# 目录

一、实习任务.............................

二、实习过程............

1.程序设计思想

(1)、

(2)、

(3)、

2.代码实现

三、程序运行结果与分析..

四、用户使用说明书和参考资料................

五、实习体会............

**一元多项式的运算**

## 实习任务

## 题目要求:

按照指数能够按照指数降序排列建立并输出多项式;

　 能够完成两个多项式的相加、相减

## 本程序的主要内容

(1)两个多项式的输入方式的选择,第一种是以字符串的形式输入一元多项式,第二种是按照系数指数的顺序依次输入多项式的指数和系数。

(2)输出两个多项式的降序排列

(3)选择对多项式进行的+，- 操作

(4)对两个多项式进行操作后显示操作后降序排列的多项式

(5)扩充功能--多项式的乘法运算

## 实习过程

## 流程图1程序流程图:

## 程序实现:

(1)选择输入的方式(字符串式输入、系数指数式输入);

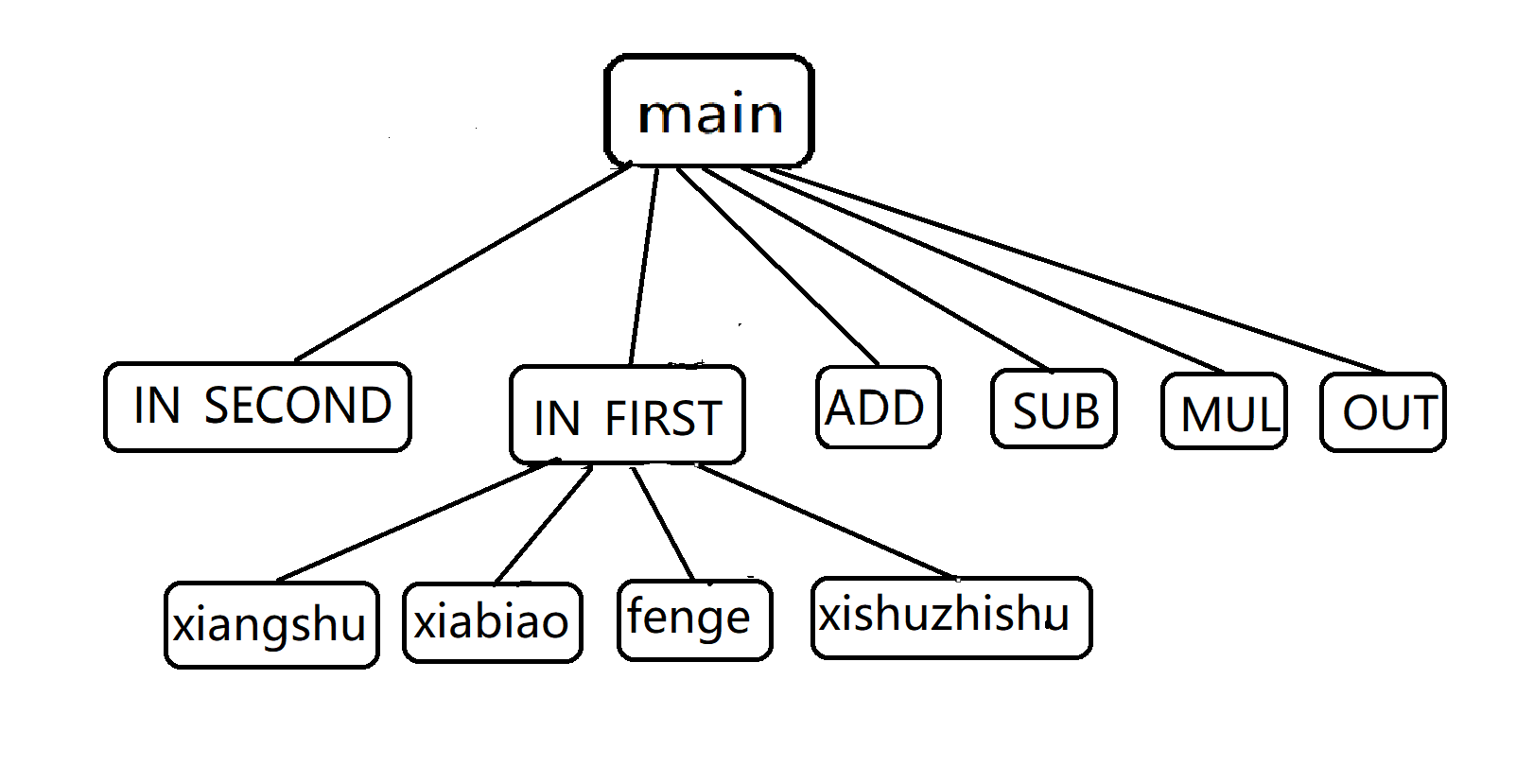
(2)输出输入的两个一元多项式的降序排列;

