Федеральное государственное автономное образовательное учреждение

высшего образования

«Нижегородский государственный университет

им. Н.И. Лобачевского»

Институт информационных технологий, математики и механики

Отчёт по лабораторной работе

Арифметические операции с полиномами.

Выполнил:

студент ИИТММ гр. 381703-1

Девликамов В.О

Проверил:

Ассистент кафедры МОСТ ИИТММ

Волокитин В.Д.

Нижний Новгород

2018 г.

Содержание

Введение……………………………………………………………………………………3

Постановка задачи…………………………………………………………………………4

Руководство пользователя………………………………………………………………...5

Руководство программиста………………………………………………………………..6

Описание структуры программы………………………………………………………6

Описание структур данных……………………………………………………………..6

Описание алгоритмов…………………………………………………………………...7

[Заключение…………………………………………………………………………………8](#_Toc499930791)

Литература……………………………………………………………………………….....9

Приложения……………………………………………………………………………….10

# Введение

В данной лабораторной работе реализованы арифметические операции над  многочленами: сложение, вычитание, умножение на константу, умножение двух полиномов. Полиномы играют ключевую роль в алгебраической геометрии, алгебре, теории узлов и других разделах математики для кодирования или выражения многочленами свойств различных объектов. Для того, чтобы работать с полиномами необходимо выполнять операции над ними.

# Постановка задачи

**Цель работы:** 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 9 (Опцианально можно расширить данное ограничение). Коэффициенты полинома - вещественные числа. Работоспособность программы необходимо проверить с помощью Google Test-ов. Кроме того, необходимо разработать пользовательское консольное приложение.

**Особенности реализации:**

1. В качестве структуры хранения полинома использовать список мономов с ненулевыми коэффициентами (односвязный или двухсвязный). Односвязный список удобнее реализовать с фиктивной головой. Элементы списка хранить упорядоченными.
2. Степень полинома хранить в "свернутом" виде, т. е. степень должна быть представлена как трехзначное число, где число сотен – это степень при переменной “x”, число десятков - степень при переменной “y”, число единиц - степень при переменной “z”.
3. Сложение полиномов осуществлять алгоритмом слияния упорядоченных массивов.
4. Вычитание полиномов допускается выполнять через сложение с умножением на константу (C = A - B = A + (-1)\*B)
5. При умножении и сложение (вычитание) необходимо следить, чтобы в итоговом полиноме были приведены подобные слагаемые и не хранилось мономов с нулевым коэффициентом.
6. Если при умножении полиномов полученные степени переменных больше 9, выводить сообщение об ошибке.
7. Считывание полинома у пользователя допускается в любом удобном для вас виде. Необходимо предоставить пользователю правила ввода данных
8. Следует учесть, что пользователь может вводить полином, не упорядочив в нем мономы.

# Руководство пользователя

# Вначале необходимо ввести полином ("Enter the polynom"). Чтобы ввести полином надо ввести количество мономов в полиноме, далее ввести коэффициент перед полиномом, после необходимо ввести степень перед x, y, z. После ввода полинома, следует выбрать операцию над введенным полиномом (1. Сложить 2 полинома, 2. Умножить на полином, 3. Вычесть 2 полинома, 4. Умножить полином на константу). Если ввели 1, 2 или 3, то после необходимо ввести полином. В случае если ввести 4, то далее необходимо ввести константу.

# Руководство программиста

**Описание структур программы.**

Программа состоит из 4-x проектов: gtest, sample, tests, polinom.

gtest содержит реализацию Google Test.

sample содержит реализацию пользовательского консольная приложения.

tests – содержит (test\_main.cpp – запускает все тесты, test\_stack.cpp – тесты для стека, test\_ariphmetic.cpp – тесты для проверки всех функций из ariphmetic.cpp)

polinom – содержит (polinom.cpp – содержит реализацию арифметических операций с

полиномами.

**Описание структур данных.**

**main.cpp**

Polinom a – полином над котором производятся арифметические операции

Polinom b – полином для вычисления арифметических операций над полиномом a

double cons – константа для вычисления арифметических операций над полиномом a

**polinom.h**

struct Monom – структура содержащая Моном

1. double coef – содержит коэффициент монома
2. int pow – степень монома
3. Monom\* next – указатель на следующий моном
4. Monom\* prev – указатель на предыдущий моном

class Polinom – класс, содержащий реализацию полинома и арифметических операций над ним

1. Monom\* pol – указатель на фиктивную голову

Класс Polinom cодержит:

* Конструктор по умолчанию
* Вставка в определенное место (для поддержания листа отсортированным)
* Вставка в конец
* Удаление монома
* Удаление дубликатов в полиноме (приведение подобных)
* Умножение на константу
* Сложение 2 полиномов
* Вычитание 2 полиномов
* Умножение 2 полиномов
* Перегруженный оператор ввода и вывода

**polinom.cpp**

Cодержит реализацию методов класса Polinom и функций, объявленных в файле polinom.h.

