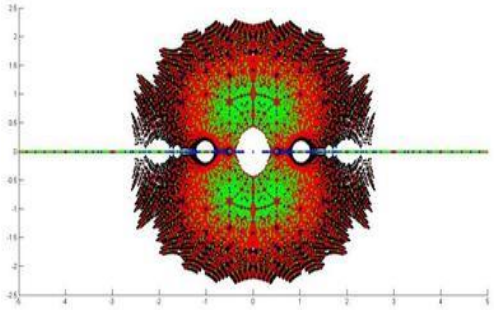


小白专场：

一元多项式的加法与乘法运算

题意理解

设计函数分别求两个一元多项式的乘积与和

已知两个多项式:

(1) $3x^4 - 5x^2 + 6x - 2$

(2) $5x^{20} - 7x^4 + 3x$

多项式和:

$5x^{20} - 4x^4 - 5x^2 + 9x - 2$

题意理解

设计函数分别求两个一元多项式的乘积与和

已知两个多项式:

(1) $3x^4 - 5x^2 + 6x - 2$

(2) $5x^{20} - 7x^4 + 3x$

多项式的乘积:

$(a+b)(c+d) = ac+ad+bc+bd$

多项式乘积:

$15x^{24}-25x^{22}+30x^{21}-10x^{20}-21x^8+35x^6-33x^5+14x^4-15x^3+18x^2-6x$

题意理解

设计函数分别求两个一元多项式的乘积与和

输入样例:

4 3 4 -5 2 6 1 -2 0
3 5 20 -7 4 3 1

$3x^4-5x^2+6x-2$

$5x^{20}-7x^4+3x$

输出样例:

15 24 -25 22 30 21 -10 20 -21 8 35 6 -33 5 14 4 -15 3 18 2 -6 1
5 20 -4 4 -5 2 9 1 -2 0

$15x^{24}-25x^{22}+30x^{21}-10x^{20}-21x^8+35x^6-33x^5+14x^4-15x^3+18x^2-6x$

$5x^{20}-4x^4-5x^2+9x-2$

求解思路

1.多项式表示
2. 程序框架
3. 读多项式
4. 加法实现
5. 乘法实现
6. 多项式输出

多项式的表示

仅表示非零项

数组：

- 编程简单、调试容易
- 需要事先确定数组大小

链表：

- 动态性强
- 编程略为复杂、调试比较困难

一种比较好的实现方法是：动态数组

下面介绍链表表示

多项式的表示

数据结构设计

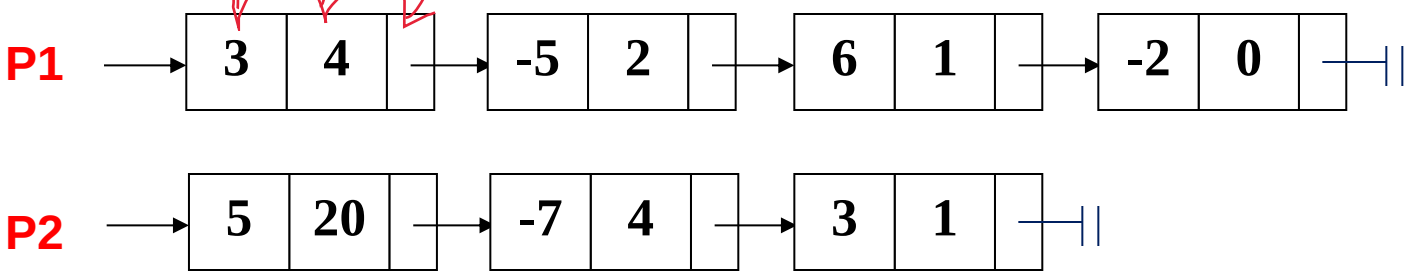
多项式

```
typedef struct PolyNode *Polynomial;
struct PolyNode {
        int coef;
        int expon;
        Polynomial link;
};
```

