1. **ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ**
   1. **Объект испытаний**

Наименование темы разработки – «Математическое описание вращений».

Наименование испытуемой программы – «QR-разложение.exe».

Программа предназначена к применению любыми пользователями ПК в целях qr-разложения методом вращений.

* 1. **Цель испытаний**

Главной целью испытаний является проверка соответствия основных характеристик программы функциональным и иным видам требований, изложенным в программном документе «Техническое задание».

* 1. **Требования к программе**

Требования к программе изложены в п. «Требования к функциональным характеристикам» Технического задания.

* 1. **Требования к программной документации**

Состав программной документации должен включать в себя:

1) техническое задание;

2) описание программы;

3) программу и методики испытаний;

4) пояснительную записку;

5) ведомость эксплуатационных документов;

6) описание применения;

7) руководство программиста;

8) руководство оператора.

* 1. **Средства и порядок испытаний**

Характеристики используемого технического средства:

• ОЗУ более 32 Мбайт

• 8 МБ видеопамяти и выше

• наличие свободного места на жестком диске более 30 Мбайт.

Перечень проверок, проводимых во время испытаний:

а) проверка комплектности программной документации;

б) проверка комплектности и состава технических и программных средств;

в) проверка соответствия технических характеристик программы;

г) проверка степени выполнения требований функционального назначения программы.

* 1. **Методы испытаний**

Проверка комплектности программной документации на программное изделие производится визуально представителем службы, ответственной за эксплуатацию. В ходе проверки сопоставляется состав и комплектность программной документации, представленной Разработчиком, с перечнем программной документации, приведенным в п. «Состав программной документации, предъявляемой на испытания» настоящего документа. Проверка считается завершенной в случае соответствия состава и комплектности программной документации, представленной Разработчиком, перечню программной документации, приведенному в указанном выше пункте. По результатам проведения проверки, представитель службы, ответственной за эксплуатацию вносит запись в Протокол испытаний – “Комплектность программной документации соответствует (не соответствует) требованиям п. «Состав программной документации, предъявляемой на испытания»” настоящего документа.

Проверка комплектности и состава технических и программных средств производится визуально представителем службы, ответственной за эксплуатацию. В ходе проверки сопоставляется состав и комплектность технических и программных средств, представленных Разработчиком, с перечнем технических и программных средств, приведенным в п. «Технические средства, используемые во время испытаний» и п. «Программные средства, используемые во время испытаний» настоящего документа. Комплектность программных средств проводится также визуально. Загрузилась операционная система, высветился логотип, версия - соответствует/не соответствует заявленной в Техническом задании и т.д. Проверка считается завершенной в случае соответствия состава и комплектности технических и программных средств, представленных Разработчиком, с перечнем технических и программных средств, приведенных в пп. «Технические средства, используемые во время испытаний» и «Программные средства, используемые во время испытаний» настоящего документа. По результатам проведения проверки представитель службы, ответственной за эксплуатацию, вносит запись в Протокол испытаний - «Комплектность технических и программных средств соответствует (не соответствует) требованиям пп. «Технические средства, используемые во время испытаний» и «Программные средства, используемые во время испытаний» настоящего документа».

Проверка работоспособности программы выполняется согласно п. «Проверка работоспособности программы» Руководства системного программиста. Проверка считается завершенной в случае соответствия состава и последовательности действий, при выполнении данной проверки, указанному выше подразделу Руководства системного программиста. По результатам проведения проверки представитель службы, ответственной за эксплуатацию вносит запись в Протокол испытаний - «п. “Проверка работоспособности программы“ выполнена».

