序

本書適合大學、科技大學與技術學院的資訊相關學系作爲入門的程式語言教材，同時也適合初學者自學之用。在全書17個章節中，涵蓋了C語言的入門基礎、資料型態、運算式、格式化的輸入與輸出、條件與流程控制、迴圈以及陣列等主題；也包含了進階的指標、字串、使用者自定資料型態、記憶體管理等主題及其相關應用。這些內容對於C語言的初學者而言已經相當足夠，更可以作爲未來學習其他程式語言的重要基礎。

爲了幫助讀者培養自己動手寫程式的能力，筆者精心設計了123個程式碼範例來進行演示。與坊間其他書籍不同之處，在於這些程式範例全部提供了完整的解析，並透過input-process-output模型與流程圖，詳細並逐步地說明解題的技巧與程式設計的過程，不但能幫助讀者們瞭解程式碼的意義與語法規則，更可以讓讀者們擁有程式設計最爲重要的思維技巧與邏輯觀念。本書更有多達315 個配合章節內容的課後練習，讀者們可以透過作答的過程檢驗自己的學習狀況，也可以累積可觀的程式設計實戰經驗。

除了各章節的範例程式與課後練習外，本書更進一步提供了19個進階的實務程式演練，涵蓋了各種眞實情境下的程式設計應用，不但能夠幫助讀者們掌握相關的邏輯觀念與設計技巧，更能夠培養出進階的程式設計能力。相關的主題包含了遊戲程式開發（例如猜數字遊戲、撲克牌的比大小遊戲、21點與五子棋遊戲等）、數據轉換與資料處理、函式庫的製作、學生成績管理、商品管理、銷售系統以及應用在即時系統（Real-Time Systems）的可排程性分析工具等主題。只要讀者們依本書內容詳細研讀，不但能夠將C語言程式設計學好，更能夠擁有未來與產業界接軌的實務能力。

本書雖力求完美，但筆者學識與經驗仍有不足，如有謬誤之處尙祈見諒並請不吝指正。

吳卓俊 junwu.tw@gmail.com
於屏東 2016年八月

光碟內容說明

爲了便利讀者的學習，本書隨附光碟中含有全書所有程式範例及實務演練的完整程式碼，以及部份課後練習所需要的資料、測試或程式檔案。光碟內含有以下的資料夾：

- Examples：此資料夾中含有ch2、ch3、…、ch17等共16個子資料夾，每個子資料夾存放有該章的範例程式碼，讀者可以自行參考。
- Projects：此資料夾存放有本書所有實務演練的完整程式，分別位於子資料夾1、2、…、19當中，請需要的讀者自行取得相關的檔案。
- Exercises：本書部份章節的課後練習需要一些額外的檔案，讀者可以在此資料夾裡的ch12與ch13這兩個子資料當中，取得所需要的資料、測試或程式檔案。

目錄

Chapter 02 您的第一個C語言程式

Chapter 03 IPO程式設計模型

Chapter 04 變數、常數與資料型態

第 1 章

電腦系統與程式語言

2016年3月，Google DeepMind公司的AlphaGo電腦圍棋軟體以四勝一負擊敗了南韓棋王李世乭。這場對弈標示了artificial intelligence（AI，人工智慧）的發展邁向了新的里程碑，人們開始注意到電腦也可以在腦力上戰勝人類，甚至開始擔心未來電腦會不會完全取代人類？在許多科幻小說與電影中，電腦系統總是被描繪爲在未來將具有超越人類能力並能自主思考的AI；這些過份聰明的電腦系統，通常並不會乖乖地爲人類服務，而是更聰明地發射核彈將人類消滅殆盡[1]，或是將人類豢養起來作爲生物電池，以提供電腦系統源源不絕的能源[2]。總之，有了太過聰明的電腦似乎不是什麼好事，就連當代最聰明的人Stephen Hawking（史蒂芬．霍金）都曾說過：「The development of full artificial intelligence could spell the end of the human race!（人工智慧發展到了極致，可能會導致人類的滅亡！）」[3]。不過請暫時先不用擔心，從在特定領域以智能打敗人類，到能夠眞正發展出擁有獨立思考能力的電腦系統，仍然還有很長的一段路要走。你可以放心地過好每一天，至少到目前爲止，電腦系統仍然是聽命於人類，乖乖地執行我們所交付的工作。當然，作爲資訊人的一份子，或許你將來也會爲實現眞正具備AI的超級電腦而努力呢！

