# 序

本書「C語言程式設計入門與實務演練」自2016年10月發行至今，已獲得多所大學、科技大學與技術學院的資訊相關學系做爲入門的程式語言教材；同時也受到許多讀者錯愛，做爲其自學之用。新版（第二版）全書共有17個章節，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，更可以做爲未來學習其他程式語言的重要基礎。此外，新版也特別爲慣用Windows作業系統的讀者，介紹並示範如何使用著名的Dev-C++來進行C語言的程式設計。

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124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323 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20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 2018年四月

# 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、…、20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ch10、ch12與ch13這三個子資料當中，取得所需要的資料、測試或程式檔案。

對於慣用Windows系統的讀者，筆者建議您先使用Dev-C++做爲主要的開發工具，待日後您累積了一定程度的開發經驗後，再轉移使用其他開發工具。除了特別說明以外，本書絕大部份的範例程式都可以在Windows及Linux與Mac OS等不同系統上使用，讀者只要在Dev-C++的選單中，使用「檔案 | 開啓舊檔」，並選擇光碟中適當的目錄與檔案，即可在Dev-C++中練習本書所有的範例程式與實務演練。

# 目錄

# 程式演練目錄

Chapter

# 02 您的第一個C語言程式

本章內容
- ▶2-1 程式設計流程
- ▶2-2 開始前的準備
- ▶2-3 在Linux/Mac OS系統中開發程式
- ▶2-4 在Windows系統中開發程式
- ▶2-5 程式碼說明

大部份的程式語言書籍都會以一個「Hello World!」的範例，做爲學習的第一個程式範例。「Hello World!」是一個console[1]模式的程式，其執行結果會在console輸出一個「Hello World!」字串。相較於現代的視窗應用程式，console模式的程式必須以文字指令進行操作，其執行結果也較爲單調乏味；但開發console模式的程式，不需要瞭解視窗應用程式背後複雜的機制與開發環境，有助於讓我們先專注學習程式設計的邏輯與概念，對於程式設計的初學者來說是較爲適合的。本章將透過C語言的「Hello World!」程式，分別在2-3節2-4節針對Windows與Linux/Mac OS系統的使用者，帶領您瞭解基本的程式開發流程。具體來說，我們將在2-3節說明如何在Linux/Mac OS系統上，使用終端機來進行C語言程式的開發；並在2-4節介紹在Windows平台上著名的免費開發工具 — Dev-C++。

## 2-1 程式設計流程

使用C語言進行程式設計（或者稱爲「寫程式」[2]）的流程十分簡單，可概分爲撰寫source code（原始程式碼）、compile（編譯）與run/execution（執行）。請參考圖2-1，其步驟包含：

(1) 以text editor（文字編輯器）撰寫source code（原始程式）；
(2) 完成後，產生副檔名 .c 的source file（原始程式檔）；
(3) 將source file（原始程式檔）交由compiler（編譯器）進行編譯；
(4) 編譯成功後，會產生object code（目的碼）；
(5) 若是編譯時發生錯誤，則重新回到步驟(1)進行debug（除錯）；

1 早期的作業系統所有的操作都必須在console中完成。所謂的console指的是作業系統中一個用以操作的管道，它只支援標準輸入與輸出，也就是只接受來自鍵盤的輸入以及回應在螢幕上的文字輸出。雖然現代的作業系統提供了較具親和力且容易操作的視窗界面，但多半仍保留有console（也就是在現代作業系統中所謂的「終端機」或是「主控台」應用程式）供我們使用，例如在Microsoft Windows系統中的「命令列提示字元」，或是在Linux/Unix/MacOS中的「terminal（終端機）」都是這一類型的操作環境。目前仍有許多的應用程式(包含許多伺服器軟體)必須在console中執行，我們將其稱為「console模式」的程式。

2 我們可以程式碼的英文(也就是code)之動名詞型式，來表達撰寫程式碼的過程，所以此步驟又常被稱為coding，或是更為口語化的「寫程式」與「寫code」等。考慮到撰寫程式碼的目的是產生可以執行的程式(program)，所以基於類似的理由，我們也常將程式碼的撰寫稱為programming，並把以此為業的人稱為programmer，因此「程式設計」(programming) 與「程式設計師」(programmer)等名詞就因應而生。至於在大陸則常以「編程」或「撰碼」稱呼coding，並以「編程員」、「程式編寫員」或「程序員」等名詞表示programmer。

