Государственное учреждение высшего профессионального образования

«Белорусский государственный технологический университет»

Кафедра информационных систем и технологий

**Лабораторная работа №4**

**Изучение режимов отображения и методов пересчета мировых координат в оконные**

**Выполнила:**

 студентка 2 курса 2 группы

Кохнович Алина Сергеевна

Минск

2016

**Ход работы**

В ходе данной работы были изучены алгоритмы и реализованы:

**Задание 1:**

**CMatrix SpaceToWindow**

**-** функция, которая возвращает матрицу пересчета координат из мировых в оконные.

CMatrix SpaceToWindow(CRectD& RS, CRect RW)

{

CMatrix matr(3,3);

double Kx = (double)(((double)(RW.right - RW.left))/((double)(RS.right - RS.left)));

double Ky = (double)(((double)(RW.bottom - RW.top))/((double)(RS.bottom - RS.top)));

matr(0,1) = matr(1,0) = matr(2,0) = matr(2,1) = 0;

matr(2,2) = 1;

matr(0,0) = Kx;

matr(0,2) = RW.left - Kx\*RS.left;

matr(1,1) = -Ky;

matr(1,2) = RW.bottom + Ky\*RS.top;

return matr;

};

**Задание 2:**

**SetMyMode**

**-** функция, устанавливающая режим отображения MM\_ANISOTROPIC и его параметры.

int SetMyMode(CDC& dc, CRect& RS, CRectD& RW)

{

int dsx = RS.right - RS.left;

int dsy = RS.top - RS.bottom;

int xsl = RS.left;

int ysl = RS.bottom;

int dwx = RW.right - RW.left;

int dwy = RW.bottom - RW.top;

int xwl = RW.left;

int ywl = RW.bottom;

int buf = dc.SetMapMode(MM\_ANISOTROPIC);

dc.SetWindowExt(dsx,dsy);

dc.SetViewportExt(dwx, -dwy);

dc.SetWindowOrg(xsl,ysl);

dc.SetViewportOrg(xwl, ywl);

return buf;

};

**Задание 3:**

**CmyPen**

- структура, которая описывает создаваемое перо.

struct CMyPen{ int PenStyle;

int PenWidth;

COLORREF color;

CMyPen()

{

PenStyle = PS\_SOLID;

PenWidth = 1;

color = RGB(0,0,0);

}

void Set(int PS, int PW, COLORREF col)

{

PenStyle = PS;

PenWidth = PW;

color = col;

}};

**Задание 4 :**

**CPlot2D-**класс, который отображает зависимость Yi=F(Xi).

class CPlot2D

{

CMatrix x; //аргумент

CMatrix y; //функция

CMatrix k; //матрица пересчета координат

CRect RW; //прямоугольник в окне

CRectD RS; //прямоугольник в области МСК

CMyPen PenLine; //перо для линий

CMyPen PenAxis; //перо для осей

public:

CPlot2D()

{

//x.RedimMatrix(3,3);

//y.RedimMatrix(3,3);

k.RedimMatrix(3,3); //конструктор по умолчанию

};

void SetParams(CMatrix& XX, CMatrix& YY, CRect& RWX); //установка параметров графика

void SetWindowRect(CRect& RWX); //установка области в окне для отображения графика

void GetWindowCoords(double xs, double ys, int &xw, int &yw);//Пересчет координаты точки из МСК

//в оконную СК

void SetPenLine(CMyPen & PLine); //Перо для рисования графика

void SetPenAxis(CMyPen& PAxis); //Перо для осей координат

void Draw(CDC& dc, int lnd1,int lnd2); //Рисование с самостоятельным пересчетом координат

void Draw1(CDC& dc, int lnd1,int lnd2);//Рисование с БЕЗ самостоятельного пересчета координат

void GetRs(CRectD & RS1); //Возвращает область графика в мировой СК

//void DrawBezier();

};

**Задание 5:**

Также было создано приложение **LibChart2D**, которое отображает множество точек на плоскости, заданных в мировой системе координат, которые содержатся в текстовом файле ReadMe.txt.

