Министерство образования и науки РФ

Нижегородский государственный университет имени Н.И. Лобачевского (Национальный исследовательский университет)

Институт Информационных Технологий Математики и Механики

Отчет на тему:

«Работа с мономами и полиномами»

Выполнил:   
 студент группы 0823-3 Панков Максим

Проверил:   
 Козинов Евгений Александрович

Нижний Новгород

2016

**Содержание**

[**Введение**](#Введение)**.3**

[**Постановка учебно-практической задачи**](#Постанов_учебно_практической_задачи)4

[**Руководство пользователя**](#Руководство_пользователя) **5**

[**Руководство программиста**](#Руководство_программиста)**7**

[**Заключение**](#Заключение)**9**

[**Список литературы**](#Список_литературы)**10**

[**Приложение**](#приложение)**11**

**Введение**

Моном - это произведение некоторого числового множителя с одной или несколькими переменными, которые взяты в неотрицательной степени. Полином - сумма мономов. Умение работать с ними очень важно даже в наше время, т.к они являются фундаментом “классической алгебры”. Данное приложение позволяет составлять из введенных ранее мономов составлять полиномы, выполнять над ними операции сложения, вычитания и умножения, а также тестировать код программы при помощи фреймворка Google Test.

**Постановка учебно-практической задачи**

Необходимо создать программу, которая выполняет следующие действия:

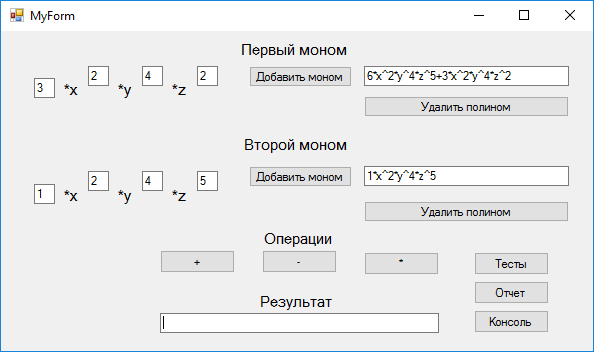
1. Составляет мономы из введенных пользователем четырех чисел (коэффициент монома и степени перед переменными x,y,z);
2. Составляет полиномы из мономов и выводит их на экран;
3. Выполняет над полиномами и мономами операции сложения, вычитания и умножения;
4. Все описанные выше действия выполняются через графическое приложение Windows Form;
5. Тестирует код программы при помощи фреймворка Google Test.

**Руководство пользователя:**

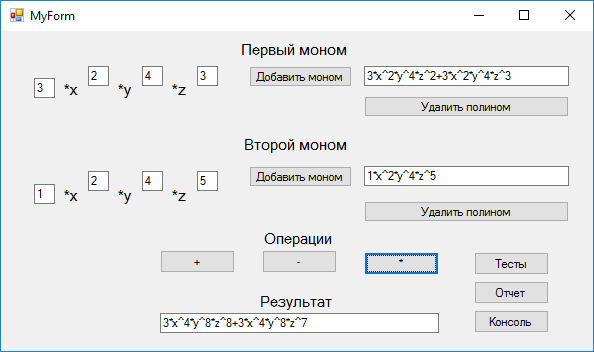
Программа предлагает набор возможностей:

1.Ввод мономов/полиномов:

Для этого необходимо запустить программу WinForm.exe и ввести желаемый моном при помощи соответствующих текстовых окон. Далее, необходимо нажать на кнопку «Добавить моном». Если пользователь хочет добавить моном (т. е создать полином), то ему необходимо повторить выше описанные действия. При необходимости, можно удалить моном/полином, нажав на кнопку «Удалить полином».

