

2.5 线性表的应用-两个多项式相加

2.5.1 问题描述

多项式 $p(x) = c_1x^{e_1} + c_2x^{e_2} + \dots + c_mx^{e_m}$



求两个多项式相加的程序

例如, $p(x)=2x^3+3.2x^5-6x+10$, $q(x)=6x+1.8x^5-2x^3+x^2-2.5x^4-5$



$$r(x)=p(x)+q(x)$$

$$r(x)=5x^5-2.5x^4+x^2+5$$

两个多项式的数据分别存放在`abc1.in`和`abc2.in`文本文件中，要求相加的结果多项式的数据存放在`abc.out`文本文件中。

`abc1.in`文件

```
4
2 3
3.2 5
-6 1
10 0
```

$$p(x)=2x^3+3.2x^5-6x+10$$

`abc2.in`文件

```
6
6 1
1.8 5
-2 3
1 2
-2.5 4
-5 0
```

$$q(x)=6x+1.8x^5-2x^3+x^2-2.5x^4-5$$




`abc.out`文件

```
第1个多项式: [[2.0, 3],[3.2, 5],[-6.0, 1],[10.0, 0]]
排序后结果:  [[3.2, 5],[2.0, 3],[-6.0, 1],[10.0, 0]]
第2个多项式: [[6.0, 1],[1.8, 5],[-2.0, 3],[1.0, 2],[-2.5, 4],[-5.0, 0]]
排序后结果:  [[1.8, 5],[-2.5, 4],[-2.0, 3],[1.0, 2],[6.0, 1],[-5.0, 0]]
相加多项式:  [[5.0, 5],[-2.5, 4],[1.0, 2],[5.0, 0]]
```

$$p(x) = c_1 x^{e_1} + c_2 x^{e_2} + \cdots + c_m x^{e_m}$$



(c_1, e_1) (c_2, e_2) \cdots (c_m, e_m)  多项式线性表



多项式项

ADT PolyClass

#多项式抽象数据类型

{

数据对象:

$\text{PolyElem} = \{(c_i, e_i) \mid 1 \leq i \leq n, c_i \in \text{float}, e_i \in \text{int}\};$

数据关系:

$r = \{\langle x_i, y_i \rangle \mid x_i, y_i \in \text{PolyElem}, i = 1, \dots, n-1\}$

基本运算:

Add(e): 将多项式项e添加到末尾。

CreateList(fname): 从fname文件中读取数据建立多项式。

getsize(): 返回多项式的项数

getitem(i): 返回序号为i的多项式项。

getdata(): 返回多项式。

Sort(): 对多项式按指数递减排序。

PolyAdd(B): 返回当前多项式与多项式B的相加结果。

} #ADT PolyClass

2.5.2 问题求解

1. 设计顺序存储结构

- 多项式的每一项用一个列表 $[c_i, e_i]$ （其中 c_i 为系数， e_i 为指数）存储，一个多项式顺序表用元素为列表 $[c_i, e_i]$ 的列表data存储。
- 例如，多项式 $p(x)=2x^3+3.2x^5-6x+10$ 的data列表为 $[[2.0, 3], [3.2, 5], [-6.0, 1], [10.0, 0]]$ 。

多项式顺序表类PolyList

```
class PolyList:                                #多项式顺序表类
    def __init__(self):                        #构造方法
        self.data=[]                          #存放多项式项的列表
    #修改基本运算算法
```

2. 设计PolyList的基本运算算法

(1) PolyList的构造方法

```
def __init__(self):  
    self.data=[]
```

#构造方法
#存放多项式项的列表

(2) 将多项式项 e 添加到末尾Add(e)

```
def Add(self,e):                                #添加一个多项式项e  
    self.data.append(e)
```