В классе **CChildView** были реализованы функции:

Работа функций демонстрируется в окне:

При нажатии Файл►F1 выполняется функция **OnGraph1()**. Данная функция рассчитывает значение функции  для  с шагом  , результатом работы является график зависимости, полученных массивов  .

void Clab4App::OnGraph1()

{

CMyPen MP;

MP.Set(PS\_SOLID, 1, RGB(255,0,0));

CMyPen MP\_LINE;

MP\_LINE.Set(PS\_SOLID, 1, RGB(0,0,0));

CPlot1.SetPenLine(MP);

CPlot1.SetPenAxis(MP\_LINE);

CRect Rect(50,50,450,450);

ifstream F1;

F1.open("F1.txt");

CMatrix M1X(F1);

CMatrix M1Y(F1);

F1.close();

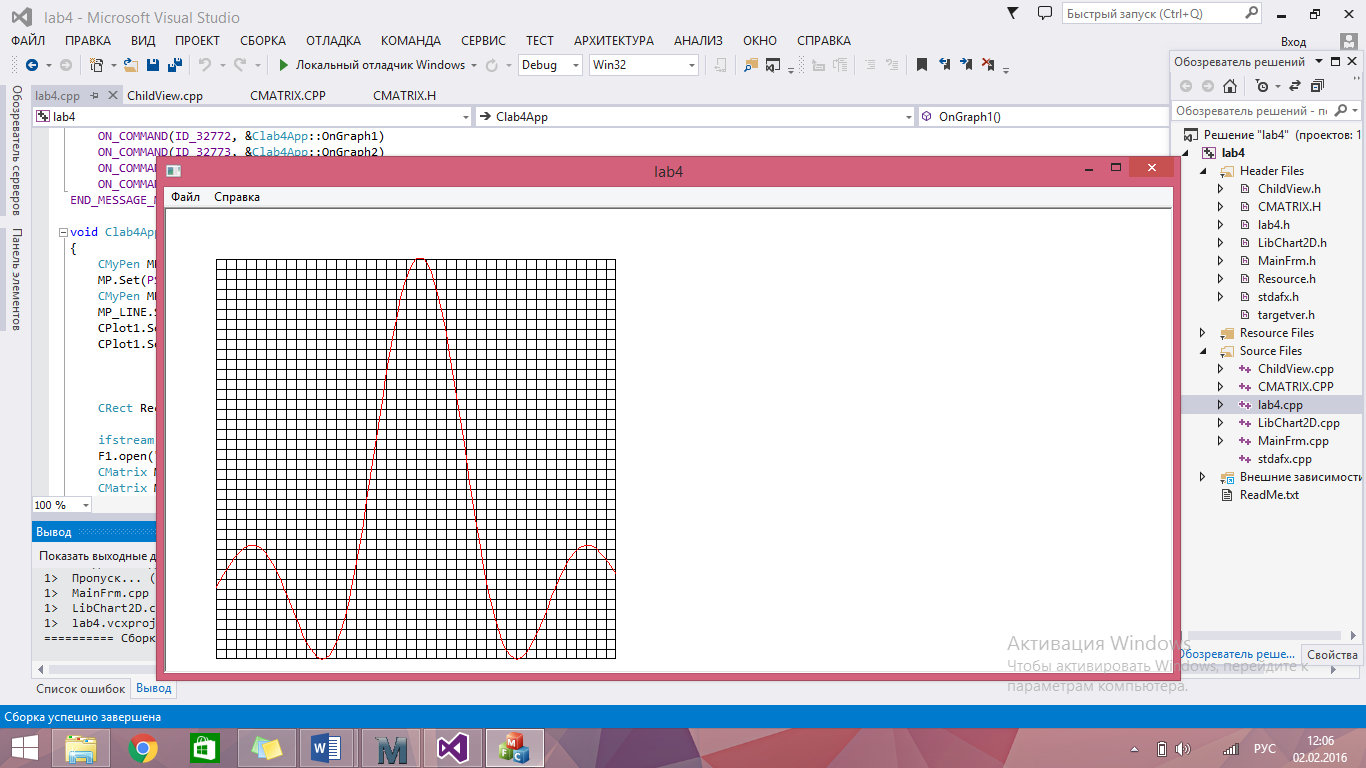
CPlot1.SetParams(M1X, M1Y, Rect);

Rect.SetRect(550, 300, 650, 750);

graf\_type = 1;

theApp.m\_pMainWnd->Invalidate();

}



При нажатии Файл►F2 выполняется функция **OnGraph2()**. Данная функция рассчитывает значение функции  для  с шагом  , результатом работы является график зависимости, полученных массивов  .

void Clab4App::OnGraph2()

{

CMyPen MP;

MP.Set(PS\_SOLID, 1, RGB(0,255,0));

CMyPen MP\_LINE;

MP\_LINE.Set(PS\_SOLID, 1, RGB(0,0,0));

CPlot2.SetPenLine(MP);

CPlot2.SetPenAxis(MP\_LINE);

CRect Rect(50,50,450,450);

ifstream F2;

F2.open("F2.txt");