(6) C語言的compiler會自動起動linker（聯結程式），將object code與library（函式庫）結合以產生符合作業系統要求的executable file（可執行檔），例如Windows平台下的EXE檔，或是Unix/Linux系統中的ELF檔[3]；

(7) 若在聯結時發生錯誤，一樣會回到步驟(1)進行debug；

(8) 聯結成功則會產生executable file（可執行檔）；

(9) 最後就可在作業平台上執行程式。

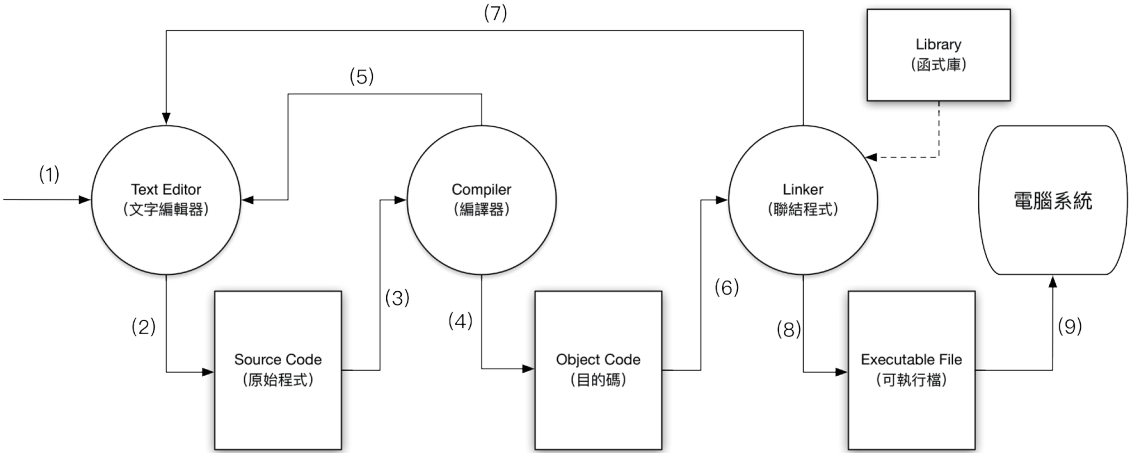


圖2-1：C語言程式的設計流程

## 2-2 開始前的準備

本節主要將爲你說明在開始進行C語言程式設計前，應準備的各種軟體工具，包含text editor（文字編輯器）以及C語言的compiler（編譯器）。我們可使用任何一套text editor（文字編輯器）來進行程式的撰寫，在這方面並沒有任何限制，你可以自行使用你所偏好的text editor軟體，例如Atom[4]、Sublime Text[5]、Microsoft Visual Studio Code[6]或是Notepad++[7]。

至於在compiler（編譯器）方面，C語言在絕大多數的作業平台上都可以找到適用的compiler，以Microsoft Windows作業系統爲例，因爲並沒有預設安裝有

3 ELF為Executable and Linkable Format，是Unix/Linux系統目前的標準可執行檔格式。
4 可參考https://atom.io。
5 可參考https://www.sublimetext.com。
6 可參考https://www.visualstudio.com/products/code-vs。
7 可參考https://notepad-plus-plus.org。

C語言的compiler，所以你可選擇安裝如MinGW[8]、Cygwin[9]或者安裝NetBeans、Eclipse、Microsoft Visual Studio或Dev-C++等IDE工具，來使用其中的C語言compiler。至於在Linux系統上，你就可以直接使用其預設已安裝好的C語言compiler — 「cc」或是「gcc」，其中cc是源自於Unix系統的C's compiler，gcc則是由GNU所提供的GNU compiler collection（GNU的編譯器工具集）。由於授權的問題，在大多數的Linux系統中已不再提供cc，而是以gcc代替；但爲了便利起見，仍然提供一個名爲cc的連結，當你在Linux系統上使用cc時，其實系統會以連結的方式幫你執行gcc。至於在Mac OS上，則可以安裝Xcode來取得C語言的編譯工具，其中也包含了cc與gcc的編譯指令；當然它們也是以連結的方式對應到Xcode開發環境所提供的C語言編譯器。

