Министерство образования и науки Российской Федерации

НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Кафедра прикладной математики

Лабораторная работа №4

по дисциплине «Численные методы»

Группа ПМ-63

Студент Майер В. А.

Преподаватель Задорожный А.Г.

Вариант 1, 2, 5

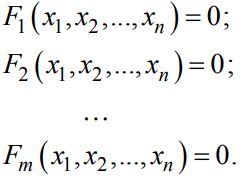
Новосибирск, 2018

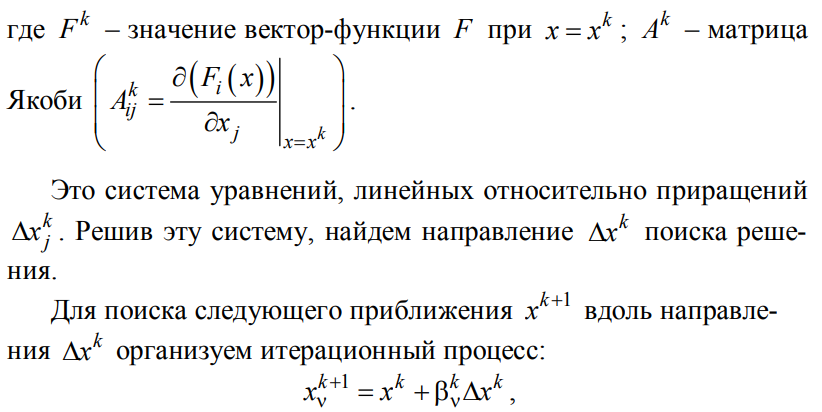
1. **Цель работы**  
   Разработать программу решения системы нелинейных уравнений методом Ньютона. Провести исследования метода для нескольких систем размерном от 2 до 10.

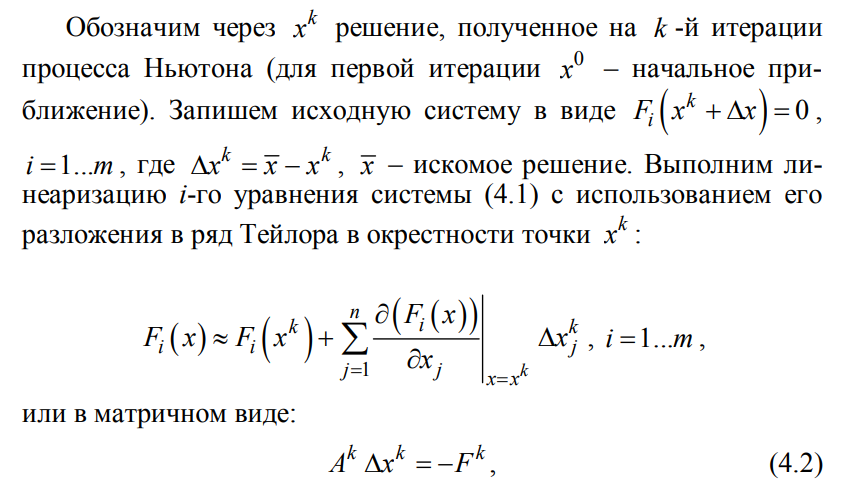
Задание:

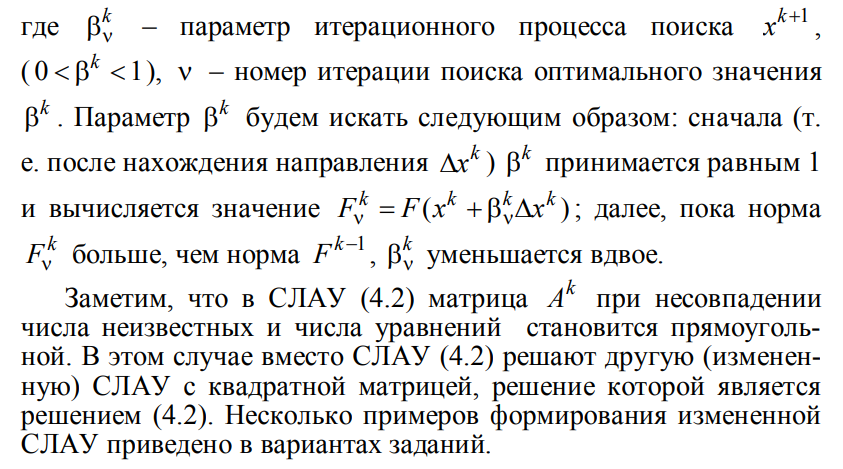
1. m ≤ n. Для нахождения , являющегося решением системы , фиксировать как нулевые те ее (n - m ) компонентов с номерами j , для которых минимальны. Производные при формировании матрицы Якоби вычислять аналитически.
2. m ≥ n. Для нахождения из системы те ее ( m - n) уравнений, для которых абсолютные значения минимальны, исключаются из системы. При вычислении нормы вектора в процессе подбора параметра учитывать все уравнения системы. Производные при формировании матрицы Якоби вычислять аналитически.
3. В задании 1 производные при формировании матрицы Якоби вычислять численно.
4. **Анализ**

Пусть дана СНУ в виде:







