

计算机高级语言上机实习报告

一元稀疏多项式运算器

班级：191174 班

学号：20161001764

姓名：牟鑫一

日期：2018.11.23

一、实习题目与要求

1.1 一元稀疏多项式运算器

- 输入并建立两个多项式;
- 多项式 a 与 b 相加, 建立和多项式 c;
- 多项式 a 与 b 相减, 建立差多项式 d;
- 输入多项式 a, b, c, d。

输出格式: 比如多项式 a 为: $A(x)=c_1x^{e_1}+c_2x^{e_2}+...+c_mx^{e_m}$ (c_i 和 e_i 分别为第 i 项的系数和指数, 且按各项指数的升幂排列, 即 $0 \leq e_1 < e_2 < ... < e_m$)。

二、需求分析

2.1 问题描述

设计一个一元稀疏多项式简单运算器。

在输入栏中依次输入多项式 A 和 B, 将其转换为功能要求中的格式, 选择要进行的运算(加法或减法)。先比较指数, 指数相同则进行系数运算, 如果两系数运算结果为 0, 则结果多项式中不存储该指数项。

2.2 系统环境

Windows、Linux、macOS 等能运行 C++ 程序的系统

2.3 运行要求

Visual Studio 2017

三、概要设计

3.1 数据结构的设计

本程序中利用带头结点的单链表来存储多项式，是线性结构。

3.2 存储结构的设计

利用带头结点的单链表来存储多项式。

3.3 算法设计

依次输入多项式 A 每一项的系数和指数，以输入 0、-1 结束，继续输入多项式 B 每一项的系数和指数，以输入 0、-1 结束，对输入的多项式 A、B 进行相加运算得到多项式 C 或进行相减运算得到多项式 D，输出多项式 C 或 D。

