

C語言程式 製作導論

這本C語言入門書，內容完整且易學習。它不但含90幾個程式範例，且涵蓋了C語言的最新功能，是一本針對初學者及富經驗的程式員而設計的好書。

PROGRAMMING IN

林縣城 譯

C 語言程式製作導論

林縣城 譯

這本C 語言入門書，內容完整
且易學習。它不但含90幾個程式
範例，且涵蓋了C 語言的最新功能
，是一本針對初學者及富經驗的
程式員而設計的好書。

儒林圖書公司 印行

版權所有
翻印必究

C 語言程式製作導論

譯 者：林 縣 城

出版者：格致圖書有限公司

總經理：儒林圖書有限公司

地 址：台北市重慶南路一段121號8樓23室

電 話：3118971-3 3144000

郵政劃撥：0106792-1 號

吉富印製有限公司承印

板橋市三民路二段正隆巷 46 弄 7 號

行政院新聞局局版台業字第4336號

中華民國七十九年二月增訂二版

定價新台幣 320 元正



目 錄

1 簡 介	1
2 一些基本概念	5
程式規劃.....	5
高階語言.....	6
作業系統.....	7
編譯程式.....	7
3 用ANSI C寫一個程式	13
習 題.....	19
4 變數、資料型態以及算術運算式	21
變數.....	21
資料型態和常數.....	23
算術運算式.....	31
習 題.....	39
5 程式迴圈	43

for 敘述	44
While 敘述	56
do 敘述	61
習 題	64
6 做決定	67
if 敘述	67
Switch 敘述	87
旗 標	91
條件運算子	95
習 題	97
7 陣 列	99
陣列元素初值的設定	111
字元陣列	112
多維陣列	117
習 題	119
8 函 數	121
引數和局部變數	124
傳回函數值	128
函數呼叫 函數再呼叫 函數	133
由上而下的程式規劃	140
函數和陣列	140
整體變數	154
自動變數及靜態變數	158
遞迴函數	161

習 題	165
9 結 構	167
函數與結構	173
設定結構的初值	183
由結構所構成的陣列	183
在結構中定義結構	187
含有陣列的結構	189
有關結構的其它變化	192
習 題	194
10 字 串	197
不定長度的字串	200
逸出字元	219
有關常數字串的進一步說明	222
字串、結構和陣列	223
字元運算	232
習 題	236
11 指 標	241
指標和結構	246
指標和函數	260
指標和陣列	266
指標運算	281
指向函數的指標	283
指標與記憶體位址	285
習 題	286

12位元運算	289
位元運算子	291
位元欄	308
習 題	314
13前端處理程式	317
# define 敘述	317
# include 敘述	335
條件性編譯	338
習 題	343
14資料型態的進一步討論	345
列舉式資料型態	345
typedef 敘述	350
資料型態轉換	353
習 題	358
15發展大型程式	359
個別編譯	359
模組間的通訊	362
16輸入與輸出	371
字元 I / O : getchar 和 putchar	372
格式化 I / O : printf 和 scanf	372
檔案 I / O	387
檔案處理上的特殊函數	392