(3)选择所要进行的操作(加、减、乘);

(4)调用函数，实现选择的操作;

(5)降序输出最终结果;

## 函数调用关系:



## 系统子程序及功能如下

(1)字符串式输入一元多项式:void IN\_FIRST();

(2)求出一元多项式中共含多少项:int xiangshu(string str);

(3)将一元多项式中的项存入字符串数组:string\* fenge(string str,int q[],int c,string strz[]);

(4)求出每一项的开始的位置:int \*xiabiao(string str,int xb[]);

(5)将多项式的系数和指数存入数组:void Cofe\_and\_Exp(string xiang[],int c,int str1[]);

(6)系数指数式输入一元多项式:void IN\_SECOND();

(7)多项式的加法运算:void ADD();

(8)多项式的减法:void SUB();

(9)多项式的乘法:void MUL();

(10)输出多项式:void OUT();

## 程序设计思想及核心代码：

(1)本问题采用顺序存储方式，在第一种输入方式中，先将字符串中的系数和指数提取出来，存入数组中，再将数据存储到STL容器中的map结构，再进行一元多项式的加、减、乘运算;在第二种输入方式中，直接将输入的系数和指数存储到map中，再进行运算。

(2)字符串式输入以及处理一元多项式;

## ①传入一个字符串形式的一元多项式。根据字符串加号和减号的数量得到多项式共有多少项，将数量作为返回值返回;

②把一元多项式的每项第一位的加号或减号所在字符串的下标返回;

③把一元多项式字符串按照加号或减号分割成一个个单独的字符串，将单独的字符串储存在字符串数组中;

④逐个分析一元多项式字符串数组里的单个项，把项的系数和指数依次存如整形数组中。

(3)系数、指数式输入一元多项式

按照屏幕中给出的例子进行输入一元多项式。

(4)按照题目要求，需要将输入的一元多项式按照降序排列的顺序进行输出，即当输入完一元多项式后，在屏幕上打印出该一元多项式的降序排列。

(5)一元多项式的运算

由于STL中的map容器是以一种(key-value)的结构进行存储根据一元多项式的定义可知，两个多项式进行运算之后，结果中是不会出现指数相同的项，由此可以判断，选择map是可以解决该问题的。

①一元多项式的加法

我们分别用map1和map2分别存储第一个多项式和第二个多项式，然后通过迭代器访问map1和map2，再定义一个map3对运算结果进行存储，比较迭代器所在位置的指数(也就是map中key的值)的大小，将指数较大的值的指数和系数存入到结果当中，迭代器相应的向后挪动一位，如果遇到指数相等，则将系数相加指数不变存到map3中。

②一元多项式的减法

用map1和map2分别存储第一个多项式和第二个多项式，然后通过迭代器访问map1和map2，再定义一个map3对运算结果进行存储，比较迭代器所在位置的指数(也就是map中key的值)的大小，将指数较大的值的指数和系数存入到结果当中，迭代器相应的向后挪动一位，如果遇到指数相等，用map1中存储的系数减map2中存储的系数，结果存放在map3中。

③一元多项式的乘法

通过双重for循环对map1中的每一项分别与map2中的每一项相乘，再存到map3中。

④结果输出

对存放在map3中的结果按照一元多项式的方式进行输出。

## 源代码

#include<bits/stdc++.h>

using namespace std;

int xb1[10000];//储存+/-的下标

int xb2[10000];

string strz1[10000];//一个字符串是一个多项式的项

string strz2[10000];

int shu1[10000]; //储存多项式的系数和指数

int shu2[10000];

int c1;

int c2;

string\* fenge(string str,int q[],int c,string strz[]){

if(c==1){

strz[0]=str;

return strz;

}

for(int i=0;i<=c;i++){

strz[i]=str.substr(q[i],q[i+1]-q[i]);}

return strz;

