflectVertic->UseVisualStyleBackColor = true;
                      this->reflectVertic->Click += gcnew System::EventHandler(this,
&MainForm::reflectVertic_Click);
                      //
                      // picture
                      //
                      this->picture->Location = System::Drawing::Point(13, 13);
                      this->picture->Name = L"picture";
                      this->picture->Size = System::Drawing::Size(543, 359);
                      this->picture->TabIndex = 4;
                      this->picture->TabStop = false;
```

```
                    //
                    // setAngle
                    //
                    this->setAngle->Location = System::Drawing::Point(13, 378);
                    this->setAngle->MaxLength = 3;
                    this->setAngle->Name = L"setAngle";
                    this->setAngle->Size = System::Drawing::Size(76, 20);
                    this->setAngle->TabIndex = 5;
                    //
                    // MainForm
                    //
                    this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);
                    this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;
                    this->ClientSize = System::Drawing::Size(568, 440);
                    this->Controls->Add(this->setAngle);
                    this->Controls->Add(this->picture);
                    this->Controls->Add(this->reflectVertic);
                    this->Controls->Add(this->reflectHorizont);
                    this->Controls->Add(this->clockwise);
                    this->FormBorderStyle =
System::Windows::Forms::FormBorderStyle::FixedSingle;
                    this->MaximizeBox = false;
                    this->Name = L"MainForm";
                    this->ShowIcon = false;
                    this->StartPosition =
System::Windows::Forms::FormStartPosition::CenterScreen;
                    this->Text = L"GDI+ learn";
                    (cli::safe_cast<System::ComponentModel::ISupportInitialize^>(this-
>picture))->EndInit();
                    this->ResumeLayout(false);
                    this->PerformLayout();

            }
#pragma endregion

private: void rectangleDraw(rectangle* rec) {

       Pen^ pen;
       Bitmap^ Img;
       Img = gcnew Bitmap(picture->Width, picture->Height);

       pen = gcnew Pen(Color::Black);
       this->g = Graphics::FromImage(Img);
       g->TranslateTransform(picture->Width / 2, picture->Height / 2);
       this->g->DrawLine(System::Drawing::Pens::Green, rec->points[0].x, rec-
>points[0].y, rec->points[1].x, rec->points[1].y);
       this->g->DrawLine(System::Drawing::Pens::Black, rec->points[1].x, rec-
>points[1].y, rec->points[2].x, rec->points[2].y);
       this->g->DrawLine(System::Drawing::Pens::Blue, rec->points[2].x, rec->points[2].y,
rec->points[3].x, rec->points[3].y);
       this->g->DrawLine(System::Drawing::Pens::Red, rec->points[3].x, rec->points[3].y,
rec->points[4].x, rec->points[4].y);

       picture->Image = Img;
}

private: System::Void reflectHorizont_Click(System::Object^ sender, System::EventArgs^ e)
{
       rect->reflectH();
       rectangleDraw(rect);
}

private: System::Void reflectVertic_Click(System::Object^ sender, System::EventArgs^ e) {
       rect->reflectV();
       rectangleDraw(rect);
```

```
}

private: System::Void clockwise_Click(System::Object^ sender, System::EventArgs^ e) {
	int angle = 0;
	if(!System::Int32::TryParse((setAngle->Text), angle))setAngle->Clear();;
	rect->rotateClockwise(angle);
	rectangleDraw(rect);
}

};
}
```

Figure.h

```
#pragma once
#include <cmath>
#include <iostream>
#define PI 3.14159265

public class Point {
public:
	int x;
	int y;

	Point() {}

	Point(int x, int y) {
		this->x = x;
		this->y = y;
	}
};

public class rectangle {
public:
	Point points[10];

	rectangle(int* points) {
		int j = 0;
		for (int i(0); i < 5; i++) {
			j = i * 2;
			this->points[i].x = points[j];
			this->points[i].y = points[j+1];
		}
	}

	rectangle() {
	}

	void rotateClockwise(float val) {
		float angle = val * PI / 180;
		for (int i(0); i < 5; i++) {
			int x1 = round((this->points[i].x * cos(angle))) - round((this-
>points[i].y * sin(angle)));
			int y1 = round((this->points[i].x * sin(angle))) + round((this-
>points[i].y * cos(angle)));
			this->points[i].x = x1;
			this->points[i].y = y1;
		}
	}

	void reflectV() {
		for (int i(0); i < 5; i++) {
			int y1 = -(round(this->points[i].y));
```

```
                this->points[i].y = y1;
            }
        }

        void reflectH() {
            for (int i(0); i < 5; i++) {
                int x1 = -(round(this->points[i].x));
                this->points[i].x = x1;
            }
        }
};
```

MainForm.cpp

```
#include "Locale"
#include "MainForm.h"
using namespace System;
using namespace System::Windows::Forms;