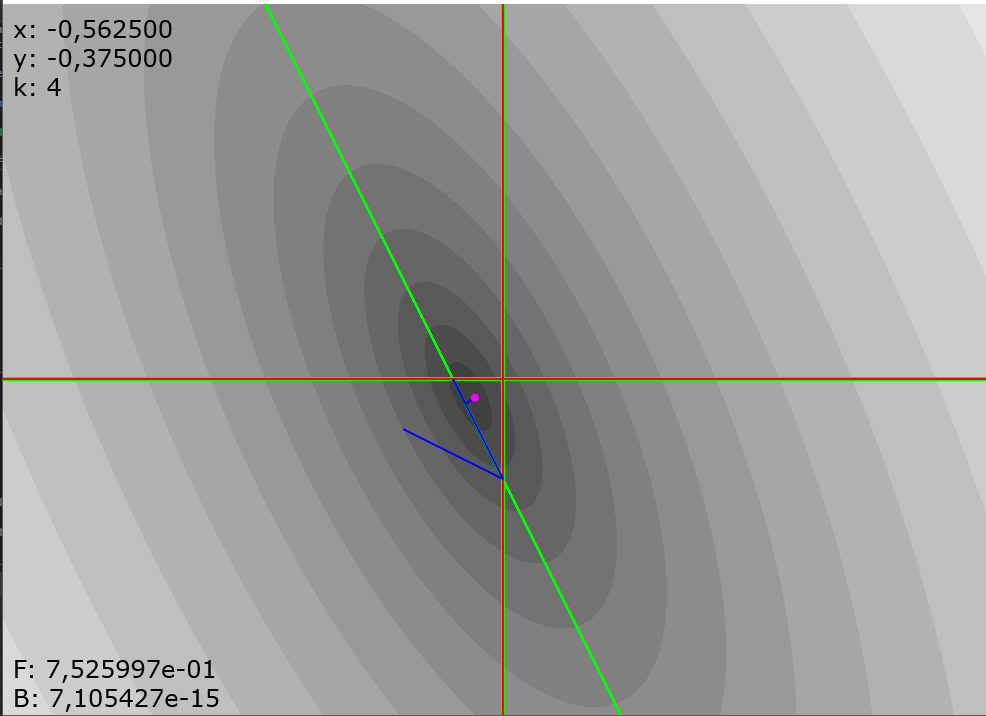


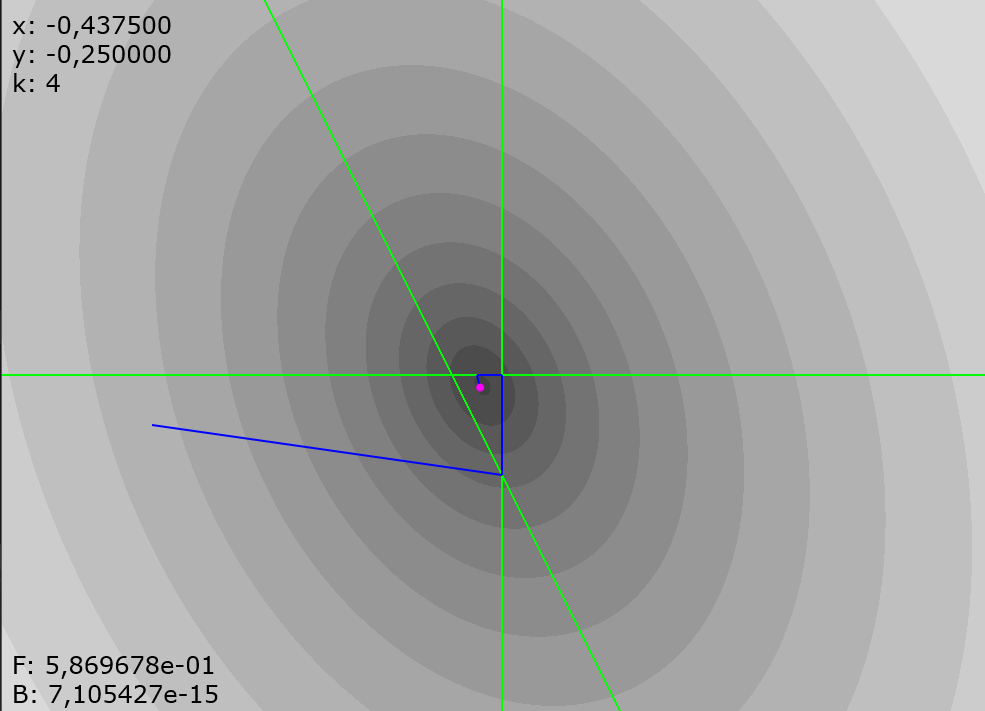
СЛАУ решаются методом Гаусса с выбором ведущего элемента.

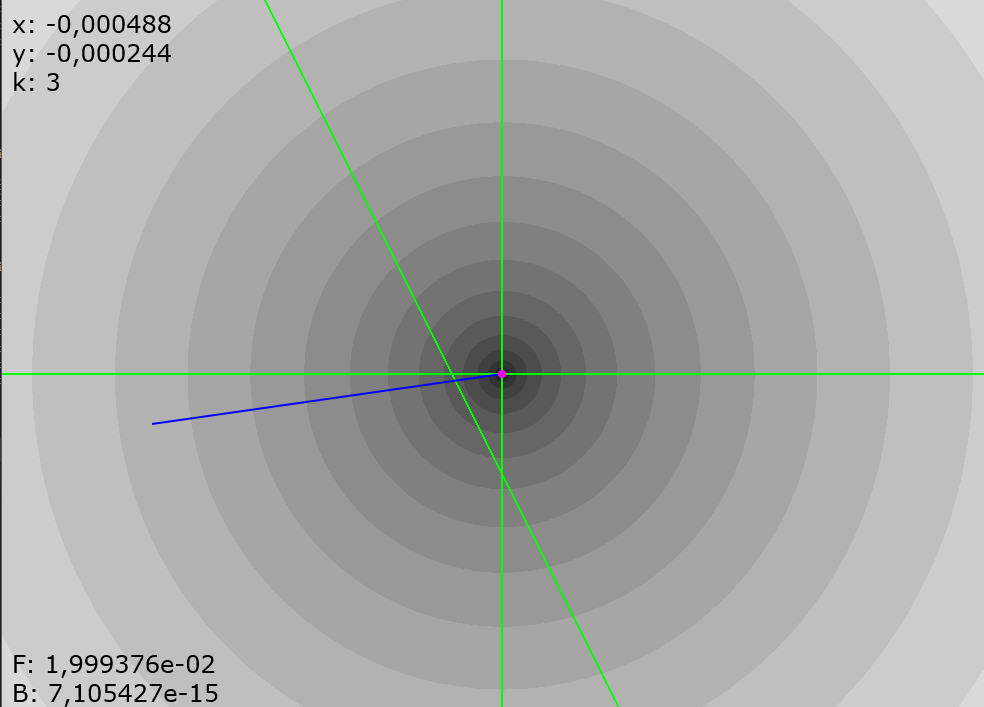
1. **Три попарно пересекающиеся прямые**

Влеяние веса на прямые

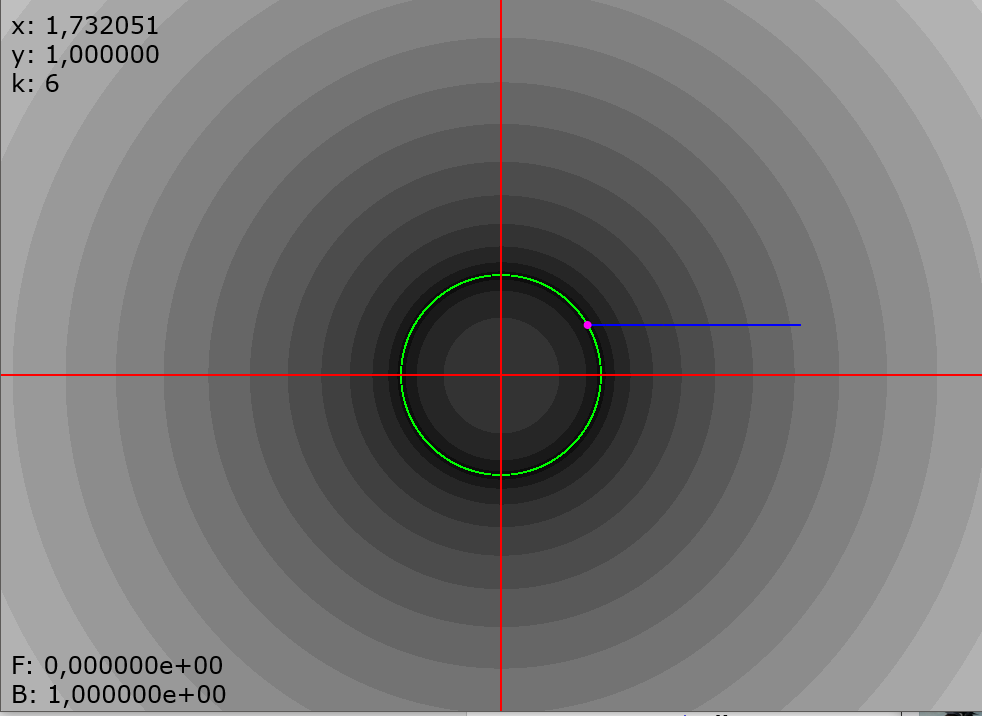
y = 0. x = 0 y = -2x – 2.