(3) 创建多项式顺序表CreateList(fname)

```
def CreateList(self,fname):          #从fname文件中读取多项式数据并添加到data
    fin=open(fname,"r")
    n=int(fin.readline().strip())
    for i in range(n):
        p=fin.readline().strip().split()
        self.data.append([float(p[0]),int(p[1])])
    fin.close()
```

(4) 返回多项式的项数 `getsize()`

```
def getsize(self):  
    return len(self.data)
```

#求多项式的项数

(5) 返回序号为*i*的多项式项

```
def __getitem__(self,i):          #求序号为i的元素  
    return self.data[i]
```

(6) 返回多项式的data列表getdata()

```
def getdata(self):  
    return self.data
```

#返回多项式列表

(7) 对多项式按指数递减排序Sort()

```
def Sort(self):                                #对data按指数递减排序  
    self.data=sorted(self.data,key=itemgetter(1),reverse=True)
```

(8) 返回当前多项式与多项式B的相加结果PolyAdd(B)

两个按指数递减排序的多项式顺序表



二路归并 + 整体创建顺序表

相加的结果多项式顺序表

用 i 、 j 分别遍历 A 和 B 中的元素，先建立一个空多项式顺序表 C ，在 i 、 j 都没有遍历完时循环，取 i 指向的 A 中元素 p ，取 j 指向的 B 中元素 q ：

- ① 若 p 元素的指数 ($p[1]$) 较大，将 p 元素添加到 C 中， i 增加1。
- ② 若 q 元素的指数 ($q[1]$) 较大，将 q 元素添加到 C 中， j 增加1。
- ③ 此时 p 、 q 元素的指数相同 ($p[1]=q[1]$)，求出它们的系数和 k ($k=p[0]+q[0]$)，如果 $k \neq 0$ ，由 k 和 $p[1]$ 新建一个元素并添加到 C 中，否则不新建结点，并将 i 、 j 均增加1。

上述循环过程结束后，若有一个多项式顺序表没有遍历完，说明余下的多项式项都是指数较小的多项式项，将它们均添加到 C 中，最后返回 C 。

def PolyAdd(self,B):	#当前多项式和多项式B的相加运算
C=PolyList()	#新建结果多项式顺序表
m=len(self.data)	#多项式A的项数
n=B.getsize()	#多项式B的项数
i,j=0,0	
while i<m and j<n:	
p,q=self.data[i],B[j]	
if p[1]>q[1]:	#将较大指数的p项添加到C中
C.Add(p)	
i+=1	
elif q[1]>p[1]:	#将较大指数的q项添加到C中
C.Add(q)	
j+=1	
else:	#两指数相同, 即p[1]=q[1]
k=p[0]+q[0]	#系数相加为k
if (k!=0):	#k不为0时添加相应项到C中
C.Add([k,p[1]])	
i+=1	
j+=1	


```
while i<m:                                #将A余下的项添加到C中
    p=self.data[i]
    C.Add(p)
    i+=1
while j<n:                                #将B余下的项添加到C中
    q=B[j]
    C.Add(q)
    j+=1
return C
```

3. 设计主程序

```
fout=open("abc.out","w+")
p=PolyList()
p.CreateList("abc1.in")
print("第1个多项式:",end=' ',file=fout)
print(p.getdata(),file=fout)
p.Sort()
print("排序后结果: ",end=' ',file=fout)
print(p.getdata(),file=fout)
q=PolyList()
q.CreateList("abc2.in")
print("第2个多项式:",end=' ',file=fout)
print(q.getdata(),file=fout)
q.Sort()
print("排序后结果: ",end=' ',file=fout)
print(q.getdata(),file=fout)
r=p.PolyAdd(q)
print("相加多项式: ",end=' ',file=fout)
print(r.getdata(),file=fout)
fout.close()
```

#创建第1个多项式顺序表p

#输出结果写入到abc.out文件中

#第1个多项式顺序表按指数递减排序

#创建第2个多项式顺序表q

#第2个多项式顺序表按指数递减排序

#r=p+q

欣
赏