習 題	404
-----------	-----

17 其他特性及較深入的主題 407

其他特性語言敘述	407
等 位	409
逗點運算子	413
變數屬性	414
命令列引數	416
動態記憶體配置	421

附 錄

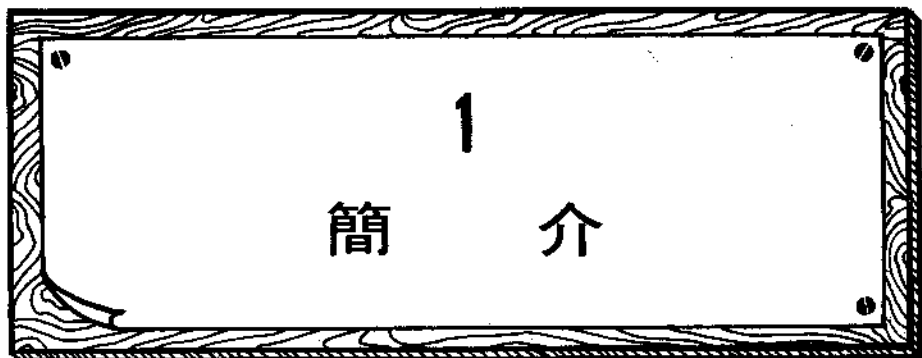
A ANSI C語言摘要 429

1.0 識別字	429
2.0 註解	430
3.0 常數	430
4.0 資料型態及宣告	434
5.0 運算式	445
6.0 儲存類別與有效範圍	463
7.0 函數	466
8.0 敘述	469
9.0 前端處理程式	475

B 標準 C 程式庫 485

標準標頭檔	485
字串函數	487
記憶體函數	489
字元函數	490

I / O函數	492
記憶體內轉換函數	499
字串對數字的轉換	500
動態記憶體配置函數	502
數學函數	503
一般性的實用函數	506
C ANSI C的主要差異處	509
1.0 識別字	509
3.0 常數	509
4.0 資料型態與宣告	511
5.0 運算式	512
7.0 函數	513
9.0 前端處理程式	514
D 常見的程式錯誤	517
E ASCII字元集	523
索 引	525



程式語言“C”是在1970年代初期，由貝爾實驗室所發展的一系列程式語言沿革而來。然而直到1970年代後期，這個程式語言才開始廣受歡迎及支持，這是因為在這之前，除了貝爾實驗室以外，尚未出現商業用途的C編譯程式。C之所以漸受歡迎，部分原因是由於UNIX作業系統同樣地受歡迎。UNIX作業系統也是在貝爾實驗室發展出來的，它用C作為標準的程式語言。事實上，UNIX作業系統本身百分之九十是用C寫成的。

隨著IBM PC及其相容電腦的風行，MS-DOS已經成為C語言最普遍的發展環境了！由於C適用於各種作業系統，因此，各家廠商都推出了他們的C語言，以期打入PC的市場。這些C語言大部份都是遵循Brian Kernighan與Dennis Ritchie所著的C語言鼻祖“The C Programming Language”（Prentice-Hall, 1978）一書中的附錄。遺憾的是，該錄附並沒有對C做一個完整且明確的定義，使得各廠商都可以依其需要來解釋C的一些特性。

1980年代初期，各種C語言的分歧已經使得C語言所定義的標準化變得迫切需要。而美國國家標準協會（American National Standards Institute, 簡稱ANSI）正是專門從事這項工作的機構，因此在1983年，ANSI成立了一個委員會來進行C語言的標準化。

有了C的ANSI定義，你便可以確信，任何一家銷售真正的ANSI C編譯程式的廠商都已經加入了ANSI標準的語言特性。這樣一來，細心的程式設計師所撰寫的C程式便可以不经修改而在具有ANSI C編譯程式的系統上執行了！

C雖是一種高階語言，但是它提供讓使用者更容易接近硬體及處理較低層次工作的能力。“C”原本是被設計來寫系統方面的程式，因此C除了是一個一般用途的結構化程式語言外，它還提供使用者更強的功能及彈性。事實上，有很多程式應用的例子，使用C語言來寫是輕而易舉的事；但是若改用其它和PASCAL、FORTRAN或BASIC等語言來處理，可能會比較困難。

本書的目的在教你如何用法C語言製作程式。在此之前，即使你從沒有接觸過C語言，那也無所謂；本書兼顧初學者和已有經驗者。假使你以前已有寫程式的經驗，你將發現C有其獨特的運作方式，跟你用過的其它程式語言可能有很大的不同。即使你已學過PASCAL語言（C表面上很像PASCAL），你也會很快地感受到C有許多獨有的特性；諸如指標（pointer）、字串和位元的處理方式。