* 1. **Проверка работоспособности программы**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назначение проверки | Номер теста | Значения входных переменных | Ожидаемый результат работы программы | Результат тестирования |
| Проверка работоспособности в рабочем диапазоне данных | 1 | 1, 1 | Программа работает корректно.  Расчет происходит. | Рисунок 1 Работа программы с числами внутри рабочего диапазона |
| 2 | чтение | Программа работает корректно.  Чтение из файла происходит. | Рисунок 2 Чтение из файла |
| 3 | Очистка,  чтение | Программа работает корректно.  Чтение из не файла происходит. | Рисунок 3 Очистка файла и последующее чтение |
| Проверка реакции программы на ввод данных вне рабочего диапазона | 4 | 77 | Программа работает корректно.  Присвоение переменной значения не происходит. | Рисунок 4 Работа программы с числами вне рабочего диапазона |
| Проверка реакции на ошибку типа входных данных | 5 | f | Программа работает корректно.  Присвоение переменной значения не происходит. | Рисунок 5 Работа программы с неверными типами данных |

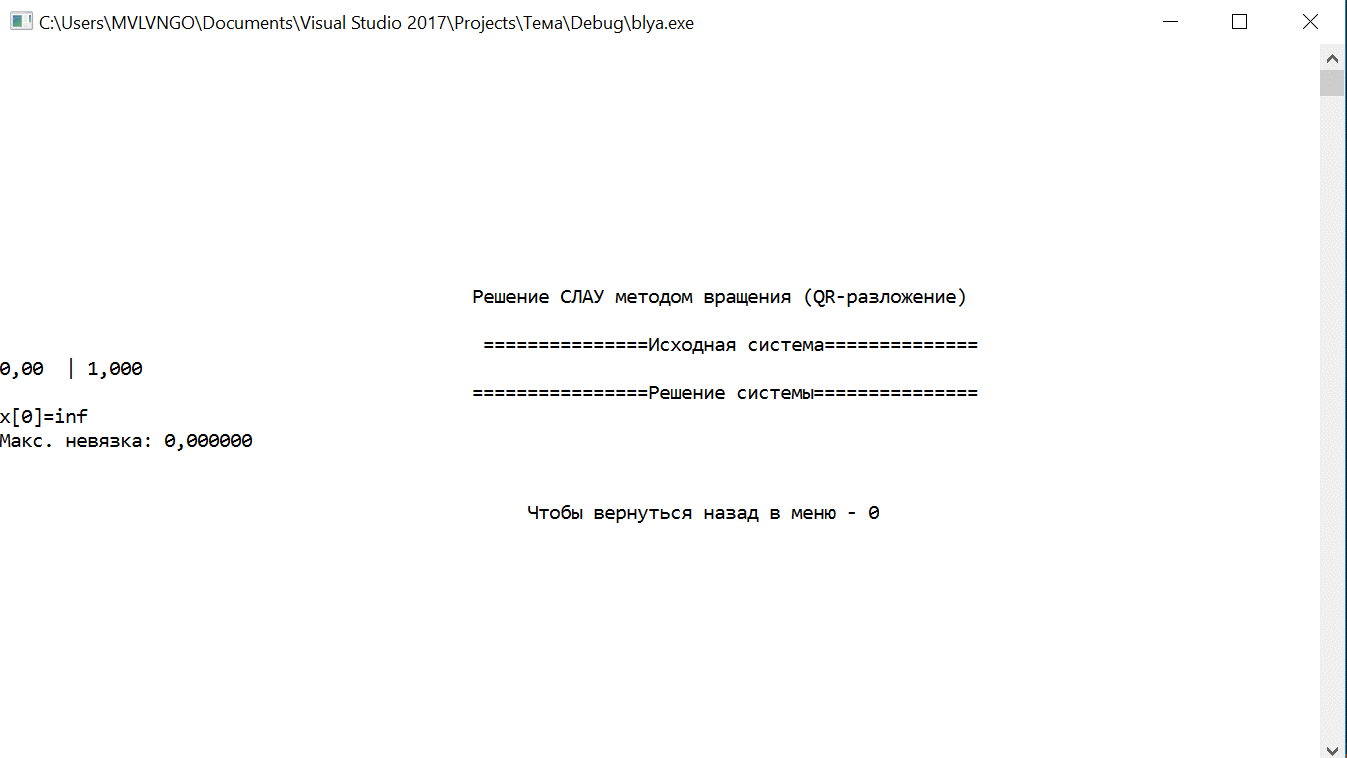


Рисунок 1 Работа программы с числами внутри рабочего диапазона

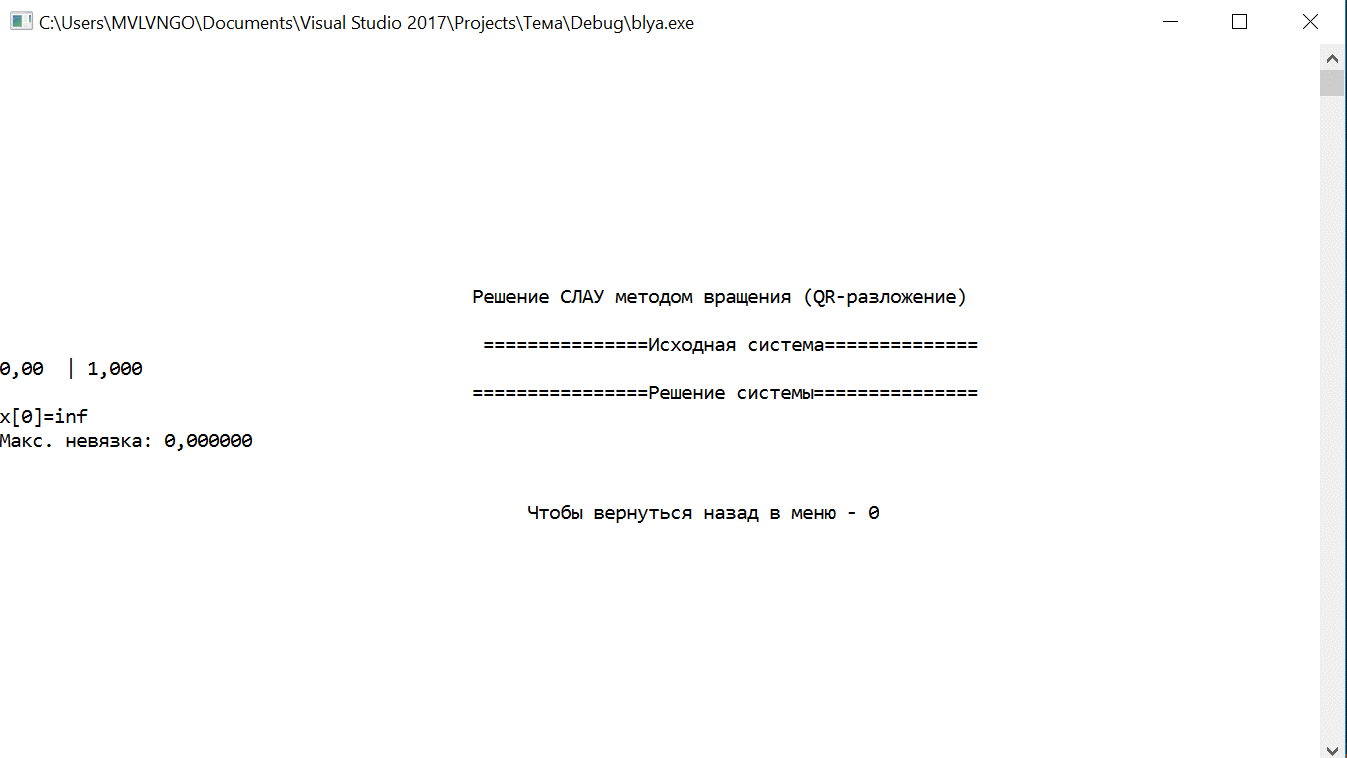


Рисунок 2 Чтение из файла

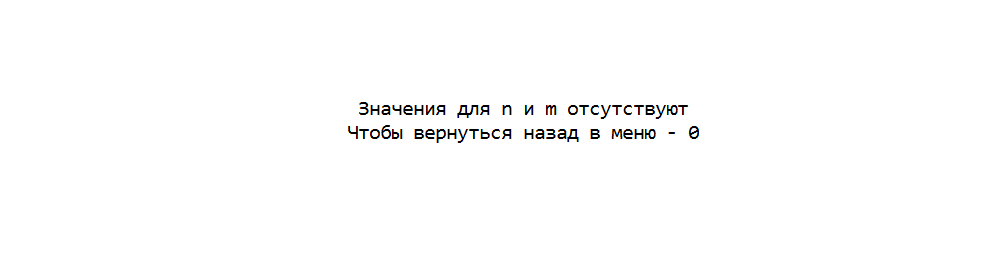


Рисунок 3 Очистка файла и последующее чтение

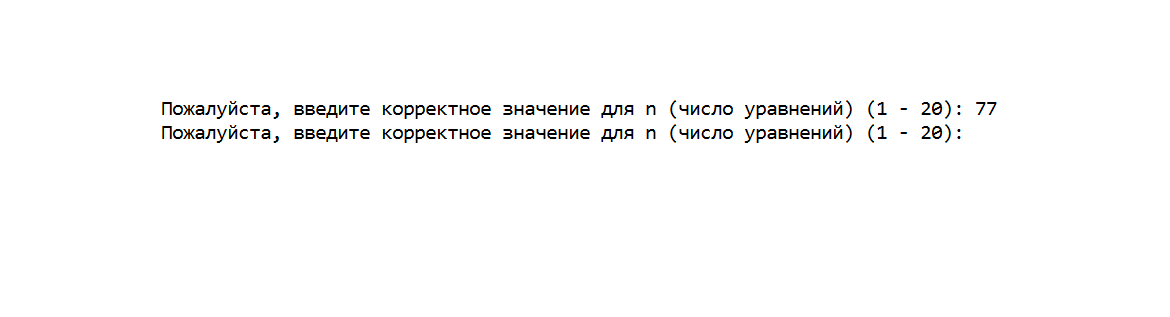


Рисунок 4 Работа программы с числами вне рабочего диапазона

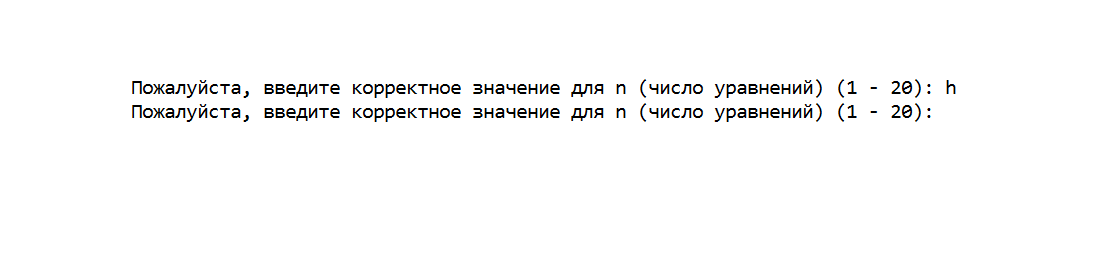


Рисунок 5 Работа программы с неверными типами данных

* 1. **Текст программы**

#include <stdio.h>

#include <conio.h>

#include <math.h>

#include <windows.h>

#include <clocale>

#include <iostream>

#include <iomanip>

#include <fstream>

using namespace std;

#define \_N 20

#define \_M 20

int back = 0;

int high;

int low;

int number1;

int number2;

const string filename = "счет.txt";

//работа с файлами

ofstream fout;

ifstream fin;

//прототипы функций общие

void printSpace(int n);

void instructions();

int askNumber(int low, int high);

int askNumber1(int low, int high);

int math(int low, int high);

int math1();

void clearFile();

class matrix {

public:

matrix(int n, int m);

~matrix();

void print\_slau();

void print\_x();

void qr();

void seta(int i, int j, float value);

void setb(int i, float value);

private:

float \*\*a;

float \*b;

float \*x;

int n, m;

};

// ------------------------------------------------------

// Конструктор

// ------------------------------------------------------

matrix::matrix(int N, int M) {

int i;

n = N; m = M;

a = new float\*[n];

for (i = 0; i<n; i++) {

a[i] = new float[m];

for (int j = 0; j<m; j++) a[i][j] = 0;

}

b = new float[n];

x = new float[n];

for (i = 0; i<n; i++) { b[i] = 0; x[i] = 0; }

}

// ------------------------------------------------------

// Деструктор

// ------------------------------------------------------

matrix::~matrix() {

for (int i = 0; i<n; i++) delete a[i];

delete[] a;

delete b; delete x;