3.4 模块设计



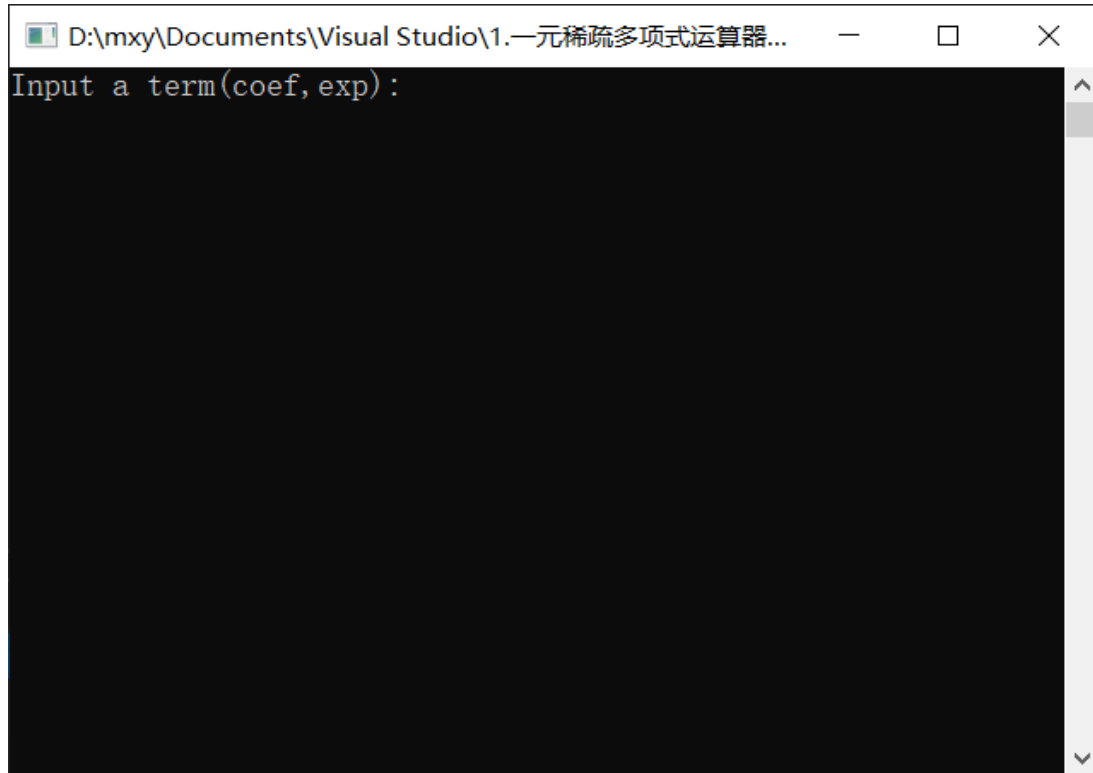
四、详细设计

4.1 类的函数成员和成员函数的设计

- 结构 Term
 - 成员函数: InsertAfter (float c, int e) 链表尾插入
 - 函数成员:
 - float coef; 系数
 - int exp; 指数
 - float c; 系数
 - int e; 指数
- 类 Polynomial 成员函数:
 - Polynomial () 构造函数, 建立空链
 - Polynomial (Polynomial&R) 复制构造函数
 - int maxOrder () 计算最大阶数
 - friend ostream&operator<<(ostream&,const Polynomial&)
 //输出多项式链表
 - friend istream&operator>>(istream&,Polynomial&)
 //导入输入的系数和指数
 - friend Polynomial operator + (Polynomial&,Polynomial&)
 //加法的重载
 - friend Polynomial operator - (Polynomial&,Polynomial&)
 //减法的重载

4.2 界面设计

4.2.1 初始:



4.2.2 输入多项式 A

$1+x+x^2+x^3+x^4+x^5$ (当输入 0 -1 时, 结束输入)



4.2.3 输入多项式 B

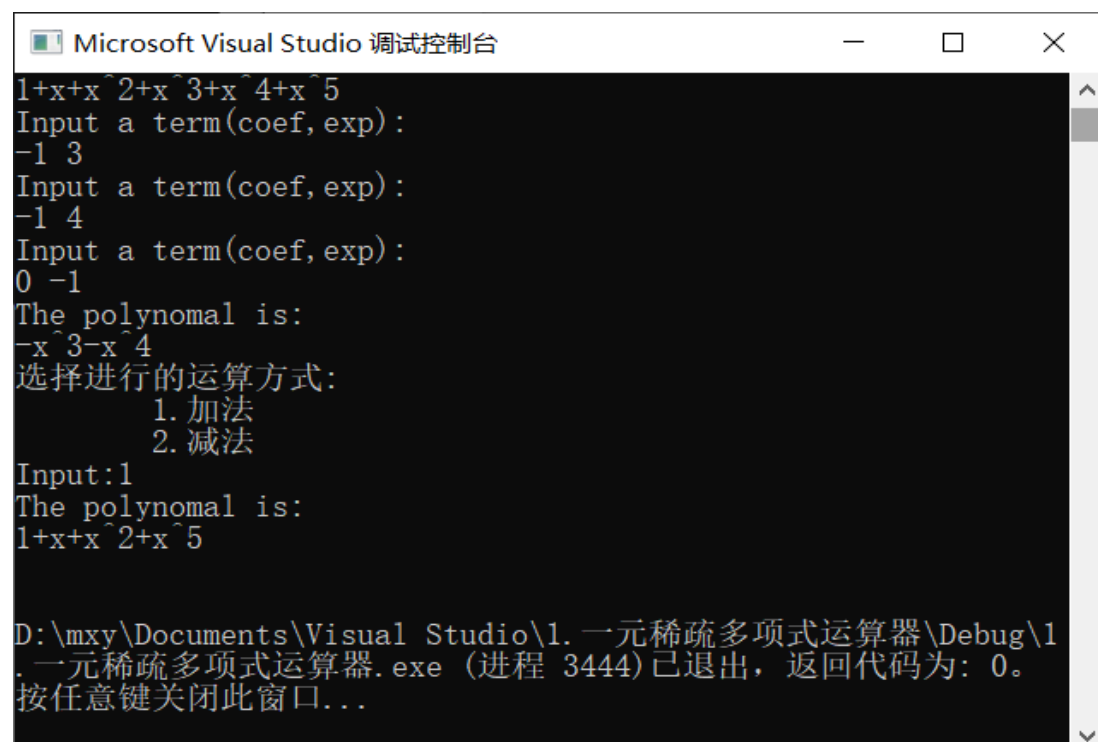
$-x^3-x^4$



```
D:\mxy\Documents\Visual Studio\1.一元稀疏多项式运算器...
Input a term(coef,exp):
0 -1
The polynomial is:
1+x+x^2+x^3+x^4+x^5
Input a term(coef,exp):
-1 3
Input a term(coef,exp):
-1 4
Input a term(coef,exp):
0 -1
The polynomial is:
-x^3-x^4
选择进行的运算方式:
    1. 加法
    2. 减法
Input:
```