[STAThread]
int main(array<String^>^ argv) {
    setlocale(LC_ALL, "rus");
    Application::EnableVisualStyles();
    Application::SetCompatibleTextRenderingDefault(false);

    CGkontrol::MainForm form; //WinFormsTest - имя вашего проекта
    Application::Run(% form);
    return 0;
}
```

3. Результат работы программы

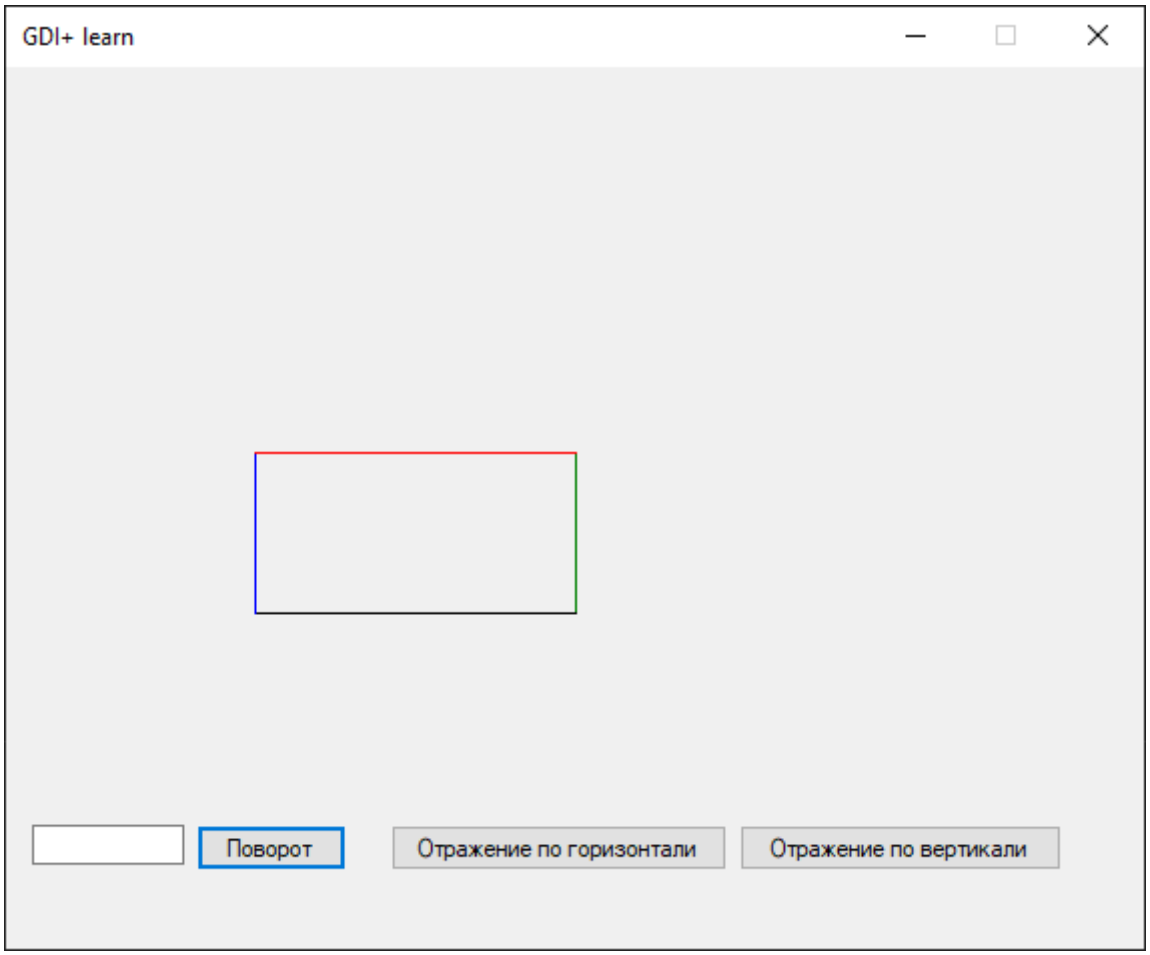

Рисунок 1 – результат выполнения программы

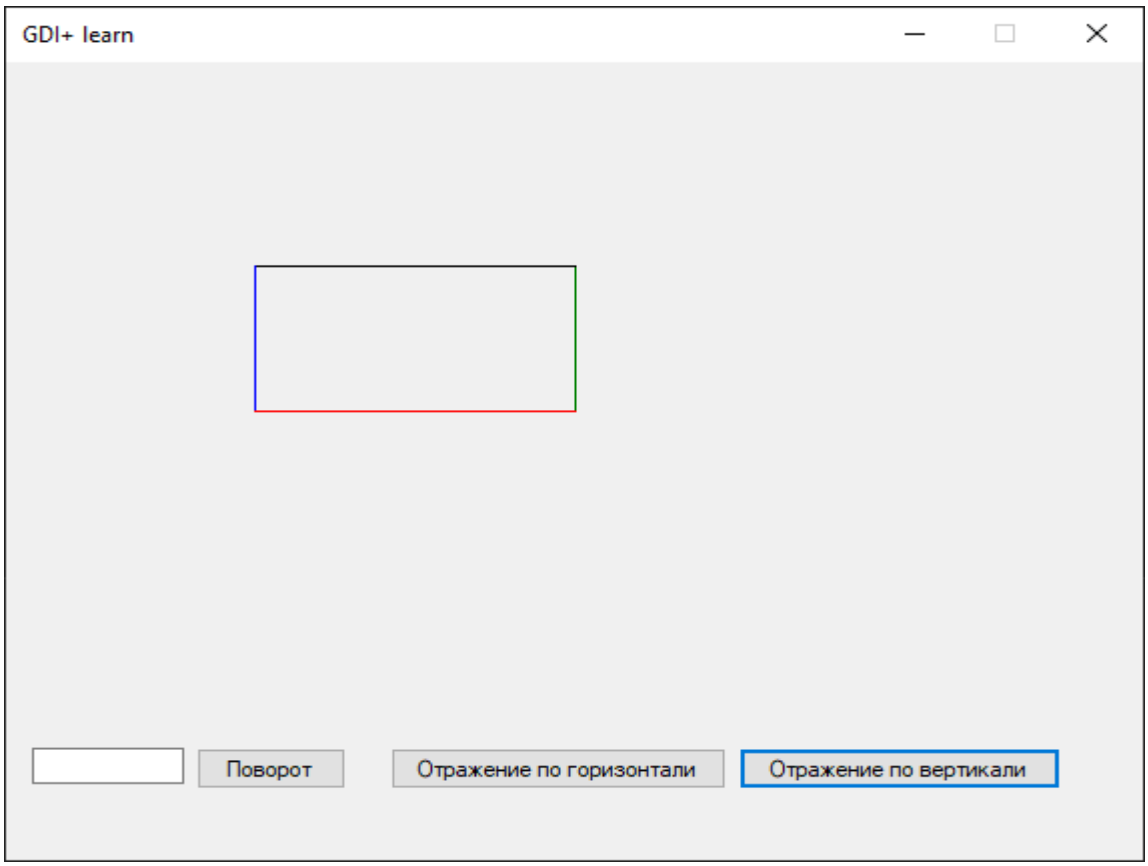