電腦系統可以簡單地區分爲hardware（硬體）與software（軟體），例如個人電腦、智慧型行動電話、平板電腦、智慧型手錶等看得到、摸得到的實體裝置就是hardware；我們可以在這些hardware上執行各式不同應用目的之software，以滿足我們各種不同的需求— 例如我們可以開啓文書編輯軟體來編輯一份文件、開啓瀏覽器來查詢資料、玩遊戲軟體以消磨時間、或是利用軟體來進行多媒體影音作品的創作等。Software是由資料以及一個或多個program（程式）所組成，可以在電腦系統上執行以完成特定的工作。簡單的software可能僅由一個program組成，而複雜的software則可能有多個program組成，其中每個program負責特定的一部份功能。對於hardware而言，一個program是一些machine code（機器碼）[4]的集合，將這些machine code加以執行，就可以解決或滿足使用者特定的問題或需求。我們在前面提到的「過份聰明的電腦系統」，就是具備先進AI的電腦系統，並且會「自己」產生各種可以用來消滅或奴役人類的program。

1 例如在Terminator系列電影（台譯為「魔鬼終結者」）中的Skynet（天網）系統。
2 例如Matrix系列電影（台譯為「駭客任務」）中的Martix（母體）。
3 可參閱http://www.bbc.com/news/technology-30290540。
4 機器碼通常係指可以在處理器硬體上直接執行的指令，又稱為機器指令。

在目前的現實生活中，電腦系統所執行的各種程式仍是由我們人類所開發的 — 我們必須遵循programming language（程式語言）的語法規則，將所要執行的各種功能撰寫出來，經過compile（編譯）產生可在電腦系統中執行的machine code後，才能交付給電腦系統執行。本章將針對computer systems（電腦系統）的基礎知識加以說明，其中包含當代電腦系統的基礎 — stored-program computer architecture（儲存程式型電腦架構），以及電腦系統的hardware與software、programming languages（程式語言），以及本書的主角 — C語言，逐一加以簡介。

1-1 儲存程式型電腦

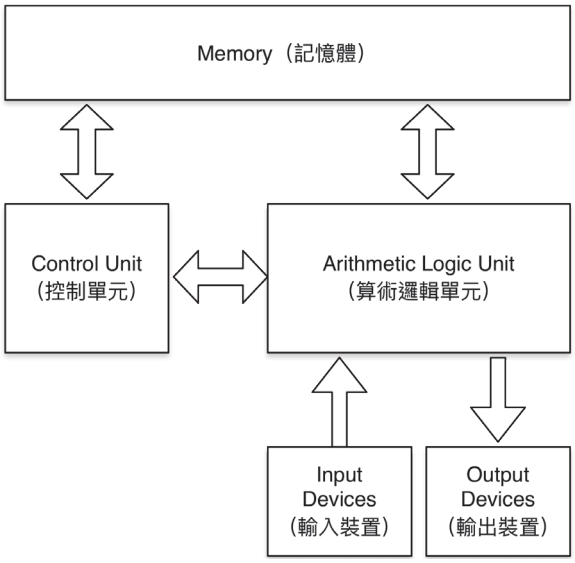
現今的電腦系統都是以stored-program computer architecture（儲存程式型電腦架構）爲基礎，此架構是由著名數學家John von Neumann（約翰．馮紐曼）於1945年所提出[1][2][3]，一般又被稱爲von Neumann model（馮紐曼模型）[5]，其主要概念是我們可以將工作事先撰寫成program並加以儲存，需要時再將其載入電腦系統執行即可完成工作。

隨著資訊科技的發展，電腦系統朝向行動化與普及化發展，有愈來愈多不同形式的電腦系統問世，除了傳統的桌上型個人電腦、筆記型電腦、各式伺服器、大型主機外，還有許多新穎的可攜式系統、嵌入式系統正融入我們的生活，包括智慧型行動電話、平板電腦、智慧型手錶、手持式/車用衛星導航系統、甚至連許多家電或生活用品內都可看到電腦系統的蹤影。雖然這些電腦系統的功能、形式皆不盡相同，但都同樣以stored-program computer architecture架構爲基礎。

在儲存程式型電腦架構中，電腦系統包含memory（記憶體）、arithmetic logic unit（ALU，算術邏輯單元）、control unit（控制單元）與input/output device（輸入/輸出裝置）等四大單元，如圖1-1所示。其中memory是用以儲存資料與program的空間，arithmetic logic unit負責執行算術與邏輯運算，例如兩個

5 一般認為在John von Neumann於1945年發表論文之前，儲存程式型電腦架構已在賓州大學的摩爾電機學院內流傳了，甚至早在1936年Konrad Zuse所提出的專利就已出現此一概念的雛型。關於儲存程式型電腦架構的概念另有一說是John Presper Eckert於1943年所創[4]。

