МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования   
**«Национальный исследовательский   
Нижегородский государственный университет им. Н.И. Лобачевского»**

**(ННГУ)**

**Институт информационных технологий, математики и механики**

Направление подготовки «Прикладная математика и информатика»

**ОТЧЕТ**

по учебной практике

**Арифметические операции с полиномами**

**Выполнила:** студентка группы

381703-1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Подпись

Ченкова А.А.

**Проверил:** ассистент кафедры

МОСТ ИИТММ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Подпись

Волокитин В. Д.

Нижний Новгород  
2019

**Содержание**

[Введение 3](#_Toc7546887)

[1 .Постановка учебно-практической задачи 4](#_Toc7546888)

[2.Руководство пользователя 5](#_Toc7546889)

[3.Руководство программиста 6](#_Toc7546890)

[*Структура программы* 6](#_Toc7546891)

[*Структура данных* 6](#_Toc7546892)

[*Описание алгоритмов* 7](#_Toc7546893)

[Заключение 9](#_Toc7546894)

[Литература 10](#_Toc7546895)

[Приложение 11](#_Toc7546896)

# Введение

**Полином** - это сумма одночленов (мономов), которые являются произведениями, состоящими из числового множителя (коэффициента) и одной или нескольких переменных, каждая из которых взята с тем или иным показателем. В данной лабораторной работе представлены алгоритмы для проведения арифметических операций над полиномами.

# 1 .Постановка учебно-практической задачи

*Формулировка задачи:*

Разработать программу, выполняющую арифметические операции с полиномами трех переменных (x, y и z): сложение, вычитание, умножение на константу, умножение двух полиномов. Считается, что полином составлен из мономов от трех переменных с ограничением на степень каждой переменой от 0 до 20. Коэффициенты полинома - вещественные числа. Работоспособность программы необходимо проверить с помощью Google Test-ов. Кроме того, необходимо разработать пользовательское консольное приложение.

# 2.Руководство пользователя

Необходимо запустить на выполнение файл “Sample.exe”, находящийся в папке “Debug”. Перед пользователем открывается прикладная программа.



Рис. 1. Ввод пользователем

Пользователю необходимо последовательно ввести два многочлена (А, В) по правилам, указанным в примере ввода. Программа выдаст коэффициенты каждого монома и степени при x,y,z для каждого монома. После чего на экран будут выведены результаты вычислений (сложение введенных полиномов, вычитание, умножение). Если размер степени переменной будет нарушен, то программа выбрасывает исключение.

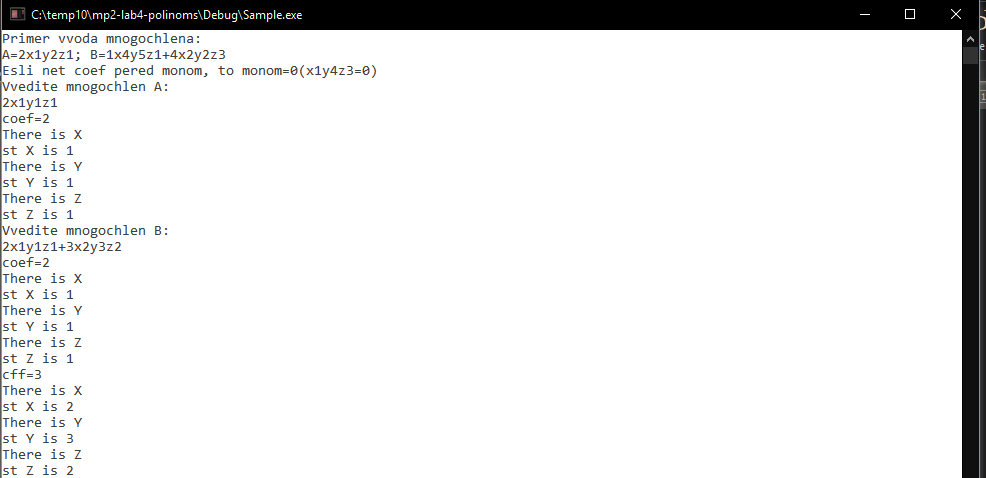


Рис.2 Ввод многочленов



Рис.3 Вычисление результатов

# 3.Руководство программиста

## *Структура программы*

Программа включает в себя следующие файлы:

* **Monom.h** - содержит объявление класса «Monom» и реализацию методов этого класса;
* **Polynoms.h** – содержит объявление класса «Polynom» и прототипы функций;
* **Polynoms.cpp** - содержит реализацию методов класса «Polynom» ;
* **main.cpp** - включает в себя прикладное приложение, демонстрирующее работу вышеназванных классов;
* **test.cpp** - содержит тесты для класса «Polynom»;
* **test\_main.cpp** - необходим для запуска Google tests.