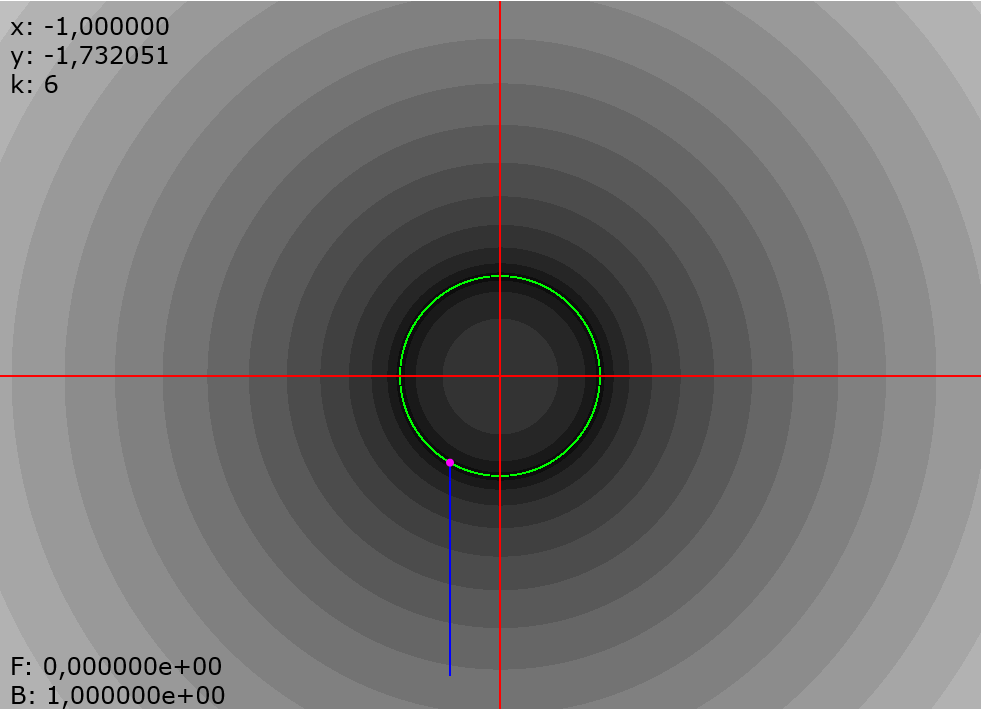






1. **N > M**





1. **Тесты с непересекающимся окружностями**

|  |  |
| --- | --- |
| Аналитически | Численно |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

1. **3-х мерное пространство с 2 функциями**

X^2 + Y^2 + Z^2 = 4

Y = X + Z

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| k | x | y | z | |b| | |F| |
| 0 | 7.000000 | -4.000000 | 8.000000 | 1.000000e+00 | 1.264358e+02 |
| 1 | 6.300781 | -4.000000 | 8.550781 | 7.812500e-03 | 1.264358e+02 |
| 2 | 4.928729 | -4.000000 | 9.333722 | 3.125000e-02 | 1.262313e+02 |
| 3 | 1.842706 | -4.000000 | 10.13693 | 1.250000e-01 | 1.247547e+02 |
| 4 | 1.842706 | 0.192161 | 10.33418 | 2.500000e-01 | 1.192288e+02 |
| 5 | -6.487821 | 0.192161 | 6.679982 | 1.000000e+00 | 1.069019e+02 |
| 6 | -3.345653 | 0.192161 | 3.537814 | 1.000000e+00 | 8.275091e+01 |
| 7 | -1.911314 | 0.192161 | 2.103475 | 1.000000e+00 | 1.974644e+01 |
| 8 | -1.398877 | 0.192161 | 1.591038 | 1.000000e+00 | 4.114655e+00 |
| 9 | -1.311051 | 0.192161 | 1.503212 | 1.000000e+00 | 5.251838e-01 |
| 10 | -1.308310 | 0.192161 | 1.500471 | 1.000000e+00 | 1.542677e-02 |
| 11 | -1.308308 | 0.192161 | 1.500468 | 1.000000e+00 | 1.502418e-05 |
| 12 | -1.308308 | 0.192161 | 1.500468 | 1.000000e+00 | 1.430678e-11 |
|  |  |  |  |  |  |
| k | x | y | z | |b| | |F| |
| 0 | 0.500000 | 0.700000 | -0.300000 | 1.000000e+00 | 3.209190e+00 |
| 1 | 1.306250 | 1.256250 | -0.300000 | 5.000000e-01 | 3.209190e+00 |
| 2 | 1.550869 | 1.250869 | -0.300000 | 1.000000e+00 | 6.736534e-01 |
| 3 | 1.540185 | 1.240185 | -0.300000 | 1.000000e+00 | 5.986736e-02 |
| 4 | 1.540144 | 1.240144 | -0.300000 | 1.000000e+00 | 2.282944e-04 |
| 5 | 1.540144 | 1.240144 | -0.300000 | 1.000000e+00 | 3.370974e-09 |
| 6 | 1.540144 | 1.240144 | -0.300000 | 1.000000e+00 | 1.665335e-16 |
|  |  |  |  |  |  |
| k | x | y | z | |b| | |F| |
| 0 | 127.0000 | 436.0000 | -139.8900 | 1.000000e+00 | 2.257907e+05 |
| 1 | 127.0000 | 266.8065 | 139.8065 | 1.000000e+00 | 2.257907e+05 |
| 2 | 127.0000 | 135.4081 | 8.408143 | 1.000000e+00 | 1.068566e+05 |
| 3 | 61.20352 | 69.61166 | 8.408143 | 1.000000e+00 | 3.453106e+04 |
| 4 | 28.10969 | 36.51783 | 8.408143 | 1.000000e+00 | 8.658352e+03 |
| 5 | 11.16332 | 19.57146 | 8.408143 | 1.000000e+00 | 2.190404e+03 |
| 6 | 1.819530 | 10.22767 | 8.408143 | 1.000000e+00 | 5.743588e+02 |
| 7 | 1.819530 | 5.542800 | 3.723270 | 1.000000e+00 | 1.746129e+02 |
| 8 | 1.819530 | 3.174155 | 1.354625 | 1.000000e+00 | 4.389607e+01 |
| 9 | 0.696015 | 2.050640 | 1.354625 | 1.000000e+00 | 1.122096e+01 |
| 10 | 0.696015 | 1.679954 | 0.983939 | 1.000000e+00 | 2.524572e+00 |
| 11 | 0.696015 | 1.628372 | 0.932357 | 1.000000e+00 | 2.748169e-01 |
| 12 | 0.696015 | 1.627333 | 0.931318 | 1.000000e+00 | 5.321372e-03 |
| 13 | 0.696015 | 1.627333 | 0.931317 | 1.000000e+00 | 2.159186e-06 |
| 14 | 0.696015 | 1.627333 | 0.931317 | 1.000000e+00 | 3.552714e-13 |