數字的加減乘除或比較其大小等工作。Control unit則是控制memory、arithmetic logic unit及input/output devices的運作，通常與arithmetic logic unit合稱爲central processing unit（CPU，中央處理器）。至於input/output裝置則負責接收資料與程式（例如透過鍵盤與滑鼠來將資料輸入電腦系統，或是從磁碟來讀取資料或程式），並且將運算後的結果傳送出去（例如螢幕、印表機或是將結果儲存在磁碟內）。



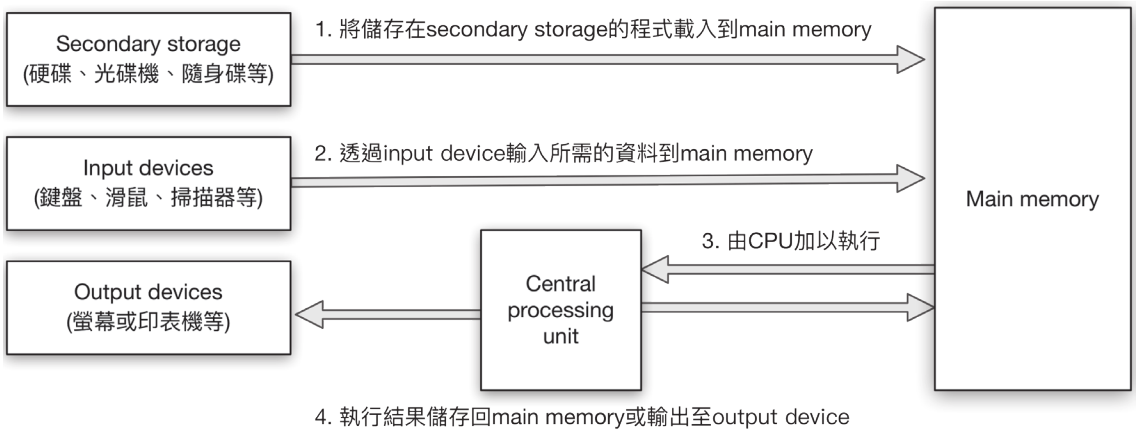
▶ 圖1-1：Stored-program computer architecture（von Neumann model）

與最早期的電腦系統比較，stored-program computer architecture內的memory除了可儲存資料外，也被用來儲存program，當我們要執行不同的program時，只需將其載入到memory中，即可由arithmetic logic unit加以執行。因此，電腦系統就可以視需要載入不同的program加以執行，就可以完成不同的工作。透過此一概念的延伸，我們可以將電腦系統區分爲hardware與software兩部份，兩者必須搭配運作才能完成特定的工作，單有hardware而無software是無法運作的，反之亦然。

1-2 電腦硬體

所謂的computer hardware（電腦硬體）係指電腦系統內實體的部份。雖然現今的電腦系統存在許多不同的形式，但絕大部份都仍符合stored-program computer architecture，其hardware元件可概分爲memory、arithmetic logic unit、

control unit與input/output device等四大單元。通常memory又可區分爲main memory（主記憶體）與secondary memory（次記憶體），其中secondary memory包含了儲存裝置，例如硬碟、軟碟與光碟機等，更常被稱爲secondary storage（次儲存體）。Arithmetic logic unit則與control unit合稱爲central processing unit（CPU），亦可簡稱爲processor（處理器）。至於input/output device則包含鍵盤、滑鼠、掃瞄器等輸入裝置與螢幕、印表機等輸出裝置。請參考圖1-2，一個典型的電腦系統要執行一個程式時，首先會將儲存在secondary memory的程式載入到main memory中，接著透過input device讓使用者輸入程式所需的資料並儲存到main memory，再由CPU執行程式並將執行結果儲存回main memory或透過output device將結果呈現給使用者。本小節將就這些相關的硬體做一簡介。



▶ 圖1-2：典型的stored-program computer硬體元件與program執行流程

1-2-1　主記憶體

Main memory（主記憶體）是用以儲存程式、資料與運算結果的硬體元件。一般又可依其特性分爲random access memory（RAM，隨機存取記憶體）與read-only memory（ROM，唯讀記憶體）兩類。RAM可以用來儲存要被執行的程式，並在執行時用以存放相關的數值、文字、甚至是圖片、影像等資料。RAM僅能暫時儲存資料，因爲它是所謂的volatile memory（揮發性記憶體），當電腦系統關閉電源或重新啓動後，原有的資料都將不復存在。相對的，ROM可長久地保有資料，稱爲nonvolatile memory（非揮發性記憶體）。Read-only memory就如同其名稱一樣，僅能讓我們讀取資料，並不能寫入新的資料，其所儲存的資料與程式是由硬體廠商事先加以燒錄，通常是負責電腦系統啓動時所要執行