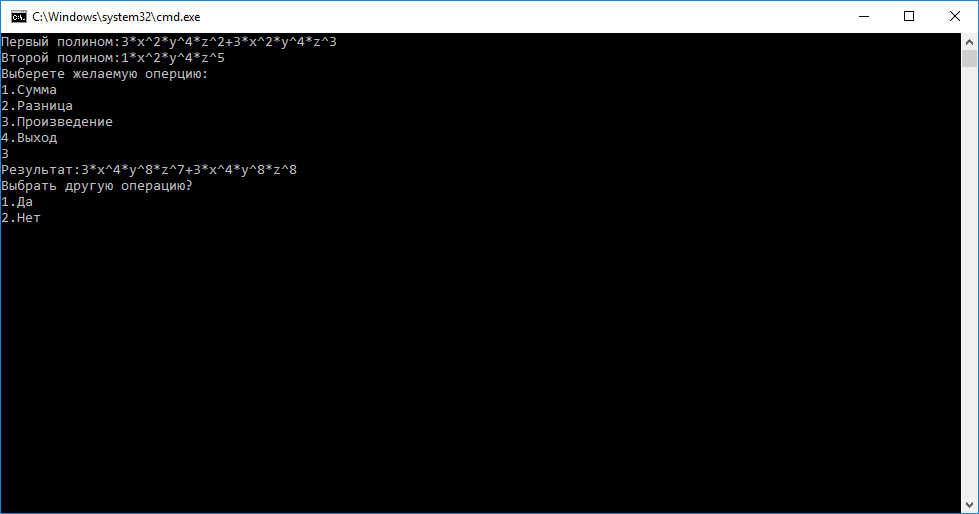


2.Арифметические операции с мономами/полиномами

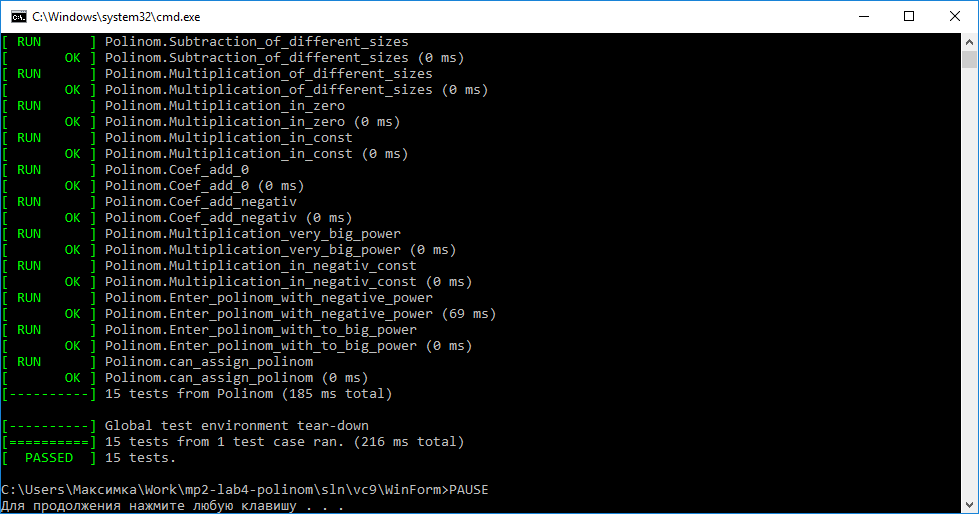
После того, как было введены необходимые данные, пользователь может выбрать необходимую арифметическую операцию, нажав на соответствующую кнопку. 3.Тестирование приложения/просмотр отчета/использование консоли:

Для этого необходимо нажать на соответствующую кнопку в графическом приложении. При этом на экране появится консоль с тестами/файл с данным отчетом/ консольный аналог программы.

Консольное приложение:



Консоль с тестами(как видим, все тесты пройдены):



**Руководство программиста**

***Описание структуры данных:***

Программа работает при помощи односвязного линейного списка мономов(мономы, в свою очередь, хранятся в виде упорядоченного массива чисел).

***Описание алгоритмов:***

[***1.Класс моном:***](#мономш)

Класс “Monom” в программе имеет следующие **методы**:

1. Monom(double coef = 0.0,int st = 0,Monom \*\_next = NULL) инициализирующий конструктор;
2. Monom(const Monom &m)-конструктор копирования;
3. Monom\* GetNext()-установка указателя на моном
4. void SetNext(Monom \*\_next)-получения указателя на моном;
5. Monom& operator=(const Monom &m)-присваивание монома(присваивает коэффициент и степени одного монома к другому моному);
6. bool operator==(const Monom &m) const-сравнение мономов(сравнивает коэффициенты и степени мономов);
7. void SetKoef(double coef)-установка численного коэффициента монома;
8. double GetKoef()-получение численного коэффициента монома;
9. void SetPower(int st)-установка степени монома;
10. int GetPower()-получение степени монома.

[***2.Класс полином:***](#полиномш)

Класс “Polinom” в программе имеет следующие **методы**: Polinom()-инициализирующий конструктор;

1. ~Polinom()-деструктор;
2. Polinom(const Polinom &pol)-конструктор копирования;
3. void AddElem(double coef, int st)-функция добавления монома в полином: функция просматривает, есть ли среди мономов в полиноме подобный к вводимому моному(с одинаковыми коэффициентами). Если есть, то эти 2 монома суммируются. Если таких мономов нет, то вводимый моном устанавливается в голову полинома;
4. void DelElem()-функция очистки полинома: функция проходит по полиному и поочередно удаляет мономы в нем;
5. Polinom& operator+(const Polinom &pol)const-перегрузка оператора “+”: функция поочередно добавляет мономы из одного в полинома в другой(при помощи функции AddElem);
6. Polinom&operator-(const Polinom &pol)const- перегрузка оператора “-”: функция поочередно добавляет мономы из одного в полинома в другой, умножая при этом коэффициенты вычитаемого полинома (при помощи функции AddElem);
7. Polinom& operator\*(const Polinom &pol) const - перегрузка оператора “\*”:в начале, функция поочередно “расшифровывает” зашифрованные ранее входящие степени мономов в полиноме(см. перегрузку ввода/вывода).Далее, при помощи AddElem, функция суммирует степени мономов и умножает степени полиномов;
8. Polinom& operator=(const Polinom &pol) - перегрузка оператора “=”:функция поочередно присваивает мономы одного полинома мономы другого полинома;
9. bool operator==(const Polinom &pol) const – перегрузка сравнения: функция поочередно сравнивает коэффициенты и степени мономов и полиномов. Если они все равны, то функция выдает “true”,а иначе “false”;
10. friend istream& operator>>(istream &s, Polinom &pol)- перегрузка ввода: функция принимает в себя введение пользователем данные, а именно коэффициенты и степени мономов. Далее она шифрует степени по принципу st=x\*100+y\*10+z (где x,y,z-степени монома). После этого эти данные переходят в AddElem;
11. friend ostream& operator<<(ostream &f, Polinom &pol) – перегрузка вывода: поочередно выводит мономы и результаты операций над мономами, расшифровывая при этом степени и учитывая такие ситуации, как пустой полином, нулевой коэффициент и степень и т.д;

За взаимодействие с пользователем отвечают консольное приложение и приложение Windows Form.Так же в программе присутствуют тесты фреймворка Google Test.

*Описание структуры программного комплекса*

[**Monom.h**](#мономш)и [**Monom.cpp**](#мономссп) **–** заголовочный и .cpp файлы, в которых находится класс Monom. [**Polinom.h**](#полиномш)и [**Polinom.cpp**](#полиномспп) **–** заголовочный и .cpp файлы, в которых находится класс Polinom. [**main-polinom.cpp**](#маинполином) **–** файл, в котором находится функция main, реализующая взаимодействие с пользователем при помощи консоли. [**test\_polinom.cpp**](#тест) **-** файл, в котором находится реализация фреймворка Google Test. [**MyForm.h**](#майформш)и [**MyForm.cpp**](#майформспп) **–** заголовочный и .cpp файлы, реализующая взаимодействие с пользователем при помощи графического интерфейса Windows Form.

