國立屏東商業技術學院 資訊工程系 程式設計(一)

5. IPO模型分析與程式設計

- IPO模型分析
- 基本輸入輸出(Basic Input and Output)
- 變數與記憶體(Variable and Memory)
- int整數資料型態
- 隨機變數(Random Number)

本章將透過簡單的範例程式，帶您瞭解C語言的輸入與輸出處理，以及變數的概念。我們將介紹一個程式設計模型 - Input/Process/Output Model，並且說明如何配合此模型進行程式設計。另外，本章也將就變數與記憶體空間的使用，做一概念性的說明。

<note tip>

程式設計模型(Programming Model)

- 所謂的<html>程式設計模型(Programming Model)</html>指的是一種思考的方法，我們可以據以構思問題的解決方案或是用以描述程式的結構。採用適當的模型，將有助於程式設計的工程化，讓程式設計如其它工程方法一樣，具有一定的進行步驟、規範與方法。

</note>

5.1 Lucky Number

請先參考以下的程式，將它加以編譯並執行，看看會得到什麼樣的結果？

h Example: luckyNumber.c

```
#include <stdio.h>

int main()
{
   int n;

   printf("Please input a number:");

   scanf("%d",&n);

   printf("Your lucky number is %d!\n", n);
}
```

其實這個程式就是一個典型的IPO模型(Input Process Output Model)的程式，先取得使用者的輸入，然後加以處理，最後將結果輸出。雖然這個模型並不適用於所有的程式設計問題，但是對初學者來說，這是一個相當值得參考的模式，同學只要熟悉這個方法，簡單的程式應該都可以應付了。下一節將說明luckyNumber.c以及IPO的細節。

5.2 IPO模型

我們先將luckyNumber.c的需求以IPO模型描述如下：

- I:取得使用者輸入的一個整數
- P:無
- O:輸出“Your lucky number is N”，N為使用者所輸入的整數

接著，我們分別就I、P、O的部份加以說明。

5.2.1 Input

在I的部份，要進行的是取得使用者輸入的資料(在luckyNumber.c中，我們所要取回的是一個整數)。我們可以使用定義在stdio.h中的scanf函式來取得使用者的輸入，其函式原型如下：

Prototype	scanf(format, memory_address)	
Header	stdio.h	
Parameters	**name**	**description**
	format	描述所欲取得的資料之格式
	memory_address	指定取回的資料所存放的記憶體位置

其中，format所指定的資料格式，是由一個字串指定，稱為<html>格式字串(format string)</html>，在luckyNumber.c中，該字串內容為“%d”，%d表示一個decimal number(10進制的整數)。因此，scanf(“%d”, &n)當中的格式字串，就表示要取回一個十進制的整數。我們將%d稱為<html>格式指定子(format specifier)</html>，當然除了%d外，還有很多格式指定子可以使用，這點在未來我們將會提供更完整的說明。

那麼使用者在執行時所取回的(來自於鍵盤)輸入，要怎麼保存呢？事實上，電腦系統在執行程式時，所有的資料都是存放在記憶體當中。因此scanf的第二個參數，就是指定所欲存放資料的記憶體位置。可是記憶體空間是由作業系統負責管理的，每當有程式要求執行時，作業系統的<html>動態記憶體配置(或稱動態記憶體管理)模組</html>會動態地分配一塊空間給該程式使用，但在絕大多數的情況下其所分配到的位置都不相同(當然也有相同的可能，只是機率比較低)。因此，在設計程式時，我們只要在程式中說明我們需要一塊記憶體空間，後續需要指定記憶體位址時，就告訴電腦要使用之前所分配給我們的那一個位址即可。

具體的做法是，先告訴電腦我們需要一個記憶體位置來存放一個整數，並且為了方便管理程式中還可能會需要的其它記憶體位址，我們必須為這個(需要空間來存放的)整數取個名字，例如以下的宣告：

```
int n;
```