CMatrix M2X(F2);

CMatrix M2Y(F2);

F2.close();

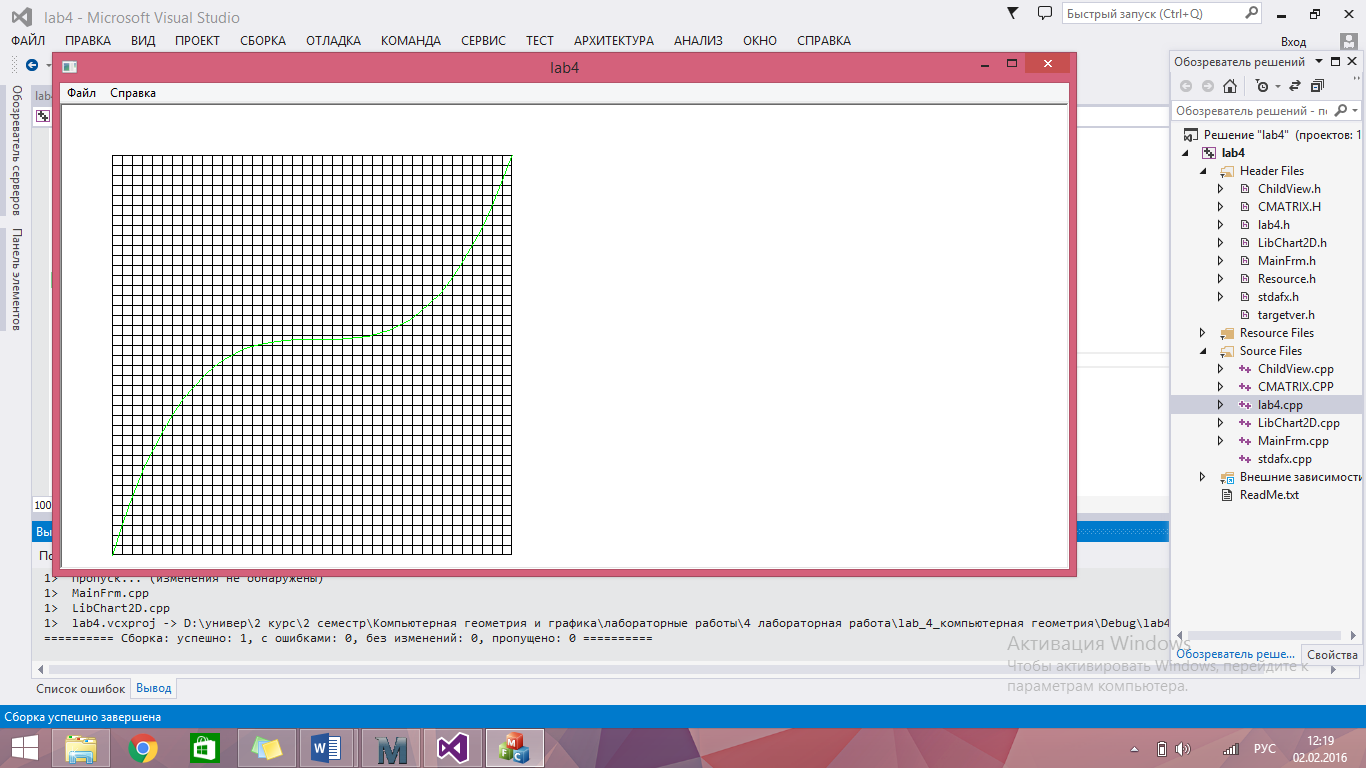
CPlot2.SetParams(M2X, M2Y, Rect);

Rect.SetRect( 100, 6, 160, 28);

graf\_type = 2;

theApp.m\_pMainWnd->Invalidate();

}



При нажатии Файл►F3 выполняется функция **OnGraph3()**. Данная функция рассчитывает значение функции  для  с шагом  , результатом работы является график зависимости, полученных массивов  .

void Clab4App::OnGraph3()

{

CMyPen MP;

MP.Set(PS\_DASHDOT, 3, RGB(255, 0, 0));

CMyPen MP\_LINE;

MP\_LINE.Set(PS\_SOLID, 1, RGB(0,0,0));

CPlot3.SetPenLine(MP);

CPlot3.SetPenAxis(MP\_LINE);

CRect Rect(0,-90,192,100);

ifstream F3;

F3.open("F3.txt");

CMatrix M3X(F3);

CMatrix M3Y(F3);

F3.close();