**Заключение**

В результате выполнения данной лабораторной работы было создано приложение, позволяющее пользователю создавать мономы и полиномы, проводить над ними простейшие арифметические операции ,а также тестировать код программы. В программе имеются как и консольное приложение, так и приложение Windows Form. Программа была протестирована на примере, некоторые операции были продемонстрированы выше.

**Список используемой литературы:**

* Гергель В.П. «Рабочие материалы к учебному курсу «Методы программирования» ННГУ, 2002
* Б. Страуструп Язык программирования С++. Специальное издание. Пер. с англ. – М.: ООО «Бином-Пресс», 2008 г. – 1104 с.: ил.
* Сайт <https://habrahabr.ru>.
* Сайт [www.cyberforum.ru](file:///C:\Users\Максимка\Desktop\www.cyberforum.ru)
* Шилдт, Герберт Полный справочник по C++, 4-е издание. : Пер. с англ. : -M. : Издательский дом "Вильямс" , 2004.-800 с.

***Приложение:***

*Monom.h*

#pragma once

#include <iostream>

class Monom

{

public:

int pow;

double koef;

Monom \*next;

Monom(double coef = 0.0, int st = 0, Monom \*\_next = NULL);

Monom(const Monom &m);

Monom\* GetNext();

void SetNext(Monom \*\_next);

Monom& operator=(const Monom &m);

bool operator==(const Monom &m) const;

void SetKoef(double coef);

double GetKoef();

void SetPower(int st);

int GetPower();

};

*Monom.cpp*

#include "monom.h"

#include <iostream>

Monom::Monom(double coef, int st, Monom \*\_next)

{

SetKoef(coef);

SetPower(st);

SetNext(\_next);

}

Monom::Monom(const Monom &m)

{

SetKoef(m.koef);

SetPower(m.pow);

SetNext(m.next);

}

Monom& Monom::operator=(const Monom &m)

{

SetKoef(m.koef);

SetPower(m.pow);

return \*this;

}

bool Monom::operator==(const Monom &m) const

{

if ((koef == m.koef) && (pow == m.pow))

{

return true;

}

else

{

return false;

}

}

void Monom::SetKoef(double coef)

{

koef = coef;

}

void Monom::SetPower(int st)

{

pow = st;

}

void Monom::SetNext(Monom \*\_next)

{

next = \_next;

}

double Monom::GetKoef()

{

return koef;

}

int Monom::GetPower()

{

return pow;

}

Monom\* Monom::GetNext()

{

return next;

}

*Polinom.h*

#pragma once

#include "monom.h"

#include <locale>

using namespace std;

class Polinom

{

public:

Monom \*pHead;

Monom\* getHead();

Polinom();

~Polinom();

Polinom(const Polinom &pol);

void AddElem(double coef, int st);

void DelElem();

Polinom& operator+(const Polinom &pol) const;

Polinom& operator-(const Polinom &pol) const;

Polinom& operator\*(const Polinom &pol) const;

Polinom& operator=(const Polinom &pol);

bool operator==(const Polinom &pol) const;

friend istream& operator>>(istream &s, Polinom &pol);

friend ostream& operator<<(ostream &f, Polinom &pol);

};

*Polinom.cpp*

#include "polinom.h"

using namespace std;

Monom\* Polinom::getHead()

{

return pHead;

}

Polinom::Polinom()

{

pHead = NULL;

}

Polinom::Polinom(const Polinom &pol)

{

Monom \*var = new Monom;

pHead = new Monom;

pHead->SetNext(NULL);

var = pol.pHead;

while (var != NULL)

{

AddElem(var->GetKoef(), var->GetPower());

var = var->GetNext();

}

}

Polinom::~Polinom()

{

/\*elem = new Monom;

while (pHead != NULL)

{

elem = pHead;

delete pHead;

pHead = elem;

}\*/

}

void Polinom::AddElem(double coef, int st)

{

if (coef == 0)

{

return;

}

Monom \*var = pHead;

bool flag = false;

Monom \*m = new Monom(coef, st, pHead);

while (var != NULL)

{

if (var->GetPower() == st)

{

flag = true;

break;

}

var = var->GetNext();

}

if (flag == true)

{

var->SetKoef(coef + var->GetKoef());

}

else

{

pHead = m;

}

}

void Polinom::DelElem()

{

Monom \*elem = new Monom;

while (pHead != NULL)

{

elem = pHead;

delete pHead;

pHead = elem;

}

}

istream &operator>>(istream &s, Polinom &pol)

{

setlocale(LC\_ALL, "Russian");

double kof;

int x, y, z;

int fp,st;

do

{

/\*cout << "Введите моном:" << endl;

cout << "Коэффициент:";\*/

s >> kof;

/\*cout << "Степень при x:"; \*/

s >> x;

/\*cout << "Степень при y:"; \*/

s >> y;

/\*cout << "Степень при z:"; \*/

s >> z;

if (x < 0 || y < 0 || z < 0)

{

throw "Степени отрицательны!";

}

if (x > 9 || y > 9 || z > 9)

{

throw "Степени имеют слишком большой размер!";

}

st = x \* 100 + y \* 10 + z;

pol.AddElem(kof, st);

cout << "Хотите добавить моном?" << endl;

cout << "1.Да" << endl;

cout << "2.Нет" << endl;

s >> fp;

if (fp == 1)

{

system("cls");

cout <<"Исходый полином:"<< pol << "+" << endl;

}

} while (fp == 1);

return s;

}

ostream& operator<<(ostream &f, Polinom &pol)

{

Polinom o(pol);

Monom \*h = o.pHead;

if (h == NULL)

{

f << h->GetKoef();

return f;

}

if (h->GetNext() == NULL)

{

if (h->GetKoef() == 0)

{

f << h->GetKoef();

return f;

}

else if (h->GetPower() == 0)

{

f << h->GetKoef();