4.2.4 选择运算方式

选择加法运算，输出运算结果



```
Microsoft Visual Studio 调试控制台
1+x+x^2+x^3+x^4+x^5
Input a term(coef,exp):
-1 3
Input a term(coef,exp):
-1 4
Input a term(coef,exp):
0 -1
The polynomial is:
-x^3-x^4
选择进行的运算方式:
    1. 加法
    2. 减法
Input:1
The polynomial is:
1+x+x^2+x^5
D:\mxy\Documents\Visual Studio\1.一元稀疏多项式运算器\Debug\1
一元稀疏多项式运算器.exe (进程 3444) 已退出，返回代码为：0。
按任意键关闭此窗口...
```

4.3 其他模块设计与实现

4.3.1 系数为 0 的情况

系数为 0 的项，不做存储

例如：系数为 0，指数任意，该项值即为 0，不做存储。

4.3.2 系数为 ± 1 的情况

只显示正负不显示字符“1”

例如：系数为-1，指数为 2，输出时应输出“-x^2”，而不是“-1x^2”。

4.3.3 减法运算

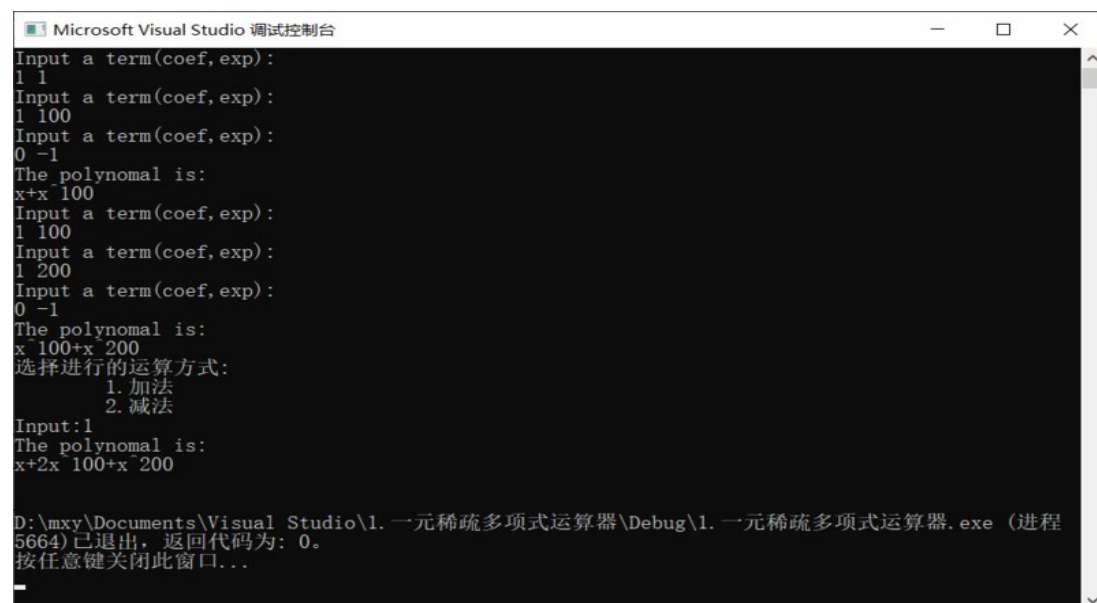
减法运算中多余的项系数取反后存储

例如：多余项系数为 1，指数为 2，存到链表尾时将系数取反存为-1，指数不变为 2

五、测试

5.1 测试一

- $(x+x^{100})+(x^{100}+x^{200})=(x+2x^{100}+x^{200})$

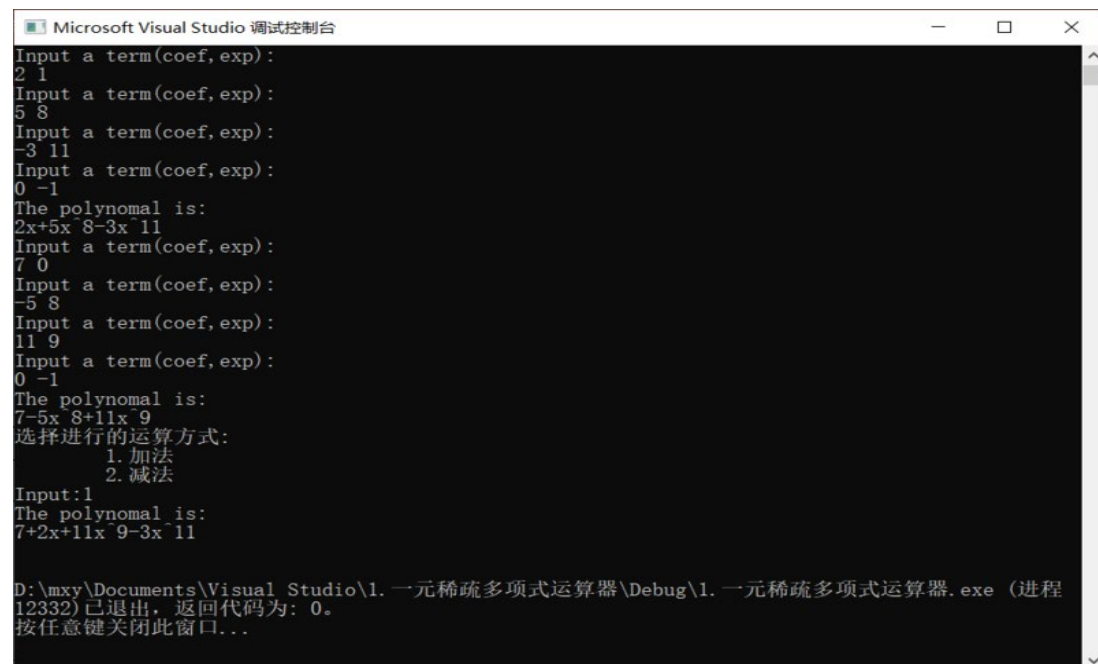


```
Microsoft Visual Studio 调试控制台
Input a term(coef,exp):
1 1
Input a term(coef,exp):
1 100
Input a term(coef,exp):
0 -1
The polynomial is:
x+x^100
