程式設計要點

鍾宜玲

軟體建構之道

- 作者：Steve McConnell
- 被公認為是程式設計界最實用的指南之一
- 寫程式就像蓋房子
- 沒有設計圖…直接蓋起來？
- 蓋到一半…一樓隔間全部打掉改成車庫..
- 這種房子沒有人敢買！
- 但是這種程式倒是很多人在使用！

需求

- 開始編寫程式時最好是能充份了解客戶的需求..
- 了解客戶需求 > 設計..

- 充份了解題意
  - Input：輸入幾筆資料？資料型態為何？
  - Output：輸出甚麼資料？型態呢？
  - 如何解題？

寫程式時的態度

- **實驗！**
- 有效程式設計關鍵：

  快速學習犯錯，每次從中獲取教訓。
- 嘗試瞭解編譯器發出的警告，而不是不管這些訊息。
- 清楚瞭解自己的程式，而不是將它編譯，看它是否能夠運作。

命名

- 錯誤範例
  - `doSomething(int inValue)`
  - `function(method)`
- **不要取一些語意不清的變數名稱**
- **不要用模糊、彈性太高的單字作為命名的一部分**
- HandellutputFile() 改為 FormatAndSave()
- **縮寫的藝術**
- 盡量用常見或是重音做為縮寫的單字
  - cls = close, del=delete, ins=insert, emp=employer

減少反向邏輯的判斷

```
if(!DoSomething() ) {
      if(!NotXXX())
         …
}
```

克服「複雜度」

- 分解。
- 謹慎地定義類別介面。
- 避免使用全域資料。
- 避免使用深度繼承。(超過3層就太多了)
- 避免使用深層迴圈和條件陳述式。

  (if裡面的if裡面的if裡面的if裡面的if...
  隔2個小時回來看都不一定看懂~)

克服「複雜度」

- 單一類別不要過於龐大。
  (可能這個類別太多任務了，請試著把它分解)
- 保持函式簡短。
- 運用慣例，節省腦力。
  - 前人走過留下的好東西要收下，不要再自創，把腦袋拿來做別的事
- 不要讓程式設計的想法侷限在你所用的語言。**以想要做的事為出發點**去思考，再想如何使用可用的程式設計工具去完成目標。

參數

- **參數不要超過七個**
  - 一般人能夠記憶數量的臨界值
- **參數最好有順序性**
  - 先用到的放前面，或是相同用途的放做一起
- **Function 應以單一用途為主**
  - 過多功能使得程式不容易 reuse
  - 命名➔可能會有CustomerAddRemoveModifySearch() 的 function name
- **盡量不要使用 ref or out 將傳入參數修改**
  - 在 function 改變變數值常常使得變數的改變很難追蹤，trace code 時需要深入 function並且瞭解其如何改變，很費工！應使用回傳方式。

試題(1)

- 寫一個函數計算當參數為 n (n很大) 時
  1 - 2 + 3 - 4 + 5 - 6 + 7 - … n 的值

程式(1)


```
long fn(long n)
{
  long temp=0;
  int i, sign=1;
  if(n<=0) {
     printf("error: n must > 0");
     exit(1);
  }
  for(i=1;i<=n;i++) {
     temp=temp+sign*i;
     sign=(-1)*sign;
  }
  return temp;
}
```

$O(n)$

在嵌入式系統的開發中，程式的運行效率很重要，能讓CPU少執行一條指令都是好的 。

程式(2)

```
long fn(long n)
{
 long temp=0;
 int j=1,i=1;
 if(n<=0){
    printf("error: n must > 0");
    exit(1);
 }
 while(j<=n){
    temp=temp+i;
    i=-i;
    i>0?i++:i--;
    j++;
 }
 return temp;
}
```

乘法改為加法指令

$O(n)$

思考

- 1 = 1
  1-2 = -1
  1-2+3 = 2
  1-2+3-4 = -2
  1-2+3-4+5 = 3
  1-2+3-4+5-6 = -3
  1-2+3-4+5-6+7 = 4
  1-2+3-4+5-6+7-8 = -4
- n為偶數： (1-2) + (3-4) + (5-6) + (7-8) …… + ((n-1)-n)
- n為奇數： (1-2) + (3-4) + (5-6) + (7-8) …+ ((n-2)-(n-1))+n

程式(3)

```
long fn(long n)
{
 if(n<=0) {
    printf("error: n must > 0");
    exit(1);
 }
 if(n%2==0)
    return (n/2)*(-1);
 else
    return (n/2)*(-1)+n;
}
```

$O(1)$

```
return n/2*(-1)+ ((n%2)?n:0) ;
```

程式(4)

$O(1)$

- 1 = 1
  1-2 = -1
  1-2+3 = 2
  1-2+3-4 = -2
  1-2+3-4+5 = 3
  1-2+3-4+5-6 = -3
  1-2+3-4+5-6+7 = 4
  1-2+3-4+5-6+7-8 = -4

```
if(n%2==0)
    return (n/2)*(-1);
else
    return (n/2)+1;
```

```
return (n%2)?(n/2)*(-1):(n/2)+1;
```

- n為偶數： (1-2) + (3-4) + (5-6) + (7-8) … + ((n-1)-n)
- n為奇數： 1+ (-2+3) + (-4+5) + (-6+7)+ …+ ((n-1)-n)

試題(2)

- 用一個函數實現兩個函數的功能

$$fn1 = \frac{n}{2!} + \frac{n}{3!} + \frac{n}{4!} + \frac{n}{5!} + \frac{n}{6!}$$

$$fn2 = \frac{n}{5!} + \frac{n}{6!} + \frac{n}{7!} + \frac{n}{8!} + \frac{n}{9!}$$

$$fn = \begin{cases} fn1, & \text{if } flag = 0 \\ fn2, & \text{if } flag = 1 \end{cases}$$

程式(1)

- 利用空間換取時間

t	0	1	2	3	4
0	2!	3!	4!	5!	6!
1	5!	6!	7!	8!	9!

```
double fn(int n, int flag)
{
  int temp=0;
  double t[2][5]={{2,6,24,120,720},{120,720,5040,40320,362880}};

  for(i=0; i<5; i++)
     temp = temp + n/t[flag][i];
  return temp;
}
```

$O(1)$

程式(2)

```
double fn(int n, int flag)
{
  double fn1=1/2+1/6+1/24+1/120+1/720;
  double fn2=1/120+1/720+1/5040+1/40320+1/362880;
  if(flag == 0)
     return n*fn1;
  else
     return n*fn2;
}
```

```
return n*(flag==0)?fn1:fn2;
```

參考資料

- http://www.facebook.com/note.php?note_id=341682696672
- http://tonycube.blogspot.com/2009/11/code-complete2.html
- http://www.dotblogs.com.tw/honorliao/archive/2010/10/09/18241.aspx
- http://www.vixual.net/blog/archives/99
  - 轉載自：http://www.vcroad.net/
- http://www.csie.ntnu.edu.tw/~u91029/Greedy.html