}

else

{

f <<h->GetKoef() << "\*x^" << h->GetPower() / 100 << "\*y^" << (h->GetPower() / 10) % 10 << "\*z^" << h->GetPower() % 10 ;

}

return f;

}

while (h->GetNext() != NULL)

{

if (h->GetKoef() == 0.0)

{

h = h->GetNext();

}

else if (h->GetPower() == 0)

{

f << h->GetKoef();

h = h->GetNext();

}

else

{

f << h->GetKoef() << "\*x^" << h->GetPower() / 100 << "\*y^" << (h->GetPower() / 10) % 10 << "\*z^" << h->GetPower() % 10;

h = h->GetNext();

}

if (h->GetNext() != NULL)

{

f << "+";

}

}

if (h->GetKoef() == 0)

{

return f;

}

else if (h->GetPower() == 0)

{

f << " + " << h->GetKoef();

}

else

{

f << " + " << h->GetKoef() << "\*x^" << h->GetPower() / 100 << "\*y^" << (h->GetPower() / 10) % 10 << "\*z^" << h->GetPower() % 10;

}

return f;

}

Polinom& Polinom::operator+(const Polinom &pol) const

{

Polinom \*result = new Polinom(\*this);

Monom \*h = pol.pHead;

while (h != NULL)

{

(\*result).AddElem(h->GetKoef(), h->GetPower());

h = h->GetNext();

}

return \*result;

}

Polinom& Polinom::operator-(const Polinom &pol) const

{

Polinom \*result = new Polinom(\*this);

Monom \*h = pol.pHead;

while (h != NULL)

{

(\*result).AddElem(-h->GetKoef(), h->GetPower());

h = h->GetNext();

}

return \*result;

}

Polinom& Polinom::operator\*(const Polinom &pol) const

{

Polinom \*result = new Polinom;

Polinom o(\*this);

for (Monom \*i = pol.pHead; i != NULL; i = i->GetNext())

{

for (Monom \*j = o.getHead(); j != NULL; j = j->GetNext())

{

int x = 0;

int y = 0;

int z = 0;

x = (i->GetPower()) % 10 + (j->GetPower()) % 10;

y = ((i->GetPower()) / 10) % 10 + ((j->GetPower()) / 10) % 10;

z = (i->GetPower()) / 100 + (j->GetPower()) / 100;

if (x > 9 || y > 9 || z > 9)

{

throw "Степени слишком большие!";

}

(\*result).AddElem(i->GetKoef() \* j->GetKoef(), i->GetPower() + j->GetPower());

}

}

return \*result;

}

Polinom& Polinom::operator=(const Polinom &pol)

{

this->DelElem();

pHead = pol.pHead;

Monom \*pol1 = pol.pHead;

Monom \*pol2 = (\*this).getHead();

pol2 = pol1;

while (pol1 != NULL)

{

pol2 = pol1;

pol1 = pol1->GetNext();

pol2 = pol2->GetNext();

}

return \*this;

}

bool Polinom:: operator==(const Polinom &pol) const

{

Polinom \*pol1 = new Polinom(pol);

if (pHead == NULL)

{

if (pol1->pHead == NULL)

return true;

else

return false;

}

Monom \*i = pHead;

Monom \*j = pol1->pHead;

int o(1);

while (i != NULL)

{

if ((i->GetKoef() != j->GetKoef()) || (i->GetPower() != j->GetPower()))

{

o = 0;

break;

}

i = i->GetNext();

j = j->GetNext();

}

if (o == 1)

return true;

else

return false;

}

*main-polinom.cpp*

#include "polinom.h"

int main()

{

setlocale(LC\_ALL, "Russian");

try

{

Polinom pol1;

Polinom pol2;

int fp, mc;

cout << "Введите первый полином:" << endl;

cin >> pol1;

system("cls");

cout << "Первый полином:" << pol1 << endl;

cout << "Введите второй полином:" << endl;

cin >> pol2;

do

{

system("cls");

cout << "Первый полином:" << pol1 << endl;

cout << "Второй полином:" << pol2 << endl;

cout << "Выберете желаемую оперцию:" << endl;

cout << "1.Сумма" << endl;

cout << "2.Разница" << endl;

cout << "3.Произведение" << endl;

cout << "4.Выход" << endl;

cin >> mc;

if (mc == 1)

{

cout << "Результат:" << pol1 + pol2 << endl;

}

if (mc == 2)

{

cout << "Результат:" << pol1 - pol2 << endl;

}

if (mc == 3)

{

cout << "Результат:" << pol1 \* pol2 << endl;

}

if (mc == 4)

{

return 0;

}

cout << "Выбрать другую операцию?" << endl;

cout << "1.Да" << endl;

cout << "2.Нет" << endl;

cin >> fp;

} while (fp == 1);

}

catch (const char\* error)

{

cout << error << endl;

}

return 0;

}

*test\_polinom.cpp*

#include <gtest.h>

#include "polinom.h"

#include "monom.h"

TEST(Polinom, Addition)

{

Polinom pol1, pol2, result, result1;

pol1.AddElem(2, 123);

pol2.AddElem(3, 123);

result = pol1 + pol2;

result1.AddElem(5, 123);

EXPECT\_EQ(true, result1 == result);

}

TEST(Polinom, Subtraction)

{

Polinom pol1, pol2, result, result1;

pol1.AddElem(3, 123);

pol2.AddElem(2, 123);

result = pol1 - pol2;

result1.AddElem(1, 123);

EXPECT\_EQ(true, result1 == result);

}

TEST(Polinom, Multiplication)

{

Polinom pol1, pol2, result, result1;

pol1.AddElem(2, 123);

pol2.AddElem(3, 432);

result = pol1 \* pol2;

result1.AddElem(6, 555);

EXPECT\_EQ(true, result1 == result);

}

TEST(Polinom, Addition\_of\_different\_sizes)

