Programming Assignment 2

Parenthesize the Expression

104303206 黃筱晴

一、題目分析與遞迴式推導

明顯此問題符合 principle of optimality。參考 Matrix-Chain Multiplication，設計類似的 Dynamic programming 演算法。

使用陣列 Num 和 Operator 作為輸入。當使用者鍵入 1-4+5*6+7*9，經過分析後，得到：

Num[]={1,4,5,6,7,9};

Operator[]={'-','+','*','+','*'};

輸入為 6 個整數和 5 個運算子。

先撰寫一函式 calculation(n1,operator,n2)來處理整數運算。例如計算 1-4=? 則呼叫

int ans=calculation(1,'-',4); //return -3

計算過程中，使用 p 和 k 兩個陣列，定義：

$p[i][j]_{1<=i<=j<=n}$：從第 i 個整數運算至第 j 個整數的運算最大值。

$k[i][j]_{1<=i<j<=n}$：從第 i 個整數運算至第 j 個整數的最佳分割方式。

依照定義推導出遞迴式：

$p[i][j]=max_{i<=l<=j-1}$\{calculation(p[i][l],Operator[l],p[l+1][j])\}

=Num[i],when i==j

k[i][j]為填入 p[i][j]時，得到 max 時的 l。

二、計算過程範例

使用 1-4+5*6+7*8 來說明計算過程。

p	1	2	3	4	5	6
1	1	-3	2	27	62	582
2		4	9	54	117	1053
3			5	30	65	585
4				6	13	117
5					7	63
6						9

k	1	2	3	4	5	6
1		1	2	2	2	2
2			2	3	3	3
3				3	3	3
4					4	5
5						5
6						

1.初始化 p[1][1]=Num[1], p[2][2]=Num[2], p[3][3]=Num[3], ...

2.從對角線 1 開始填入，

$$p[1][2]=\max_{1<=l<=1}\{calculation(p[1][l],Operator[l],p[l+1][1])\}$$

= calculation(p[1][2], Operator[1], p[2][1]) //l=1

= 1-4 =(-3)

p[2][3],p[3][4],p[4][5],p[5][6]做法相同。

3.計算對角線 2

$$p[1][3]=\max_{1<=l<=2}\{calculation(p[1][l],Operator[l],p[l+1][1])\}$$

$$= \max\{calculation(p[1][1], Operator[1], p[2][3]),$$

$$calculation(p[1][2], Operator[2], p[3][3])\}$$

$$= \max\{1-9=(-8),(-3)+5=2\}=2 \qquad //l=2$$

p[2][4],p[3][5],p[4][6]做法相同。

4.用同樣方式計算對角線 3

5.用同樣方式計算對角線 4

6.最後用同樣方式計算對角線 5，p[1][6]就是這串輸入的運算最大值。

分析矩陣 k 找出最佳分割方式

1.分解從第 1 個整數算到第 6 個整數。k[1][6]=2，從 Num[2]和 Num[3]做分割。

(1-4)+(5*6+7*8)

2.繼續分解從第 3 個整數算到第 6 個整數。k[3][6]=3，從 Num[3]和 Num[4]做分割。

(1-4)+(5*(6+7*8))

3.繼續分解從第 4 個整數算到第 6 個整數。k[4][6]=5，從 Num[5]和 Num[6]做分割。

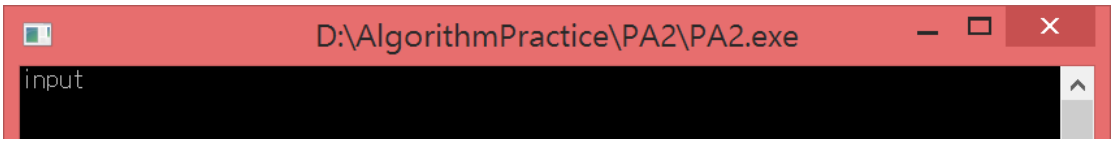
(1-4)+(5*((6+7)*8))

三、Pseudo-Code：

```
//計算
MaxCalculation(int Num[],char Operator[],int size)
   int p[][],k[][];
   for(int i=1;i<=size;i++) p[i][i]=Num[i];
   for(int line=1;line<=size-1;line++)
      int i=1;
      int j=i+line;
      while(i<=size && j<=size)
         p[i][j]=max_{i<=l<=j-1}{   calculation(p[i][l],Operator[l],p[l+1][j]) };
         k[i][j]=l;
         i++;j++;
//印答案
Ans(int p[][],int k[][],int size,int start,int end)
   if(end==start)  顯示 p[start][end] ;                              // p[start][end]=p[start][start]=Num[start]
   else
      char flag=0;// 0->要印括弧      1->不需要印括弧
      int divide= k[start][end];
      if(某些情形不需要印括弧)    flag=1;
      if(flag==0) cout<<"(";    Ans(p,k,size,start,divide);
      cout<<Operator[divide];
      Ans(p,k,size,divide+1,end);      if(flag==0) cout<<")";
```

四、程式測試

點擊執行檔 PA2.exe 執行測試程式。程式一開始便提示使用者輸入 input。



輸入：1-4+5*6+7*9

得到輸出：p 矩陣、k 矩陣、最大運算結果與其分割方法(已經去除多餘括弧)。