本章習題

一、簡答題

1. 請列舉stored-program computer architecture所包含的四大單元為何？
2. 請說明何謂電腦系統的input/output device，並分別列舉兩個例子。
3. 請說明hardware與software的差異。
4. 請說明RAM與ROM的差異。
5. 請解釋何謂nonvolatile memory。
6. 請計算12MB的記憶體空間等於多少位元？
7. 請說明何謂memory cell？為什麼memory cell的大小與電腦系統的執行效率有關聯？
8. 請說明何謂secondary storage？你最常使用的secondary storage是什麼？
9. 請說明電腦系統中file與directory的關係。
10. 請說明CPU的主要功用為何？
11. 請列舉computer software常見的分類有哪些？
12. 請說明何謂commercial off the shelf？
13. 請說明program、programming以及programmer的意義分別為何？
14. 請說明第一代程式語言與第二代程式語言的差異為何？
15. Assembly language需要何種軟體工具，才能將其原始程式轉換為電腦系統可以執行的程式？
16. 為什麼low-level programming language的效率比high-level programming language更好？
17. 請說明high-level programming language與low-level programming language的差異為何？
18. 請解釋compilation與interpretation的差異。
19. Object code與executable code有何差異？要怎麼做才能將object code轉變成executable code？
20. Very high-level programming language與high-level programming language的主要差異為何？

21. 請說明high-level programming language的程式開發流程。
22. 請說明使用IDE來開發軟體的好處為何？
23. 請說明為何C語言是一個portability很好的語言？
24. 請列舉C語言有哪些標準？為何標準化對於軟體開發十分重要？

本章習題

想看看更高收益的伺服器市場能否有什麼發展，那麼現在就切換到Linux平台，應該是你最好的選擇！

若不能一次到位，直接改換成Linux作爲你的作業平台，那麼筆者建議可以安裝Cygwin，讓你在Microsoft Winows中得到類似Unix/Linux的操作環境，並且直接以terminal（終端機）進行程式的開發。Terminal是一個簡單的工具軟體，可以提供我們以console模式的各種指令來進行系統內的各項工作。在這樣的環境中，你可以使用vi、vim、emacs、pico或joe進行程式碼的編輯，然後使用cc或gcc進行編譯。如果你對於vi這一類的console模式下的text editor實在是用不習慣，那麼也可以搭配如Atom、Sublime Text等視窗界面的text editor進行程式編輯；但最終的編譯動作還是建議以文字模式的cc或gcc指令進行。當然，如果你擁有Unix/Linxu伺服器的帳號，也可以在Microsoft Windows中，使用telnet/SSH client的軟體（例如putty），透過網路連上伺服器直接使用Unix/Linux的console模式來開發C語言的程式。

準備好開發的工具後，以下我們將以Linux系統爲例，說明如何進行程式的開發，請依照下面的說明，來完成您的第一支C語言程式 -- Hello World!

2-3 編輯原始程式

由於C語言的原始程式檔案格式爲純文字，所以你可以使用任一套text editor（文字編輯器）來編寫程式，如前一小節所述。現在，請使用你偏好的text editor來編輯一個名爲「Hello.c」的檔案，並鍵入以下內容：

Example 2-1：第一個C語言程式範例

Location:[CDROM]/Examples/ch2
Filename:Hello.c

```
1  /* This is my first C program */
2  #include <stdio.h>
3
4  int main()
5  {
6       printf("Hello World!\n");
7  }
```

2-4 編譯與執行程式

C語言的source code（原始程式碼）只是一般的text file（文字檔），而非executable file（可執行檔），如果沒有經過編譯是無法執行的。簡單來說，text file是人類看得懂的編碼方式，而executable file必須是電腦能夠理解的機器指令，通常會表達爲binary file（二進位檔）。所以我們需要有一個compiler（編譯器）來將source code轉換成電腦認識的binary file。以Linux系統爲例，我們可以使用下面的指令來進行編譯：

```
[1:18 user@ws example]$ cc Hello.c
```

注意

上述例子中，最前面的[1:18 user@ws example]$爲所謂的prompt（提示字串），其中包含現在的系統時間、使用者帳號、伺服器名稱以及目前的工作目錄，接在prompt後面的才是我們所輸入的指令，以上面這個例子來說，其所輸入的指令爲「cc Hello.c」。