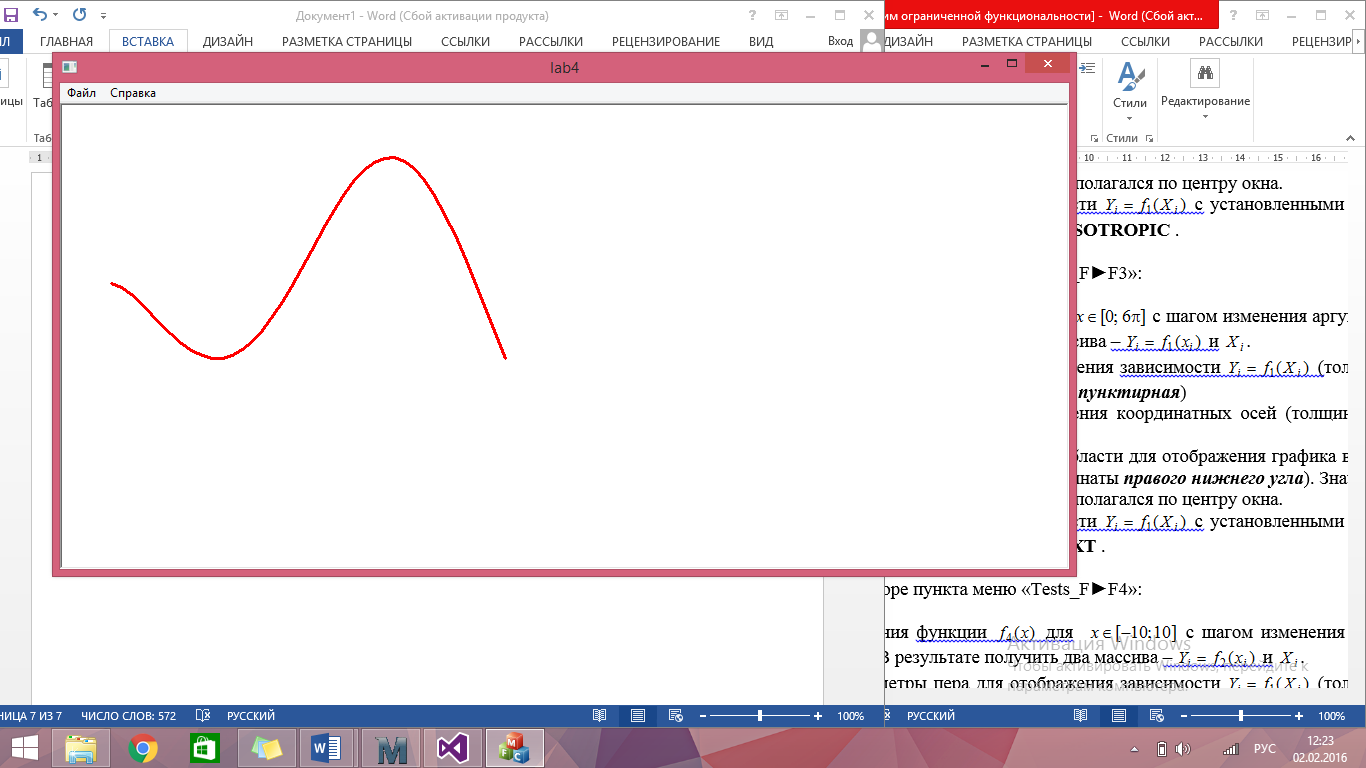
CPlot3.SetParams(M3X, M3Y, Rect);

Rect.SetRect( 50, 34, 160, 62);

graf\_type = 3;

theApp.m\_pMainWnd->Invalidate();

}



При нажатии Файл►F4 выполняется функция **OnGraph4()**. Данная функция рассчитывает значение функции  для  с шагом  , результатом работы является график зависимости, полученных массивов  .

void Clab4App::OnGraph4()

{

CMyPen MP;

MP.Set(PS\_SOLID, 2, RGB(255,0,0));

CMyPen MP\_LINE;

MP\_LINE.Set(PS\_SOLID, 1, RGB(0,0,0));

CPlot4.SetPenLine(MP);

CPlot4.SetPenAxis(MP\_LINE);

CRect Rect(50,50,450,450);

ifstream F4;

F4.open("F4.txt");

CMatrix M4X(F4);

CMatrix M4Y(F4);

F4.close();