因此，本書在此給不同平台的學習者，提供以下的建議：

- **Linux平台**：如果你是Linux（或是其他Unix-like的平台，例如freeBSD、illumos/Solaris或是Darwin/OS X等）的使用者，那麼你大概不必另行安裝相關的C語言開發與編譯的工具。你可以直接在terminal（終端機）裡以console模式的vi、vim、emacs、pico或是joe等text editor編寫程式，並使用cc或gcc進行編譯即可。這是筆者最推薦的開發環境之一。我們將在2-3節，針對如何在Linux系統上開發C語言的程式進行說明。

- **Mac OS平台**：Mac OS其實也可以算是Unix-like的平台之一，它也有提供terminal（終端機）軟體，讓我們可以進行console模式的各項操作。但其預設並未提供C語言的編譯工具，你必須先安裝免費的Xcode開發工具才能取得。有了Xcode所提供的cc與gcc後，你就可以如同前述的Linux環境一樣，進行C語言的程式設計與編譯等工作。我們將在後續2-3節介紹如何在Mac OS系統上開發C語言的程式進行說明。

- **Microsoft Windows平台**：如果在Windows系統上安裝Cygwin或MinGW，就可以在Winows系統中得到類似Unix/Linux的操作環境，並參考2-3節的說明，學習如何使用terminal（終端機）來進行程式的開發。除此之外，

8 MinGW是Minimalist GNU for Windows的縮寫，顧名思義是一套在Windows環境中的GNU工具集，其中也包含了C語言的compiler與相關工具，可參考http://www.mingw.org。

9 Cygwin其實是一套在Windows環境中提供Unix-like的操作環境，其中當然也包含了GNU的C語言compiler，可參考http://www.cygwin.com。

我們也將在2-4節介紹一套在Windows系統中著名的開發工具Dev-C++，讓慣用Windows的使用者可以更爲容易地進行C語言程式設計的學習。請參考2-4節的說明，下載並安裝Dev-C++。

以下我們在2-3節先以Linux/Mac OS系統爲例，說明如何進行程式的開發；並在後續的2-4節介紹如何使用在Windows系統中的Dev-C++來進行C語言程式的開發。請慣用不同系統的讀者，自行參考相關的介紹。

## 2-3 在Linux/Mac OS系統中開發程式

本節針對慣用Linux或Mac OS的讀者，說明如何進行C語言程式的開發。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。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：你的第一個C語言程式

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  }
```

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，您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]
```

本節針對Linux與Mac OS開發C語言程式的過程，提供了詳細的說明。建議讀者依照這些步驟實際操作一遍，以掌握開發C語言程式的要領。本書隨附光碟也收錄了全書所有的程式範例原始程式，你也可以直接載入這些範例，以節省自行輸入的時間。

## 2-4 在Windows系統中開發程式

本節將針對慣用Windows的使用者，介紹並使用Dev-C++來進行C語言程式的開發。Dev-C++是由Bloodshed Software公司的Colin Laplace所推出的一套C++語言的整合開發環境（Integrated Development Environment，IDE）軟體，可以在同一個環境中進行C語言原始程式的編寫、編譯與執行等開發工作。讀者可以至SourceForge上取得由Orwell所維護的最新版本（截至2018年4月，Orwell所釋出的Dev-C++最新版本爲5.11版）。有興趣的讀者可以在https://sourceforge.net/projects/orwelldevcpp/files/latest/download，下載到最新版本的Dev-C++並以滑鼠雙擊後，起動其安裝程序。當你完成Dev-C++的安裝後，接下來請依照以下的說明，使用Dev-C++來完成Example 2-1的程式碼編寫、編譯與執行等動作：