若沒有任何的錯誤訊息產生，那麼就表示你已經順利地完成了編譯的動作。如果你有看到任何的錯誤訊息，那麼請再仔細檢查一下是否source code有輸入錯誤的地方，修改後請再次進行編譯，直到沒有問題爲止。

cc預設會將編譯好的程式，在目前目錄下產生一個名爲a.out的檔案。請檢查你的目錄下是否多了一個名爲a.out的檔案？請用以下指令執行這個可執行檔：

```
[1:18 user@ws example]$ ./a.out
```

如果一切順利，你應該可以看到以下的輸出的結果

```
[1:18 user@ws example]$ ./a.out
Hello World!
[1:18 user@ws example]$
```

其實cc是C's compiler的縮寫，是內建於Unix系統上的C語言編譯器，不過因爲授權問題，在Linux系統上大多是連結到另一個常見的C語言編譯器gcc，其全名爲GNU compiler collection（其中不但有C語言的編譯器，也有其他程式語言

的編譯器）。換句話說，在大部份的Linux系統上，cc與gcc其實都是相同的一個檔案 — gcc。請使用以下指令再次進行編譯與執行，你應該會得到完全一樣的結果：

```
[1:18 user@ws example]$ gcc Hello.c
[1:18 user@ws example]$ ./a.out
Hello World!
[1:18 user@ws example]$
```

你還可以使用編譯器的「-o」參數，來指定所輸出的可執行檔檔名，例如以下的例子，將Hello.c編譯成爲Hello並加以執行：

```
[1:18 user@ws example]$ gcc Hello.c -o Hello
[1:18 user@ws example]$ ./Hello
Hello World!
[1:18 user@ws example]$
```

2-5 程式碼說明

本節說明前述的Example 2-1的Hello.c的內容。

2-5-1 程式進入點

首先這種在console模式下執行的程式，大多具備以下的架構：

```
int main()
{
   程式碼
}
```

這種console模式的程式在執行時，電腦系統會將其載入到主記憶體，並從程式當中特定的位置開始一行一行、逐行地執行程式碼。這個所謂的特定位置，也就是指程式首先被執行的地方，我們稱之爲entry point（程式進入點）。事實上，絕大多數的程式語言都有類似的機制，用以啓動程式的執行，其中console模式的C語言程式，其entry point就是程式中的main function（函式）；關於function以後會詳細說明。在前面的程式碼中，第一行的「int main()」就是在

本章將透過簡單的範例程式，帶你瞭解C語言程式的輸入與輸出處理。我們將介紹一個programming model（程式設計模型）[1] — input-process-output model（IPO模型），並且說明如何配合此模型進行程式設計。另外，本章也將就簡單的變數與記憶體空間的使用，做一概念性的說明。輸入與輸出是許多程式最爲基礎的功能，在上一章當中，我們介紹了printf()函式讓你可以進行資料的輸出；在本章中我們將介紹scanf()函式以取得使用者的輸入。爲了增進你對於程式設計的興趣，我們也將在本章中介紹亂數的使用，並以簡單的程式說明相關輸入、輸出以及資料處理的應用。

在開始之前，請先編輯下面這個檔名爲「LuckyNumber.c」的程式碼，並且將它加以編譯與執行，看看會得到什麼樣的結果？

Example 3-1：印出一個Lucky Number！


Location: [CDROM]/Examples/ch3
Filename: LuckyNumber.c

```
 1  #include <stdio.h>
 2
 3  int main()
 4  {
 5     int num;
 6     int lucky;
 7     printf("Please input a number:");
 8     scanf("%d",&num);
 9     lucky = num+1;
10     printf("Your lucky number is %d!\n", lucky);
11  }
```

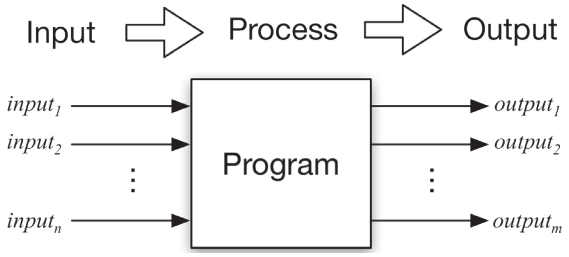
3-1 IPO模型

電腦程式的目的是爲了要解決特定的問題或滿足特定的需求，而programmer的工作就是要達成此一目的。有經驗的programmer首先會考慮的是解決問題或滿足需求的「方法」，其次才是思考如何以特定的程式語言來實作這個「方法」。但是對於初學者而言，由於欠缺設計程式的經驗，往往會陷入不知道如何構思「方法」的困境。因此，本章將介紹一個簡單的programming