}

int xiangshu(string str){

int c=0;

for(int i=0;i<str.length();i++){

if(str.substr(i,1)=="+"||str.substr(i,1)=="-")

c++;

}

for(int i=0;i<str.length()-1;i++){

if(str.substr(i,2)=="^-")

c--;

}

return c;

}

int \*xiabiao(string str,int xb[]){

int n=0;

for(int i=0;i<str.length();i++){ if(str.substr(i,1)=="+"||str.substr(i,1)=="-")

if(str[i-1]!='^')

xb[n++]=i;

}

return xb;

}

void Cofe\_and\_Exp(string xiang[],int c,int str1[]){

int n=0;

for(int i =0;i<c;i++){

string ss = xiang[i];

int h1 = ss.find("^");

int h2 = ss.length();

int h3 = ss.find("X");

if(h3==-1){

str1[n++] = atoi(ss.c\_str());

str1[n++] = 0;

}

else{

if(h1==-1){

if(h3!=1){

str1[n++] = atoi(ss.c\_str());

str1[n++] = 1;

}

else{

if(ss[0]=='+')

str1[n++] = 1;

else

str1[n++] = -1;

str1[n++]= 1;

}

}

else{

if(h3!=1){

str1[n] = atoi(ss.c\_str());

n++;

string sss = ss.substr(h1+1,h2-h1-1);

str1[n] = atoi(sss.c\_str());

n++;

}else{

if(ss[0]=='+'){

str1[n++] = 1;

}

else{

str1[n++] = -1;

}

string sss = ss.substr(h1+1,h2-h1-1);

str1[n++] = atoi(sss.c\_str());}}}}}

int IN\_FIRST(){

string str1,str2;

printf("请输入第一个字符串:\n");

getline(cin,str1);

if(str1.substr(0,1)!="-")

str1='+'+str1;

c1 =xiangshu(str1);

int \*q1 = xiabiao(str1,xb1);

string \*p1 = fenge(str1,q1,c1,strz1);

Cofe\_and\_Exp(p1,c1,shu1);

printf("请输入第二个字符串:\n");

getline(cin,str2);

if(str2.substr(0,1)!="-")

str2='+'+str2;

c2 =xiangshu(str2);

int \*q2 = xiabiao(str2,xb2);

string \*p2 = fenge(str2,q2,c2,strz2);

Cofe\_and\_Exp(p2,c2,shu2);

return 0;

}

map<int,int,greater<int> > m1;

map<int,int,greater<int> > m2;

map<int,int,greater<int> > m3;

void IN\_SECOND(){

int k,n;

bool f1=false;

bool f2=false;

printf("请输入第一个一元多项式:\n");

while(scanf("%d",&k)&&k!=0){

scanf("%d",&n);

m1[n]+=k;

}

for(map<int,int,greater<int> >::iterator it=m1.begin();it!=m1.end();it++){

if(it!=m1.begin()){

if(it->second>0){

printf("+");

}

}

int k=it->second;

int n=it->first;

if(k!=1&&k!=-1){

if(n==0){

printf("%d",k);

}else if(n==1){

printf("%dx",k);

}else{

printf("%dx^%d",k,n);

}

}else if(k==1){

if(n==0){

printf("%d",k);

}else if(n==1){

printf("x");

}else{

printf("x^%d",n);

}

}else if(k==-1){

if(n==0){

printf("%d",k);

}else if(n==1){

printf("-x");

}else{

printf("-x^%d",n);

}

}

}

cout<<endl;

printf("请输入第二个一元多项式:\n");

while(scanf("%d",&k)&&k!=0){

scanf("%d",&n);

m2[n]+=k;

}

for(map<int,int,greater<int> >::iterator it=m2.begin();it!=m2.end();it++){

if(it!=m2.begin()){

if(it->second>0){

printf("+");

}

}

int k=it->second;

int n=it->first;

if(k!=1&&k!=-1){

if(n==0){

printf("%d",k);

}else if(n==1){

printf("%dx",k);

}else{

printf("%dx^%d",k,n);

}

}else if(k==1){

if(n==0){

printf("%d",k);

}else if(n==1){

printf("x");

}else{

printf("x^%d",n);

}

}else if(k==-1){

if(n==0){

printf("%d",k);

}else if(n==1){

printf("-x");

}else{

printf("-x^%d",n);

}

}

}

cout<<endl;