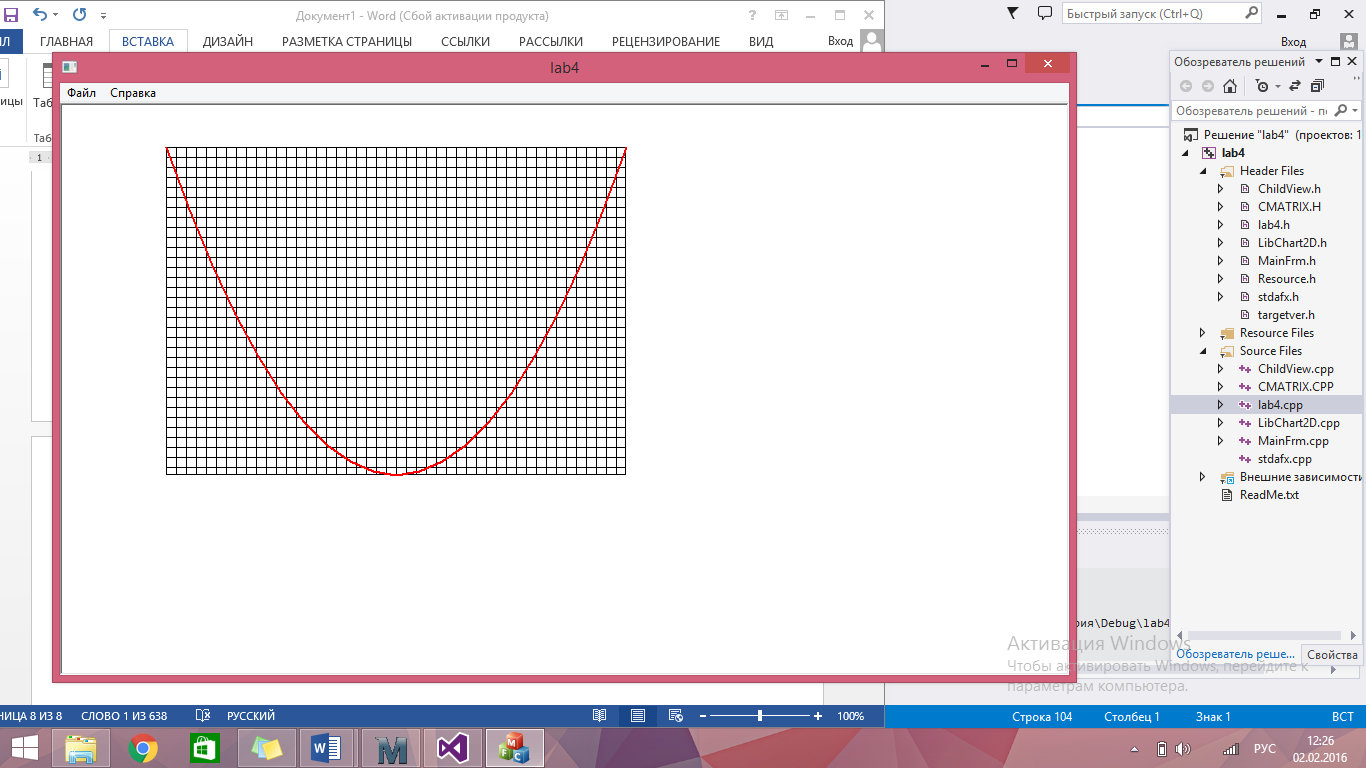
Rect.SetRect( 100, 34, 560, 362);

CPlot4.SetParams(M4X, M4Y, Rect);

graf\_type = 4;

theApp.m\_pMainWnd->Invalidate();

}



Листинг функции **OnGraph4()**:

При нажатии Файл►Графики выполняется функция **OnGraph()**. Данная функция отображает сразу четыре графика, которые были созданы ранее.

void Clab4App::OnGraph()

{

CMyPen MP1;

CMyPen MP2;

CMyPen MP3;

CMyPen MP4;

MP1.Set(PS\_SOLID, 1, RGB(0,0,0));

MP2.Set(PS\_SOLID, 1, RGB(0,0,0));

MP3.Set(PS\_SOLID, 1, RGB(0,0,0));

MP4.Set(PS\_SOLID, 1, RGB(0,0,0));

CMyPen MP1\_LINE;

CMyPen MP2\_LINE;

CMyPen MP3\_LINE;

CMyPen MP4\_LINE;

MP1\_LINE.Set(PS\_SOLID, 2, RGB(250,0,0));

CPlot1.SetPenLine(MP1\_LINE);

MP2\_LINE.Set(PS\_SOLID, 3, RGB(250,0,0));

CPlot2.SetPenLine(MP2\_LINE);

MP3\_LINE.Set(PS\_SOLID, 2, RGB(250,0,0));

CPlot3.SetPenLine(MP3\_LINE);