1 所謂的programming model（程式設計模型）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，讓程式設計如其他工程方法一樣，具有一定的進行步驟、規範與方法。

model（程式設計模型） — input-process-output model（IPO模型，輸入-處理-輸出模型）[1][2]，希望能幫助初學者迅速擁有程式設計的基本能力。雖然IPO模型並不能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，只要熟悉這個方法，很多簡單的程式都可以迎刃而解。

所謂的IPO模型是把一個程式的運作分成三個階段：input（輸入）、process（處理）與output（輸出），如圖3-1所示：

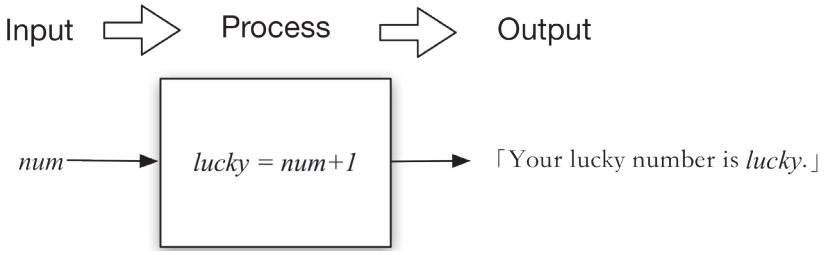


▶ 圖3-1：IPO模型概念圖

各階段說明如下：

- Input階段：取得使用者輸入的1至n筆資料（有些程式在執行時，並不需要使用者輸入的資料，因此也可以略過此階段）。
- Process階段：依據需求進行資料的處理。
- Output階段：依據處理的結果，輸出1至m筆資料[2]。

你應該已經完成Example 3-1的練習了吧！Example 3-1的程式在執行時，會要求使用者輸入一個整數，然後將它加一後，輸出爲使用者的幸運數字。如果你還沒有完成這個練習，建議你先停下閱讀，先去把程式編輯完成並且進行編譯與執行，然後好好地觀察一下它的執行結果，再繼續閱讀本章後續的內容。依據Example 3-1的執行結果，我們可以IPO模型將它表達如圖3-2所示：



▶ 圖3-2：Example 3-1的LuckyNumber.c所對應之IPO模型

2　當raw data（原始資料）經處理後具有特定意義時，亦被稱為information（資訊）。

換個方式來說，使用IPO模型來構思程式時，LuckyNumber.c的各個階段可以設計如下（爲了方便起見，我們使用「I」、「P」與「O」分別代表input、process與output階段）：

- I ：取得使用者輸入的一個整數num；
- P：進行lucky=num+1的運算，也就是將使用者所輸入的整數加一後，作爲lucky的值；
- O：輸出「Your lucky number is lucky」，其中lucky是上一階段所計算出的結果。

接下來，我們分別就input、process以及output階段分別加以說明。

3-1-1 Input階段

在input的部份，我們所要進行的是取得使用者輸入的資料(在本例中，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，其函式原型如表3-1所示：

▶ 表3-1：scanf() 的prototype

Prototype（原型）	scanf(format , memory_address)	
用途	依指定格式取回使用者的輸入。	
Header File（標頭檔）	stdio.h	
Parameters（參數）	名稱	說明
	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

注意

C語言提供了許多function（函式）供我們使用，所謂的function prototype（函式原型）指的是function的名稱以及其輸入與輸出的描述。在篇幅允許的情況下，我們將爲你整理function所需要載入的header file（函式標頭檔）以及其輸入與輸出的部份。對於function而言，其輸入的部份我們稱之爲parameter（參數）；輸出的部份則稱爲return value（傳回值）。要特別注意的是，爲了幫助程式設計的初學者，在此我們所提供的scanf()的prototype尚未符合C語言的語法要求，甚至也忽略了其傳回值的部份，我們只是先以比較簡單的方式讓讀者們理解。本書後續將視學習進程，逐步以符合C語言語法要求的方式來表達各個函式的原型。

從表3-1中可得知scanf()對應的標頭檔是stdio.h，要記得使用#include 來加以載入，如LuckyNumber.c的第1行：

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

至於在parameter（參數）的部份則有format與memory_address，其中format是用以指定所欲取得的資料之格式，memory_address則是指定要使用哪個記憶體位置來保存使用者所輸入的資料。其實scanf()的函式名稱就是取自於scan with format的縮寫，也就是「依格式來取得使用者輸入的資料」之意。因此，其format參數就是用以指定格式。該格式是以一個string（字串）來指定，稱之爲format string（格式字串）。以LuckyNumber.c第8行爲例：

```
scanf("%d",&num);
```