}

void ADD(){

map<int,int>::iterator it1=m1.begin();

map<int,int>::iterator it2=m2.begin();

while(it1!=m1.end()&&it2!=m2.end()){

int x1=it1->first;

int x2=it2->first;

int y1=it1->second;

int y2=it2->second;

if(x1>x2){

m3[x1]=y1;

it1++;

}else if(x1<x2){

m3[x2]=y2;

it2++;

}else{

int x=x1;

int y=y1+y2;

if(y!=0){

m3[x]=y;

}

it1++;

it2++;

}

}

if(it1==m1.end()&&it2!=m2.end()){

while(it2!=m2.end()){

m3[it2->first]=it2->second;

it2++;

}

}else if(it1!=m1.end()&&it2==m2.end()){

while(it1!=m1.end()){

m3[it1->first]=it1->second;

it1++;

}

}

}

void SUB(){

map<int,int>::iterator it1=m1.begin();

map<int,int>::iterator it2=m2.begin();

while(it1!=m1.end()&&it2!=m2.end()){

int x1=it1->first;

int x2=it2->first;

int y1=it1->second;

int y2=it2->second;

if(x1>x2){

m3[x1]=y1;

it1++;

}else if(x1<x2){

m3[x2]=-y2;

it2++;

}else{

int x=x1;

int y=y1-y2;

if(y!=0){

m3[x]=y;

}

it1++;

it2++;

}

}

if(it1==m1.end()&&it2!=m2.end()){

while(it2!=m2.end()){

m3[it2->first]=it2->second;

it2++;

}

}else if(it1!=m1.end()&&it2==m2.end()){

while(it1!=m1.end()){

m3[it1->first]=it1->second;

it1++;

}

}

}

void MUL(){

for(map<int,int>::iterator it1=m1.begin();it1!=m1.end();it1++){

for(map<int,int>::iterator it2=m2.begin();it2!=m2.end();it2++){

int x1=it1->first;

int k1=it1->second;

int x2=it2->first;

int k2=it2->second;

int k=k1\*k2;

int x=x1+x2;

m3[x]+=k;

}

}

}

void OUT(){

if(m3.empty()){

printf("0");

}

for(map<int,int,greater<int> >::iterator it=m3.begin();it!=m3.end();it++){

if(it!=m3.begin()){

if(it->second>0){

printf("+");

}

}

int k=it->second;

int n=it->first;

if(k!=1&&k!=-1){

if(n==0){

printf("%d",k);

}else if(n==1){

printf("%dx",k);

}else{

printf("%dx^%d",k,n);

}

}else if(k==1){

if(n==0){

printf("%d",k);

}else if(n==1){

printf("x");

}else{

printf("x^%d",n);

}

}else if(k==-1){

if(n==0){

printf("%d",k);

}else if(n==1){

printf("-x");

}else{

printf("-x^%d",n);

}

}

}

}

int main(){

printf("~~~~~~~~~~~~~~欢迎使用一元多项式计算器~~~~~~~~~~~~~~\n");

printf("~~ 目录 ~~\n");

printf("~~ 一元多项式的加法运算 ~~\n");

printf("~~ 一元多项式的减法运算 ~~\n");

printf("~~ \*扩充功能\* ~~\n");

printf("~~ 一元多项式的乘法运算 ~~\n");

printf("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n");

printf("请选择您要输入一元式的格式:\n");

printf("如果选择方式一请输入‘1’(※输入格式：'k0X^n0+'k1X^n1')\n");

printf("如果选择方式二请输入‘2’(※输入格式：1 2 3 4 0(1,3表示系数部分,2,4表示指数部分,最终以0表示停止输入))\n");

int temp;

scanf("%d",&temp);

getchar();

if(temp==1){

IN\_FIRST();

for(int i=0;i<c1\*2;i+=2){

m1[shu1[i+1]]+=shu1[i];

}

printf("第一个一元多项式降序排列为:\n");

for(map<int,int,greater<int> >::iterator it=m1.begin();it!=m1.end();it++){

if(it!=m1.begin()){

if(it->second>0){

printf("+");

}

}

int k=it->second;

int n=it->first;

if(k!=1&&k!=-1){

if(n==0){

printf("%d",k);

}else if(n==1){

printf("%dx",k);

}else{

printf("%dx^%d",k,n);

}

}else if(k==1){

if(n==0){

printf("%d",k);

}else if(n==1){

printf("x");

