Министерство образования и науки Российской Федерации

Федеральное государственное автономное

Образовательное учреждение высшего образования

«Национальный исследовательский Нижегородский   
государственный университет им. Н.И. Лобачевского»

Институт информационных технологий, математики и механики

**Отчет по лабораторной работе**

**"Разбор арифметических выражений"**

**Выполнил:**

Утюгов Денис  
Студент группы 381906-2

**Проверила:**

Усова Марина Андреевна

Нижний Новгород

2020

**Содержание:**

1. Введение
2. Постановка учебно-практической задачи
3. Руководство пользователя
4. Руководство программиста
5. Заключение
6. Список литературы
7. Приложение

**Введение**

Наряду с привычным вычислительным применением компьютеры широко используются и для аналитической обработки данных. Среди примеров таких приложений – компьютерное доказательство теорем, логический вывод, анализ текстовой информации и многое другое. Среди таких примеров и задача обработки полиномов, задаваемых в общей аналитической форме. Полиномы являются хорошо изученной областью математики (алгебра полиномов), которая широко используется в приложениях (аппроксимация экспериментальных данных, построение функциональных зависимостей и т.п.). Лабораторная работа направлена на изучение методов компьютерной обработки полиномов. С этой целью в лабораторной работе изучаются различные варианты структуры хранения и разрабатываются программы для обработки полиномов. Основной учебной целью работы является практическое освоение методов организации структур хранения данных с помощью списков. В ходе выполнения лабораторной работы разрабатывается общая форма представления линейных списков, разрабатываются программы работы со списками, которые могут быть использованы и в других областях приложений.

**Постановка учебно-практической задачи**

В рамках лабораторной работы ставится задача реализации программы, способной выполнять алгебраические действия с полиномами с помощью структуры данных «линейный список».

**Руководство пользователя**

Для проверки работоспособности программы выполните следующие действия:

1. Запустите программу с помощью клавиши F5
2. Введите первый полином следуя инструкциям программы
3. Введите второй полином следуя инструкциям программы
4. Посмотрите на результат работы программы, который будет выведен автоматически после ввода двух полиномов.

**Руководство программиста**

Методы классов «Моном» и «Полином», описание алгоритмов.

**Описание класса TMonom:**

class TMonom;

typedef TMonom\* pMonom;

class TMonom

{

protected:

double coef;

int deg;

pMonom pnext;

public:

TMonom(double \_coef = 1, int \_deg = 0, pMonom \_pnext = NULL);

double getcoef() const;

int getdeg() const;

pMonom getpnext() const;

void setcoef(double co);

void setdeg(int de);

void setpnext(pMonom pn);

};

**Описание класса TPolinom:**

class TPolinom

{

protected:

pMonom pFirst;

public:

TPolinom();

void InsMonom(double coef, int deg);

int DelMonom(int deg);

void Transformation();

TPolinom SUMMA1(TPolinom& op2);

void SUMMA2(TPolinom& op2);

friend istream& operator >> (istream& is, TPolinom& pol);

};

**Описание алгоритмов**

**Класс TMonom:**

double getcoef() const – возвращает значение коэффициента

int getdeg() – возвращает свёрнутую степень

pMonom getpnext() const – возвращает указатель на следующее звено списка

void setcoef(double co) – получает значение коэффициента монома, которое будет передано в функцию

void sestdeg(int de) – получает значение степени монома, которое будет передано в функцию

void setpnext(pMonom pn) – получает значение указателя монома, которое будет передано в функцию

**Класс TPolinom:**

void InsMonom(double coef, int deg) – Вставка монома с коэффициентом coef и степенью deg.

int DelMonom(int deg) – Удаление монома со степенью deg.

void Transformation – обход полинома и печать результата

void SUMMA1(TPolinom &op2) – сложение полиномов без изменения слагаемых.

void SUMMA2(TPolinom &op2) – сложение полиномов с изменением первого слагаемого.

**Заключение**

В ходе выполнения лабораторной работы мной был разработан класс, позволяющий производить алгебраические операции с полиномами, для выполнения задачи использовалась алгебраическая структура «линейный список»

**Литература**

Под редакцией Гергеля В.П. Учебно-методическое пособие. – Нижний Новгород: Нижегородский госуниверситет, 2017. – 105с.

