顺序表实现多项式相加

代码：

#include<stdlib.h>

#include<stdio.h>

#define ERROR 0;

#define OK 1;

#include <iostream>

#include <cstdio>

#include <cstring>

#include <cstdlib>

using namespace std;

typedef int ElemType;

typedef struct

{

int n;

int maxLength;

ElemType\* element;

} SeqList;

typedef int Status;

Status Init(SeqList\* L, int mSize)

{

L->maxLength = mSize;

L->n = 0;

L->element = (ElemType\*)malloc(sizeof(ElemType) \* mSize);

if (!L->element)

return ERROR;

return OK;

}

Status Find(SeqList L, int i, ElemType\* x)

{

if (i<0 || i>L.n - 1)

return ERROR;

\*x = L.element[i];

return OK;

}

Status Insert(SeqList\* L, int i, ElemType x)

{

int j;

if (i<-1 || i>L->n - 1)

return ERROR;

if (L->n == L->maxLength)

return ERROR;

for (j = L->n - 1; i > i; j++)

L->element[j + 1] = L->element[j];

L->element[i + 1] = x;

L->n = L->n + 1;

return OK;

}

Status Delete(SeqList\* L, int i) {

int j;

if (i<0 || i>L->n - 1)

return ERROR;

for (j = i + 1; j < L->n; j++)

L->element[j - 1] = L->element[j];

L->n--;

return OK;

}

Status Output(SeqList L)

{

int i;

if (!L.n)

return ERROR;

for (i = 0; i <= L.n - 1; i++)

printf("%d", L.element[i]);

printf("\n");

return OK;

}

void Destroy(SeqList\* L)

{

(\*L).n = 0;

(\*L).maxLength = 0;

free((\*L).element);

}

void main()

{

int i;

SeqList list;

Init(&list, 10);

for (i = 0; i < 9; i++)

Insert(&list, i - 1, i);

Output(list);

Delete(&list, 0);

Output(list);

Destroy(&list);

system("pause");

}

typedef struct node {

int coef, exp;

node\* next;

}\*LinkList, node;

void InitList(LinkList& L)

{

L = new node;

L->exp = -1;

L->coef = -1;

L->next = NULL;

}

int n, x, y;

void Input(LinkList& h1, LinkList& h2)

{

LinkList l1 = h1;

LinkList l2 = h2;

printf("第一个多项式的项数为：\n");

scanf\_s("%d", &n);

printf("降序依次输入系数与指数：\n");

for (int i = 0; i < n; i++)

{

scanf\_s("%d%d", &x, &y);

node\* p1;

p1 = new node;

p1->coef = x;

p1->exp = y;

p1->next = NULL;

l1->next = p1;

l1 = l1->next;

}

printf("\n第二个多项式的项数为：\n");

scanf\_s("%d", &n);

printf("降序依次输入系数与指数：\n");

for (int i = 0; i < n; i++)

{

scanf\_s("%d%d", &x, &y);

node\* p2;

p2 = new node;

p2->coef = x;

p2->exp = y;

p2->next = NULL;

l2->next = p2;

l2 = l2->next;

}

}

void Output(LinkList h) {

h = h->next;

if (h == NULL) {

printf("0 0");

}

else {

int cnt = 0;

while (h != NULL) {

if (!cnt) {

printf("%d x^%d | ", h->coef, h->exp);

}

else {

printf("%d x^%d | ", h->coef, h->exp);

}

cnt++;

h = h->next;

}

}

}

LinkList add(LinkList h1, LinkList h2)

{

LinkList r1 = h1;

LinkList r2 = h2;

LinkList l3;

InitList(l3);

LinkList h3 = l3;

if (h1->next == NULL)

return r2;

if (h2->next == NULL)

return r1;

r1 = r1->next;

r2 = r2->next;

while (r1 != NULL && r2 != NULL)

{

if (r1->exp > r2->exp) {

l3->next = r1;

l3 = l3->next;

r1 = r1->next;

}

else if (r1->exp < r2->exp) {

l3->next = r2;

l3 = l3->next;

r2 = r2->next;

}

else {

node\* tmp = new node;

tmp->exp = r1->exp;

tmp->coef = r1->coef + r2->coef;

if (tmp->coef == 0)

{

r1 = r1->next;

r2 = r2->next;

}

else {

r1 = r1->next;

r2 = r2->next;

l3->next = tmp;

l3 = l3->next;

}

}

}

if (r1 != NULL) {

while (r1 != NULL) {

l3->next = r1;

l3 = l3->next;

r1 = r1->next;