1. **Зависимость от шага при численном диф.**

Последний тест из 5 пункта.

x^2 + y^2 = 4

(x+1)^2 + y^2 = 1

Начальная точка (1; -1)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Шаг  dx | k | x | y | |b| | |F| |
| 1e-1 | 28 | -2.000000 | -0.000000 | 1.000000e+00 | 3.093441e-14 |
| 1e-2 | 28 | -2.000000 | -0.000000 | 1.000000e+00 | 2.920541e-14 |
| 1e-3 | 28 | -2.000000 | -0.000000 | 1.000000e+00 | 2.983341e-14 |
| 1e-4 | 28 | -2.000000 | -0.000000 | 1.000000e+00 | 2.904713e-14 |
| 1e-5 | 28 | -2.000000 | -0.000000 | 1.000000e+00 | 2.983341e-14 |
| 1e-6 | 28 | -2.000000 | -0.000000 | 1.000000e+00 | 2.904713e-14 |
| 1e-7 | 28 | -2.000000 | -0.000000 | 1.000000e+00 | 3.093441e-14 |
| 1e-8 | 28 | -2.000000 | -0.000000 | 1.000000e+00 | 3.391400e-14 |
| 1e-9 | 29 | -2.000000 | -0.000000 | 1.192093e-07 | 2.154176e-14 |
| 1e-10 | 60 | -2.000000 | -0.000000 | 9.765625e-04 | 4.466575e-14 |
| 1e-11 | 22 | -2.000000 | 0.000000 | 9.765625e-04 | 2.220446e-16 |
| 1e-12 | 1000 | -2.000000 | -0.000051 | 5.960464e-08 | 1.829515e-09 |
| 1e-13 | 26 | -1.999997 | -0.000129 | 7.105427e-15 | 1.285495e-05 |
| 1e-14 | 53 | -1.999985 | -0.007001 | 7.105427e-15 | 2.206370e-05 |
| 1e-15 | 9 | -1.988966 | -0.166179 | 7.105427e-15 | 1.735110e-02 |

Текст Программы

/\*lab4.h\*/

#pragma once

#include "stdafx.h"

typedef void(\*Gause)(std::vector<std::vector<double>> &A, std::vector<double> &F, std::vector<double> &X);

using namespace std;

//#define Anali

#define Test3

#pragma region Tests

В этом регионе все тесты на куча строк которые не стоит вставлять

#pragma endregion

class chm4

{

public:

chm4()

{

InItFunction(Func);

InItJacobi(AnalJ);

h = k = 0;

Move.resize(3);

Move[0] = D2D1::Point2F(500, 375);

Move[1] = D2D1::Point2F(-10, 10);

Move[2] = D2D1::Point2F(7.5, -7.5);

}

~chm4() {}

int maxiter, m, n;

vector<double> x, dx;

double e1, e2, h;

double NormF0, NormF;

double FMax;

vector<vector<double>> Jacobi;

vector<double> F;

vector<D2D1\_POINT\_2F> Target;

vector<D2D1\_POINT\_2F> Move;

double b;

int k;

void TwoCicle()

{

auto hinstLib = LoadLibrary(TEXT("LDU.dll"));

Gause gause = (Gause)GetProcAddress(hinstLib, "Gause");

k = 0;

int p = min(m, n);

F.resize(p);

dx.resize(n);

Jacobi.resize(p);

for (size\_t i = 0; i < p; i++)

{

Jacobi[i].resize(p);

F[i] = -Func[i](x);

}

NormF = NormF0 = Norm(F);

b = 1;

D2D1\_POINT\_2F o;

if (n == 2)

{

o = D2D1::Point2F(x[0], x[1]);

Target.push\_back(o);

}

log.open("log.txt", ios::app);

log << "------------------------------------------" << endl;

PrintLog();

while (CheckEnd())

{

Fbrake();

NormF = Norm(F);

gause(Jacobi, F, dx);

if (Norm(dx) == 0)

{

k++; break;

}

FindB();

Sum(x, dx, b);

if (n == 2)

{

o = D2D1::Point2F(x[0], x[1]);

Target.push\_back(o);

}

k++;

PrintLog();

}

log.close();

Shadow();

}

void Init()

{

ifstream in("info.txt");

in >> e1 >> e2 >> maxiter;

in >> n >> m;

in.close();

in.open("v0.txt");

x.resize(n);

for (size\_t i = 0; i < n; i++)

in >> x[i];

FMax = GetNorm(-13.0, -13.0);

if (!FMax || FMax == -1)

FMax = GetNorm(-13.0, 13.0);

}

D2D1\_POINT\_2F Result()

{

if (n != 2)

return D2D1::Point2F(0.0f, 0.0f);

else

return D2D1::Point2F(x[0], x[1]);

}

double GetNorm(double X, double Y)

{

double sum = 0;

vector<double> r(2);

r[0] = X;

r[1] = Y;

for (size\_t i = 0; i < m; i++)

{

double l = pow(Func[i](r), 2);

if (l < 1e-3)

return -1;

else

sum += l;

}

return sqrt(sum);

}

private:

function<double(vector<double>)> Func[3];

function<double(vector<double>)> AnalJ[3][2];

vector<int> index;

ofstream log;

struct br

{

double f;

int i;

};

double Norm(vector<double> x)

{

double sum = 0;

for (size\_t i = 0; i < x.size(); i++)

sum += pow(x[i], 2);

return sqrt(sum);

}

bool CheckEnd()

{

return k < maxiter && NormF / NormF0 >= e1 && b >= e2;

}

void FindB()

{

b = 2;

double norm = NormF;

vector<double> F(m);

vector<double> a;

do

{

b /= 2;

for (size\_t i = 0; i < m; i++)

{

a = x;

Sum(a, dx, b);