**Приложение**

**Файл monom.h:**

#pragma once

#ifndef \_MONOM\_H

#define \_MONOM\_H

#include <iostream>

using namespace std;

class TMonom;

typedef TMonom\* pMonom;

class TMonom

{

protected:

double coef;

int deg;

pMonom pnext;

public:

TMonom(double \_coef = 1, int \_deg = 0, pMonom \_pnext = NULL);

double getcoef() const;

int getdeg() const;

pMonom getpnext() const;

void setcoef(double co);

void setdeg(int de);

void setpnext(pMonom pn);

};

TMonom::TMonom(double \_coef, int \_deg, pMonom \_pnext)

{

coef = \_coef;

deg = \_deg;

pnext = \_pnext;

}

double TMonom::getcoef() const

{

return coef;

}

int TMonom::getdeg() const

{

return deg;

}

pMonom TMonom::getpnext() const

{

return pnext;

}

void TMonom::setcoef(double c)

{

coef = c;

}

void TMonom::setdeg(int d)

{

deg = d;

}

void TMonom::setpnext(pMonom pn)

{

pnext = pn;

}

#endif

**Файл TPolinom.h:**

#pragma once

#ifndef \_POLINOM\_H

#define \_POLINOM\_H

#include <iostream>

#include "monom.h"

using namespace std;

const double EPS = 0.00000001;

class TPolinom

{

protected:

pMonom pFirst;

public:

TPolinom();

void InsMonom(double coef, int deg);

int DelMonom(int deg);

void Transformation();

TPolinom SUMMA1(TPolinom& op2);

void SUMMA2(TPolinom& op2);

friend istream& operator >> (istream& is, TPolinom& pol);

};

TPolinom::TPolinom()

{

pMonom p = new TMonom(0, -1, NULL);

pFirst = p;

p->setpnext(pFirst);

}

void TPolinom::InsMonom(double coef, int deg)

{

pMonom next, present;

next = present = pFirst;

if (coef != 0)

{

if (next->getpnext() != pFirst)

{

next = next->getpnext();

while (next->getdeg() > deg)

{

present = next;

next = next->getpnext();

}

}

if (next->getdeg() == deg)

{

double CurrentCoef;

CurrentCoef = next->getcoef() + coef;

if (CurrentCoef <= EPS && CurrentCoef >= -EPS)

DelMonom(deg);

else

next->setcoef(CurrentCoef);

}

else

{

pMonom pm = new TMonom(coef, deg, next);

present->setpnext(pm);

}

}

}

int TPolinom::DelMonom(int deg)

{

pMonom next, present;

int result = -1;

next = present = pFirst;

if (next->getpnext() == pFirst)

return result;

else

{

next = next->getpnext();

while (next->getdeg() != deg && next->getdeg() != -1)

{

present = next;

next = next->getpnext();

}

if (next->getdeg() == -1)

return result;

else

{

present->setpnext(next->getpnext());

result = next->getdeg();

delete next;

next = NULL;

}

}

return result;

}

void TPolinom::Transformation()

{

int i = 0, flag = 1, flag2 = 1, beginflag = 1;

pMonom p;

p = pFirst;

if (p->getpnext() != pFirst)

{

p = p->getpnext();

while (p->getdeg() != -1)

{

flag = 0;

flag2 = 0;

if (p->getcoef() > 0 && (abs(p->getcoef()) != 1 || (p->getdeg() / 100 == 0 && (p->getdeg() % 100) / 10 == 0 && (p->getdeg() % 10) == 0)))

{

if (!beginflag) cout << " +";

beginflag = 0;

cout << p->getcoef();

}

else

if (p->getcoef() > 0 && abs(p->getcoef()) == 1)

{

if (!beginflag) cout << " +";

beginflag = 0;

flag = 1;

}

else

if (p->getcoef() < 0 && (abs(p->getcoef()) != 1 || (p->getdeg() / 100 == 0 && (p->getdeg() % 100) / 10 == 0 && (p->getdeg() % 10) == 0)))

{

beginflag = 0;

cout << " " << p->getcoef();

}

else

{

beginflag = 0;

cout << " -";

flag = 1;

}

if (p->getdeg() / 100 != 0)

{

if (!flag) cout << "\*";

else flag = 0;

if (p->getdeg() / 100 == 1)

cout << "x";

else

cout << "x^" << p->getdeg() / 100;

flag2 = 1;

}

if ((p->getdeg() % 100) / 10 != 0)

{

if (!flag) cout << "\*";

else flag = 0;

if ((p->getdeg() % 100) / 10 == 1)

cout << "y";

else

cout << "y^" << (p->getdeg() % 100) / 10;

flag2 = 1;

}

if ((p->getdeg() % 10) != 0)

{

if (!flag) cout << "\*";

else flag = 0;

if ((p->getdeg() % 10) == 1)

cout << "z";

else

cout << "z^" << (p->getdeg() % 10);

flag2 = 1;

}

p = p->getpnext();

}

}

else

cout << "Zero" << endl;

}

TPolinom TPolinom::SUMMA1(TPolinom& op2)

{

TPolinom res;

pMonom Previous1, Current1, Previous2, Current2;

double SumCoef;

Previous1 = Current1 = pFirst;