C語言的每一個特性在本書中會一一介紹，附帶地會舉出一個小而完整的程式例子來說明每個特性；這種方式反映出本書編寫時特別強調而採用的原則——“例舉教學法”。假使你所使用的電腦系統有提供C編譯程式，那麼你最好能將本書中每個程式輸入電腦執行看看，並比較其結果是否與書中所說的一致。如此，不但你將學會C語言及其語法，同時也會對鍵入、編譯及執行C程式的過程相當熟悉。

傳授ANSI C語言所採用的方式，是先提出一個問題，想出一個用電腦來解決的方法，然後着手用C語言寫成程式。採用這種方式在適當時機引用及說明解決某個特定問題，必需使用的新的語言

架構。

你將發現本書從頭至尾都很強調程式的可讀性 (*readability*)。因為作者堅信所設計的程式，一定要讓設計者或其它的人很容易閱讀才行。根據經驗及常識判斷，你會發覺這樣的程式比較容易寫、除錯及修改；況且發展可讀性高的程式是遵守結構化程式設計原則的自然結果。

本書是為教學而編寫的，每一章的題材內容都是根據前面已討論過的章節，因此最好是按章節順序閱讀，避免跳著看。在進行下一章節時，先把每一章後面的習題練習一下。

第二章說明一些關於高階程式語言的基本術語及編譯程式的過程，以確保爾後每章所討論的語言能一致而不會混淆不清。從第三章開始循序漸進介紹 C 語言，在第十六章之前，所有 C 語言的必要特性都已涵蓋，第十六章將深入探討 C 語言中的輸入／輸出動作。最後在第十七章將介紹 C 語言中一些比較高級或奧妙的特性。

附錄 A 列有 C 語言的語法以供參考。附錄 B 摘錄了本書所介紹之 C 語言及 ANSI 版本 C 語言之間的主要差異。在附錄 C 中你會發現一些常見的程式錯誤。此外，並附有索引，以便你能很快地找到你所要知道的 C 語言中某種特性說明及程式範例。

本書並不侷限於某個提供 C 語言的電腦系統或作業系統，但是既然 C 最常在 UNIX 作業系統下執行，因此本書特別說明在 UNIX 下如何使用 C 語言。書中敘述如何在 UNIX 下編譯及執行 C 程式，另外在附錄 D 至 F 中也列出許多 UNIX 程式庫常式 *library routines*，說明 C C (C 語言編譯程式) 命令及 *lint* 程式。附錄 D 中所列的程式庫在大多數非 UNIX 系統下也能使用。

一旦你精通本書的內容之後，你可能希望能進一步閱讀 *Topics in C programming* (Kochan & Wood)。在該書中，你會見到指標與結構的詳細說明，以及如何運用各種不同的 C 程式庫。該

4 C 語言程式製作導論

書其他的一些章節會說明如何用 UNIX 系統的 `curses` 套裝軟體來撰寫與終端機獨立的程式，如何使用 `make` 功能，以及如何為 C 程式除錯。附帶一提，該書是以 UNIX 為導向的，不過非 UNIX 的使用者也會發現其中大多數的內容也非常有用。

2

一些基本概念

本章描述一些在你學習如何寫作 C 程式之前，必需了解的基本名詞；順便對使用高階層語言撰寫程式的情形做一個概說，也討論一下編譯 C 程式的過程。

程式規劃

電腦實際上是非常笨的機器，要它做什麼，它就只能做什麼。大部分的電腦系統只能執行非常基本的動作，例如把某數加上 1 或測試某數是否等於 0 等，這些基本動作中頂多和加 1 或測試是否為 0 一樣複雜而已；這些基本動作構成所謂的電腦的**指令集**（**instruction set**）。一些電腦的指令個數是有限制的，例如 DEC PDP-8 電腦只有幾十個指令；但是有些電腦，例如 DEC VAX 却有上百個指令。