MP4\_LINE.Set(PS\_SOLID, 3, RGB(250,0,0));

CPlot4.SetPenLine(MP4\_LINE);

CPlot1.SetPenAxis(MP1);

CPlot2.SetPenAxis(MP2);

CPlot3.SetPenAxis(MP3);

CPlot4.SetPenAxis(MP4);

CRect Rect(50,50,350,250);

ifstream F1;

F1.open("F1.txt");

CMatrix M1X(F1);

CMatrix M1Y(F1);

F1.close();

ifstream F2;

F2.open("F2.txt");

CMatrix M2X(F2);

CMatrix M2Y(F2);

F2.close();

ifstream F3;

F3.open("F3.txt");

CMatrix M3X(F3);

CMatrix M3Y(F3);

F3.close();

ifstream F4;

F4.open("F4.txt");

CMatrix M4X(F4);

CMatrix M4Y(F4);

F4.close();

CPlot1.SetParams(M1X, M1Y, Rect);

Rect.SetRect(50, 350, 350, 550);

CPlot2.SetParams(M2X, M2Y, Rect);

Rect.SetRect( 450, 50, 760, 250);

CPlot3.SetParams(M3X, M3Y, Rect);

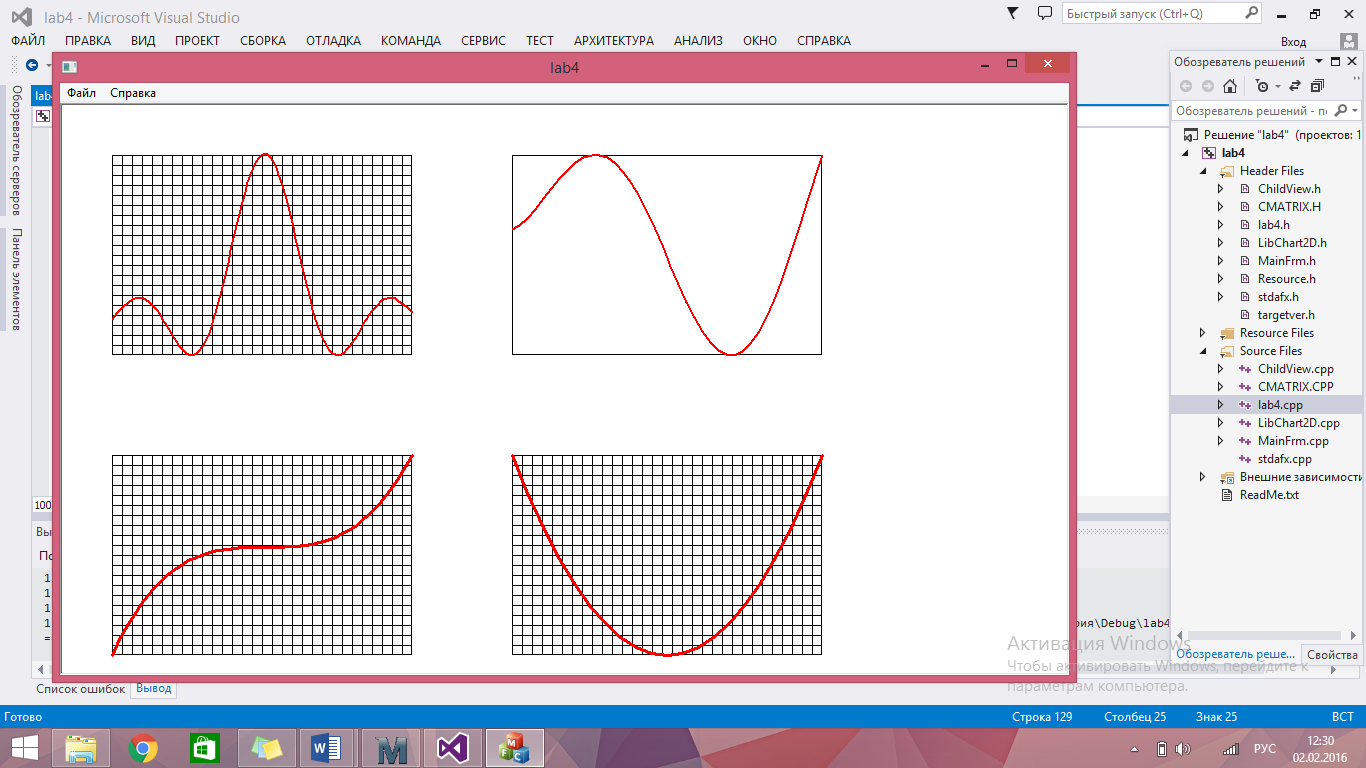
Rect.SetRect( 450, 350, 760, 550);

CPlot4.SetParams(M4X, M4Y, Rect);

graf\_type = 5;

theApp.m\_pMainWnd->Invalidate();

}



**Листинг программы:**

// lab4.cpp : Defines the class behaviors for the application.