1. 起動Dev-C++後，在選單中選取「檔案 | 開新檔案 | 原始碼」，建立一個新的原始程式。你也可以在Dev-C++的工具列上，直接以滑鼠點擊圖示，或者是使用「Ctrl+N」快速鍵，來建立一個新的原始程式。完成上述動作後，你將會在畫面中央得到一個新增的「新文件」，如圖2-2所示，請在此處將Example 2-1的程式碼加以輸入。爲了便利讀者輸入程式，我們在此將Example 2-1再列示如下：

### Example 2-1：你的第一個C語言程式

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  }
```

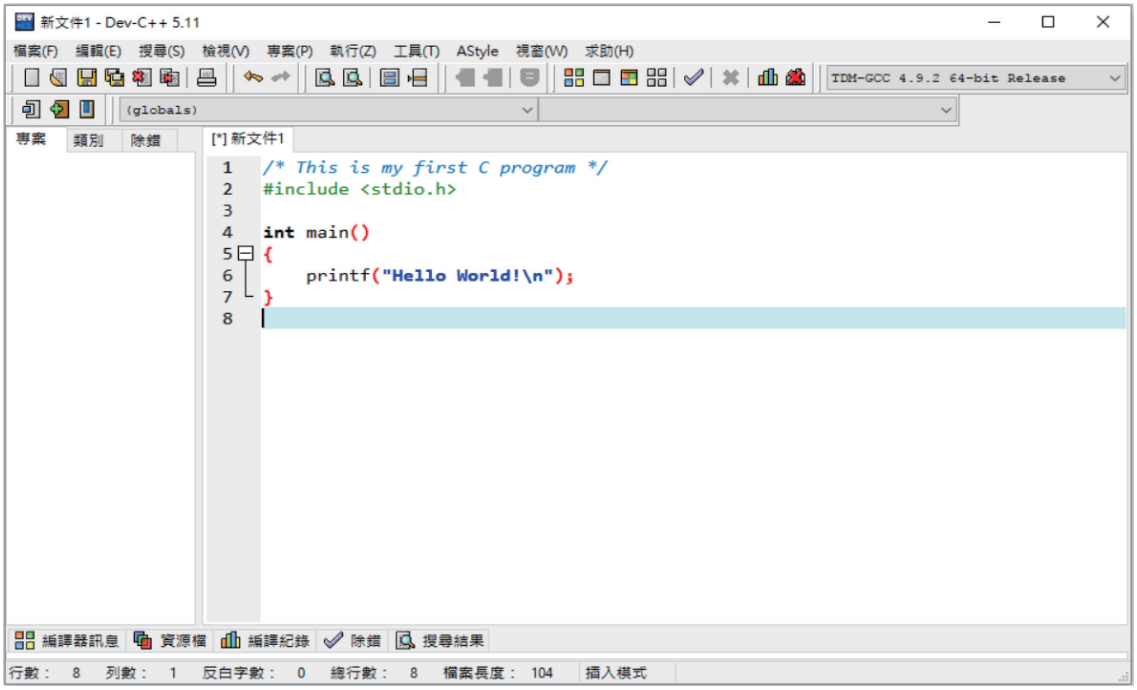


圖2-2：在Dev-C++裡編寫原始程式碼。

2. 當您將Example 2-1的原始程式碼在Dev-C++裡編寫完成後，就可以使用編譯器將其轉換爲可執行檔，請在選單中選取「執行 | 編譯」，或是使用「F9」快速鍵進行編譯。要注意的是，當你第一次進行編譯前，Dev-C++會先要求你將原始程式碼存檔後，才能進行編譯。請建立或指定適當的目錄來存放你的程式，不過要特別注意的是Dev-C++預設會將檔案儲存爲C++語言的原始程式，而不是C語言的原始程式。請參考圖2-3，爲你的原始程式命名爲「hello.c」，並且選擇存檔類型爲「C source files (*.c)」，完成後請按下「存檔」按鈕。後續，Dev-C++就會開始進行鍵進行編譯。

3. 原始程式碼的編譯結果，將會顯示在程式碼下方的「編譯記錄」區域，如圖2-4所示。一旦完成程式碼的編譯後，你就可以得到一個檔名爲「hello.exe」的可執行檔。

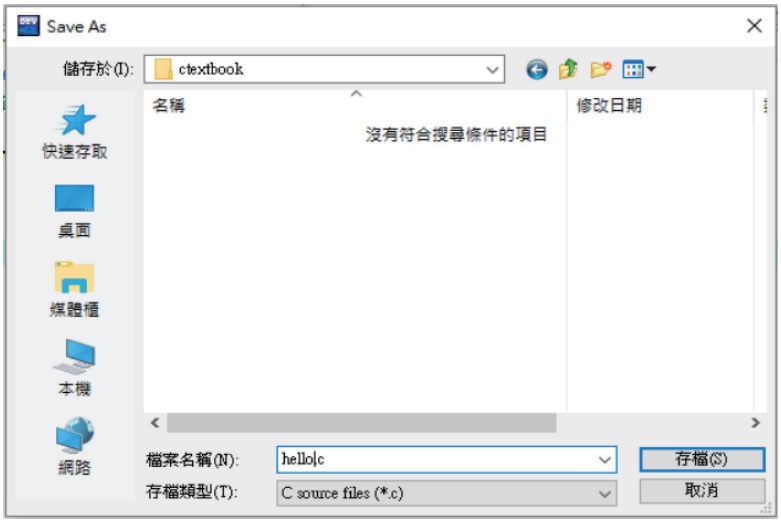


圖2-3：在編譯前先進行原始程式的存檔。

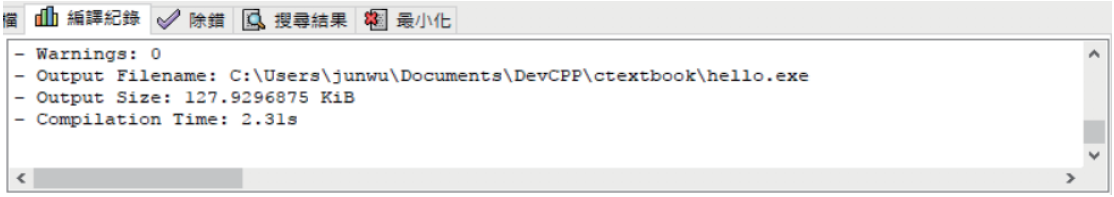


圖2-4：編譯結果顯示於程式碼編輯區下方的「編譯記錄」區。