{

Polinom pol1, pol2, result, result1;

pol1.AddElem(2, 123);

pol1.AddElem(3, 123);

pol2.AddElem(3, 123);

result = pol1 + pol2;

result1.AddElem(8, 123);

EXPECT\_EQ(true, result1 == result);

}

TEST(Polinom, Subtraction\_of\_different\_sizes)

{

Polinom pol1, pol2, result, result1;

pol1.AddElem(5, 123);

pol1.AddElem(3, 123);

pol2.AddElem(2, 123);

result = pol1 - pol2;

result1.AddElem(6, 123);

EXPECT\_EQ(true, result1 == result);

}

TEST(Polinom, Multiplication\_of\_different\_sizes)

{

Polinom pol1, pol2, result, result1;

pol1.AddElem(5, 123);

pol1.AddElem(3, 123);

pol2.AddElem(2, 423);

result = pol1 \* pol2;

result1.AddElem(16, 546);

EXPECT\_EQ(true, result1 == result);

}

TEST(Polinom, Multiplication\_in\_zero)

{

Polinom pol1, pol2, result, result1;

pol1.AddElem(2, 123);

pol2.AddElem(0, 000);

result = pol1 \* pol2;

result1.AddElem(0, 000);

EXPECT\_EQ(true, 0 == result1.getHead());

}

TEST(Polinom, Multiplication\_in\_const)

{

Polinom pol1, pol2, result, result1;

pol1.AddElem(2, 123);

pol2.AddElem(3, 000);

result = pol1 \* pol2;

result1.AddElem(6, 123);

EXPECT\_EQ(true, result1 == result);

}

TEST(Polinom, Coef\_add\_0)

{

Polinom pol;

pol.AddElem(0, 125);

EXPECT\_EQ(true, pol.getHead() == 0);

}

TEST(Polinom, Coef\_add\_negativ)

{

Polinom pol;

pol.AddElem(-1, 125);

EXPECT\_EQ(true, pol.getHead()->GetKoef() == -1);

}

TEST(Polinom, Multiplication\_very\_big\_power)

{

Polinom pol1;

Polinom pol2;

pol1.AddElem(1, 666);

pol2.AddElem(2, 666);

ASSERT\_ANY\_THROW(pol1\*pol2);

}

TEST(Polinom, Multiplication\_in\_negativ\_const)

{

Polinom pol1, pol2, result, result1;

pol1.AddElem(2, 123);

pol2.AddElem(-3, 000);

result = pol1 \* pol2;

result1.AddElem(-6, 123);

EXPECT\_EQ(true, result1 == result);

}

TEST(Polinom, Enter\_polinom\_with\_negative\_power)

{

Polinom pol1;

stringstream c;

c << "2 -1 0 0 ";

ASSERT\_ANY\_THROW(c >> pol1);

}

TEST(Polinom, Enter\_polinom\_with\_to\_big\_power)

{

Polinom pol1;

stringstream c;

c << "2 15 0 0 ";

ASSERT\_ANY\_THROW(c >> pol1);

}

TEST(Polinom, can\_assign\_polinom)

{

Polinom \*pol1 = new Polinom;

Polinom \*pol2 = new Polinom;

pol1->AddElem(5, 123);

pol2->AddElem(7, 543);

ASSERT\_NO\_THROW(pol1 = pol2);

EXPECT\_EQ(pol1, pol2);

}

*MyForm.cpp*

#include "MyForm.h"

using namespace System;

using namespace System::Windows::Forms;

[STAThreadAttribute]

void Main(array<String^>^ args) {

Application::EnableVisualStyles();

Application::SetCompatibleTextRenderingDefault(false);

WinForm::MyForm form;

Application::Run(%form);

}

*MyForm.h*

#pragma once

#include <stdio.h>

#include "polinom.h"

#include <iostream>

#include <sstream>

#include <gtest.h>

#include <windows.system.h>

using namespace std;

namespace WinForm {

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;

}

}

protected:

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

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

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

private: System::Windows::Forms::TextBox^ textBox3;

private: System::Windows::Forms::TextBox^ textBox4;

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

private: System::Windows::Forms::TextBox^ textBox5;

private: System::Windows::Forms::Button^ button2;

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

private: System::Windows::Forms::TextBox^ textBox10;

private: System::Windows::Forms::TextBox^ textBox9;

private: System::Windows::Forms::TextBox^ textBox8;

private: System::Windows::Forms::TextBox^ textBox7;

private: System::Windows::Forms::Button^ button4;

private: System::Windows::Forms::TextBox^ textBox6;

private: System::Windows::Forms::Button^ button3;

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

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

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

private: System::Windows::Forms::Label^ label6;

private: System::Windows::Forms::Label^ label7;

private: System::Windows::Forms::Label^ label8;

private: System::Windows::Forms::Label^ label9;

private: System::Windows::Forms::Button^ button5;

private: System::Windows::Forms::Button^ button6;

private: System::Windows::Forms::Button^ button7;

private: System::Windows::Forms::Label^ label10;

private: System::Windows::Forms::TextBox^ textBox11;

private: System::Windows::Forms::Button^ button8;

private: System::Windows::Forms::Button^ button9;

private: System::Windows::Forms::Button^ button10;

protected:

private:

/// <summary>

/// Требуется переменная конструктора.

/// </summary>

System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Обязательный метод для поддержки конструктора - не изменяйте

/// содержимое данного метода при помощи редактора кода.