}else{

printf("x^%d",n);

}

}else if(k==-1){

if(n==0){

printf("%d",k);

}else if(n==1){

printf("-x");

}else{

printf("-x^%d",n);

}

}

}

cout<<endl;

for(int i=0;i<c2\*2;i+=2){

m2[shu2[i+1]]+=shu2[i];

}

printf("第二个一元多项式降序排列为:\n");

for(map<int,int,greater<int> >::iterator it=m2.begin();it!=m2.end();it++){

if(it!=m2.begin()){

if(it->second>0){

printf("+");

}

}

int k=it->second;

int n=it->first;

if(k!=1&&k!=-1){

if(n==0){

printf("%d",k);

}else if(n==1){

printf("%dx",k);

}else{

printf("%dx^%d",k,n);

}

}else if(k==1){

if(n==0){

printf("%d",k);

}else if(n==1){

printf("x");

}else{

printf("x^%d",n);

}

}else if(k==-1){

if(n==0){

printf("%d",k);

}else if(n==1){

printf("-x");

}else{

printf("-x^%d",n);

}

}

}

cout<<endl;

}else{

IN\_SECOND();

}

printf("如果想进行加法操作请输入'+';\n");

printf("如果想进行减法操作请输入'-';\n");

printf("如果想进行乘法操作请输入'\*';\n");

printf("请选择您要进行的操作:");

char c;

cin>>c;

switch(c){

case '+':ADD();break;

case '-':SUB();break;

case '\*':MUL();break;

}

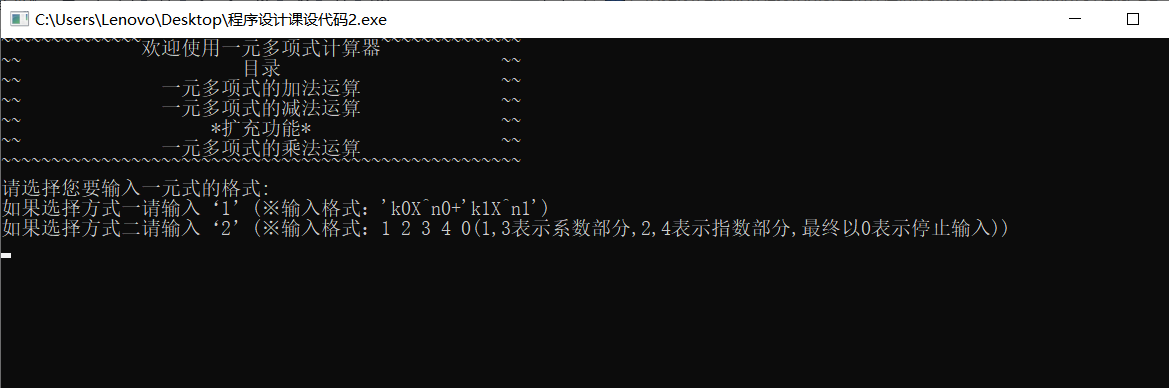
printf("运算结果结果为:\n");

OUT();

