國立屏東大學 資訊工程學系 程式設計(一)

5. IPO模型分析與程式設計

- IPO模型分析
- scanf()函式
- 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。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>

請先編輯luckyNumber.c的程式碼，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);
}
```

5.1 IPO模型與需求分析

luckyNumber.c這個程式是一個典型的IPO模型(Input Process Output Model)的程式：先取得使用者的輸入(input)，然後加以處理(process)，最後將結果輸出(output)，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。

我們可以將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。

5.1.1 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

scanf()對應的標頭檔是stdio.h，要記得以#include<stdio.h>加以載入。在參數的部份，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4.4.3小節末要求你去try的答案，趕快去try吧！

5.1.2 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。

5.1.3 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。

5.2 IPO程式設計

前一節，以luckyNumber.c為例，說明了其需求可以IPO模型加以分析與描述。本節將介紹如何以IPO分析

來完成相關的程式設計工作，首先還是以luckyNumber.c為例，示範IPO程式設計的方法，後續再以其它範例做更詳細的說明。

5.2.1 Lucky Number

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：

```
// 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：

```
// 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! 後續我們將繼續練習如何使用IPO分析與consoleIPO framework來完成程式設計。

5.2.2 Lucky Number 2

現在讓我們動手以IPO程式設計方法來寫寫程式，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]
```

<note tip>Eenie meenie minie mo (衣泥～咪泥～媽泥～繆) </note>

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。

5.2.2.1 IPO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stdlib.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，又稱為亂數產生器，定義在stdlib.h中，可以幫我們隨機產生一個介於0，RAND_MAX之間的整數數值，RAND_MAX也定義在stdlib.h中，在不同系統平台上其值可能會有變化，您可以使用下列的程式碼片段來取得該值：

```
printf("The value of RAND_MAX is %d.\n", RAND_MAX);
```

當然，您必須先載入stdlib.h才能正確地編譯與執行。以Mac OS X 10.8.4為例，RAND_MAX的值為2147483647，那就表示rand()函式可回傳一個介於0到2147483647的整數數值。

試試下面的程式片段：

```
printf("A random number %d is generated by rand() function.\n", rand() );
```

它可以隨機產生一個整數值。但是若執行多幾次呢？結果又會如何？那如果在一個程式內使用多次rand()又會如何呢？例如：

```
printf("A random number %d is generated by rand() function.\n", rand() );
printf("A random number %d is generated by rand() function.\n", rand() );
printf("A random number %d is generated by rand() function.\n", rand() );
printf("A random number %d is generated by rand() function.\n", rand() );
```

有沒有發現每次的數值都一樣？這是因為rand()的必須參考一個種子數(seed number)來產生隨機變數(又稱為亂數)，因為我們並沒有在程式中設定這個種子數的數值，因此其結果就等同於每次都設定其為預設值。換言之，每次執行時其種子數皆相同，因此自然會產生一樣的亂數。為了解決這個問題，我們可以使用srand()函式來設定rand()函式所需參考的種子數srand()也定義在stdlib.h中，其參數為所欲設定的種子數，例如以下的程式碼片段，設定亂數產生器之種子數為3：

```
srand(3);
printf("A random number %d is generated by rand() function.\n", rand() );
```

多執行幾次看看，您會發現結果還是一樣！這是因為我們還是每次都設定了一樣的數值(也就是以3做為種子數)，所以我們還需要使用別的方法來讓程式每次執行都可以設定不同的種子數。定義在time.h標頭檔中的time()函式，若傳入NULL參數，可以傳回自1970年1月2日00:00:00起迄今過了多少時間(其值是以秒為單位)。由於時間是一直變動的，所以每次你呼叫time(NULL)其傳回值也就不盡相同(除非在同一秒內呼叫多次)。我們可以利用這個函式來做為初始值的設定。

```
srand( time(NULL) );
printf("A random number %d is generated by rand() function.\n", rand() );
```

執行看看結果是否每次產生的亂數還會一樣嗎？

<note tip> srand(time(NULL))的呼叫方式，會先執行裡面的time(NULL)取得傳回值後做為參數再呼叫srand()就如同數學的函數一樣，假設兩個funciton f(x)=2x與g(y)=y+3f(g(3))的值必須先計算g(3)的值再傳給f做為參數，意即g(3)=6f(g(3))=f(6)=12 </note>

考慮以下的程式片段：

```
srand(time(NULL));
x=rand();
```

現在x的數值是一個介於0到RAND_MAX的整數值了，但接下來要解決的問題是luckyNumber2.c要求的是一個介於0到9的整數，我們要如何處理呢？我們可以讓電腦幫我們做一個除法，讓x除以10，其餘數必定是介於0到9之間的一個整數。除了常用的加減乘除外，電腦系統還提供一個特別的數學運算子－餘除，意即進行除法但傳回其餘數數值。在C語言中，餘除運算子是以%來表示，因此，我們可修改上面的程式碼如下：

```
srand(time(NULL));
x=rand();
```

```
n = x % 10;
```