其format string的內容爲"%d"，表示所要取得的資料爲一個decimal number（意即10進制的整數）。因此，「scanf("%d", &num);」當中的format string，就表示要取回一個10進制的整數。我們將%d稱爲format specifier（格式指定子）。當然在C語言中除了%d外，還有很多format specifier可以使用，本書後續將會提供更完整的說明。

注意

在C語言中，一個string（字串）的內容，不論是printf()要輸出的string或是scanf()要指定格式的format string必須使用雙引號「" "」將其包裹，例如第2章中的hello.c所使用到的「printf("Hello World!\n");」，或是LuckyNumber.c中的「scanf("%d",&num);」的format string，都必須使用雙引號標示其字串內容。

至於程式在執行時所取回的輸入（由使用者從鍵盤輸入），又該如何保存起來呢？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，就是指定所欲用來存放資料的記憶體位置。但是記憶體空間是由作業系統負責管理，每當有程式要求使用記憶體空間時，作業系統的動態記憶體配置(或稱動態記憶體管理)模組就會動態地分配一塊空間給該程式使用，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，後續需要指定該記憶體位址時，只要告訴電腦要使用之前所分配給我們的那一個位址即可。

Precedence（優先順序）	Operator（運算子）	Unary/Binary（一元/二元運算）	Associativity（關聯性）
4	+	binary	left associative
	-	binary	
5	=	binary	right associative
	*=	binary	
	/=	binary	
	%=	binary	
	+=	binary	
	-=	binary	

註：precedence數值愈小者優先順序愈高

5-9 IPO程式設計實務演練

程式演練 3

台灣坪轉換為公制單位

坪（ping）爲台灣慣用的面積單位，1坪的面積等於3.3058平方公尺，而1平方公尺等於0.3025坪。請使用C語言設計一個工具程式，讓使用者輸入坪（ping）的數量，然後將其轉換爲平方公尺（m2）後輸出。

延續慣例，我們先爲這個程式命名，由於此程式是提供坪與平方公尺的轉換，因此筆者建議可以命名爲「Ping2SquareMeter.c」若是「ping2mm.c」。這個程式仍然可以使用IPO模型進行分析：

- I ：要求使用者輸入坪的數量；
- P ：計算出對應的平方公尺面積；
- O：將計算後的結果輸出。

其實這個程式，對你而言應該並不困難，我們可以很快地將更細部的IPO分析建立起來：

- H：stdio.h
- V：float ping, mm;
- I ：要求使用者輸入坪的數量，並儲存到變數ping；
  - Prompt string:「How many ping?」
  - scanf(" %f", &ping);
- P ：計算出對應的平方公尺面積；
  - mm=ping*3.3058;
- O：將計算後的結果輸出。
  - printf("%f ping = %f mm \n", ping,mm);

這個程式並不困難，你應該已經有能力可以自行開發完成。對於這一類型的程式來說，其程式架構非常單純，可以使用IPO模型完成其開發，其中比較需要思考的應該是程式所需要的變數及其型態爲何，這會對整個程式有很大的影響。試想，如果這個題目的ping與mm變數都被宣告爲int的話，整個程式不就無法產生正確的結果了嗎？需要完整程式參考的讀者，可以在本書隨附光碟中的「Projects/3」目錄中找到。

程式演練 4

Trappist-1星球公民證號驗證

Trappist-1星球（唸做特拉普一號星球）位於水瓶座附近，距離地球約40光年，是在星際大探險時期最早展開人類移民的星球之一。Trappist-1星球以含有豐富的K-286氦礦產聞名於世，目前約有20萬人口，其中5萬人爲東龍貿易公司的員工與星際艦隊成員，享有一級星際公民身份；其餘15萬人絕大多數從事K-286氦礦開採工作，如果能持續開採工作滿10年就可以取得二級星際公民身份。一級與二級星際公民，將可享有免費醫療及教育費用的減免。

目前星際醫療站位於Trappist-1星球首府的Kum Khum市的醫療飛船上的機器人需要你的幫忙，由於機器人身上負責驗證公民證號的程式故障，請你使用C語言設計一個程式並安裝在機器人身上，以幫助他們完成公民證號的驗證碼之計算（計算完成後印出驗證碼）。