}

}

if (r2 != NULL) {

while (r2 != NULL) {

l3->next = r2;

l3 = l3->next;

r2 = r2->next;

}

}

return h3;

}

int main()

{

LinkList h1, h2;

InitList(h1);

InitList(h2);

Input(h1, h2);

printf("\n两个多项式分别为：\n");

Output(h1);

printf("\n");

Output(h2);

LinkList h4 = mul(h1, h2);

LinkList h3 = add(h1, h2);

printf("\n多项式相乘结果为：\n");

Output(h4);

printf("\n");

printf("多项式相加结果为：\n");

Output(h3);

printf("\n");

system("pause");

return 0;

}

代码逻辑思路：

要实现用顺序表进行多项式的相加，主要分为两部分，一是定义顺序表，二是定义多项式。

1. 定义顺序表

使用typedef int， status find，status insert，status delete，status output，status destroy对顺序表进行初始化操作.

1. 定义多项式

先对多项式进行初始化，然后要定义多项式指数与系数的输入与输出方式。

最后用main函数实现。

问题及总结：

在实现多项式加法时，没有考虑到h1与h2为空的情况，这种情况下需要直接返回r2或者r1，然后当r1大于r2或者r2大于r1时，指向下一个节点。

单链表实现多项式相加

代码：

#include <stdio.h>

#include <stdlib.h>

#include <iostream>

using namespace std;

typedef int ElemType;

typedef int Status;

#define OK 1

#define ERROR 0

typedef struct PNode {

float coef;

int index;

struct PNode\* next;

}PNode, \* Polynomial;

Status InitPolyn(Polynomial& p) {

p = new PNode;

if (!p) return ERROR;

p->next = NULL;

return OK;

}

void CreatePolyn(Polynomial& p, int n) {

PNode\* l = p;

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

PNode\* q = new PNode;

q->next = NULL;

cout << "请输入多项式系数：";

cin >> q->coef;

cout << "请输入多项式指数：";

cin >> q->index;

l->next = q;

l = q;

}

}

void AddPolyn(Polynomial& La, Polynomial& Lb) {

PNode\* pa = La->next;

PNode\* pb = Lb->next;

PNode\* pc = La;

PNode\* temp\_pa = NULL;

PNode\* temp\_pb = NULL;

while (pa && pb) {

if (pa->index == pb->index) {

if (!(pa->coef + pb->coef)) {

temp\_pa = pa;

temp\_pb = pb;

pa = pa->next;

pb = pb->next;

delete temp\_pa;

delete temp\_pb;

}

else {

pa->coef = pa->coef + pb->coef;

pc->next = pa;

pc = pa;

pa = pa->next;

temp\_pb = pb;

pb = pb->next;

delete temp\_pb;

}

}

else if (pa->index < pb->index) {

pc->next = pa;

pc = pa;

pa = pa->next;

}

else {

pc->next = pb;

pc = pb;

pb = pb->next;

}

}

pc->next = pa ? pa : pb;

delete Lb;

}

Status ShowPolyn(Polynomial p) {

PNode\* q = p->next;

if (!q) return ERROR;;

while (q) {

cout << "多项式系数为：" << q->coef << endl;

cout << "多项式指数为：" << q->index << endl;

q = q->next;

}

return OK;

}

int main() {

Polynomial La, Lb;

InitPolyn(La);

CreatePolyn(La, 3);

InitPolyn(Lb);

CreatePolyn(Lb, 4);

AddPolyn(La, Lb);

ShowPolyn(La);

}

代码逻辑及主要问题：

首先还是要初始化多项式与链表，

在涉及到系数比较时问题较多，

当系数相等时，将pb,pa均后移一位，并销毁他们之前指向的节点，pc保持不变。

若系数相加不为0，则将指数相加的结果保存在pa, 并先将pa赋值给pc->next, 再将pa赋值给pc;之后, pa, pb均后移, 并销毁pb之前指向的节点

若系数相加不为0，则将指数相加的结果保存在pa, 并先将pa赋值给pc->next, 再将pa赋值给pc;之后, pa, pb均后移, 并销毁pb之前指向的节点

若pa系数小于pb系数，则先将pa赋值给pc->next, 再将pa赋值给pc；之后pa后移一位，pb不变

若pb系数小于pa系数，则先将pb赋值给pc->next, 再将pb赋值给pc；之后pb后移一位，pa不变

循环结束后，若pa未到达表尾，则将pa所指节点插入到"和多项式"链表中；pb同理