F[i] = Func[i](a);

}

norm = Norm(F);

} while (norm > NormF && b > e2);

}

void Sum(vector<double> &a, vector<double> &b, double c)

{

if (b.size() < a.size())

{

for (size\_t i = 0; i < index.size(); i++)

a[index[i]] += c \* b[i];

return;

}

for (size\_t i = 0; i < a.size(); i++)

a[i] += c \* b[i];

}

double ChisJ(int i, int j, vector<double> x)

{

double oo = 1e-2;

x[j] -= oo/2;

double X1 = Func[i](x);

x[j] += oo;

double X2 = Func[i](x);

return (X2 - X1) / oo;

}

void Fbrake()

{

index.resize(0);

if (n == m)

{

for (size\_t i = 0; i < m; i++)

{

for (size\_t j = 0; j < n; j++)

#ifdef Anali

Jacobi[i][j] = AnalJ[i][j](x);

#else

Jacobi[i][j] = ChisJ(i, j, x);

#endif // Anali

F[i] = -Func[i](x);

}

}

else

{

if (m > n)

{

vector<br> brf(m - n);

for (size\_t i = 0; i < m - n; i++)

brf[i].f = Func[i](x);

sort(brf.begin(), brf.end(), [](br i, br j) -> bool { return (i.f < j.f); });

for (size\_t i = m - n, i2 = i - 1, i3 = 0; i < m; i++, i3++)

{

if (abs(Func[i](x)) < abs(brf[i2].f))

{

F[i3] = -brf[i2].f;

for (size\_t j = 0; j < n; j++)

#ifdef Anali

Jacobi[i3][j] = AnalJ[brf[i2].i][j](x);

#else

Jacobi[i3][j] = ChisJ(brf[i2].i, j, x);

#endif

brf[i2].f = Func[i](x);

brf[i2].i = i;

sort(brf.begin(), brf.end(), [](br i, br j) -> bool { return (i.f < j.f); });

}

else

{

F[i3] = -Func[i](x);

for (size\_t j = 0; j < n; j++)

#ifdef Anali

Jacobi[i3][j] = AnalJ[i][j](x);

#else

Jacobi[i3][j] = ChisJ(i, j, x);

#endif

}

}

}

else

{

vector<br> brf(n - m);

for (size\_t i = 0; i < n - m; i++)

brf[i].f = MaxJacobi(i);

sort(brf.begin(), brf.end(), [](br i, br j) -> bool { return (i.f < j.f); });

for (size\_t i = n - m, i2 = i - 1, i3 = 0; i < n; i++, i3++)

{

double p = MaxJacobi(i);

double ggr = brf[i2].f;

if (p <= ggr)

{

F[i3] = -Func[i3](x);

index.push\_back(brf[i2].i);

for (size\_t j = 0; j < m; j++)

#ifdef Anali

Jacobi[j][i3] = AnalJ[j][brf[i2].i](x);

#else

Jacobi[j][i3] = ChisJ(j,brf[i2].i, x);

#endif

brf[i2].f = p;

brf[i2].i = i;

sort(brf.begin(), brf.end(), [](br i, br j) -> bool { return (i.f < j.f); });

}

else

{

F[i3] = -Func[i3](x);

index.push\_back(i);

for (size\_t j = 0; j < m; j++)

#ifdef Anali

Jacobi[j][i3] = AnalJ[j][i](x);

#else

Jacobi[j][i3] = ChisJ(j, i, x);

#endif // Anali

}

}

}

}

}

double MaxJacobi(int j)

{

if (!n)

return 0;

#ifdef Anali

double max = abs(AnalJ[0][j](x));

#else

double max = abs(ChisJ(0, j, x));

#endif // Anali

for (size\_t i = 1; i < m; i++)

{

#ifdef Anali

double p = abs(AnalJ[i][j](x));

#else

double p = abs(ChisJ(i, j, x));

#endif // Anali

if (max < p)

max = p;

}

return max;

}

void Shadow()

{

if (n != 2)

return;

else

Sum(dx, x, 1);

}

void PrintLog()

{

log << "k: ";

// auto qq = log.precision(15);

log << k << " " << fixed;

for (size\_t i = 0; i < x.size(); i++)

{

log << x[i]<<" ";

}

log<< scientific << "b: " << b << " F: "<<NormF << endl;

// log.precision(qq);

}

};

LDU.dll

// dllmain.cpp : Определяет точку входа для приложения DLL.

#include "stdafx.h"

#include "LDU.cpp"

Matrix::Gause::Gause<double> gause;

extern "C"

{

\_declspec(dllexport) void Gause(std::vector<std::vector<double>> &A, std::vector<double> &F, std::vector<double> &X)

{

gause.A = A;

gause.f = F;

Matrix::Gause::CountUpTrianMatrix(gause);

Matrix::Gause::GausBack(gause);

X = gause.x;

}

}

BOOL APIENTRY DllMain( HMODULE hModule,

DWORD ul\_reason\_for\_call,

LPVOID lpReserved

)

{

setlocale(LC\_ALL, "Russian");

switch (ul\_reason\_for\_call)

{

case DLL\_PROCESS\_ATTACH:

case DLL\_THREAD\_ATTACH:

case DLL\_THREAD\_DETACH:

case DLL\_PROCESS\_DETACH:

break;

}

return TRUE;

}

// LDU.cpp : Определяет экспортированные функции для приложения DLL.

//

#include "stdafx.h"

namespace Matrix

{

enum class Errors

{

noError,

ErrorOpenFile,

MatrixNotCount

};

enum class Format

{

LDU,

Profile

};

template<class T>

class Matrix

{

public:

Matrix() {}

~Matrix() {}

std::vector<T> al, au, di, f, x;

std::vector<size\_t> ia;

size\_t n, m;

Format format;

Errors error;

private:

};

namespace Gause

{

template<class T>

class Gause : Matrix<T>

{

public:

std::vector<std::vector<T>> A;

std::vector<double> f, x;

Gause()

{

this->format = Format::Profile;

error = Errors::noError;

}

~Gause() {}

Errors error;

private:

size\_t n; //Размерность матрицы

};

template<class T>

void GausBack(Gause<T> &matrix)

{

std::vector<T> &X = matrix.x;

std::vector<T> &F = matrix.f;

std::vector<std::vector<T>> &A = matrix.A;

auto n = F.size();

X.resize(n);

if (matrix.error != Errors::noError)

return;

for (int i = n - 1; i >= 0; i--)

{

double sum = 0;

for (size\_t j = i + 1; j < n; j++)

sum += A[i][j] \* X[j];

X[i] = (F[i] - sum) / A[i][i];

}

}

template<class T>

void CountUpTrianMatrix(Gause<T> &matrix)

{

if (matrix.error != Errors::noError)

return;

std::vector<std::vector<T>> &A = matrix.A;

std::vector<T> &F = matrix.f;

auto n = F.size();

for (size\_t i = 0; i < n; i++)