//

#include "stdafx.h"

#include "afxwinappex.h"

#include "afxdialogex.h"

#include "lab4.h"

#include "MainFrm.h"

#include "CMATRIX.H"

#include "LibChart2D.h"

#include "math.h"

#include <iostream>

#include <fstream>

#ifdef \_DEBUG

#define new DEBUG\_NEW

#endif

using namespace std;

// Clab4App

BEGIN\_MESSAGE\_MAP(Clab4App, CWinApp)

ON\_COMMAND(ID\_APP\_ABOUT, &Clab4App::OnAppAbout)

ON\_COMMAND(ID\_FILE\_GRAPHICS, &Clab4App::OnGraph)

ON\_COMMAND(ID\_32772, &Clab4App::OnGraph1)

ON\_COMMAND(ID\_32773, &Clab4App::OnGraph2)

ON\_COMMAND(ID\_32774, &Clab4App::OnGraph3)

ON\_COMMAND(ID\_32775, &Clab4App::OnGraph4)

END\_MESSAGE\_MAP()

void Clab4App::OnGraph1()

{

CMyPen MP;

MP.Set(PS\_SOLID, 1, RGB(255,0,0));

CMyPen MP\_LINE;

MP\_LINE.Set(PS\_SOLID, 1, RGB(0,0,0));

CPlot1.SetPenLine(MP);

CPlot1.SetPenAxis(MP\_LINE);

CRect Rect(50,50,450,450);

ifstream F1;

F1.open("F1.txt");

CMatrix M1X(F1);

CMatrix M1Y(F1);

F1.close();

CPlot1.SetParams(M1X, M1Y, Rect);

Rect.SetRect(550, 300, 650, 750);

graf\_type = 1;

theApp.m\_pMainWnd->Invalidate();

}

void Clab4App::OnGraph2()

{

CMyPen MP;

MP.Set(PS\_SOLID, 1, RGB(0,255,0));

CMyPen MP\_LINE;

MP\_LINE.Set(PS\_SOLID, 1, RGB(0,0,0));

CPlot2.SetPenLine(MP);

CPlot2.SetPenAxis(MP\_LINE);

CRect Rect(50,50,450,450);

ifstream F2;

F2.open("F2.txt");

CMatrix M2X(F2);

CMatrix M2Y(F2);

F2.close();

CPlot2.SetParams(M2X, M2Y, Rect);

Rect.SetRect( 100, 6, 160, 28);

graf\_type = 2;

theApp.m\_pMainWnd->Invalidate();

}

void Clab4App::OnGraph3()

{

CMyPen MP;

MP.Set(PS\_DASHDOT, 3, RGB(255, 0, 0));

CMyPen MP\_LINE;

MP\_LINE.Set(PS\_SOLID, 1, RGB(0,0,0));

CPlot3.SetPenLine(MP);

CPlot3.SetPenAxis(MP\_LINE);

CRect Rect(0,-90,192,100);

ifstream F3;

F3.open("F3.txt");

CMatrix M3X(F3);

CMatrix M3Y(F3);

F3.close();

CPlot3.SetParams(M3X, M3Y, Rect);

Rect.SetRect( 50, 34, 160, 62);

graf\_type = 3;

theApp.m\_pMainWnd->Invalidate();

}

void Clab4App::OnGraph4()

{

CMyPen MP;

MP.Set(PS\_SOLID, 2, RGB(255,0,0));

CMyPen MP\_LINE;

MP\_LINE.Set(PS\_SOLID, 1, RGB(0,0,0));

CPlot4.SetPenLine(MP);

CPlot4.SetPenAxis(MP\_LINE);

CRect Rect(50,50,450,450);

ifstream F4;

F4.open("F4.txt");

CMatrix M4X(F4);

CMatrix M4Y(F4);

F4.close();

Rect.SetRect( 100, 34, 560, 362);

CPlot4.SetParams(M4X, M4Y, Rect);

graf\_type = 4;

theApp.m\_pMainWnd->Invalidate();

}

// Clab4App construction

void Clab4App::OnGraph()

{

CMyPen MP1;

CMyPen MP2;

CMyPen MP3;

CMyPen MP4;

MP1.Set(PS\_SOLID, 1, RGB(0,0,0));