## }

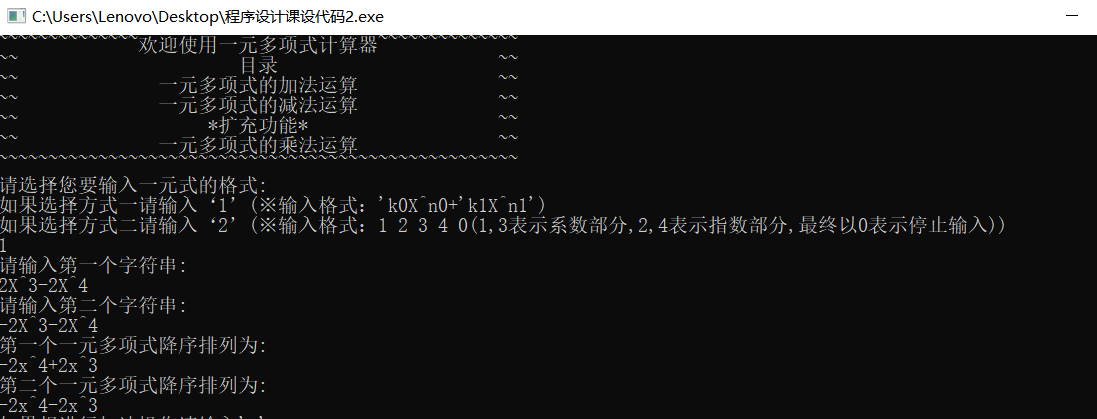
## 程序运行结果与分析

1. 目录项，以及选择输入一元式的格式：

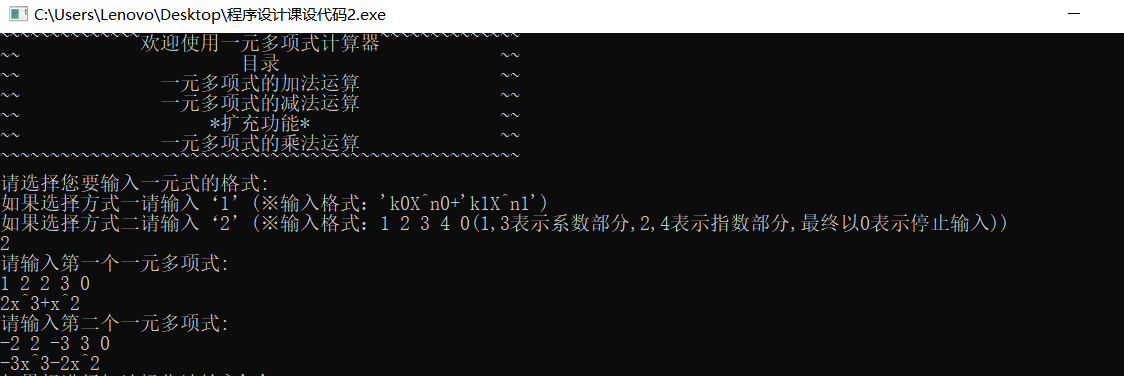


3.1目录

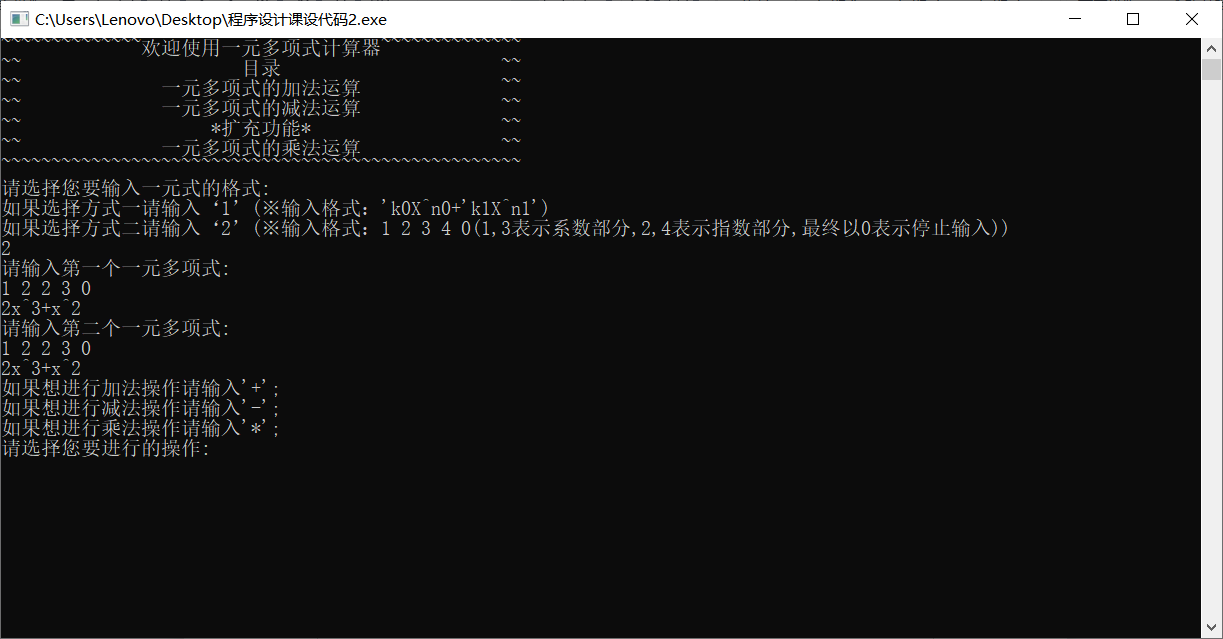
2.一元多项式输入后，需要将输入的一元多项式按照指数的降序排列进行输出，运行结果如下：