{

//Опредиление главного Элемента в столбце

T Max = A[i][i];

size\_t index = i;

for (size\_t j = i + 1; j < n; j++)

{

if (abs(A[j][i]) > abs(Max))

{

Max = A[j][i];

index = j;

}

}

//Проверка единственность

if (!Max)

{

matrix.error = Errors::MatrixNotCount;

return;

}

//Свап строки

if (i != index)

{

A[index].swap(A[i]);

T buf = F[i];

F[i] = F[index];

F[index] = buf;

}

//Постройка верхней треугольной матрицы

for (size\_t j = i + 1; j < n; j++)

{

if (!A[j][i])

continue;

T c = A[j][i] / Max;

A[j][i] = 0.;

for (size\_t l = i + 1; l < n; l++)

A[j][l] -= A[i][l] \* c;

F[j] -= F[i] \* c;

}

}

}

template<class T>

Gause<T> Create(std::vector<double> &al, std::vector<double> &au, std::vector<double> &di, std::vector<size\_t> &ia)

{

Gause<T> m;

std::vector<std::vector<T>> &A = m.A;

if (!di.size())

{

m.Error = Errors::MatrixNotCount;

return;

}

else

{

A.resize(m.n);

for (size\_t i = 0; i < m.n; A[i++].resize(m.n));

}

m.n = di.size();

auto n = m.n;

for (size\_t i = 0; i < n; i++)

A[i][i] = di[i];

for (size\_t i = 0; i < n; i++)

{

for (size\_t m = ia[i], j = i - ia[i + 1] + m; m < ia[i + 1]; m++, j++)

{

A[i][j] = al[m];

A[j][i] = au[m];

}

}

}

}

namespace LDU

{

template<class T>

class LDU: Matrix<T>

{

public:

LDU<T>()

{

this->format = Format::LDU;

}

~LDU<T>() {}

std::vector<T> al, au, di, f, x;

std::vector<size\_t> ia;

size\_t n, m;

Format format;

Errors error;

};

template<class T>

void SaveFile(LDU<T> &matrix)

{

}

template<class T>

void CountX(LDU<T> &matrix, std::vector<T> F)

{

size\_t n = matrix.n;

std::vector<T> &D = matrix.di;

std::vector<T> &U = matrix.au;

std::vector<T> &L = matrix.al;

std::vector<T> &X = matrix.x;

Errors Error = matrix.error;

std::vector<size\_t> &ia = matrix.ia;

if (Error != Errors::noError)

return;

//Поиск Y

std::vector<T> &Y = X;

for (size\_t i = 0; i < n; i++)

{

double sum = 0;

for (size\_t m = ia[i],

j = i - ia[i + 1] + m;

m < ia[i + 1]; m++)

sum += Y[j++] \* L[m];

Y[i] += F[i] - sum;

}

//Поиск Z

std::vector<T> &Z = X;

for (size\_t i = 0; i < n; i++)

Z[i] /= D[i];

//Поиск X

for (int i = n - 1; i >= 0; i--)

for (int j = ia[i + 1] - ia[i] - 1,

j1 = i - j - 1,

j2 = ia[i + 1] - j - 1;

j >= 0; j--)

X[j1++] -= X[i] \* U[j2++];

}

template<class T>

void CountLDU(LDU<T> &matrix)

{

size\_t n = matrix.n;

size\_t m = matrix.m;

auto &D = matrix.di;

auto &U = matrix.au;

auto &L = matrix.al;

auto &Error = matrix.error;

auto &ia = matrix.ia;

double CompareNumber;

if (std::is\_same<T, double>::value)

CompareNumber = pow(10, 15);

else

CompareNumber = pow(10, 7);

for (size\_t i = 0; i < n; i++)

{

double sum = 0;

for (size\_t j = ia[i], j1 = i - ia[i + 1] + j; j < ia[i + 1]; j++)

{

double sum1 = 0, sum2 = 0;

for (size\_t k = min(j - ia[i], ia[i - ia[i + 1] + j + 1] - ia[i - ia[i + 1] + j]),

i1 = j - k,

i2 = i - ia[i + 1] + j - k,

i3 = ia[i - ia[i + 1] + j + 1] - k;

k > 0; k--)

{

sum1 += L[i1] \* D[i2] \* U[i3];

sum2 += L[i3++] \* D[i2++] \* U[i1++];

}

L[j] = (L[j] - sum1) / D[j1];

U[j] = (U[j] - sum2) / D[j1];

sum += L[j] \* D[j1++] \* U[j];

}

//Если сделать проверка на inf после деления на 0, то можем получить не то что нам нужно

//Т.к. при разность (D[i] - sum) может получиться не 0, а число близкое к нему(разность нецелого типа)

//и в таком случае при делении числа на эту разность может получиться не inf.

if (abs((D[i] - sum)) < abs(D[i] / CompareNumber))

{

Error = Errors::MatrixNotCount;

return;

}

D[i] -= sum;

}

}

template <class T>

void CountLDU(LDU<T> &matrix, std::vector<T> &al, std::vector<T> &au, std::vector<T> &di, std::vector<size\_t> &ia)

{

size\_t n = matrix.n;

size\_t m = matrix.m;

auto &D = matrix.di;

auto &U = matrix.au;

auto &L = matrix.al;

auto &Error = matrix.error;

ia.\_Copy\_alloc(matrix.ia.get\_allocator());

double CompareNumber;

if (std::is\_same<T, double>::value)

CompareNumber = pow(10, 15);

else

CompareNumber = pow(10, 7);

for (size\_t i = 0; i < n; i++)

{

double sum = 0;

for (size\_t j = ia[i], j1 = i - ia[i + 1] + j; j < ia[i + 1]; j++)

{

double sum1 = 0, sum2 = 0;

for (size\_t k = min(j - ia[i], ia[i - ia[i + 1] + j + 1] - ia[i - ia[i + 1] + j]),

i1 = j - k,

i2 = i - ia[i + 1] + j - k,

i3 = ia[i - ia[i + 1] + j + 1] - k;

k > 0; k--)

{

sum1 += L[i1] \* D[i2] \* U[i3];

sum2 += L[i3++] \* D[i2++] \* U[i1++];

}

L[j] = (al[j] - sum1) / D[j1];

U[j] = (au[j] - sum2) / D[j1];

sum += L[j] \* D[j1++] \* U[j];

}

//Если сделать проверка на inf после деления на 0, то можем получить не то что нам нужно

//Т.к. при разность (D[i] - sum) может получиться не 0, а число близкое к нему(разность нецелого типа)

//и в таком случае при делении числа на эту разность может получиться не inf.

if (abs((D[i] - sum)) < abs(D[i] / CompareNumber))

{

Error = Errors::MatrixNotCount;

return;