Input a term(coef,exp):
1 100
Input a term(coef,exp):
1 200
Input a term(coef,exp):
0 -1
The polynomial is:
x^100+x^200
选择进行的运算方式:
1. 加法
2. 减法
Input:1
The polynomial is:
x+2x^100+x^200

D:\mxy\Documents\Visual Studio\1. 一元稀疏多项式运算器\Debug\1. 一元稀疏多项式运算器.exe (进程
5664)已退出。 返回代码为: 0。
按任意键关闭此窗口...
```

5.2 测试二

- $(2x+5x^8-3x^{11})+(7-5x^8+11x^9)=(7+2x+11x^9-3x^{11})$

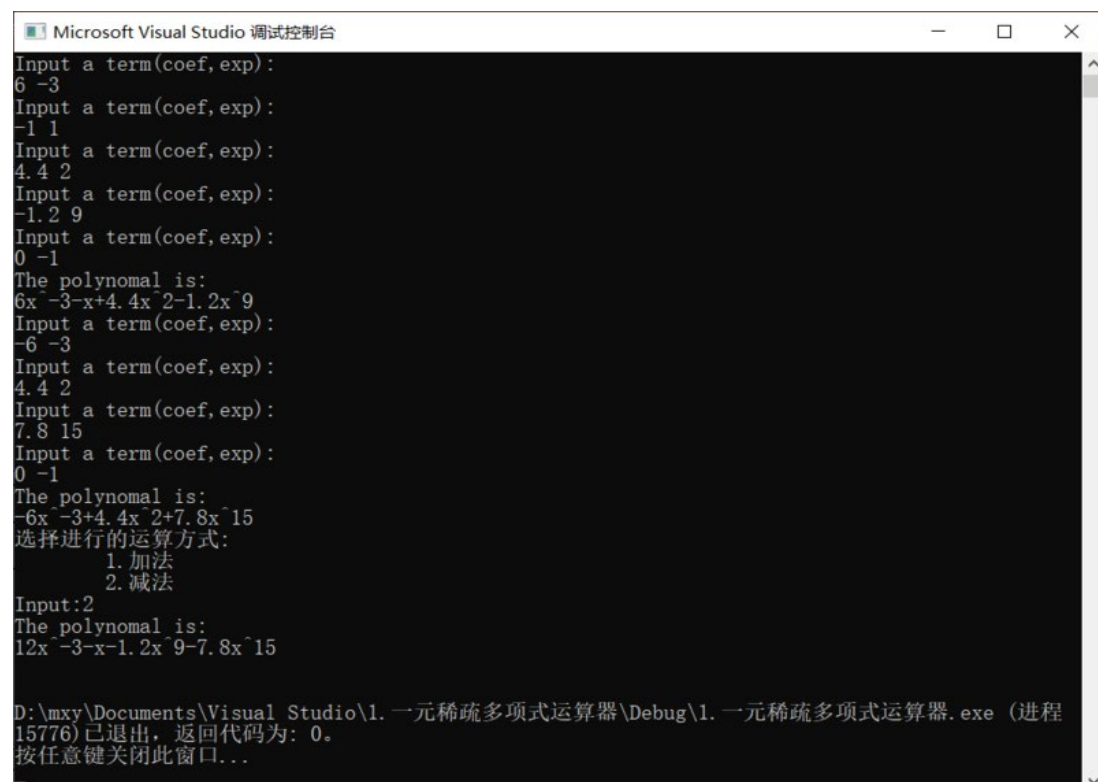


```
Microsoft Visual Studio 调试控制台
Input a term(coef,exp):
2 1
Input a term(coef,exp):
5 8
Input a term(coef,exp):
-3 11
Input a term(coef,exp):
0 -1
The polynomial is:
2x+5x 8-3x 11
Input a term(coef,exp):
7 0
Input a term(coef,exp):
-5 8
Input a term(coef,exp):
11 9
Input a term(coef,exp):
0 -1
The polynomial is:
7-5x 8+11x 9
选择进行的运算方式:
1. 加法
2. 减法
Input:1
The polynomial is:
7+2x+11x 9-3x 11

D:\mxy\Documents\Visual Studio\1. 一元稀疏多项式运算器\Debug\1. 一元稀疏多项式运算器.exe (进程
12332)已退出, 返回代码为: 0。
按任意键关闭此窗口...
```