如此一來，變數n的數值就會是一個介於0到9之間的整數值了。

<note important> 注意□C語言是一個左值(left value)的程式語言，意即任何數學運算式都是先計算等號右邊的數值，再將其值指派給等號左邊。例如n=x與x=n其運算結果是不一樣的。假設n與x的數值原本分別為3與5□n=x會把等號右邊的x的數值指派給左邊的n□因此其結果使得n與x的數值皆為5；同理□x=n會使兩者的值皆為3□ </note>

最後，讓我們把上面的程式碼片段加到luckyNumber2.c的Process Section中，不要忘記在Header File Section內載入所需的標頭檔：

h luckyNumber2.c

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

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

   // Input Section

   // Process Section
   srand(time(NULL));
   x=rand();
   n = x % 10;

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

至此□luckyNumber2.c以IPO程式設計方法完成了程式開發。

5.2.3 Lucky Number 3

本章最後再示範一次IPO程式設計，請參考以下的要求：

試寫一C語言程式luckyNumber3.c□讓使用者輸入一個最小值與最大值，然後程式輸出一個介於兩者間

的一個幸運數字(介於最小值與最大值之間)。

程式執行時應該要有以下的輸出畫面：

```
[15:17 user@ws example] ./a.out
Please input the minimal number: 3
Please input the maximal number: 6
Eenie meenie minie mo...
Your lucky number is 5!
[15:17 user@ws example] ./a.out
Please input the minimal number: 13
Please input the maximal number: 73
Eenie meenie minie mo...
Your lucky number is 23!
[15:17 user@ws example]
```

首先讓我們以IPO分析這個程式的要求：

- I:要求使用者輸入兩個整數，一個是最小值，一個是最大值
- P:想辦法產生一個介於最小值與最大值之間的隨機變數
- O:把那個變數加以輸出

接著放入更多細節：

- I:
  1. 以scanf()取得最小值，放到變數min中
  2. 以scanf()取大最大值，放到變數max中
- P:
  1. 以目前時間設定亂數種子數
  2. 想辦法產生一個介於min與max間的隨機變數，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

我們還可以再詳細一點，把目前已經學會的C語言程式語法加到IPO分析中：

- I:
  1. scanf(“%d”, &min);
  2. scanf(“%d”, &max);
- P:
  1. srand(time(NULL));
  2. x = 介於min與max的隨機變數
  3. n = x;
- O:
  1. printf(“Eenie meenie minie mo...\n”);
  2. printf(“Your lucky number is %d!\n”, n);

我們將Head File Section與Variable Declaration Section也加入其中，並放入適當的程式碼(分別以H和D表示這兩個Sections)。

- H:
  1. #include <stdio.h>
  2. #include <stdlib.h>
  3. #include <time.h>
- D:
  1. int n;
  2. int x;
  3. int min;
  4. int max;
- I:
  1. scanf(“%d”, &min);
  2. scanf(“%d”, &max);
- P:
  1. srand(time(NULL));
  2. x = 介於min與max的隨機變數
  3. n = x;
- O:
  1. printf(“Eenie meenie minie mo...\n”);
  2. printf(“Your lucky number is %d!\n”, n);

注意到上述的分析中，絕大部份都是已經可以直接加入到ConsoleIPO framework中的程式碼，只剩下如何產生介於min與max的隨機變數這個問題□max-min+1可以表達介於min合max之間共有幾個數字，假設min與max分別為3與5，那麼x必須是介於3到5之間的數值，且5-3+1共有3個數字在3與5之間(也就是3,4,5)。在前一小節的討論中，我們已經知到rand()的結果餘除10即可產生介於0到9之間的亂數，那麼如果讓rand()餘除(5-3+1)，也就是rand()%3□就可以產生0、1或2這三個可能的數值。接著再把產生出的數值加上min(也就是3)，那麼原本可能的0, 1, 2加上3之後就成了3,4,5這三個可能的數值。因此我們可以得到以下的程式碼：

```
x = min + (rand()%((max-min)+1))
```

x就成為了介於min到max間的亂數了！將這個程式碼寫入我們的IPO分析中，並且也順便加入提示使用者輸入的提示字串：

- H:
  1. #include <stdio.h>
  2. #include <stdlib.h>
  3. #include <time.h>
- D:
  1. int n;
  2. int x;
  3. int min;
  4. int max;
- I:
  1. printf(“Please input the minimal number:”);
  2. scanf(“%d”, &min);
  3. printf(“Please input the maximal number:”);
  4. scanf(“%d”, &max);
- P:

    1. srand(time(NULL));
    2. x = min+rand()%(max-min+1);
    3. n = x;
- O:
    1. printf(“Eenie meenie minie mo...\n”);
    2. printf(“Your lucky number is %d!\n”, n);

只要把這個IPO分析的結果套用在ConsoleIPO framework中，luckyNumber3.c就可以輕易地完成，其程式碼如下:

h luckyNumber3.c

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

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

// Input Section
   printf("Please input the minimal number:");
   scanf("%d", &min);
   printf("Please input the maximal number:");
   scanf("%d", &max);

// Process Section
   srand(time(NULL));
   x = min+rand()%(max-min+1);
   n = x;

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

看完了luckyNumber.c、luckyNumber2.c與luckyNumber3.c，相信您應該已經可以掌握IPO程式設計的方法了。

5.3 Excerices

作業3

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: 193849

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

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