}

// ------------------------------------------------------

// Вывод СЛАУ на экран

// ------------------------------------------------------

void matrix::print\_slau() {

for (int i = 0; i<n; i++) {

for (int j = 0; j<m; j++) printf("%.2f ", a[i][j]);

printf("| %.3f ", b[i]);

printf("\n");

}

}

// ------------------------------------------------------

// Печать решения и невязки решения

// ------------------------------------------------------

void matrix::print\_x() {

float max = 0, h;

for (int i = 0; i<n; i++) {

h = 0;

for (int j = 0; j<n; j++) h = h + x[j] \* a[i][j];

if (max<fabs(b[i] - h)) max = fabs(b[i] - h);

printf("x[%i]=%.3f ", i, x[i]);

if (i == 5) printf("\n");

}

printf("\nМакс. невязка: %f\n", max);

}

// ------------------------------------------------------

// Установить значение матрицы a[][]

// ------------------------------------------------------

void matrix::seta(int i, int j, float value) {

a[i][j] = value;

}

// ------------------------------------------------------

// Установить значение вектора b[]

// ------------------------------------------------------

void matrix::setb(int i, float value) {

b[i] = value;

}

// ------------------------------------------------------

// QR-алгоритм

// ------------------------------------------------------

void matrix::qr() {

int l, k;

float c[\_N][\_M];

float s[\_N][\_M];

float akk, akl, alk, all, bk, bl;

// Прямой ход

for (k = 0; k<n - 1; k++) { // "Большой" шаг (исключение переменных)

for (l = k + 1; l<n; l++) { // "Малый" шаг

c[k][l] = a[k][k] / (sqrt(a[k][k] \* a[k][k] + a[l][k] \* a[l][k]));

s[k][l] = a[l][k] / (sqrt(a[k][k] \* a[k][k] + a[l][k] \* a[l][k]));

// Умножение матрицы a[][] на T[k][l]

akk = a[k][k]; alk = a[l][k]; akl = a[k][l]; all = a[l][l];

a[k][k] = akk\*c[k][l] + alk\*s[k][l];

a[k][l] = akl\*c[k][l] + all\*s[k][l];

a[l][k] = -akk\*s[k][l] + alk\*c[k][l];

a[l][l] = -akl\*s[k][l] + all\*c[k][l];

// Вектор свободных членов умножается на T[k][l]

bk = b[k]; bl = b[l];

b[k] = bk\*c[k][l] + bl\*s[k][l];

b[l] = -bk\*s[k][l] + bl\*c[k][l];

}

}

// Теперь матрица a[][] -- верхняя диагональная.

// Обратный ход

float h;

x[n - 1] = b[n - 1] / a[n - 1][n - 1];

for (l = (n - 1); l >= 1; l--) {

h = b[l - 1];

for (k = (l + 1); k <= n; k++) h = h - x[k - 1] \* a[l - 1][k - 1];

x[l - 1] = h / a[l - 1][l - 1];

}

}

void printSpace(int n) {

system("cls");

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

cout << endl;

}

void instructions()

{

printSpace(10);

cout << setw(114) << "QR-разложение матрицы — представление матрицы в виде произведения унитарной (или ортогональной матрицы) и \n" << setw(114) <<"верхнетреугольной матрицы. QR-разложение является основой одного из методов поиска собственных векторов и \n" << setw(80) << "чисел матрицы — QR-алгоритма.\n\n";

do {

cout << setw(79) << "Чтобы вернуться назад в меню - 0";

back = \_getch();

printSpace(10);

} while (back != 48);

}

int math(int low, int high)

{

int n, m;

int i;

printSpace(10);

int choose = askNumber(1,20);

n = choose;

number1 = n;

printSpace(10);

int choose1 = askNumber1(1, 20);

m = choose1;

number2 = m;

fout.open(filename);

fout << number1 << " " << number2;

fout.close();

printSpace(10);

matrix A(n, m);

for (i = 0; i<n; i++) for (int j = 0; j<m; j++) A.seta(i, j, i + j);

for (i = 0; i<n; i++) A.setb(i, 1);

cout << setw(90) << "Решение СЛАУ методом вращения (QR-разложение)\n\n";

cout << setw(90) << "===============Исходная система==============\n";