上述的程式碼，稱為<html>變數宣告(variable declaration)</html>，表達了我們需要一個記憶體空間來存放一個稱為n的整數。在程式設計的術語裡，這個整數n被稱為<html>變數(variable)</html>，換句話說，以上的宣告會要到一塊記憶體空間來存放變數n，

當程式執行時，會產生一個稱為symbol table的表格，用來記錄所有的變數所分配到的記憶體空間，例如

symbol	type	address
n	int	100

<note important>

變數的相對與真實記憶體位址

變數n所分配到的位置，其實每次都是固定的！但是每次程式執行時，由作業系統的動態記憶體配置(或稱動態記憶體管理)模組所分配給程式的記憶體區塊是不固定的，Compiler會幫我們標記並分配記憶體空間給所需的變數，但是以相對位址來指定。假設Compiler將變數n分配在+100的記憶體位址，就代表從作業系統動態分配給程式的空間的開頭處往後加上100，才是變數n真正儲存的位址。若程式某次執行時，被分配到記憶體空間2500到3524之處，那麼變數n就位於2500+100＝2600的位址上。由於每次程式取得的空間不同，所以變數每次分配到的真實(又稱為絕對)記憶體空間自然也就不會一樣。您應該會在計算機概論、作業系統、程式語言、編譯器與系統程式等課程，學習到關於記憶體管理的更多細節。後續本文將不特別區分變數之真實與相對位址。

</note>

當變數宣告完成後，我們可以使用&運算子來表達變數所在的記憶體位址，例如變數n所存放的位置，可以在程式碼中以&n來表示。在前述的例子中，變數n的記憶體位置為100，意即&n為100。所以當我們以scanf(“%d”, &n)來取得一個整數時，該整數的值(value)將會被存放到記憶體編號100的位置。假設使用者輸入的整數是3，那麼：從程式的角度來看，存放在記憶體位址100的變數n的值為3，也就是n=3，&n=100。

因此，scanf(“%d”, &n)就可以把使用者所輸入的數值存放到變數n所在的記憶體位址了。再強調一次，在程式碼中n代表一個變數，&n則代表這個變數的數值所存放的記憶體位址。

<note important>

使用scanf()常犯的錯誤

初學者(以及粗心的人)在使用scanf()，有一個常犯的錯誤：將欲存放的記憶體位址寫成了變數。例如scanf(“%d”, n)，這就會要求電腦將使用者的輸入存放到一個(非預期的)記憶體位址(其值為n的數值)。延續前述的假設n=3且&n=100，那麼scanf(“%d”,n)的結果就把資料寫入到了記憶體位址3的地方，而不是變數n所在的位址。

</note>

通常，我們不會單獨地使用scanf()，試試看將luckyNumber.c中第7行移除(更好的方法是先將它註解起來)，程式執行時會發生什麼事？若是您已經知道程式要求您輸入一個整數，那應該就沒有問題；但若使用者並不知道(或是忘記了)，那麼他可能會認會程式當掉了！因為執行後，電腦沒有任何反應。所以我們通常會在使用scanf()取得輸入前，加一行printf()所印出的字串，利用這個字串的內容來提示使用者該做些什麼、或是該輸入什麼樣式的資料。請將第7行加回程式，再看看下面的執行結果：

```
[14:47 user@ws example]./a.out
```

```
Please input a number: 3
Your lucky number is 3!
[14:47 user@ws example]
```

我們將這種字串稱為提示字串，通常不會以\n結尾(不過這與程式的要求或喜好有關)。如果還有同學不瞭解\n的意義，那表示您還沒有得到4.3.3小節末要你去try的答案，趕快去try吧！