常系数

指数

多项式中的下一项



程序框架搭建

```
int main()
{
    读入多项式1
    读入多项式2
    乘法运算并输出
    加法运算并输出

    return 0;
}
```

需要设计的函数：

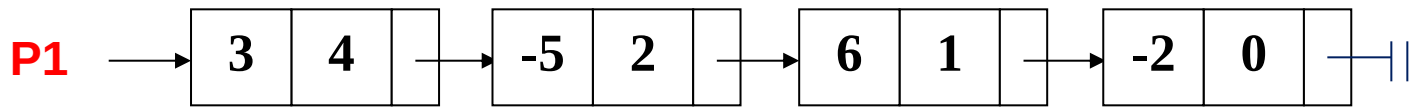
- 读一个多项式
- 两多项式相乘
- 两多项式相加
- 多项式输出

```
int main()
{
    Polynomial P1, P2, PP, PS;

    P1 = ReadPoly();
    P2 = ReadPoly();
    PP = Mult( P1, P2 );
    PrintPoly( PP );
    PS = Add( P1, P2 );
    PrintPoly( PS );

    return 0;
}
```

如何读入多项式

```
Polynomial ReadPoly()
{
    ……
    scanf("%d", &N);
    ……
    while ( N-- ) {
          scanf("%d %d", &c, &e);
          Attach(c, e, &Rear);
     }
     …..
     return P;
}
```

4 3 4 -5 2 6 1 -2 0



如何读入多项式

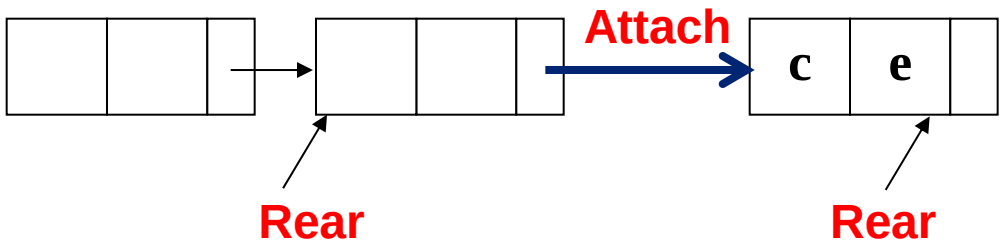
```
Polynomial ReadPoly()
{
    ……
    scanf("%d", &N);
    ……
    while ( N-- ) {
          scanf("%d %d", &c, &e);
          Attach(c, e, &Rear);
     }
     …..
     return P;
}
```

Rear初值是多少？

两种处理方法：

1. Rear初值为NULL

   在Attach函数中根据Rear是否为NULL做不同处理



如何读入多项式

```
Polynomial ReadPoly()
{
    ……
    scanf("%d", &N);
    ……
    while ( N-- ) {
          scanf("%d %d", &c, &e);
          Attach(c, e, &Rear);
     }
     …..
     return P;
}
```

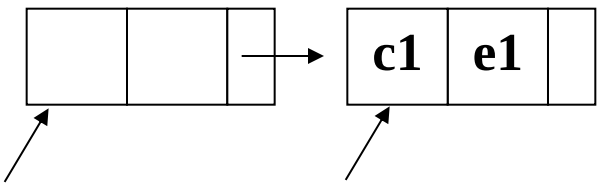
链表的尾部节点

Rear初值是多少？

两种处理方法：

1. Rear初值为NULL
2. Rear指向一个空结点

讲一个个节点连接成链表



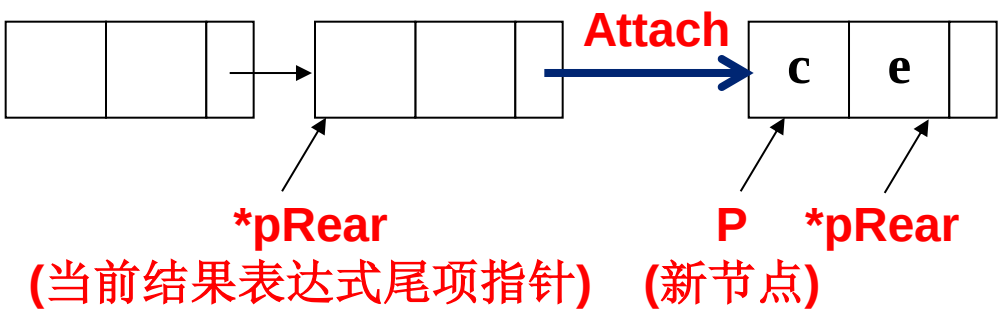
Rear
(一开始指向空结点)