A.print\_slau();

cout << setw(90) << "================Решение системы===============\n";

A.qr();

A.print\_x();

cout << setw(90) << "\n\n";

do {

cout << setw(80) << "Чтобы вернуться назад в меню - 0";

back = \_getch();

printSpace(10);

} while (back != 48);

return 0;

}

int math1()

{

fin.open(filename);

fin >> number1;

fin >> number2;

int n = number1;

int m = number2;

fin.close();

if (n == 0)

{

printSpace(10);

cout << setw(81) << "Значения для n и m отсутствуют \n";

}

else

{

int i;

printSpace(10);

matrix A(n, m);

for (i = 0; i < n; i++) for (int j = 0; j < m; j++) A.seta(i, j, i + j);

for (i = 0; i < n; i++) A.setb(i, 1);

cout << setw(90) << "Решение СЛАУ методом вращения (QR-разложение)\n\n";

cout << setw(90) << "===============Исходная система==============\n";

A.print\_slau();

cout << setw(90) << "================Решение системы===============\n";

A.qr();

A.print\_x();

cout << setw(90) << "\n\n";

}

do {

cout << setw(80) << "Чтобы вернуться назад в меню - 0";

back = \_getch();

printSpace(10);

} while (back != 48);

return 0;

}

// ------------------------------------------------------

// Main

// ------------------------------------------------------

int main() {

system("color F0");

setlocale(LC\_ALL, "Russian");

printSpace(10);

do

{

stop:

cout << setw(95) << "!!!Математическое описание вращений. Добро пожаловать!!!\n\n";

cout << setw(82) << "Чтобы продолжить - нажмите enter\n";

cout << setw(78) << "Чтобы выйти - нажмите esq";

int action = \_getch();

printSpace(10);

if (action == 13)

{

do

{

cout << setw(81) << "Чтобы получить справку - нажмите i \n";

cout << setw(80) << "Чтобы начать расчеты - нажмите r \n";

cout << setw(81) << "Чтобы считать из файла - нажмите f \n";

cout << setw(81) << "Чтобы очистить счет - нажмите b \n\n";

cout << setw(86) << "Чтобы вернуться назад в главное меню - нажмите 0";

int choose = \_getch();

switch (choose)

{

case 98: {

clearFile();

break;

}

case 105: {

instructions();

break;

}

case 114: {

math(1,20);

break;

}

case 48:

{

printSpace(10);

goto stop;

}

case 102:

{

math1();

break;

}

default: {

printSpace(10);

cout << setw(67) << "Вы ошиблись!";

Sleep(2000);

printSpace(10);

}

}

} while (1);

}

else if (action == 27)

{

printSpace(10);

cout << setw(72) << "До новых встреч!!!";

Sleep(2000);

return 0;

}

else {

printSpace(10);

cout << setw(67) << "Вы ошиблись!";

Sleep(2000);

printSpace(10);

}

} while (1);

}

bool checkIsDigit(char \*str)

{

while (\*str != 0)

{

if (!isdigit(\*str++))

return false;

}

return true;

}

int askNumber(int low, int high)

{

int number;

do {

cout << setw(86) << "Пожалуйста, введите корректное значение для n (число уравнений)" << " (" << low << " - " << high << "): ";

char num\_ch[100];

cin >> num\_ch;

if (checkIsDigit(num\_ch))

number = atoi(num\_ch);

else number = -1;

} while (number > high || number < low);

return number;

}

int askNumber1(int low, int high)

{

int number;

do {

cout << setw(86) << "Пожалуйста, введите корректное значение для m (число неизвестных)" << " (" << low << " - " << high << "): ";

char num\_ch[100];

cin >> num\_ch;

if (checkIsDigit(num\_ch))

number = atoi(num\_ch);

else number = -1;

} while (number > high || number < low);

return number;

}

// Очистка файла

void clearFile()

{

system("cls");

for (int i = 0; i < 10; i++)

cout << endl;

number1 = 0;

number2 = 0;

fout.open(filename);

fout << number1;

fout << number2;

fout.close();

}