5.2.2 Process

在luckyNumber.c這個例子中，並沒有需要對使用者所輸入的資料做任何的處理，因此我們先忽略此部份。

5.2.3 Output

最後luckyNumber.c要求輸出一個使用者的幸運數字。當然，這個幸運數字就是使用者自己輸入的數字(命運是掌握在自己的手裡啊！)，因此我們只要想辦法將使用者剛輸入的數字再輸出即可。請參考程式第11行：

```
printf("Your lucky number is %d!\n", n);
```

我們又再次使用了定義在stdio.h中的printf()函式，但這次有些不一樣的地方，請先參考以下的函式原型：

Prototype	printf(format, values)	
Header	stdio.h	
Parameters	**name**	**description**
	format	描述所欲輸出的資料之格式
	values	要輸出的數值

第一個參數是格式字串，第二個則是數值。其實printf()是取print with format之意，將指定在後的數值套用在指定的格式字串上，加以輸出。也就是說，將格式字串的內容原封不動地輸出，但其中若有格式指定子(format specifier)則以後續的數值來套用。因此printf("Your lucky number is %d!\n", n)就是將變數n的數值代入到格式字串中再加以輸出。假設n的值為3(這個數值是使用者輸入的)，那麼最後輸出的內容為"Your lucky number is 3!"

有沒有注意到前述函數原型中，參數values是以複數的方式呈現！沒錯，這表示可以輸出一個或一個以上的數值資料。若有兩筆或兩筆以上的資料時，數值與數值間必須以一個逗號(,)來加以分隔。請參考以下的例子：

```
printf("The value of n is %d, and n+6=%d.\n", n, n+6);
```

這個例子不但輸出了兩個數值，其中第二個數值還是一個運算的結果。

5.3 consoleIPO Framework

讓我們再看一次luckyNumber.c

h Example: luckyNumber.c

```
#include <stdio.h>

int main()
{
   int n;

   printf("Please input a number:");

   scanf("%d",&n);

   printf("Your lucky number is %d!\n", n);
}
```

先前的(較粗略的)的IPO模型分析如下：

- I:取得使用者輸入的一個整數
- P:無
- O:輸出“Your lucky number is N”，N為使用者所輸入的整數

現在我們已經討論過luckyNumber.c程式的細節，讓我們把學到的知識應用到IPO分析，讓我們的模型更加明確、更加接近編寫程式的需求：

- I:取得使用者輸入的一個整數，把它放入變數n
- P:無
- O:輸出“Your lucky number is”以及變數n的數值

OK，現在讓我們看看該如何完成這個程式，首先把C語言程式的console framework寫下來:

```
int main()
{
}
```

我們再加上一些註解，讓這個consoleFramework更接近IPO模型：

```
int main()
{
   // Input Section


   // Process Section


   // Output Section
```

```
}
```

這樣還不夠，再加上寫程式可能會用到的函式庫定義檔的載入及變數的宣告這兩個部份：

h console IPO framework

```
// Header File Section


int main()
{
   // Variable Declaration Section

   // Input Section


   // Process Section


   // Output Section

}
```

我們把這個framework稱為consoleIPO。現在把前面的IPO分析套用在這個framework上：首先是I的部份“ 取得使用者輸入的一個整數，把它放入變數n “ ，利用scanf()函式，我們可以寫出以下的程式:

```
// Header File Section


int main()
{
   // Variable Declaration Section

   // Input Section
   scanf("%d", &n); //注意！是□n不是n!

   // Process Section


   // Output Section

}
```

因為P的部份不用處理，所以直接處理O的部份 ” 輸出”Your lucky number is“以及變數n的數值 “ ， 利用printf()將程式碼加到Output Section中：

```c
// Header File Section


int main()
{
   // Variable Declaration Section

   // Input Section
   scanf("%d", &n); //注意！是□n□不是n!

   // Process Section


   // Output Section
   printf("Your lucky number is %d!\n", n); //注意！是n□不是&n!
}
```