星際公民證號一共有9位數，全部由數字組成，由左至右編號，最左邊爲1號位置，最右邊爲9號位置，其驗證碼的計算方式如下：

(1)前八碼中，單數位置的數字的值加總。
(2)前八碼中，雙數位置的數字的值加總。
(3)將第(1)項的結果乘以5，並與第(2)項的結果加總。
(4)將第(3)項的結果減7後取個位數，即爲驗證碼。

請將你所計算出來的驗證碼印出，機器人將會把這個驗證碼再交給其他的程式負責判斷該驗證碼所代表的意義（0-4代表非公民，5-7代表二級公民，8-9代表一級公民）。爲了Trappist-1星球與銀河系和平，更爲了人類的未來，這個程式你可得好好地寫出來！

在檔案名稱方面，建議你可以使用「GenerateVerificationCode.c」或是簡短一點的「Verify.c」。這個程式仍然可以使用IPO模型進行分析：

- I ：要求使用者輸入9位數的公民證號；
- P ：計算出對應的驗證碼；
- O：將驗證碼輸出。

目前來說，這個程式其實有一點難度，主要的困難應該是在於該如何取得一個9位數的數字？如果使用int型態的方式取得這個數字，後續又該如何將一個int切割成9個獨立的數字（依各個位數切割）？以便能進行驗證碼的計算。聰明的讀者可能已經有人想到了下面的做法：

```
int citizenID;  // 代表銀河公民證號的變數
int d1, d2, d3, d4, d5, d6, d7, d8, d9; // d1 ~ d9代表第1到第9位數
scanf("%d", &citizenID);  // 讓使用者輸入的銀河公民證號
d1=citizenID/100000000;  // 把citizenID除以100000000得到第1位數
// 將citizenID的值扣除掉第1位數後再除以10000000，以得到第2位數
d2=(citizenID = (citizenID-d1*100000000))/10000000;
// 將citizenID的值扣除掉第2位數後再除以10000000，以得到第3位數
d3=(citizenID = (citizenID-d2*10000000))/1000000;
d4=(citizenID = (citizenID-d3*1000000))/100000; // 其餘依此類推
d5=(citizenID = (citizenID-d4*100000))/10000;
d6=(citizenID = (citizenID-d5*10000))/1000;
d7=(citizenID = (citizenID-d6*1000))/100;
d8=(citizenID = (citizenID-d7*100))/10;
d9=(citizenID = (citizenID-d8*10));
```

如果你看不懂這個方法，可以將它寫成一個程式，一步一步測試看看其執行結果，相信可以幫助你了解這個程式（不是筆者不願意幫你解釋，而是覺得你應該已經有能力看懂這個程式）。筆者再提供另一個方法給你參考：

```
char d1, d2, d3, d4, d5, d6, d7, d8, d9;
scanf("%c%c%c%c%c%c%c%c%c",
       &d1, &d2, &d3, &d4, &d5, &d6, &d7, &d8, &d9;
```

```
d1-=48; d2-=48; d3-=48; d4-=48;
d5-=48; d6-=48; d7-=48; d8-=47; d9-=48;
```

這個方法改用9個字元來讀取使用者的輸入（使用9個字元的好處之一是可以確保使用者的確地輸入了9個數字），接著我們讓每個字元減去48（因爲48是阿拉伯數字0的ASCII碼，讓每個輸入的資料都減掉48的目的，就是讓d1到d9的變數能夠等於使用者所輸入的數字「值」。於是，後面就可以容易地進行處理。我們以第二種方式，來繼續我們的IPO分析：

(1)前八碼中，單數位置的數字的值加總。
(2)前八碼中，雙數位置的數字的值加總。
(3)將第(1)項的結果乘以5，並與第(2)項的結果加總。
(4)將第(3)項的結果減7後取個位數，即爲驗證碼。

- H：stdio.h
- V：
  - char d1, d2, d3, d4, d5, d6, d7, d8, d9;
  - int vcode;
- I：要求使用者輸入9位數的公民證號；
  - scanf("%c%c%c%c%c%c%c%c%c", &d1, &d2, &d3, &d4, &d5, &d6, &d7, &d8, &d9);
- P：計算出對應的驗證碼；
  - vcode=(((d1+d3+d5+d7)*5+(d2+d4+d6+d8))-7)%10
- O：將驗證碼輸出。
  - printf("%d\n", vcode);

後續的程式實作就留給你作爲練習。如果需要完整的程式，可以參考本書隨附光碟中的「Projects/4」目錄中找到。

註：本章5-7頁所討論的程式碼「printf("x / y = %d\n", x/y);」問題，其答案是將程式碼更正如下：

```
printf("x / y = %f\n", x/(double)y );
```

如何？與你的答案是否一致？