5.3 测试三

- $(6x^{-3}-x+4.4x^2-1.2x^9)-(-6x^{-3}+4.4x^2+7.8x^{15})=(12x^{-3}-x-1.2x^9-7.8x^{15})$



```
Microsoft Visual Studio 调试控制台
Input a term(coef,exp):
6 -3
Input a term(coef,exp):
-1 1
Input a term(coef,exp):
4.4 2
Input a term(coef,exp):
-1.2 9
Input a term(coef,exp):
0 -1
The polynomial is:
6x -3-x+4.4x 2-1.2x 9
Input a term(coef,exp):
-6 -3
Input a term(coef,exp):
4.4 2
Input a term(coef,exp):
7.8 15
Input a term(coef,exp):
0 -1
The polynomial is:
-6x -3+4.4x 2+7.8x 15
选择进行的运算方式:
1. 加法
2. 减法
Input:2
The polynomial is:
12x -3-x-1.2x 9-7.8x 15

D:\mxy\Documents\Visual Studio\1. 一元稀疏多项式运算器\Debug\1. 一元稀疏多项式运算器.exe (进程
15776)已退出, 返回代码为: 0。
按任意键关闭此窗口...
```


六、附录

6.1 程序源代码:

```
#include "pch.h"
#include <iostream>
#include <math.h>
using namespace std;

struct Term {
    float coef;           //系数
    int exp;              //指数
    Term *link;
    Term(float c, int e, Term *next = NULL)
    {
        coef = c; exp = e; link = next;
    }
    Term *InsertAfter(float c, int e);
    friend ostream& operator<<(ostream&, const Term&);
};

class Polynomial {
public:
    Polynomial() { first = new Term(0, -1); } //建立空链表
    Polynomial(Polynomial&R);
    int maxOrder();
    Term *getHead() const { return first; } //取得多项式单链表的表头指针
private:
    Term *first;
    friend ostream& operator<<(ostream&, const Polynomial&);
    friend istream& operator>>(istream&, Polynomial&);
    friend Polynomial operator+(Polynomial&, Polynomial&);
    friend Polynomial operator-(Polynomial&, Polynomial&);
};

Term*Term::InsertAfter(float c, int e) { //在当前由this指针指示的项后面插入一个新项
    link = new Term(c, e, link);
    return link;
};

ostream&operator<<(ostream&out, const Term&x) { //输出一个项x的内容到输出流out中
    if (x.coef == 0.0) return out; //系数为0返回输出流
```

```

    if (x.coef != 1 && x.coef != -1) out << x.coef;    //系数不为1或-1，输出系数
    switch (x.exp) {
    case 0: {
        if (fabs(x.coef) == 1) out << x.coef;    //如果指数为0且系数为1或-1，直接输出系数
    } break;
    case 1: {
        if (x.coef == -1) out << "-x";    //指数为1的情况下，如果系数为-1则输出-x
        else out << "x";    //否则输出x
    } break;
    default: {
        if (x.coef == -1) out << "-x^" << x.exp;
        else out << "x^" << x.exp;
    } break;
    }
    return out;
};