最後，把程式中所有使用到的變數加到Variable Declaration Section中，還有因為scanf()與printf()是定義在stdio.h標頭檔，所以也要它加到Header File Section中：

```c
// Header File Section
#include <stdio.h>

int main()
{
   // Variable Declaration Section
   int n;

   // Input Section
   scanf("%d", &n); //注意！是□n□不是n!

   // Process Section


   // Output Section
   printf("Your lucky number is %d!\n", n); //注意！是n□不是&n!
}
```

完成了！是不是很簡單？您還可以在Input Section中再加上提示字串，這樣一切就更完美了! 後續我們將練習如何使用IPO分析與consoleIPO framework來完成程式設計。

5.4 Luck Number 2 隨遇則安

現在讓我們動手寫寫程式，請參考以下的要求：

試寫一C語言程式luckyNumber2.c□告訴使用者他的幸運數字(介於0~9)是多少?

程式執行時應該要有以下的輸出畫面：

```
[15:17 user@ws example] ./a.out
Eenie meenie minie mo...
Your lucky number is 5!
[15:17 user@ws example] ./a.out
Eenie meenie minie mo...
Your lucky number is 8!
[15:17 user@ws example] ./a.out
Eenie meenie minie mo...
Your lucky number is 0!
[15:17 user@ws example]
```

(衣泥～咪泥～媽迷～繆)

注意到了嗎？每次執行時輸出的結果是不一樣的，就好像丟骰子一樣，沒人知道會出現幾點？那就隨遇而安吧！那麼，這個程式該怎麼寫呢？讓我們用IPO模型來試試看吧。

5.4.1 IPO模型分析

讓我們分析一下這個程式的要求：

1. 它沒有要使用者輸入資料的部份
2. 要像丟骰子一樣，輸出一個介於0~9的數字

以IPO模型可以表達如下：

- I:無
- P:想辦法產生一個介於0~9的隨機變數
- O:把那個變數加以輸出

當然，我們還可以放入更多細節：

- I:無
- P:想辦法產生一個介於0~9的隨機變數，並將其數值放入變數n
- O:
    1. 輸出”Eenie meenie minie mo...”
    2. 輸出“Your lucky number is”以及n的數值

因為還不會使用C語言產生隨機變數，所以我們可以暫時將程式簡化為：

- I:無
- P:
    1. 假設變數x是一個介於0~9的隨機變數
    2. 將x的數值放入變數n□也就是n=x
- O:
    1. 輸出“Eenie meenie minie mo...”

2. 輸出“Your lucky number is”以及n的數值

依據現在的IPO分析，結合consoleIPO framework，我們可以得到下列程式片段：

```
// Header File Section
#include <stdio.h>

int main()
{
   // Variable Declaration Section
   int n;
   int x;

   // Input Section

   // Process Section
   n=x;

   // Output Section
   printf("Eenie meenie minie mo...\n");
   printf("Your lucky number is %d!\n", n); //注意！是n，不是&n!
}
```

現在，只要想辦法解決“ 如何讓x是介於0~9的隨機變數 ”這個問題，程式就可以完成了，C語言已經預先幫我們開發了許多有用的函式，只要能善用這些函式，程式的開發將會是一件很容易的事情。現在這個問題也不例外，我們可以使用定義在time.h中的rand()函式，來完成luckyNumber2.c這個程式，rand()是一個隨機變數產生器，可以幫我們產生介於0～1之間的數值

h luckyNumber2.c

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main()
{
   int n;
   int x;

   srand(time(NULL));
   x = rand();
   n = x % 10;
   printf("Eenie meenie minie mo...\n");
   printf("Your lucky number is %d!\n", n);
}
```

From:
https://junwu.nptu.edu.tw/dokuwiki/ - **Jun Wu的教學網頁**
國立屏東大學資訊工程學系
CSIE, NPTU
Total: 194548

Permanent link:
https://junwu.nptu.edu.tw/dokuwiki/doku.php?id=c:biovar

Last update: **2019/07/02 15:01**