4. 接下來請在選單中選取「執行 | 執行」或是使用「F10」快速鍵，來執行這個已編譯完成的hello.exe可執行檔。一旦我們對Dev-C++下達了執行的命令後，Dev-C++會起動一個「命令提示字元」的應用程式（也就是Microsoft Windows系統的Console環境），並在此程式中執行我們所編譯完成的hello.exe可執行檔，其執行畫面如圖2-5所示。您可以看到此程式輸出了「Hello World!」字串，並在按下任意鍵後結束。

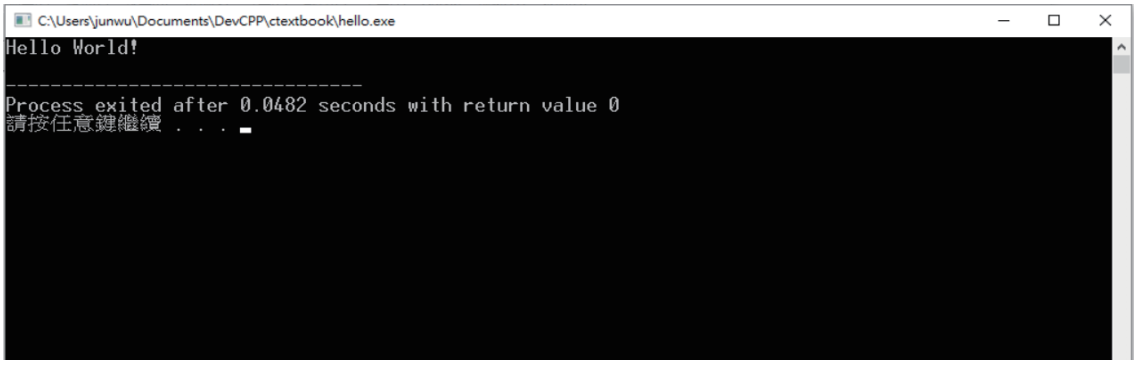


圖2-5：Dev-C++使用命令提示字元來顯示執行結果。

我們已經使用Dev-C++示範了Example 2-1的hello.c程式的開發與執行過程，建議讀者依照前述的步驟實際進行一遍，以掌握Dev-C++的使用方式。本書後續的程式範例，也都可以使用Dev-C++來進行開發，在本書隨附的光碟中，也將提供所有程式範例的原始程式，您可以使用Dev-C++來載入這些程式[10]，並加以測試其執行結果。關於本書隨附光碟的詳細使用說明，請參考「光碟內容說明」。

## 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起動程式的執行，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()」就是在定義這個entry point，其後緊接著一對大括號{}，電腦會從左大括號開始，一行一行地加以執行，直到遇到右大括號爲止。因此，一個簡單的C語言程式，就是將欲由電腦執行的功能，以符合C語言語法規則的方式寫成程式碼，並且是要依序寫在int main()的大括號內。前面所介紹到的Example 2-1（也就是hello.c程式），就是一個console模式下的程式，在後續的小節裡，我們將繼續說明hello.c這個程式的細節。

10 使用Dev-C++選單中的「檔案 | 開啓舊檔」，就可以載入這些程式範例的原始程式。

## 2-5-2 註解

在Example 2-1 hello.c的開頭處，有以下這一行：