## *Структура данных*

**Класс Monom**

Поля класса:

* + int power; //степень
  + double cf; //коэффициент
  + Link pointer; //указатель на след. моном

Методы класса:

* Monom(int \_power = 0, double \_cf = 0, Link \_pointer = NULL); //конструктор инициализации
* void SetPower (int); //установка степени
* void SetCf (double); //установка коэффициента
* void SetPointer (Link); //установка указателя
* int GetPower (void) const; //получить степень
* double GetCf (void) const; //получить коэффициент
* Monom\* GetPointer (void) const; //получить следующий моном

**Класс Polynoms**

Поля класса:

* Link Head; //указатель на голову

Методы класса:

* Polynom(); //конструктор без параметров
* Polynom(string); //конструктор с параметром
* void loadString(string);// ввод строки
* void Add(int,double);//вставка монома
* Polynom& operator+(Polynom); //оператор сложения
* Polynom& operator-(Polynom); //оператор вычитания
* Polynom& operator\*(Polynom); //оператор умножения полиномов
* bool operator==(const Polynom&) const;//оператор сравнения
* double calc(double x, double y, double z);//вычисление значения
* void Print();//печать

## *Описание алгоритмов*

В программе были реализованы алгоритмы и перегрузки следующих операций: сложение, вычитание, умножение и сравнение.

1. Оператор сравнения

bool Polynom::operator== (const Polynom& op2) const -возвращает true, если последовательное сравнение степеней и коэффициентов полиномов совпадает, иначе возвращает false.

1. Сложение полиномов

Polynom &Polynom::operator+ (Polynom op2) –два списка последовательно сливаются в третий

1. Вычитание полиномов

Polynom &Polynom::operator- (Polynom op2) ­–прибавление к первому полиному второго, коэффициенты которого умножены на «-1»

1. Умножение двух полиномов

Polynom &Polynom::operator\*(Polynom op2) –последовательное умножение коэффициентов каждого монома первого полинома на коэффициенты мономов второго полинома и аналогичное сложение степеней

1. Функция вставки

void Polynom::Add(int power, double cf) – функция на вход принимается коэффициент и степень в формате трехзначного числа, находит нужное место для монома и вставляет его на это место

# Заключение

В результате лабораторной работы была разработана программа, выполняющая арифметические операции над полиномами трех переменных (x, y и z), создано пользовательское консольное приложение, дающее возможность пользователю вводить полиномы, предварительно не упорядочив в них мономы. Работа программы протестирована с помощью Google Tests.

# Литература

1. Шилдт, Г. С++ для начинающих: самоучитель / Шилдт, Г. - Изд-во: Эком, 2013г.

# Приложение

# Сложение полиномов

void Polynom::Add(int power, double cf) {

init();

Link prev = Head, cur = Head, tmp; //prev=указатель на пред cur=указатель на данный эл списка

if (cf != 0){

while ((cur->GetPointer()->GetPower() != -1)/\*не бегать по кругу\*/ && (cur->GetPointer()->GetPower() > power)) //найти нужное место для добавляемого монома

{

cur = cur->GetPointer();

}

cur = cur->GetPointer();

while (prev->GetPointer() != cur)

prev = prev->GetPointer();

if (cur->GetPower() == power){

if ((cur->GetCf() + cf) != 0)

cur->SetCf(cur->GetCf() + cf);

else {

prev->SetPointer(cur->GetPointer());

cur->SetPointer(NULL);

delete cur;

}

}

else{

tmp = new Monom(power, cf, cur);

prev->SetPointer(tmp);

}

}

}

Polynom &Polynom::operator+(Polynom op2) {

init();

Polynom res;

res.init();

Link cur = res.Head->GetPointer();

Link point = Head->GetPointer(); //бегает по левому операнду

while (point->GetPower() != -1)

{

res.Add(point->GetPower(), point->GetCf());

point = point->GetPointer();

}

point = op2.Head->GetPointer(); // теперь бегает по правому

while (point->GetPower() != -1)

{

res.Add(point->GetPower(), point->GetCf());

point = point->GetPointer();

}

return res;

}

# Вычитание полиномов

Polynom &Polynom::operator-(Polynom op2) {

init();

Polynom res;

res.init();

Link cur = res.Head->GetPointer(); /\*бегает по результату\*/

Link point = Head->GetPointer(); //бегает по левому операнду

while (point->GetPower() != -1)

{

res.Add(point->GetPower(), point->GetCf());

point = point->GetPointer();

}

point = op2.Head->GetPointer(); // теперь бегает по правому

while (point->GetPower() != -1)

{

res.Add(point->GetPower(), -(point->GetCf()));

point = point->GetPointer();

}

return res;

}

# Умножение полиномов

Polynom &Polynom::operator\*(Polynom op2){

init();

Polynom res;

res.init();

Link cur1 = Head->GetPointer();

Link cur2 = op2.Head->GetPointer();

while (cur1->GetPower() != -1)

{

while (cur2->GetPower() != -1)

{

res.Add( cur1->GetPower() + cur2->GetPower() ,

cur1-> GetCf()\*cur2->GetCf());

cur2 = cur2->GetPointer();

}

cur2 = cur2->GetPointer();

cur1 = cur1->GetPointer();

}

return res;

}

# Сравнение полиномов

bool Polynom::operator==(const Polynom&op2) const {

if (Head == nullptr)

{

if (op2.Head == nullptr)

return true;

return false;

}

bool res = true;

Link cur1 = Head->GetPointer();

Link cur2 = op2.Head->GetPointer();

while ((cur1->GetPower() != -1) && (cur2->GetPower() != -1)) {

if ((abs(cur1->GetCf() - cur2->GetCf())) ||

(cur1->GetPower() != cur2->GetPower()))

res = false;

if (((cur1->GetPower() == -1) && (cur2->GetPower() != -1)) ||

((cur1->GetPower() != -1) && (cur2->GetPower() == -1)))

res = false;

cur1 = cur1->GetPointer();

cur2 = cur2->GetPointer();

}

return res; }