**Описание алгоритмов**

Умножение полинома на константу: Идем по списку полинома и умножаем коэффициент монома на константу.

Сложение двух полиномов: Будем идти двумя указателями: первый указывает на первый полином, второй указывает на второй. Передвигаем тот указатель, который указывает на моном с наибольшей степенью, если степени одинаковые, то складываем коэффициенты перед мономами и на каждом шаге вставляем данный моном в новый полином, который и будет результатом сложения 2 полиномов.

Вычитание двух полиномов: Реализуем через сложение и умножения на константу (A + (-1)\*B)

Умножение двух полиномов: Будем идти двумя указателями: первый указывает на первый полином, второй указывает на второй. Перемножим все мономы из 1 полинома на все мономы 2 полинома, передвигая указатель (На каждом шаге будем двигать 2 указатель, а после того как он придет в конец 2 полинома, передвинем 1 указатель, будем повторять пока 1 указатель не придет в конец полинома). Также на каждом шаге результат будем вставлять в определенное место, для поддержания полинома отсортированным, в новый полином, который и будет результатом умножения 2 полиномов.

# Заключение

В результате проделанной работы были достигнуты следующие результаты:

1. Разработан программу производящую арифметические операции над полиномом.
2. Функции тестируются с помощью Google Tests.
3. Программа написана на языке C++.
4. Коэффициенты полинома ­- вещественные числа, а также полином составлен из мономов от 3 переменных (существует ограничение на степень переменной от 0 до 9)
5. Элементы списка хранятся в упорядоченном виде.
6. В качестве структуры хранения полинома использован двухсвязный список мономов.

# Литература

1. Многочлены [https://ru.wikipedia.org/wiki/Многочлен]

**Приложения**

Polinom Polinom::operator\*(double v) const {

Polinom ans;

if (v < eps) return ans;

ans = \*this;

Monom\* temp = ans.pol->next;

while (temp != nullptr) {

temp->coef = temp->coef \* v;

temp = temp->next;

}

return ans;

}

Polinom Polinom::operator+(const Polinom& p) const {

Polinom ans;

Monom\* a = pol->next;

Monom\* b = p.pol->next;

while (true) {

if (a == nullptr) {

if (b == nullptr) {

break;

}

else {

while (b != nullptr) {

ans.ins(b->coef, b->pow);

b = b->next;

}

break;

}

}

else if (b == nullptr) {

while (a != nullptr) {

ans.ins(a->coef, a->pow);

a = a->next;

}

break;

}

else {

if (a->pow == b->pow) {

double coef = a->coef + b->coef;

if (coef < eps) {

a = a->next;

b = b->next;

}

else {

ans.ins(coef, a->pow);

a = a->next;

b = b->next;

}

}

else if (a->pow > b->pow) {

ans.ins(a->coef, a->pow);

a = a->next;

}

else {

ans.ins(b->coef, b->pow);

b = b->next;

}

}

}

return ans;

}

Polinom Polinom::operator-(const Polinom &p) const {

return \*this + p\*(-1.0);

}

Polinom Polinom::operator\*(const Polinom &p) const {

Polinom ans;

Monom\* a = pol->next;

Monom\* b = p.pol->next;

while (a != nullptr) {

Monom\* temp = b;

while (temp != nullptr) {

int pow\_x = (a->pow) / 100 + (temp->pow) / 100;

int pow\_y = ((a->pow) / 10 % 10) + ((temp->pow) / 10 % 10);

int pow\_z = (a->pow) % 10 + (temp->pow) % 10;

if (pow\_x > 9 || pow\_y > 9 || pow\_z > 9) throw "Error";

int pow = pow\_x \* 100 + pow\_y \* 10 + pow\_z;

double coef = a->coef \* temp->coef;

ans.ins(coef, pow);

temp = temp->next;

}

a = a->next;

}

ans.del\_dublicate();

return ans;

}