```
/* This is my first C program */
```

其實這一行程式碼對於程式的執行沒有任何作用，只是所謂的comment（註解）。在程式語言中，凡是註解的部份，編譯器在進行編譯時都會略過此一部份，因此註解並不會對程式的執行造成任何的影響，其目的只是爲您所撰寫的程式做一些說明。通常註解的內容不外乎版本、版權宣告、作者資訊、撰寫日期及程式碼的說明等。在C語言中註解使用的方式有以下兩種方式:

(1) 單行註解：以「//」開頭到行尾的部份，皆視爲註解。這種單行註解，可以放在每一行的開頭，或是在行中其他位置。例如：

```
// 這是C的註解
int i; // 宣告變數i
// int maybe wrong here;
```

(2) 多行註解：以「/*」開頭到「*/」結尾的部份，皆視爲註解。這種方式可將多行的內容都視爲註解。例如：

```
/* 這也是C的註解,可用於單行的註解 */
/* 這種方式也可以應用在多行的註解
   對於需要較多說明的時候
   可以使用這種方式*/
```

特別要說明的是，註解有時不一定只是做程式碼的說明，還可以暫時把可能有問題的程式碼加以註解，以便進行程式的偵錯，這在程式設計上是常用的一種除錯的方式。

## 2-5-3 印出字串

最後要說明的是在hello.c的main function（函式）內，唯一的一行程式碼：

```
printf("Hello World!\n");
```

像這樣的一行程式碼，我們稱之爲statement（敘述），也就是要告訴電腦要幫我們執行的工作爲何。在main function中，可以有一行以上的statement，

每一行都必須以分號「;」結尾。「println("Hello World!\n");」是一個標準的statement，其目的爲輸出"Hello World!字串到螢幕上。

其實printf也是一個function，請思考以下的例子：

```
f(x)=2x+5
```

明顯地，這是一個數學的函數，唸做「f of x」，其函數名稱爲「f」，並且有一個輸入的parameter（參數）名爲「x」，它會計算「2x+5」的值並加以傳回，例如f(3)的值爲11或者f(5)的值爲15。對於C語言來說，printf也是一個function（函式）[11]，我們可以把它思考爲以下的型式：