}

D[i] = di[i] - sum;

}

}

template<class U>

bool Open(std::string path, std::vector<U> &a, size\_t len)

{

std::ifstream ia;

ia.exceptions(std::ifstream::failbit | std::ifstream::badbit);

try

{

ia.open(path);

a.resize(len);

for (size\_t i = 0; i < len; i++)

ia >> a[i];

ia.close();

}

catch (std::ifstream::failure e)

{

std::cout << "Error Open File: " << path << std::endl;

return false;

}

return true;

}

template<class T>

void OpenFile(LDU<T> &matrix, std::string path)

{

int error = 0;

size\_t n, m;

if (path.size())

path += "/";

std::vector<size\_t> info(2);

error +=

!Open(path + "info.txt", info, 2);

n = info[0], m = info[1];

matrix.x.resize(n);

error +=

!Open(path + "al.txt", matrix.al, m) +

!Open(path + "au.txt", matrix.au, m) +

!Open(path + "di.txt", matrix.di, n) +

!Open(path + "ia.txt", matrix.ia, n + 1) +

!Open(path + "F.txt", matrix.f, n);

if(error)

matrix.error = Errors::ErrorOpenFile;

}

template<class T>

LDU<T> Create(std::string path)

{

LDU<T> m;

OpenFile(m, path);

CountLDU(m);

return m;

}

template<class T>

LDU<T> Create(std::vector<double> &al, std::vector<double> &au, std::vector<double> &di, std::vector<size\_t> &ia)

{

LDU<T> m;

CountLDU(m, al, au, di, ia);

return m;

}

}

}

/\*basewin.h\*/

#pragma once

#include "stdafx.h"

template <class DERIVED\_TYPE>

class BaseWindow

{

public:

static LRESULT CALLBACK WindowProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam)

{

DERIVED\_TYPE \*pThis = NULL;

if (uMsg == WM\_NCCREATE)

{

CREATESTRUCT\* pCreate = (CREATESTRUCT\*)lParam;

pThis = (DERIVED\_TYPE\*)pCreate->lpCreateParams;

SetWindowLongPtr(hwnd, GWLP\_USERDATA, (LONG\_PTR)pThis);

pThis->m\_hwnd = hwnd;

}

else

{

pThis = (DERIVED\_TYPE\*)GetWindowLongPtr(hwnd, GWLP\_USERDATA);

}

if (pThis)

{

return pThis->HandleMessage(uMsg, wParam, lParam);

}

else

{

return DefWindowProc(hwnd, uMsg, wParam, lParam);

}

}

BaseWindow() : m\_hwnd(NULL) { }

BOOL Create(

PCWSTR lpWindowName,

DWORD dwStyle,

DWORD dwExStyle = 0,

int x = CW\_USEDEFAULT,

int y = CW\_USEDEFAULT,

int nWidth = 1000,

int nHeight = 750,

HWND hWndParent = 0,

HMENU hMenu = 0

)

{

WNDCLASS wc = { 0 };

wc.lpfnWndProc = DERIVED\_TYPE::WindowProc;

wc.hInstance = GetModuleHandle(NULL);

wc.lpszClassName = ClassName();

RegisterClass(&wc);

m\_hwnd = CreateWindowEx(

dwExStyle, ClassName(), lpWindowName, dwStyle, x, y,

nWidth, nHeight, hWndParent, hMenu, GetModuleHandle(NULL), this

);

return (m\_hwnd ? TRUE : FALSE);

}

HWND Window() const { return m\_hwnd; }

protected:

virtual PCWSTR ClassName() const = 0;

virtual LRESULT HandleMessage(UINT uMsg, WPARAM wParam, LPARAM lParam) = 0;

HWND m\_hwnd;

};

Project.cpp

// Project3.cpp : Определяет точку входа для приложения.

//

#include "stdafx.h"

#include "basewin.h"

#include "Lab4.h"

chm4 lab;

template <class T> void SafeRelease(T \*\*ppT)

{

if (\*ppT)

{

(\*ppT)->Release();

\*ppT = NULL;

}

}

class MainWindow : public BaseWindow<MainWindow>

{

ID2D1Factory \*pFactory;

IDWriteFactory \*pDWriteFactory;

ID2D1HwndRenderTarget \*pRenderTarget;

ID2D1BitmapRenderTarget \*pRenderBitmap;

ID2D1SolidColorBrush \*pBrush;

IDWriteTextFormat \*pTextFormat;

ID2D1Bitmap \*pBitmap;

HRESULT CreateGraphicsResources();

void DiscardGraphicsResources();

void OnPaint();

public:

MainWindow() : pFactory(NULL), pRenderTarget(NULL), pBrush(NULL)

{

}

PCWSTR ClassName() const { return L"4m4 window"; }

LRESULT HandleMessage(UINT uMsg, WPARAM wParam, LPARAM lParam);

};

// Recalculate drawing layout when the size of the window changes.

HRESULT MainWindow::CreateGraphicsResources()

{

HRESULT hr = S\_OK;

if (pRenderTarget == NULL)

{

RECT rc;

GetClientRect(m\_hwnd, &rc);

D2D1\_SIZE\_U size = D2D1::SizeU(rc.right, rc.bottom);

hr = pFactory->CreateHwndRenderTarget(

D2D1::RenderTargetProperties(),

D2D1::HwndRenderTargetProperties(m\_hwnd, size),

&pRenderTarget);

if (SUCCEEDED(hr))

{

const D2D1\_COLOR\_F color = D2D1::ColorF(1.0f, 1.0f, 0.0f);

hr = pRenderTarget->CreateSolidColorBrush(color, &pBrush);

}

static const WCHAR msc\_fontName[] = L"Verdana";

static const FLOAT msc\_fontSize = 24;

HRESULT hr = S\_OK;

if (SUCCEEDED(hr))

{

// Create a DirectWrite factory.

hr = DWriteCreateFactory(

DWRITE\_FACTORY\_TYPE\_SHARED,

\_\_uuidof(pDWriteFactory),

reinterpret\_cast<IUnknown \*\*>(&pDWriteFactory)

);

}

if (SUCCEEDED(hr))

{

// Create a DirectWrite text format object.

hr = pDWriteFactory->CreateTextFormat(

msc\_fontName,

NULL,

DWRITE\_FONT\_WEIGHT\_NORMAL,

DWRITE\_FONT\_STYLE\_NORMAL,

DWRITE\_FONT\_STRETCH\_NORMAL,

msc\_fontSize,

L"", //locale

&pTextFormat

);

}

hr = pRenderTarget->CreateCompatibleRenderTarget(D2D1::SizeF(1000.0f, 750.0f),&pRenderBitmap);

hr = pRenderTarget->CreateBitmap(D2D1::SizeU(1000,750), D2D1::BitmapProperties(), &pBitmap);

}

return hr;

