题解

```c
#include <stdio.h>
#include <stdlib.h>
#include <stddef.h>

typedef struct PolyNode *Polynomial;
struct PolyNode
{
    int ceof;
    int expon;
    Polynomial link;
};

Polynomial ReadPoly();
void Attach(int c,int e,Polynomial *pRear);
Polynomial Add(Polynomial P1,Polynomial P2);
Polynomial Mult(Polynomial P1,Polynomial P2);
void PrintPoly(Polynomial P);


int main()
{
    Polynomial P1,P2,PP,PS;
    P1=ReadPoly();
    P2=ReadPoly();
    PP=Mult(P1,P2);
    PrintPoly(PP);
    PS=Add(P1,P2);
    PrintPoly(PS);
}

Polynomial ReadPoly()
{
    Polynomial PHead,Rear,t;
    int c,e,n;

    scanf("%d",&n);
    PHead=(Polynomial)malloc(sizeof(struct PolyNode));
    PHead->link=NULL;
    Rear=PHead;

    while(n--)
    {
        scanf("%d %d",&c,&e);
        Attach(c,e,&Rear);
    }

    t=PHead;
    PHead=PHead->link;
    free(t);
    return PHead;
}

void Attach(int c,int e,Polynomial *pRear)
{
    Polynomial P;
    P=(Polynomial)malloc(sizeof(struct PolyNode));
    P->ceof=c;
    P->expon=e;
    P->link=NULL;
    (*pRear)->link=P; /* 把新建的结点连接在尾结点Rear后面 */
    *pRear=P; /* 将Rear指向新建结点，即尾结点 */
}

Polynomial Add(Polynomial P1,Polynomial P2)
{
    Polynomial PAns,t1,t2,Rear,temp;
    int sum;

    t1=P1;
    t2=P2;

    PAns=(Polynomial)malloc(sizeof(struct PolyNode));
    Rear=PAns;
```

```
    while(t1&&t2)
    {
        if(t1->expon==t2->expon)
        {
            sum=t1->ceof+t2->ceof;
            if(sum)
            {
                Attach(sum,t1->expon,&Rear);
            }
           t1=t1->link;
           t2=t2->link;
        }
        else if(t1->expon>t2->expon)
        {
            Attach(t1->ceof,t1->expon,&Rear);
            t1=t1->link;
        }
        else if(t1->expon<t2->expon)
        {
            Attach(t2->ceof,t2->expon,&Rear);
            t2=t2->link;
        }
    }

    while(t1)
    {
        Attach(t1->ceof,t1->expon,&Rear);
        t1=t1->link;
    }

    while(t2)
    {
        Attach(t2->ceof,t2->expon,&Rear);
        t2=t2->link;
    }

    Rear->link=NULL; /* Rear始终指向尾结点 */
    /* 释放最开始建立的空结点 */
    temp=PAns;
    PAns=PAns->link;
    free(temp);

    return PAns;
}

Polynomial Mult(Polynomial P1,Polynomial P2)
{
    Polynomial P,t1,t2,Rear,t;
    int c,e;

    if(!P1||!P2)
    {
        return NULL;
    }

    t1=P1;
    t2=P2;

    P=(Polynomial)malloc(sizeof(struct PolyNode));
    P->link=NULL;
    Rear=P;

    /* 将t2中每一项与t1中的第一项相乘并放入结果链表 */
    while(t2)
    {
        Attach(t1->ceof*t2->ceof,t1->expon+t2->expon,&Rear);
        t2=t2->link;
    }

    t1=t1->link;

    while (t1)
    {
        t2=P2;
        Rear=P;
        while(t2)
        {
            e=t1->expon+t2->expon;
            c=t1->ceof*t2->ceof;
```

```
            while(Rear->link&&Rear->link->expon>e) /* Rear->link为空时，跳出循环 */
            {
                Rear=Rear->link;
            }
            if(Rear->link&&Rear->link->expon==e)
            {
                if(Rear->link->ceof+c)
                {
                    Rear->link->ceof+=c;
                }
                else
                {
                    t=Rear->link;
                    Rear->link=t->link;
                    free(t);
                }
            }
            else
            {
                t=(Polynomial)malloc(sizeof(struct PolyNode));
                t->ceof=c;
                t->expon=e;
                t->link=Rear->link;
                Rear->link=t;
                Rear=Rear->link;
            }
            t2=t2->link;
        }
        t1=t1->link;
    }
    t2=P;
    P=P->link;
    free(t2);
    return P;
}

void PrintPoly(Polynomial P)
{
    Polynomial temp;
    temp=P;
    int flag=0;

    if(!temp)
    {
        printf("0 0\n");
        return  ;
    }

    while(temp)
    {
        if(!flag)
        {
            flag=1;
        }
        else
        {
            printf(" ");
        }
        printf("%d %d",temp->ceof,temp->expon);
        temp=temp->link;
    }

    printf("\n");
}
```