Previous2 = Current2 = op2.pFirst;

if (Current1->getpnext() != pFirst && Current2->getpnext() != op2.pFirst)

{

Current1 = Current1->getpnext();

Current2 = Current2->getpnext();

while (Current1->getdeg() != -1 && Current2->getdeg() != -1)

{

if (Current1->getdeg() > Current2->getdeg())

{

res.InsMonom(Current1->getcoef(), Current1->getdeg());

Previous1 = Current1;

Current1 = Current1->getpnext();

}

else

if (Current1->getdeg() < Current2->getdeg())

{

res.InsMonom(Current2->getcoef(), Current2->getdeg());

Previous2 = Current2;

Current2 = Current2->getpnext();

}

else

{

SumCoef = Current1->getcoef() + Current2->getcoef();

if (SumCoef)

res.InsMonom(SumCoef, Current1->getdeg());

Previous1 = Current1;

Current1 = Current1->getpnext();

Previous2 = Current2;

Current2 = Current2->getpnext();

}

}

}

else

{

Current1 = Current1->getpnext();

Current2 = Current2->getpnext();

}

while (Current1->getdeg() != -1)

{

res.InsMonom(Current1->getcoef(), Current1->getdeg());

Current1 = Current1->getpnext();

}

while (Current2->getdeg() != -1)

{

res.InsMonom(Current2->getcoef(), Current2->getdeg());

Current2 = Current2->getpnext();

}

return res;

}

void TPolinom::SUMMA2(TPolinom& op2)

{

pMonom Previous1, Current1, Previous2, Current2;

double SumCoef;

Previous1 = Current1 = pFirst;

Previous2 = Current2 = op2.pFirst;

if (Current1->getpnext() != pFirst && Current2->getpnext() != op2.pFirst)

{

Current1 = Current1->getpnext();

Current2 = Current2->getpnext();

while (Current2->getdeg() != -1)

{

if (Current1->getdeg() > Current2->getdeg())

{

Previous1 = Current1;

Current1 = Current1->getpnext();

}

else

if (Current1->getdeg() < Current2->getdeg())

{

Previous1->setpnext(Current2);

Previous2->setpnext(Current2->getpnext());

Current2->setpnext(Current1);

Previous1 = Current2;

Current2 = Previous2->getpnext();

}

else

{

SumCoef = Current1->getcoef() + Current2->getcoef();

if (SumCoef)

{

Current1->setcoef(SumCoef);

Previous1 = Current1;

Current1 = Current1->getpnext();

}

else

{

Current1 = Current1->getpnext();

DelMonom(Current2->getdeg());

}

Previous2 = Current2;

Current2 = Current2->getpnext();

}

}

}

else

if (Current1->getpnext() == pFirst && Current2->getpnext() != op2.pFirst)

{

while (Previous2->getpnext() != pFirst)

Previous2 = Previous2->getpnext();

Previous2->setpnext(pFirst);

pFirst->setpnext(op2.pFirst->getpnext());

op2.pFirst->setpnext(op2.pFirst);

}

}

istream& operator >> (istream& is, TPolinom& pol)

{

int i, choise = 1, degX, degY, degZ, deg;

double coef;

do {

cout << "Input the coefficient before monomial: ";

do { is >> coef; } while (coef == 0);

cout << "Input the degree of X: ";

do { is >> degX; } while (degX < 0 || degX > 9);

cout << "Input the degree of Y: ";

do { is >> degY; } while (degY < 0 || degY > 9);

cout << "Input the degree of Z: ";

do { is >> degZ; } while (degZ < 0 || degZ > 9);

deg = degX \* 100 + degY \* 10 + degZ;

pol.InsMonom(coef, deg);

cout << "If you want to add more, press 1. If not - 0;" << endl;

is >> choise;

} while (choise != 0);

return is;

}

#endif

**Файл Main.cpp**

#include <iostream>

#include "polinom.h"

using namespace std;

int main()

{

TPolinom P1, P2, P3;

cin >> P1;

cout << "MONOM 1 = ";

P1.Transformation();

cout << endl;

cin >> P2;

cout << "MONOM 2 = ";

P2.Transformation();

cout << endl;

cout << "SUM 1 = ";

P3 = P1.SUMMA1(P2);

P3.Transformation();

cout << endl;

cout << "MONOM 1 = ";

P1.Transformation();

cout << endl;

cout << "MONOM 2 = ";

P2.Transformation();

cout << endl;

cout << endl;

cout << endl;

cout << "SUM 2 = ";

P1.SUMMA2(P2);

P1.Transformation();

cout << endl;

cout << "MONOM 1 = ";

P1.Transformation();

cout << endl;

cout << "MONOM 2 = ";

P2.Transformation();

cout << endl;

return 0;

}