}

void MainWindow::DiscardGraphicsResources()

{

SafeRelease(&pRenderTarget);

SafeRelease(&pBrush);

}

void MainWindow::OnPaint()

{

HRESULT hr = CreateGraphicsResources();

if (SUCCEEDED(hr))

{

pRenderBitmap->BeginDraw();

pRenderBitmap->Clear(D2D1::ColorF(D2D1::ColorF::White));

float f;

pBrush->SetColor(D2D1::ColorF(1.0f, 1.0f, 1.0f));

for (double i = lab.Move[1].x, i1 = 0; i < lab.Move[1].y; i += 0.02, i1++)

{

for (double j = lab.Move[2].x, j1 = 0; j > lab.Move[2].y; j -= 0.02, j1++)

{

f = lab.GetNorm(i, j) / lab.FMax;

if(f < 0)

{

pBrush->SetColor(D2D1::ColorF(0, 1, 0));

pRenderBitmap->FillRectangle(D2D1::RectF(i1 - 1, j1 - 1, i1 + 1, j1 + 1), pBrush);

}

else

{

// f \*= 3;

f = pow(f, 0.333);

f = (int)(f \* 20) / 20.0;

pBrush->SetColor(D2D1::ColorF(f, f, f));

pRenderBitmap->FillRectangle(D2D1::RectF(i1, j1, i1 + 1, j1 + 1), pBrush);

}

}

}

pBrush->SetColor(D2D1::ColorF(1, 0, 0));

pRenderBitmap->FillRectangle(D2D1::RectF(0, lab.Move[0].y - 1, 1000, lab.Move[0].y + 1), pBrush);

pRenderBitmap->FillRectangle(D2D1::RectF(lab.Move[0].x - 1, 0, lab.Move[0].x + 1, 750), pBrush);

pBrush->SetColor(D2D1::ColorF(0, 0, 1));

D2D1\_POINT\_2F o, o1;

o = lab.Target[0];

for (size\_t i = 1; i < lab.Target.size(); i++)

{

o1 = o;

o = lab.Target[i];

pRenderBitmap->DrawLine(D2D1::Point2F(o1.x \*50 + lab.Move[0].x, o1.y \* -50 + lab.Move[0].y),

D2D1::Point2F(o.x \* 50 + lab.Move[0].x, o.y \* -50 + lab.Move[0].y), pBrush, 2);

}

if (lab.dx.size() == 2 && lab.NormF / lab.NormF0 < lab.e1)

{

o.x = lab.dx[0];

o.y = lab.dx[1];

pBrush->SetColor(D2D1::ColorF(0, 0, 0.5));

pRenderBitmap->DrawLine(D2D1::Point2F(o1.x \* 50 + lab.Move[0].x, o1.y \* -50 + lab.Move[0].y),

D2D1::Point2F(o.x \* 50 + lab.Move[0].x, o.y \* -50 + lab.Move[0].y), pBrush, 1);

}

pBrush->SetColor(D2D1::ColorF(1, 0, 1));

pRenderBitmap->FillEllipse(D2D1::Ellipse(D2D1::Point2F(lab.x[0] \* 50 + lab.Move[0].x, lab.x[1] \* -50 + lab.Move[0].y), 4, 4), pBrush);

pBrush->SetColor(D2D1::ColorF(0, 0, 0));

wchar\_t Coor[100];

swprintf\_s(Coor, L"x: %lf\ny: %lf\nk: %i\n\0", lab.x[0], lab.x[1], lab.k - 1);

pRenderBitmap->DrawTextW(Coor, wcslen(Coor), pTextFormat, D2D1::RectF(10, 10, 300, 500), pBrush);

swprintf\_s(Coor, L"F: %e\nB: %e\0", lab.NormF, lab.b);

pRenderBitmap->DrawTextW(Coor, wcslen(Coor), pTextFormat, D2D1::RectF(10, 650, 300, 740), pBrush);

//pRenderTarget->FillRectangle(rectangle, pBrush);

pRenderBitmap->GetBitmap(&pBitmap);

pRenderBitmap->EndDraw();

PAINTSTRUCT ps;

BeginPaint(m\_hwnd, &ps);

pRenderTarget->BeginDraw();

pRenderTarget->DrawBitmap(pBitmap, D2D1::RectF(0, 0, 1000, 750), 1, D2D1\_BITMAP\_INTERPOLATION\_MODE\_NEAREST\_NEIGHBOR, D2D1::RectF(0, 0, 1000, 750));

hr = pRenderTarget->EndDraw();

if (FAILED(hr) || hr == D2DERR\_RECREATE\_TARGET)

{

DiscardGraphicsResources();

}

EndPaint(m\_hwnd, &ps);

}

}

int WINAPI wWinMain(HINSTANCE hInstance, HINSTANCE, PWSTR, int nCmdShow)

{

MainWindow win;

lab.Init();

lab.TwoCicle();

if (!win.Create(L"4m4", WS\_OVERLAPPED | WS\_CAPTION | WS\_SYSMENU ))

return 0;

ShowWindow(win.Window(), nCmdShow);

// Run the message loop.

MSG msg = { };

while (GetMessage(&msg, NULL, 0, 0))

{

TranslateMessage(&msg);

DispatchMessage(&msg);

}

return 0;

}

void MoveSkipY(int o, double l)

{

lab.Move[0].y -= o;

lab.Move[2].x -= l;

lab.Move[2].y -= l;

}

void MoveSkipX(int o, double l)

{

lab.Move[0].x -= o;

lab.Move[1].x += l;

lab.Move[1].y += l;

}

LRESULT MainWindow::HandleMessage(UINT uMsg, WPARAM wParam, LPARAM lParam)

{

switch (uMsg)

{

case WM\_CREATE:

if (FAILED(D2D1CreateFactory(

D2D1\_FACTORY\_TYPE\_SINGLE\_THREADED, &pFactory)))

{

return -1; // Fail CreateWindowEx.

}

return 0;

case WM\_DESTROY:

DiscardGraphicsResources();

SafeRelease(&pFactory);

PostQuitMessage(0);

return 0;

case WM\_PAINT:

OnPaint();

return 0;

case WM\_KEYDOWN:

switch (wParam)

{

case VK\_UP:

MoveSkipY(-50, -1);

InvalidateRect(m\_hwnd, NULL, FALSE);

break;

case VK\_DOWN:

MoveSkipY(50, 1);

InvalidateRect(m\_hwnd, NULL, FALSE);

break;

case VK\_LEFT:

MoveSkipX(-50, -1);

InvalidateRect(m\_hwnd, NULL, FALSE);

break;

case VK\_RIGHT:

MoveSkipX(50, 1);

InvalidateRect(m\_hwnd, NULL, FALSE);

break;

default:

break;

}

return 0;

}

return DefWindowProc(m\_hwnd, uMsg, wParam, lParam);

}