Rear

如何读入多项式

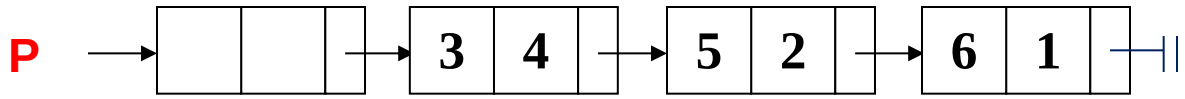
```
void Attach( int c, int e, Polynomial *pRear )
{       Polynomial P;

        P = (Polynomial)malloc(sizeof(struct PolyNode));
        P->coef = c;    /* 对新结点赋值  */
        P->expon = e;
        P->link = NULL;
        (*pRear)->link = P;
        *pRear = P;       /* 修改pRear值 */
}
```

将一个个节点连接成链表



如何读入多项式

```
Polynomial ReadPoly()
{
    Polynomial  P, Rear, t;
    int c, e, N;

    scanf("%d", &N);
    P = (Polynomial)malloc(sizeof(struct PolyNode)); /* 链表头空结点 */
    P->link = NULL;
    Rear = P;
    while ( N-- ) {
          scanf("%d %d", &c, &e);
          Attach(c, e, &Rear);              /* 将当前项插入多项式尾部 */
     }
     t = P;  P = P->link;  free(t);         /* 删除临时生成的头结点 */
     return P;
}
```



如何将两个多项式相加

```
Polynomial Add( Polynomial P1, Polynomial P2 )
{
        ……
        t1 = P1; t2 = P2;
        P = (Polynomial)malloc(sizeof(struct PolyNode)); P->link = NULL;
        Rear = P;
        while (t1 && t2) {
                if (t1->expon == t2->expon) {
                        …..
                }
                else if (t1->expon > t2->expon) {
                        ……
                }
                else {
                        ……
                }
        }
        while (t1) {
                ……
        }
        while (t2) {
                …..
        }
        ……..
        return P;
}
```

如何将两个多项式相乘

方法：

1.将乘法运算转换为加法运算

将P1当前项(ci,ei)乘P2多项式，再加到结果多项式里

```
t1 = P1; t2 = P2;
P = (Polynomial)malloc(sizeof(struct PolyNode)); P->link = NULL;
Rear = P;
while (t2) {
        Attach(t1->coef*t2->coef, t1->expon+t2->expon, &Rear);
        t2 = t2->link;
}
```

2.逐项插入

将P1当前项($c1_i$,$e1_i$)乘P2当前项（$c2_i$,$e2_i$),并插入到结果多项式中。关键是要找到插入位置

初始结果多项式可由P1第一项乘P2获得（如上）

如何将两个多项式相乘

```
Polynomial Mult( Polynomial P1, Polynomial P2 )
{           …….
            t1 = P1; t2 = P2;
            …….
            while (t2) {        /* 先用P1的第1项乘以P2，得到P */
            ……..
            }
            t1 = t1->link;
            while (t1) {
                        t2 = P2; Rear = P;
                        while (t2) {
                                    e = t1->expon + t2->expon;
                                    c = t1->coef * t2->coef;
                                    ……….
                                    t2 = t2->link;
                        }
                        t1 = t1->link;
            }
            …….
}
```