/// </summary>

void InitializeComponent(void)

{

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

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

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

this->textBox3 = (gcnew System::Windows::Forms::TextBox());

this->textBox4 = (gcnew System::Windows::Forms::TextBox());

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

this->textBox5 = (gcnew System::Windows::Forms::TextBox());

this->button2 = (gcnew System::Windows::Forms::Button());

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

this->textBox10 = (gcnew System::Windows::Forms::TextBox());

this->textBox9 = (gcnew System::Windows::Forms::TextBox());

this->textBox8 = (gcnew System::Windows::Forms::TextBox());

this->textBox7 = (gcnew System::Windows::Forms::TextBox());

this->button4 = (gcnew System::Windows::Forms::Button());

this->textBox6 = (gcnew System::Windows::Forms::TextBox());

this->button3 = (gcnew System::Windows::Forms::Button());

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

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

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

this->label6 = (gcnew System::Windows::Forms::Label());

this->label7 = (gcnew System::Windows::Forms::Label());

this->label8 = (gcnew System::Windows::Forms::Label());

this->label9 = (gcnew System::Windows::Forms::Label());

this->button5 = (gcnew System::Windows::Forms::Button());

this->button6 = (gcnew System::Windows::Forms::Button());

this->button7 = (gcnew System::Windows::Forms::Button());

this->label10 = (gcnew System::Windows::Forms::Label());

this->textBox11 = (gcnew System::Windows::Forms::TextBox());

this->button8 = (gcnew System::Windows::Forms::Button());

this->button9 = (gcnew System::Windows::Forms::Button());

this->button10 = (gcnew System::Windows::Forms::Button());

this->SuspendLayout();

//

// label1

//

this->label1->AutoSize = true;

this->label1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 11.25F));

this->label1->Location = System::Drawing::Point(237, 9);

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

this->label1->Size = System::Drawing::Size(114, 18);

this->label1->TabIndex = 0;

this->label1->Text = L"Первый моном";

//

// textBox1

//

this->textBox1->Location = System::Drawing::Point(33, 47);

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

this->textBox1->Size = System::Drawing::Size(21, 20);

this->textBox1->TabIndex = 1;

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

//

// textBox2

//

this->textBox2->Location = System::Drawing::Point(87, 35);

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

this->textBox2->Size = System::Drawing::Size(21, 20);

this->textBox2->TabIndex = 2;

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

//

// textBox3

//

this->textBox3->Location = System::Drawing::Point(141, 35);

this->textBox3->Name = L"textBox3";

this->textBox3->Size = System::Drawing::Size(21, 20);

this->textBox3->TabIndex = 3;

this->textBox3->TextChanged += gcnew System::EventHandler(this, &MyForm::textBox3\_TextChanged);

//

// textBox4

//

this->textBox4->Location = System::Drawing::Point(196, 35);

this->textBox4->Name = L"textBox4";

this->textBox4->Size = System::Drawing::Size(21, 20);

this->textBox4->TabIndex = 4;

this->textBox4->TextChanged += gcnew System::EventHandler(this, &MyForm::textBox4\_TextChanged);

//

// button1

//

this->button1->Location = System::Drawing::Point(248, 35);

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

this->button1->Size = System::Drawing::Size(103, 21);

this->button1->TabIndex = 5;

this->button1->Text = L"Добавить моном";

this->button1->UseVisualStyleBackColor = true;

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

//

// textBox5

//

this->textBox5->Location = System::Drawing::Point(363, 35);

this->textBox5->Name = L"textBox5";

this->textBox5->Size = System::Drawing::Size(205, 20);

this->textBox5->TabIndex = 6;

this->textBox5->TextChanged += gcnew System::EventHandler(this, &MyForm::textBox5\_TextChanged);

//

// button2

//

this->button2->Location = System::Drawing::Point(363, 65);

this->button2->Name = L"button2";

this->button2->Size = System::Drawing::Size(205, 21);

this->button2->TabIndex = 7;

this->button2->Text = L"Удалить полином";

this->button2->UseVisualStyleBackColor = true;

this->button2->Click += gcnew System::EventHandler(this, &MyForm::button2\_Click);

//

// label2

//

this->label2->AutoSize = true;

this->label2->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 11.25F));

this->label2->Location = System::Drawing::Point(240, 104);

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

this->label2->Size = System::Drawing::Size(111, 18);

this->label2->TabIndex = 8;

this->label2->Text = L"Второй моном";

//

// textBox10

//

this->textBox10->Location = System::Drawing::Point(33, 153);

this->textBox10->Name = L"textBox10";

this->textBox10->Size = System::Drawing::Size(21, 20);

this->textBox10->TabIndex = 9;

this->textBox10->TextChanged += gcnew System::EventHandler(this, &MyForm::textBox10\_TextChanged);

//

// textBox9

//

this->textBox9->Location = System::Drawing::Point(87, 140);

this->textBox9->Name = L"textBox9";

this->textBox9->Size = System::Drawing::Size(21, 20);

this->textBox9->TabIndex = 10;

this->textBox9->TextChanged += gcnew System::EventHandler(this, &MyForm::textBox9\_TextChanged);

//

// textBox8

//

this->textBox8->Location = System::Drawing::Point(141, 140);

this->textBox8->Name = L"textBox8";

this->textBox8->Size = System::Drawing::Size(21, 20);

this->textBox8->TabIndex = 11;

this->textBox8->TextChanged += gcnew System::EventHandler(this, &MyForm::textBox8\_TextChanged);

//

// textBox7

//

this->textBox7->Location = System::Drawing::Point(196, 140);

this->textBox7->Name = L"textBox7";

this->textBox7->Size = System::Drawing::Size(21, 20);

this->textBox7->TabIndex = 12;

this->textBox7->TextChanged += gcnew System::EventHandler(this, &MyForm::textBox7\_TextChanged);

//

// button4

//

this->button4->Location = System::Drawing::Point(248, 135);

this->button4->Name = L"button4";

this->button4->Size = System::Drawing::Size(103, 21);

this->button4->TabIndex = 13;

this->button4->Text = L"Добавить моном";

this->button4->UseVisualStyleBackColor = true;

this->button4->Click += gcnew System::EventHandler(this, &MyForm::button4\_Click);

//

// textBox6

//

this->textBox6->Location = System::Drawing::Point(363, 135);

this->textBox6->Name = L"textBox6";

this->textBox6->Size = System::Drawing::Size(205, 20);

this->textBox6->TabIndex = 14;

this->textBox6->TextChanged += gcnew System::EventHandler(this, &MyForm::textBox6\_TextChanged);