```

```

Polynomial::Polynomial(Polynomial&R) {    //用已有多项式对象R初始化当前多项式对象
    first = new Term(0, -1);
    Term *destptr = first, *srcptr = R.getHead()->link;
    while (srcptr != NULL) {
        destptr->InsertAfter(srcptr->coef, srcptr->exp);
        srcptr = srcptr->link;
        destptr = destptr->link;
    }
};

```

```

int Polynomial::maxOrder() {    //计算最大阶数
    Term *current = first;
    while (current->link != NULL) current = current->link;
    return current->exp;
};

```

```

istream&operator>>(istream&in, Polynomial&x) { //输入多项式
    Term *rear = x.getHead();
    float c;
    int e;
    while (1) {
        cout << "Input a term(coef,exp):" << endl;
        in >> c >> e;
        if (c == 0 && e == -1) break;    //系数为0，指数为-1时输入结束
        rear = rear->InsertAfter(c, e);
    }
};

```

```

        return in;
};

ostream&operator<<(ostream&out, Polynomial&x) { //输出得到的和/差多项式链表
    Term *current = x.getHead()->link;
    cout << "The polynomial is:" << endl;
    bool isEnd = true;
    while (current != NULL) {
        if (isEnd == false && current->coef > 0.0) out << "+"; //输入未结束则输出一个 "+"
        isEnd = false;
        out << *current; //输出当前项
        current = current->link;
    }
    out << endl;
    return out;
};

```

```

Polynomial operator + (Polynomial&A, Polynomial&B) { //加法的重载
    Term*pa, *pb, *pc, *p;
    float temp;
    Polynomial C; pc = C.first;
    pa = A.getHead()->link; pb = B.getHead()->link;
    while (pa != NULL && pb != NULL) {
        if (pa->exp == pb->exp) {
            temp = pa->coef + pb->coef;
            if (fabs(temp) > 0.001) //系数相加不为0
                pc = pc->InsertAfter(temp, pa->exp);
            pa = pa->link; pb = pb->link;
        }
        else if (pa->exp < pb->exp) {
            pc = pc->InsertAfter(pa->coef, pa->exp);
            pa = pa->link;
        }
        else {
            pc = pc->InsertAfter(pb->coef, pb->exp);
            pb = pb->link;
        }
    }
    if (pa != NULL) p = pa; //处理链剩余部分
    else p = pb;
    while (p != NULL) {
        pc = pc->InsertAfter(p->coef, p->exp);
        p = p->link;
    }
}

```

```

    }
    return C;
};

Polynomial operator - (Polynomial&A, Polynomial&B) { //减法的重载
    Term*pa, *pb, *pd, *p;
    float temp;
    Polynomial D; pd = D.first;
    pa = A.getHead()->link; pb = B.getHead()->link;
    while (pa != NULL && pb != NULL) {
        if (pa->exp == pb->exp) {
            temp = pa->coef - pb->coef;
            if (fabs(temp) > 0.001)
                pd = pd->InsertAfter(temp, pa->exp);
            pa = pa->link; pb = pb->link;
        }
        else if (pa->exp < pb->exp) {
            pd = pd->InsertAfter(pa->coef, pa->exp);
            pa = pa->link;
        }
        else {
            pd = pd->InsertAfter(pb->coef, pb->exp);
            pb = pb->link;
        }
    }
    if (pa != NULL) p = pa;
    else p = pb;
    while (p != NULL) {
        pd = pd->InsertAfter(-(p->coef), p->exp);
        p = p->link;
    }
    return D;
};

int main() {
    int a;
    Polynomial A, B, C, D;
    cin >> A;
    cout << A;
    cin >> B;
    cout << B;
    cout << "选择进行的运算方式:" << endl;
    cout << "1. 加法" << endl;
    cout << "2. 减法" << endl;

```

```
cout << "Input:";
cin >> a;
if (a == 1) {
    C = A + B;
    cout << C << endl;
}
else {
    D = A - B;
    cout << D << endl;
}
return 0;
}
```