3.1-1字符串式输入方式

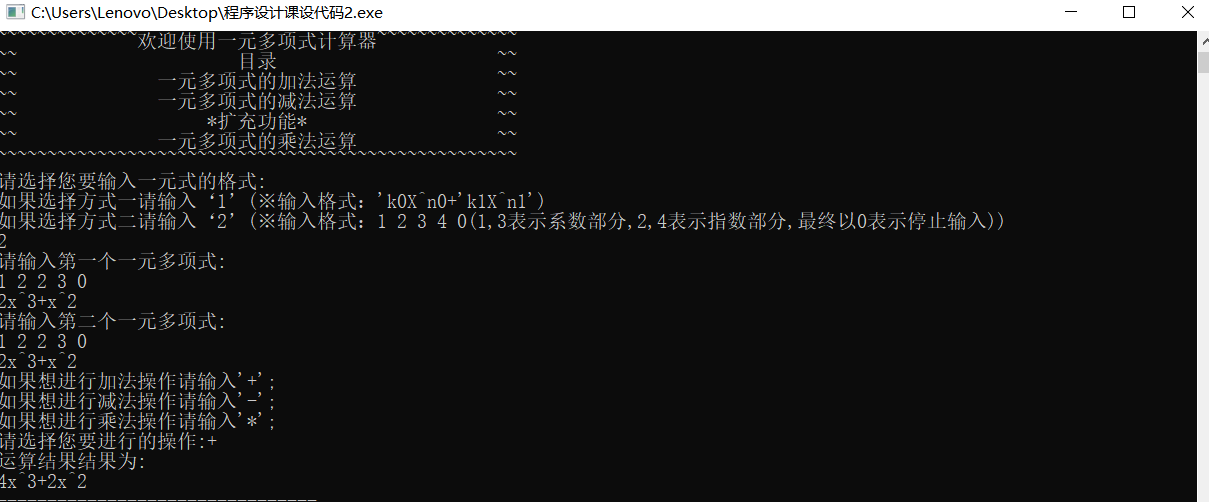


3.2-2系数指数输入方式

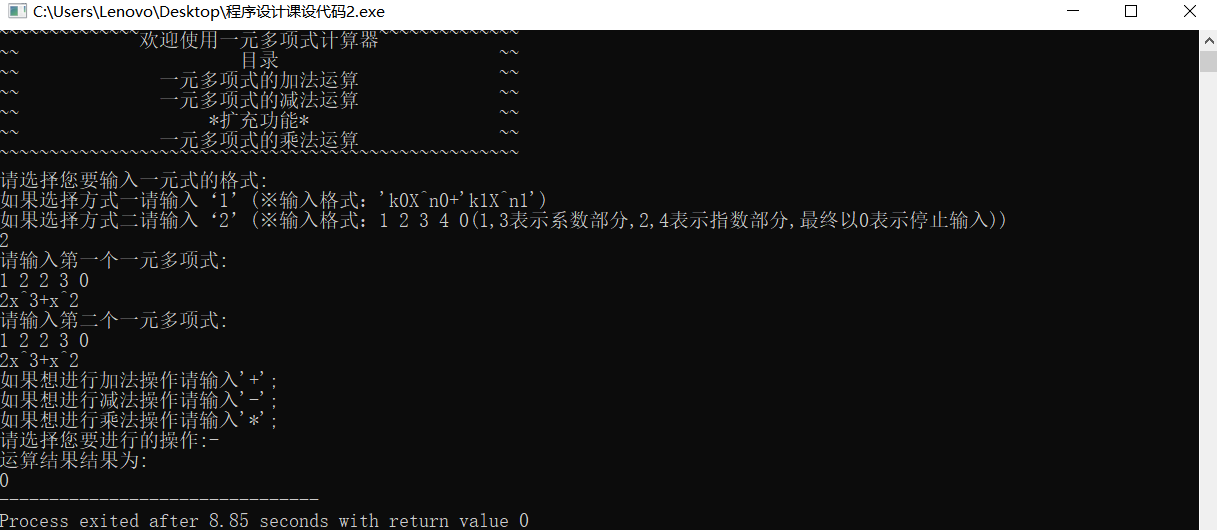
3.选择所要进行的运算，加法输入+，减法输入-，乘法输入\*：

3.3运算的选择

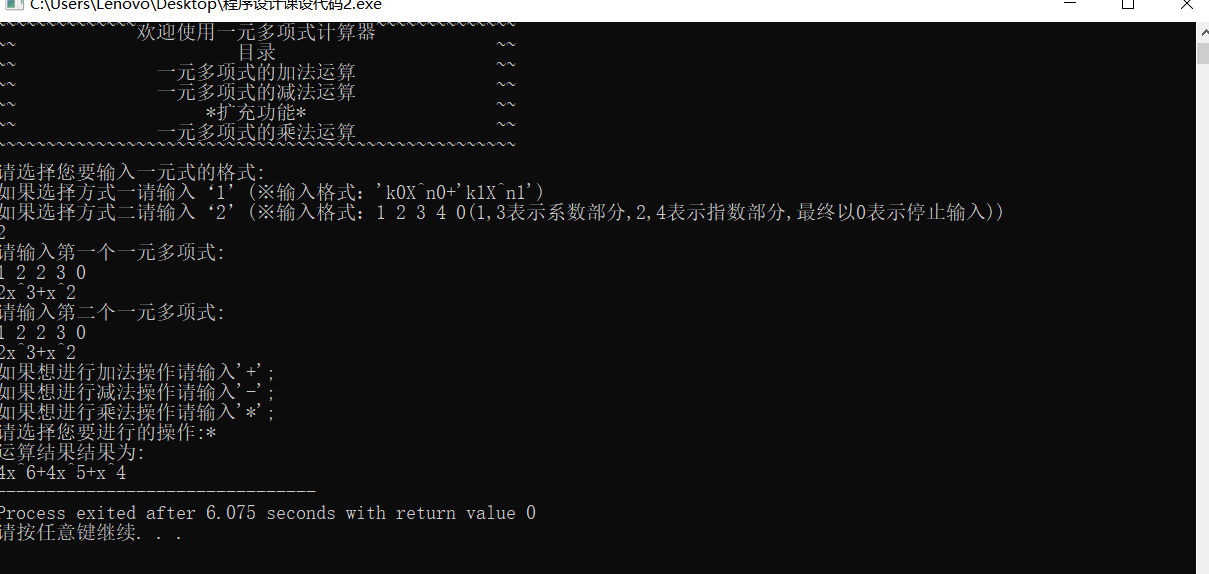
4.将最终的运算结果进行输出：



3.4-1加法运算结果



3.4-2减法运算结果



3.4-3乘法运算结果

## 用户使用说明书和参考资料

1.用户使用说明书

(1)本系统在DEV C++软件上运行，实现功能的语言为C++。

(2)运行程序后第一步会出现目录以及选择输入一元多项式的格式，并且会给出每个输入格式的样例，请根据样例进行输入一元多项式。

(3)输入之后会自动输出输入的多项式的降序排列

(4)将两个要进行运算的多项式输入完成之后，屏幕上会提示本系统可进行的运算，以及使用每种运算所要输入的字符，按照提示输入对应的字符

(5)输出运算后的结果

2.参考资料