MP2.Set(PS\_SOLID, 1, RGB(0,0,0));

MP3.Set(PS\_SOLID, 1, RGB(0,0,0));

MP4.Set(PS\_SOLID, 1, RGB(0,0,0));

CMyPen MP1\_LINE;

CMyPen MP2\_LINE;

CMyPen MP3\_LINE;

CMyPen MP4\_LINE;

MP1\_LINE.Set(PS\_SOLID, 2, RGB(250,0,0));

CPlot1.SetPenLine(MP1\_LINE);

MP2\_LINE.Set(PS\_SOLID, 3, RGB(250,0,0));

CPlot2.SetPenLine(MP2\_LINE);

MP3\_LINE.Set(PS\_SOLID, 2, RGB(250,0,0));

CPlot3.SetPenLine(MP3\_LINE);

MP4\_LINE.Set(PS\_SOLID, 3, RGB(250,0,0));

CPlot4.SetPenLine(MP4\_LINE);

CPlot1.SetPenAxis(MP1);

CPlot2.SetPenAxis(MP2);

CPlot3.SetPenAxis(MP3);

CPlot4.SetPenAxis(MP4);

CRect Rect(50,50,350,250);

ifstream F1;

F1.open("F1.txt");

CMatrix M1X(F1);

CMatrix M1Y(F1);

F1.close();

ifstream F2;

F2.open("F2.txt");

CMatrix M2X(F2);

CMatrix M2Y(F2);

F2.close();

ifstream F3;

F3.open("F3.txt");

CMatrix M3X(F3);

CMatrix M3Y(F3);

F3.close();

ifstream F4;

F4.open("F4.txt");

CMatrix M4X(F4);

CMatrix M4Y(F4);

F4.close();

CPlot1.SetParams(M1X, M1Y, Rect);

Rect.SetRect(50, 350, 350, 550);

CPlot2.SetParams(M2X, M2Y, Rect);

Rect.SetRect( 450, 50, 760, 250);

CPlot3.SetParams(M3X, M3Y, Rect);

Rect.SetRect( 450, 350, 760, 550);

CPlot4.SetParams(M4X, M4Y, Rect);

graf\_type = 5;

theApp.m\_pMainWnd->Invalidate();

}

Clab4App::Clab4App()

{

// TODO: replace application ID string below with unique ID string; recommended

// format for string is CompanyName.ProductName.SubProduct.VersionInformation

SetAppID(\_T("lab4.AppID.NoVersion"));

// TODO: add construction code here,

// Place all significant initialization in InitInstance

}

// The one and only Clab4App object

Clab4App theApp;

// Clab4App initialization

BOOL Clab4App::InitInstance()

{

CWinApp::InitInstance();

flag = false;

EnableTaskbarInteraction(FALSE);

// AfxInitRichEdit2() is required to use RichEdit control

// AfxInitRichEdit2();

// Standard initialization

// If you are not using these features and wish to reduce the size

// of your final executable, you should remove from the following

// the specific initialization routines you do not need

// Change the registry key under which our settings are stored

// TODO: You should modify this string to be something appropriate

// such as the name of your company or organization

SetRegistryKey(\_T("Local AppWizard-Generated Applications"));

// To create the main window, this code creates a new frame window

// object and then sets it as the application's main window object

CMainFrame\* pFrame = new CMainFrame;

if (!pFrame)

return FALSE;

m\_pMainWnd = pFrame;

// create and load the frame with its resources

pFrame->LoadFrame(IDR\_MAINFRAME,

WS\_OVERLAPPEDWINDOW | FWS\_ADDTOTITLE, NULL,

NULL);

pFrame->ShowWindow(SW\_SHOW);

pFrame->UpdateWindow();

return TRUE;

}

int Clab4App::ExitInstance()

{

return CWinApp::ExitInstance();

}

class CAboutDlg : public CDialogEx

{

public:

CAboutDlg();

enum { IDD = IDD\_ABOUTBOX };

protected:

virtual void DoDataExchange(CDataExchange\* pDX);

protected:

DECLARE\_MESSAGE\_MAP()

};

CAboutDlg::CAboutDlg() : CDialogEx(CAboutDlg::IDD)

{

}

void CAboutDlg::DoDataExchange(CDataExchange\* pDX)

{

CDialogEx::DoDataExchange(pDX);

}

BEGIN\_MESSAGE\_MAP(CAboutDlg, CDialogEx)

END\_MESSAGE\_MAP()

void Clab4App::OnAppAbout()

{

CAboutDlg aboutDlg;

aboutDlg.DoModal();

}