#include<iostream>

#include"max.h"

#include"string"

#include"math.h"

#include"vector"

#include"map"

#include <unordered\_set>

#include <algorithm>

#include <cstdlib>

#include <sstream>

using namespace std;

//下面的main函数请勿修改

#include <stdio.h>

#include <stdlib.h>

//存放多项式某项的结点结构

struct node

{

  int exp ;  // 表示指数

  int coef ; //表示系数

  struct node \*next;  //指向下一个结点的指针

};

typedef  struct node \* PNODE ;

/\*

  函数功能：生成多项式

  函数名：createPoly

  函数参数：无

  返回值：指向多项式的头指针

\*/

PNODE insert(PNODE poly,int ins\_coef,int ins\_exf){

    PNODE head=poly;

    PNODE  p=(PNODE)malloc(sizeof(struct node));

    p->coef=ins\_coef;

    p->exp=ins\_exf;

    p->next=NULL;

    if(head->next==NULL){

       head->next=p;

       return head;

    }

    PNODE pre=head;

    PNODE cur\_1=head->next;//第一层搜索使用

     PNODE cur\_2=head->next;

    while(cur\_1!=NULL){//先搜索有没有可合并的同类项

        if(cur\_1->exp==p->exp){

            cur\_1->coef+=p->coef;

            free(p);

            return head;

        }

        cur\_1=cur\_1->next;

    }

       while(cur\_2!=NULL&&cur\_2->exp<p->exp){

        pre=cur\_2;

        cur\_2=cur\_2->next;

       }

       p->next=cur\_2;

       pre->next=p;

    return head;

}

PNODE createPoly(void)

{

    //在此处填写代码，能实现创建一个多项式并返回多项式头指针的函数

    //注意：头指针不存放多项式的项。

    /\*\*\*\*\*\*\*\*\*\*  Begin \*\*\*\*\*\*\*\*\*\*/

    PNODE poly=(PNODE) malloc(sizeof(struct node));

   poly->exp=0;

   poly->coef=0;

   poly->next=NULL;

   int ins\_coef=0;

   int ins\_exf=0;

   scanf("%d %d",&ins\_coef,&ins\_exf);

   while(ins\_coef!=0){

   poly= insert(poly, ins\_coef, ins\_exf);

   scanf("%d %d",&ins\_coef,&ins\_exf);

   }

   return poly;

    /\*\*\*\*\*\*\*\*\*\*  End  \*\*\*\*\*\*\*\*\*\*/

}

/\*

   函数功能：进行多项式相加

   函数名：addPoly

   函数参数：polyAddLeft ：加法左边多项式头指针, polyAddRight：加法右边多项式头指针

   返回值：指向结果多项式的头指针

\*/

void deleteData(PNODE polyAddRight,int key){

    PNODE pre=polyAddRight;

      PNODE cur=pre->next;

      while(cur!=NULL){

        if(cur->coef==key){

            PNODE tem=cur;

            pre->next=cur->next;

            cur=cur->next;

            free(tem);

        }

        else{

            pre=cur;

            cur=cur->next;

        }

      }

      return ;

}

PNODE addPoly(PNODE polyAddLeft , PNODE polyAddRight)

{

    //在此处填写代码，能实现创两个多项式相加并返回结果多项式头指针的函数

    /\*\*\*\*\*\*\*\*\*\*  Begin \*\*\*\*\*\*\*\*\*\*/

    PNODE first= polyAddLeft;

    PNODE pre\_first=first ;

    PNODE cur\_first=first->next;

    PNODE second= polyAddRight;

    while(pre\_first->next!=NULL){

        PNODE cur\_second= polyAddRight->next;//不断更新cur\_second

        PNODE tem=cur\_first;//要移除的节点

        cur\_first=cur\_first->next;

        pre\_first->next=cur\_first;

        bool k=false;//标志检查合并是否成功

        while(cur\_second!=NULL){

          if(cur\_second->exp==tem->exp){

               cur\_second->coef+=tem->coef;

               k=true;

               break;

          }

          cur\_second=cur\_second->next;

        }

        if(!k){

            polyAddRight=insert(polyAddRight,tem->coef,tem->exp);

        }

  free(tem);

    }

    deleteData( polyAddRight,0);

    return  polyAddRight;

    /\*\*\*\*\*\*\*\*\*\*  End \*\*\*\*\*\*\*\*\*\*/

}

/\*

   函数功能：输出多项式

   函数名：printPoly

   函数参数：待输出多项式的头指针poly

   返回值：无

\*/

 void printPoly(PNODE poly)

 {

    //在此处填写代码，能实现按格式输出多项式的功能，输出格式样例见说明

    /\*\*\*\*\*\*\*\*\*\*  Begin \*\*\*\*\*\*\*\*\*\*/

    if(poly->next==NULL) {

        printf("Error !this is empty()");

        return;

    }

    PNODE pre=poly;

    PNODE cur=pre->next;

    while(cur->next!=NULL){

     printf("%dx^%d+",cur->coef,cur->exp);

     cur=cur->next;

    }

    printf("%dx^%d",cur->coef,cur->exp);

    return ;

    /\*\*\*\*\*\*\*\*\*\*  End \*\*\*\*\*\*\*\*\*\*/

}

void destroyPoly(PNODE poly)

{

//释放存储多项式的链表空间

 PNODE pre=poly;

 pre->next=NULL;

    PNODE cur=pre->next;

    while(cur!=NULL){

        PNODE tem=cur;

        cur=cur->next;

        free(tem);

    }

}

int main(void){

PNODE polyAddLeft =createPoly();

printPoly(polyAddLeft);

PNODE polyAddRight=createPoly();

printPoly( polyAddRight);

printf("\n");

  polyAddRight=addPoly( polyAddLeft ,  polyAddRight);

  printPoly( polyAddRight);

  printf("\n");

  destroyPoly(polyAddRight);

  printf("\n");

   printPoly( polyAddRight);

}