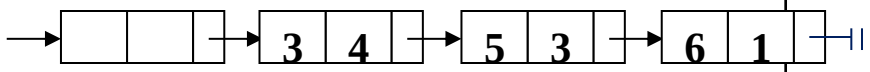

浙江大学计算机科学与技术学院

如何将两个多项式相乘

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

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

            t1 = P1; t2 = P2;
            P = (Polynomial)malloc(sizeof(struct PolyNode)); P->link = NULL;
            Rear = P;
            while (t2) {                    /* 先用P1的第1项乘以P2，得到P */
                        Attach(t1->coef*t2->coef, t1->expon+t2->expon, &Rear);
                        t2 = t2->link;
            }
            t1 = t1->link;
            while (t1) {
                        t2 = P2; Rear = P;
                        while (t2) {
                                    ……
                                    t2 = t2->link;
                        }
                        t1 = t1->link;
            }
            ……
}
```

如何将两个多项式相乘

```
Polynomial Mult( Polynomial P1, Polynomial P2 )
{
        .....
        while (t1) {
                t2 = P2; Rear = P;
                while (t2) {
                        e = t1->expon + t2->expon;
                        c = t1->coef * t2->coef;
                        while (Rear->link && Rear->link->expon > e)
                                Rear = Rear->link;
                        if (Rear->link && Rear->link->expon == e) {
                                ........
                        }
                        else {
                                ........
                        }
                        t2 = t2->link;
                }
                t1 = t1->link;
        }
        .....
}
```

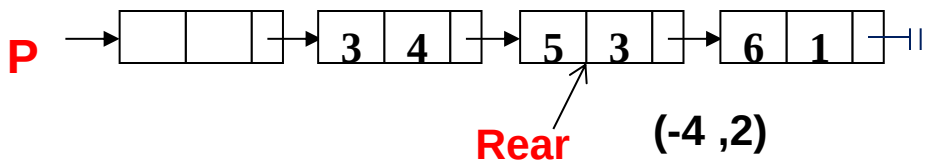


插入: (-4 ,2)

浙江大学计算机科学与技术学院

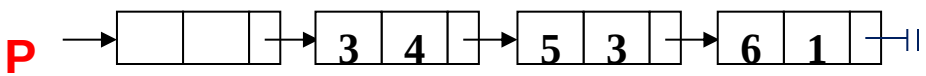
如何将两个多项式相乘

```
Polynomial Mult( Polynomial P1, Polynomial P2 )
{           .......
                    while (Rear->link && Rear->link->expon > e)
                            Rear = Rear->link;
                     if (Rear->link && Rear->link->expon == e) {
                            if (Rear->link->coef + c)
                                    Rear->link->coef += c;
                            else {
                                    t = Rear->link;
                                    Rear->link = t->link;
                                    free(t);
                            }
                     }
                     else {
                            t = (Polynomial)malloc(sizeof(struct PolyNode));
                            t->coef = c; t->expon = e;
                            t->link = Rear->link;
                            Rear->link = t; Rear = Rear->link;
                     }
           .......
}
```



如何将两个多项式相乘

```
Polynomial Mult( Polynomial P1, Polynomial P2 )
{           …….
            t1 = P1; t2 = P2;
            …….
            while (t2) { /* 先用P1的第1项乘以P2，得到P */
            ……..
            }
            t1 = t1->link;
            while (t1) {
                        t2 = P2; Rear = P;
                        while (t2) {
                                    e = t1->expon + t2->expon;
                                    c = t1->coef * t2->coef;
                                    ……….
                                    t2 = t2->link;
                        }
                        t1 = t1->link;
            }
            t2 = P;  P = P->link;  free(t2);

            return P;
}
```



如何将多项式输出

```
void PrintPoly( Polynomial P )
{   /* 输出多项式 */
        int flag = 0;                   /* 辅助调整输出格式用 */

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

        while ( P ) {
                if (!flag)
                        flag = 1;
                else
                        printf(" ");
                printf("%d %d", P->coef, P->expon);
                P = P->link;
        }
        printf("\n");
}
```