//

// button3

//

this->button3->Location = System::Drawing::Point(363, 170);

this->button3->Name = L"button3";

this->button3->Size = System::Drawing::Size(205, 21);

this->button3->TabIndex = 15;

this->button3->Text = L"Удалить полином";

this->button3->UseVisualStyleBackColor = true;

this->button3->Click += gcnew System::EventHandler(this, &MyForm::button3\_Click);

//

// label3

//

this->label3->AutoSize = true;

this->label3->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 11.25F));

this->label3->Location = System::Drawing::Point(60, 49);

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

this->label3->Size = System::Drawing::Size(21, 18);

this->label3->TabIndex = 16;

this->label3->Text = L"\*x";

//

// label4

//

this->label4->AutoSize = true;

this->label4->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 11.25F));

this->label4->Location = System::Drawing::Point(114, 49);

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

this->label4->Size = System::Drawing::Size(21, 18);

this->label4->TabIndex = 17;

this->label4->Text = L"\*y";

//

// label5

//

this->label5->AutoSize = true;

this->label5->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 11.25F));

this->label5->Location = System::Drawing::Point(168, 49);

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

this->label5->Size = System::Drawing::Size(22, 18);

this->label5->TabIndex = 18;

this->label5->Text = L"\*z";

//

// label6

//

this->label6->AutoSize = true;

this->label6->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 11.25F));

this->label6->Location = System::Drawing::Point(60, 155);

this->label6->Name = L"label6";

this->label6->Size = System::Drawing::Size(21, 18);

this->label6->TabIndex = 19;

this->label6->Text = L"\*x";

//

// label7

//

this->label7->AutoSize = true;

this->label7->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 11.25F));

this->label7->Location = System::Drawing::Point(114, 155);

this->label7->Name = L"label7";

this->label7->Size = System::Drawing::Size(21, 18);

this->label7->TabIndex = 20;

this->label7->Text = L"\*y";

//

// label8

//

this->label8->AutoSize = true;

this->label8->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 11.25F));

this->label8->Location = System::Drawing::Point(168, 155);

this->label8->Name = L"label8";

this->label8->Size = System::Drawing::Size(22, 18);

this->label8->TabIndex = 21;

this->label8->Text = L"\*z";

//

// label9

//

this->label9->AutoSize = true;

this->label9->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 11.25F));

this->label9->Location = System::Drawing::Point(260, 198);

this->label9->Name = L"label9";

this->label9->Size = System::Drawing::Size(76, 18);

this->label9->TabIndex = 22;

this->label9->Text = L"Операции";

//

// button5

//

this->button5->Location = System::Drawing::Point(159, 219);

this->button5->Name = L"button5";

this->button5->Size = System::Drawing::Size(75, 23);

this->button5->TabIndex = 23;

this->button5->Text = L"+";

this->button5->UseVisualStyleBackColor = true;

this->button5->Click += gcnew System::EventHandler(this, &MyForm::button5\_Click);

//

// button6

//

this->button6->Location = System::Drawing::Point(261, 219);

this->button6->Name = L"button6";

this->button6->Size = System::Drawing::Size(75, 23);

this->button6->TabIndex = 24;

this->button6->Text = L"-";

this->button6->UseVisualStyleBackColor = true;

this->button6->Click += gcnew System::EventHandler(this, &MyForm::button6\_Click);

//

// button7

//

this->button7->Location = System::Drawing::Point(363, 221);

this->button7->Name = L"button7";

this->button7->Size = System::Drawing::Size(75, 23);

this->button7->TabIndex = 25;

this->button7->Text = L"\*";

this->button7->UseVisualStyleBackColor = true;

this->button7->Click += gcnew System::EventHandler(this, &MyForm::button7\_Click);

//

// label10

//

this->label10->AutoSize = true;

this->label10->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 11.25F));

this->label10->Location = System::Drawing::Point(256, 261);

this->label10->Name = L"label10";

this->label10->Size = System::Drawing::Size(80, 18);

this->label10->TabIndex = 26;

this->label10->Text = L"Результат";

//

// textBox11

//

this->textBox11->Location = System::Drawing::Point(159, 282);

this->textBox11->Name = L"textBox11";

this->textBox11->Size = System::Drawing::Size(279, 20);

this->textBox11->TabIndex = 27;

this->textBox11->TextChanged += gcnew System::EventHandler(this, &MyForm::textBox11\_TextChanged);

//

// button8

//

this->button8->Location = System::Drawing::Point(473, 221);

this->button8->Name = L"button8";

this->button8->Size = System::Drawing::Size(75, 23);

this->button8->TabIndex = 28;

this->button8->Text = L"Тесты";

this->button8->UseVisualStyleBackColor = true;

this->button8->Click += gcnew System::EventHandler(this, &MyForm::button8\_Click);

//

// button9

//

this->button9->Location = System::Drawing::Point(473, 250);

this->button9->Name = L"button9";

this->button9->Size = System::Drawing::Size(75, 23);

this->button9->TabIndex = 29;

this->button9->Text = L"Отчет";

this->button9->UseVisualStyleBackColor = true;

this->button9->Click += gcnew System::EventHandler(this, &MyForm::button9\_Click);

//

// button10

//

this->button10->Location = System::Drawing::Point(473, 279);

this->button10->Name = L"button10";

this->button10->Size = System::Drawing::Size(75, 23);

this->button10->TabIndex = 30;

this->button10->Text = L"Консоль";

this->button10->UseVisualStyleBackColor = true;

this->button10->Click += gcnew System::EventHandler(this, &MyForm::button10\_Click);

