

資料結構實習

11/17 作業報告

Stack 實作中序轉後序計算結果

班級：資訊二甲

學號：D1109023

姓名：楊孟憲

Contents

1	引言	3
2	題目敘述	4
3	作法	4
4	執行結果	7
5	心得與討論	7

1 引言

如何將中序轉為後序？

我們可以使用 Stack 的特質來實作，實作過程如下：將 $(a + b) \times (a + d)$ 轉為後序。

元素	堆疊	輸出
((-
a	(a
+	(+	a
b	(+	ab
)	-	ab+
*	*	ab+
(* (ab+
c	* (ab+c
+	* (+	ab+c
d	* (+	ab+cd
)	*	ab+cd+
-	-	ab+cd+*

Figure 1: 圖一

如何計算後序結果？

假設有一個後序為 $123*+$ ，當遇到運算子時，就會將前兩個數字做運算，並解放回堆疊裡，以此類推。

1. $123*$

2. $16+$

3. 7

2 題目敘述

題意說明：讀入資料檔 (1117hw.txt)，檔案中是多個中序的運算式所組成的字串，請利用堆疊 (Stacks) 的原理來撰寫一個程式，將檔案中的字串分別讀入轉換成後序運算式，並且輸出運算結果。

3 作法

宣告說明：st 存放 operator，res 存放 numbers，resPrefix 為後序結果，cnt 為左括號數量。先將中序的運算子讀入，並且使用 for 迴圈遍歷每一個字元。當遇到數字的時候繼續累加數字。

否則，當 haveNum == true 時，將 now Push 進 res，如果沒有左括號（包含當前），判斷 operator 的優先層級（乘除大於加減），如果 Stack 裡的比當前的大就先拿出來做後序，如果當前字元是 ')' 的話將 st 的運算符號拿出，直到左括號或堆疊為空為止（範例 1）。

迴圈執行完成後要將 st 剩餘的 operator 處理完（範例 2）

範例 1

```
1 for(const char i : ss) {  
2     if(i >= '0' && i <= '9') {  
3         now *= 10;  
4         now += (int)(i - '0');  
5         haveNum = true;
```

```

6     resPrefix += i;
7 }
8
9 else {
10     if(haveNum) {
11         res.Push({val, now, '\0'});
12         now = 0;
13         haveNum = false;
14     }
15     if(i != ')') {
16         if(i == '(') {
17             cnt++;
18         }
19         if(cnt <= 0) {
20             if(!st.IsEmpty() && getPriority(st.Top().oper) >=
                getPriority(i)) {
21                 res.Push(st.Top());
22                 resPrefix += st.Top().oper;
23                 st.Pop();
24             }
25         }
26         st.Push({op, 0, i});
27     }
28
29     else {
30         cnt--;
31         while(st.Top().oper != '(' && !st.IsEmpty()) {
32             res.Push(st.Top());
33             resPrefix += st.Top().oper;
34             st.Pop();
35         }
36         st.Pop();
37     }

```

```
38 }
39 }
```

範例 2

```
1 if(haveNum) res.Push({val, now, '\0'});
2 while (!st.IsEmpty()) {
3     res.Push(st.Top()); // 彈出堆疊中剩餘的運算符
4     resPrefix += st.Top().oper;
5     st.Pop();
6 }
```

將中序轉換後會得到一個堆疊 `res` 以及一個字串 `resPrefix`，將 `res` 依照後序運算原理操作便可得到結果（範例 3）

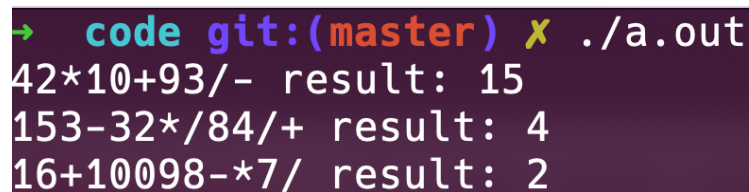
範例 3

```
1 mystack Postfix;
2 for(int i = 0; i <= res.top; i++) {
3     if(res.a[i].type == op) {
4         int a = Postfix.Pop().value;
5         int b = Postfix.Pop().value;
6         swap(a, b);
7         if(res.a[i].oper == '+') {
8             a += b;
9         } else if(res.a[i].oper == '-') {
10            a -= b;
11        } else if(res.a[i].oper == '*') {
12            a *= b;
13        } else {
14            a /= b;
15        }
16        Postfix.Push({val, a, '\0'});
17    }
18 }
```

```
17 | } else {  
18 |     Postfix.Push({val, res.a[i].value, '\0'});  
19 | }  
20 | }
```

4 執行結果

輸入輸出結果：(圖二)

A terminal window with a dark purple background. The prompt is 'code git:(master) X ./a.out'. Below the prompt, three lines of output are shown: '42*10+93/- result: 15', '153-32*/84/+ result: 4', and '16+10098-*7/ result: 2'.

```
→ code git:(master) X ./a.out  
42*10+93/- result: 15  
153-32*/84/+ result: 4  
16+10098-*7/ result: 2
```

Figure 2: 圖二

5 心得與討論

這次的作業的先備知識需要先了解中序轉後序的原理以及利用後序計算結果，這讓我瞭解了如何例用 Stack 實作四則運算的題目，我覺得十分有挑戰性。實作過後我將結果丟到 ZeroJudge 上測試也沒有什麼問題，我覺得資料結構的能力以及對程式的手感有大幅提升。