Рисунок 2 – результат отражения по вертикали

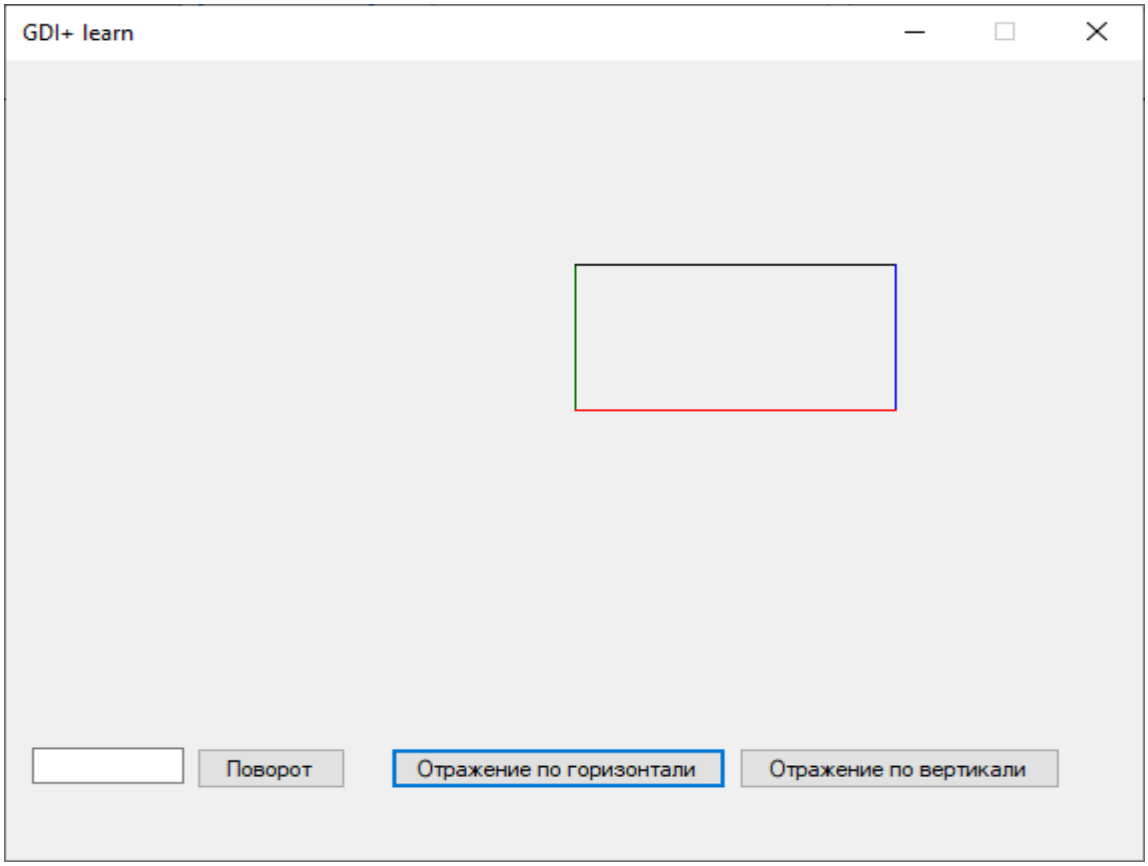

Рисунок 3 – результат отражения по горизонтали

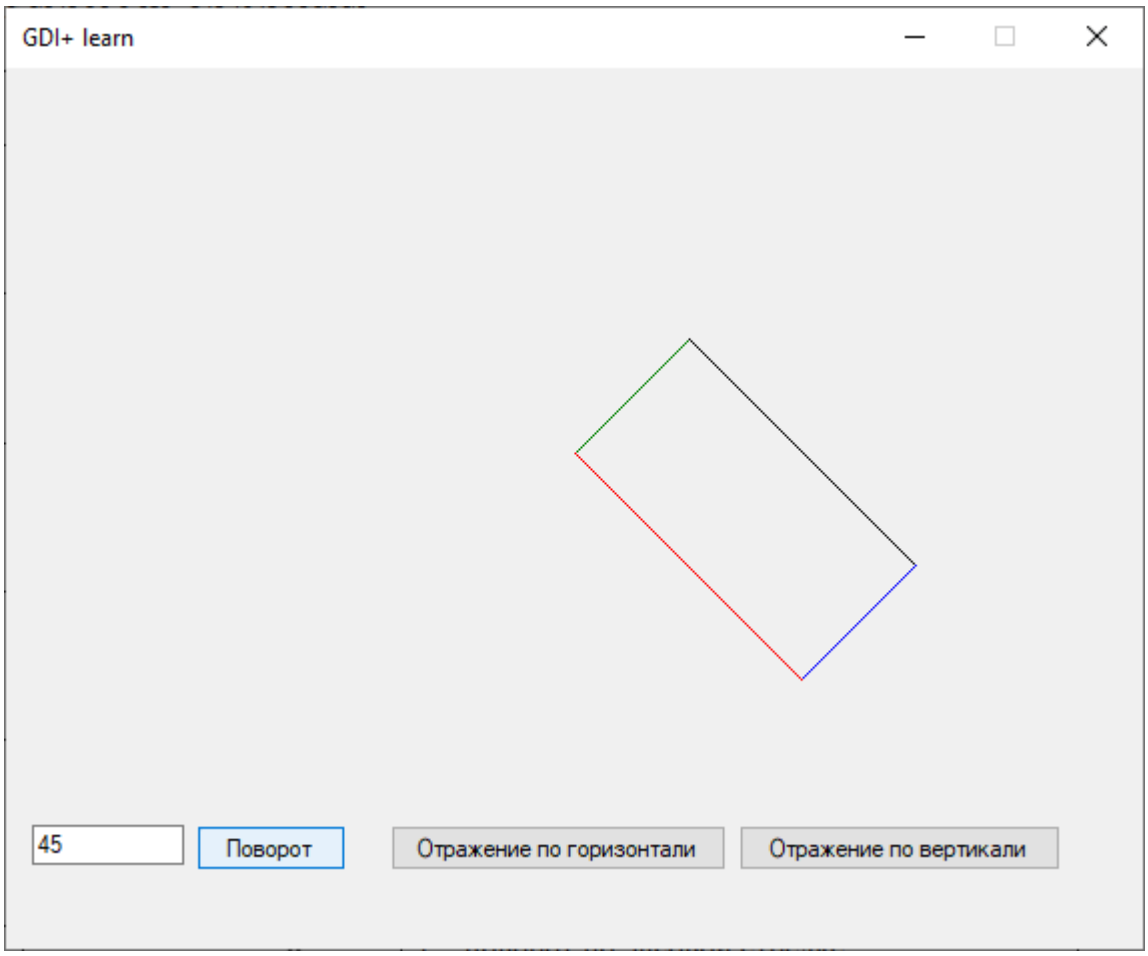

Рисунок 5 – результат поворота фигуры