//

// MyForm

//

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

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

this->ClientSize = System::Drawing::Size(592, 320);

this->Controls->Add(this->button10);

this->Controls->Add(this->button9);

this->Controls->Add(this->button8);

this->Controls->Add(this->textBox11);

this->Controls->Add(this->label10);

this->Controls->Add(this->button7);

this->Controls->Add(this->button6);

this->Controls->Add(this->button5);

this->Controls->Add(this->label9);

this->Controls->Add(this->label8);

this->Controls->Add(this->label7);

this->Controls->Add(this->label6);

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

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

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

this->Controls->Add(this->button3);

this->Controls->Add(this->textBox6);

this->Controls->Add(this->button4);

this->Controls->Add(this->textBox7);

this->Controls->Add(this->textBox8);

this->Controls->Add(this->textBox9);

this->Controls->Add(this->textBox10);

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

this->Controls->Add(this->button2);

this->Controls->Add(this->textBox5);

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

this->Controls->Add(this->textBox4);

this->Controls->Add(this->textBox3);

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

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

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

this->Cursor = System::Windows::Forms::Cursors::Default;

this->Name = L"MyForm";

this->Text = L"MyForm";

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

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

Polinom \*pol1;

Polinom \*pol2;

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

{

pol1 = new Polinom;

pol2 = new Polinom;

}

private: System::Void label4\_Click(System::Object^ sender, System::EventArgs^ e)

{

}

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

{

try

{

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

int x = Convert::ToInt32(textBox2->Text);

int y = Convert::ToInt32(textBox3->Text);

int z = Convert::ToInt32(textBox4->Text);

if (x < 0 || y < 0 || z < 0)

{

throw "Отрицательная степень не допустима! ";

}

if (x > 9 || y > 9 || z > 9)

{

throw "Слишком большая степень не допустима! ";

}

int N = x \* 100 + y \* 10 + z;

pol1->AddElem(kof, N);

string str;

stringstream s;

s << (\*pol1);

getline(s, str);

textBox5->Text = gcnew String(str.c\_str());

}

catch (const char\* error)

{

textBox5->Text = "Некоректная степень не допустима!";

}

}

private: System::Void button2\_Click(System::Object^ sender, System::EventArgs^ e)

{

textBox5->Text = "";

pol1 = new Polinom;

}

private: System::Void button4\_Click(System::Object^ sender, System::EventArgs^ e)

{

try

{

double koef = Convert::ToDouble(textBox10->Text);

int x = Convert::ToInt32(textBox9->Text);

int y = Convert::ToInt32(textBox8->Text);

int z = Convert::ToInt32(textBox7->Text);

if (x < 0 || y < 0 || z < 0)

{

throw "Отрицательная степень не допустима! ";

}

if (x > 9 || y > 9 || z > 9)

{

throw "Слишком большая степень не допустима! ";

}

int N = x \* 100 + y \* 10 + z;

pol2->AddElem(koef, N);

string str;

stringstream c;

c << (\*pol2);

getline(c, str);

textBox6->Text = gcnew String(str.c\_str());

}

catch (const char\* error)

{

textBox6->Text = "Некоректная степень не допустима!";

}

}

private: System::Void button3\_Click(System::Object^ sender, System::EventArgs^ e)

{

textBox6->Text = "";

pol1 = new Polinom;

}

private: System::Void button5\_Click(System::Object^ sender, System::EventArgs^ e)

{

Polinom res = (\*pol1) + (\*pol2);

string str;

stringstream c;

c << res;

getline(c, str);

textBox11->Text = gcnew String(str.c\_str());

}

private: System::Void button6\_Click(System::Object^ sender, System::EventArgs^ e)

{

Polinom res = (\*pol1) - (\*pol2);

string str;

stringstream c;

c << res;

getline(c, str);

textBox11->Text = gcnew String(str.c\_str());

}

private: System::Void button7\_Click(System::Object^ sender, System::EventArgs^ e)

{

try

{

Polinom res = (\*pol1) \* (\*pol2);

string str;

stringstream c;

c << res;

getline(c, str);

textBox11->Text = gcnew String(str.c\_str());

}

catch (const char\* error)

{

textBox11->Text = "Некоректная степень не допустима!";

}

}

private: System::Void button8\_Click(System::Object^ sender, System::EventArgs^ e)

{

system("test.bat");

/\*STARTUPINFO si;

PROCESS\_INFORMATION pi;

ZeroMemory(&si, siz eof(si));

si.cb = sizeof(si);

ZeroMemory(&pi, sizeof(pi));

char text[] = "test.bat";

wchar\_t wtext[100];

mbstowcs(wtext, text, strlen(text) + 1);

LPWSTR name = wtext;

if (!CreateProcess(NULL, name, NULL, NULL, FALSE, 0, NULL, NULL, &si, &pi))

{

}

WaitForSingleObject(pi.hProcess, INFINITE);

CloseHandle(pi.hProcess);

CloseHandle(pi.hThread);

\*/

}

private: System::Void button9\_Click(System::Object^ sender, System::EventArgs^ e)

{

system("Otchet.docx");

}

private: System::Void button10\_Click(System::Object^ sender, System::EventArgs^ e)

{

system("sample.exe");

}

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 textBox3\_TextChanged(System::Object^ sender, System::EventArgs^ e){}

private: System::Void textBox4\_TextChanged(System::Object^ sender, System::EventArgs^ e){}

private: System::Void textBox5\_TextChanged(System::Object^ sender, System::EventArgs^ e){}

private: System::Void textBox6\_TextChanged(System::Object^ sender, System::EventArgs^ e){}

private: System::Void textBox7\_TextChanged(System::Object^ sender, System::EventArgs^ e){}

private: System::Void textBox8\_TextChanged(System::Object^ sender, System::EventArgs^ e){}

private: System::Void textBox9\_TextChanged(System::Object^ sender, System::EventArgs^ e){}

private: System::Void textBox10\_TextChanged(System::Object^ sender, System::EventArgs^ e){}

private: System::Void textBox11\_TextChanged(System::Object^ sender, System::EventArgs^ e){}

};

}