```
printf(s)=將s輸出到螢幕上
```

換句話說，printf函式的作用是在取得輸入的string（字串）s後，將其輸出到螢幕上。這裡所謂的string是指character（字元）的組合，並由雙引號標註。例如：

「"Hello World\n"」是正確的字串

「Hello World\n」缺乏雙引號標註，並不是正確的字串

「"Hello World\n」缺乏右方的雙引號標註，所以不是正確的字串

可是在「"Hello World!\n"」當中的「\n」又是什麼呢? 其實很簡單，請你自己修改一下hello.c的程式，將其中的「\n」拿掉，再編譯與執行看看有何差異，你不就可以得到答案了嗎？

## 2-5-4 函式標頭檔

在C語言中，已預先定義好許多有用的function（函式），我們可以視需要在程式中選擇使用這些function，來完成特定的工作。這些function依功能或性質，被分類存放在不同的library（函式庫）中。C語言也提供許多function header file（函式標頭檔），其副檔名爲.h，用以定義性質或功能相關的function。在程式內使用特定的function前，你必須明確地告訴C語言的編譯器，將含有該function定義的header file載入，這個動作就是透過「#include」這個preprocessor

11 本書為了區別起見，將數學上的function譯做「函數」，將程式語言中的function譯做「函式」。

directive（前置處理器命令）[12]，來將特定的function header file加以載入。在Example 2-1 hello.c中的：

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

就是要求載入「stdio.h」檔案，其中定義了與標準輸入輸出(standard input/output)相關的函式定義。例如printf()函式就包含在stdio.h的定義當中，所以我們必須先將stdio.h載入，後續compiler才能正確地處理printf()函式。

## 2-5-5 敘述

經過上述的說明，相信你對於基本的C語言程式設計已經有了初步的概念。我們最後利用本節定義一下C語言的程式架構：

- 每個C語言程式都必須要有main()[13]，在main()函式中，以一組大括號「{ }」把要執行的程式碼包裹起來。
- 在main()中的程式碼，又被稱爲statement（敘述），每行敘述必須以分號「;」做爲結尾。例如在hello.c程式中的main()函式，其中就包含了一行「printf("Hello World!\n");」的statement。
- Statement依其功用，在C語言中又可在區分爲以下幾種類別（本書後續將分別加以介紹）：
  - Expression statement（運算敘述），進行算術與邏輯等各式運算的敘述（包含函式呼叫亦屬於此類），詳如第5章。
  - Selection statement（選擇敘述），又稱爲conditional statement（條件敘述），依特定條件改變程式執行動線的敘述，詳如第7章。
  - Iteration statement（重複敘述，或稱迴圈敘述），使得特定程式碼可以反覆執行的敘述，詳如第8章；
  - Compound statement（複合敘述），可搭配selection statement、iteration statement等，以一組大括號將多個statement組合在一起，讓原本的單一敘述改以一個擁有多個敘述的複合敘述取代。此部份將於本書第7、8等章節中介紹。

12 所謂的preprocessor（前置處理器）是一個電腦程式，在compiler進行編譯前執行。所有以#開頭的程式碼，都會被視為交付給preprocessor的directive（命令）。

13 當然，日後我們會設計一些不具備main()的程式，但它們僅做為函式庫或其他用途，不能被加以執行。

- Jump statement（跳躍敘述），用以改變程式執行的動線的敘述，將於本書第7、8等章節中介紹。
- Labeled statement（標籤敘述）：配合jump statement所使用的標籤，將於本書第8章中介紹。

# 本章習題

## 簡答題

1. 請說明C語言的程式開發流程。
2. 請說明在Linux系統上的C語言編譯器為何？
3. 請說明Dev-C++可以幫助開發人員進行哪些操作？
4. 何謂程式的entry point？C語言程式的entry point是什麼？
5. C語言提供兩種進行註解的方式，請分別加以說明。
6. 註解是程式中不會被執行的部份。請說明為何需要在程式中寫註解？
7. 何謂function header？為何要載入它？又要如何載入？

## 程式練習題

1. 設計一個C語言的程式用以印出「This is my very first C program!」，並請將其source code儲存於檔案First.c。如果你在編譯的過程中遇到錯誤，請將錯誤訊息列出，並說明你是如何解決這些問題。
2. 設計一個C語言的程式，輸出你的英文名字以及「I am glad to be a C programmer!」，請參考以下的執行結果（其中XXX為你的英文名字）：

```
Hi, My name is XXX.
I am glad to be a C programmer!
```

請將這個程式命名為Glad.c，並將其可執行檔命名為Glad。同樣地，如果你在編譯的過程中遇到錯誤，請將錯誤訊息列出，並說明你是如何解決這些問題。