

## 实验报告要求包含：

### 一、需求分析

设计一个一元稀疏多项式简单计算器；  
由于其为稀疏多项式，采用链表的数据结构；

### 二、概要设计

设置一个链表用于结构类型；  
采用链式存储，由于其表达稀疏多项式，采用顺序表占用内存大，浪费资源；  
设计一个结构体，用于表示多项式中的其中一项；  
采用数组的方式输入测试数据；  
设计了五个函数，分别用于利用结构体数组生成多项式链表，相加，相减，求导，合理的输出多项式五个功能；  
由于每次输入测试需要重新输入测试数据，本程序采用数组输入数据，即，在 main（）函数中定义好测试数据的数组，程序直接运行即可，此与直接输入的逻辑相同，相比与输入数据而言并没有程序上变化；

### 三、详细设计

详细 C/C++代码；

源代码已附

思考题伪代码。

Function MultiplyPolynomials(PolyA: Pointer to Node, PolyB: Pointer to Node): Pointer to Node

// 创建一个新的链表用于存储结果

ResultHead: Pointer to Node = NULL

CurrentResult: Pointer to Node = NULL

// 遍历多项式 A 的每个节点

For Each NodeA in PolyA

// 遍历多项式 B 的每个节点

For Each NodeB in PolyB

// 计算当前系数和指数的乘积

ProductCoefficient= NodeA.Coefficient \* NodeB.Coefficient

ProductExponent = NodeA.Exponent + NodeB.Exponent

// 创建新的节点存储乘积结果

NewNode = CreateNode(ProductCoefficient, ProductExponent)

// 如果结果是链表的第一个节点

If ResultHead is NULL

ResultHead = NewNode

CurrentResult = NewNode

```

Else
    // 将新节点添加到结果链表的末尾
    CurrentResult.Next = NewNode
    CurrentResult = NewNode

// 返回结果链表的头指针
Return ResultHead
End Function

```

#### 四、调试分析与测试结果

1. 由于每次输入测试需要重新输入测试数据，本程序采用数组输入数据，即，在 main（）函数中定义好测试数据的数组，程序直接运行即可，此与直接输入的逻辑相同，相比与输入数据而言并没有程序上的变化；
  2. 输出程序时，经常出现输出不完善的情况，比如说输出  $0 * x$  的 0 次方；或者多项式的末尾仅有一个加号；
- 最终测试结果如下

```

=====测试结果=====
(-3.1 x**(11.0) + 5.0 x**(8.0) + 2.0 x**(1.0)) + (11.0 x**(9.0) + -5.0 x**(8.0) + 7.0) = (-3.1 x**(11.0) + 11.0 x**(9.0) + 2.0 x + 7.0)
(-1.2 x**(9.0) + 4.4 x**(2.0) + -1.0 x + 6.0 x**(-3.0)) - (7.8 x**(15.0) + 4.4 x**(2.0) + -6.0 x**(-3.0)) = (-7.8 x**(15.0) + -1.2 x**(9.0) + -1.0 x + 12.0 x**(-3.0))
(1.0 x**(5.0) + 1.0 x**(4.0) + 1.0 x**(3.0) + 1.0 x**(2.0) + 1.0 x + 1.0) + (-1.0 x**(4.0) + -1.0 x**(3.0)) = (1.0 x**(5.0) + 1.0 x**(2.0) + 1.0 x + 1.0)
(1.0 x**(3.0) + 1.0 x) + (-1.0 x**(3.0) + -1.0 x**(1.0)) = (0)
(1.0 x**(100.0) + 1.0 x) + (1.0 x**(200.0) + 1.0 x) = (1.0 x**(200.0) + 2.0 x**(100.0) + 1.0 x)
(1.0 x**(3.0) + 1.0 x**(2.0) + 1.0 x) + (0) = (1.0 x**(3.0) + 1.0 x**(2.0) + 1.0 x + 0)
PS C:\Users\pc\Desktop\c_test> 

```

#### 五、附录

源代码 2\_1.c