(1)数据结构(C语言版) 严蔚敏 吴伟民 清华大学出版社2020.11

(2)算法笔记 胡凡 曾磊 机械工业出版社 2016.07

## 小组分工情况

本小组成员：孙赫，王立恒，苏浩

对于本次课设小组成员一起进行了深入探究:

首先我们对该问题进行了一个具体的分析，了解了如何对一元多项式的输入和存储，还分析了应该用什么结构去进行多项式的运算。认清了一元多项式的运算的主要是对一元多项式的存储，一元多项式的运算规则以及对指数大小的排序。

其次：我们共同完成菜单项，再使用switch函数使用户可以选择输入格式以及所进行的操作

小组分工中，代码阶段：苏浩负责设计目录格式代码，编写输出部分的代码，王立恒负责一元多项式以字符串形式输入时，处理字符串，将字符串中的系数和指数存放到数组当中，供运算时使用，孙赫负责一元多项式以(系数、指数)形式的输入，以及编写一元多项式的加、减、乘运算的代码，代码完成后，由我们三人共同完成代码的优化。编写报告阶段，苏浩负责对报告进行排版，对功能的描述，编写用户说明书；王立恒负责做出程序的图形设计，描述模块的具体实现方法；孙赫负责程序的运行结果分析，描述模块的实现方法，汇总后完成实验报告。