爲了利用電腦來解決問題，我們必須用電腦的基本指令來表達解決問題的方法。一個電腦程式實際上是一組用來解決問題的指令，所採用的方法稱爲**演算法**（**algorithm**）。例如，假使我們想要發展程式測試某數是奇數或是偶數，那麼這一組解決此問題的**敘述**（**statement**）就是程式，而用來測試某數是奇數或是偶數的方法就是演算法了。通常要發展一個程式以解決某個問題，先要將解決

方法化成一個演算法，然後再根據演算法寫成程式。因此，上面問題的演算法可寫成：“首先將此數除以 2，若餘數是 0，則此數是偶數；否則為奇數”。有了這個演算法，我們就可使用某種電腦語言（例如 BASIC、PASCAL 或 C）來寫程式以便在電腦系統上執行。

高階語言

剛發明電腦時，撰寫程式的唯一途徑是用二進位數（binary number），直接對應機器指令和記憶體中的位址。接下來發展出組合語言（assembly language），程式師可不用二進位數來撰寫程式，而代之以較高階的組合語言來撰寫。組合語言以符號名稱（symbolic name）來代表每個機器指令和記憶體位址，而由一個稱為組譯程式（Assembler）的系統軟體來將組合語言程式中的符號碼格式轉換成機器指令。

因為組合語言的敘述和機器指令有一對一的對應關係，故被視為低階語言。程式員仍然需要學習某個電腦系統的指令集方能寫出組合語言程式，這樣的程式是不具有轉移性的（transportable）；即不能將其直接搬到另外一個電腦系統上執行，而必須重新用另一個電腦系統的組合語言撰寫，此乃由於每個電腦有其不同的指令集之故。

隨後發展出許多的高階語言，其中 FORTRAN 是第一個，程式員不必再顧慮某個電腦的指令集，而可用較複雜或高階的指令來撰寫程式。一個 FORTRAN 語言的指令或稱為敘述，隱含著許多個機器指令，而不像組合語言的指令和機器指令間是一對一的關係。

某個高階語言語法（syntax）的標準化，意指用此語言撰寫的程式應該和機器無關；換言之，此程式可在任何提供此種語言的機器上執行，其只需稍加修改或根本不用修改。

爲了要將高階語言程式轉換成某個電腦能接受的機器指令，必須有特殊的系統程式來做轉換的工作，那就是編譯程式。

作業系統

在討論編譯程式之前，我們先來介紹作業系統（也是一個系統程式）所扮演的角色。

作業系統控制電腦系統的整個運作，所有的輸入／輸出動作均經由作業系統處理。另外，作業系統必須管理系統的資源（如中央處理機、記憶體、檔案、磁碟機和印表機等等）及分配程式執行的時間及順序。

所謂作業系統就是用來控制電腦系統所有作業的一個程式。在電腦系統上所執行的所有 I/O（輸入／輸出）作業都要透過作業系統。作業系統也必須管理電腦系統的資源，同時也必須處理程式的執行。

目前最受歡迎的作業系統之一是 UNIX，是由貝爾實驗室發展的，UNIX 是一個很不尋常的作業系統，它已經在許多不同種類的電腦系統上運作。以往作業系統都是針對某種電腦而設計，而由於 UNIX 主要是用高階語言 C 寫成並不遷就於某個電腦結構，所以 UNIX 可以輕而易舉地轉移至很多不同的電腦系統上運作。

Microsoft 公司的 MS-DOS 是另一種相當流行的作業系統。這種系統主要用在 IBM PC 以及與其相容的電腦上。

編譯程式

編譯程式是一種軟體程式，在原理上它與你在本書所見到的程式並沒有什麼不同，不過它比較複雜一些。

編譯程式是一個系統程式，負責分析某種電腦語言的程式，將之轉換成適合你所使用的電腦系統執行的形式。

圖 2 - 1 說明從鍵入、編譯到執行以 C 語言撰寫程式的步驟以及所用的典型 UNIX 命令。

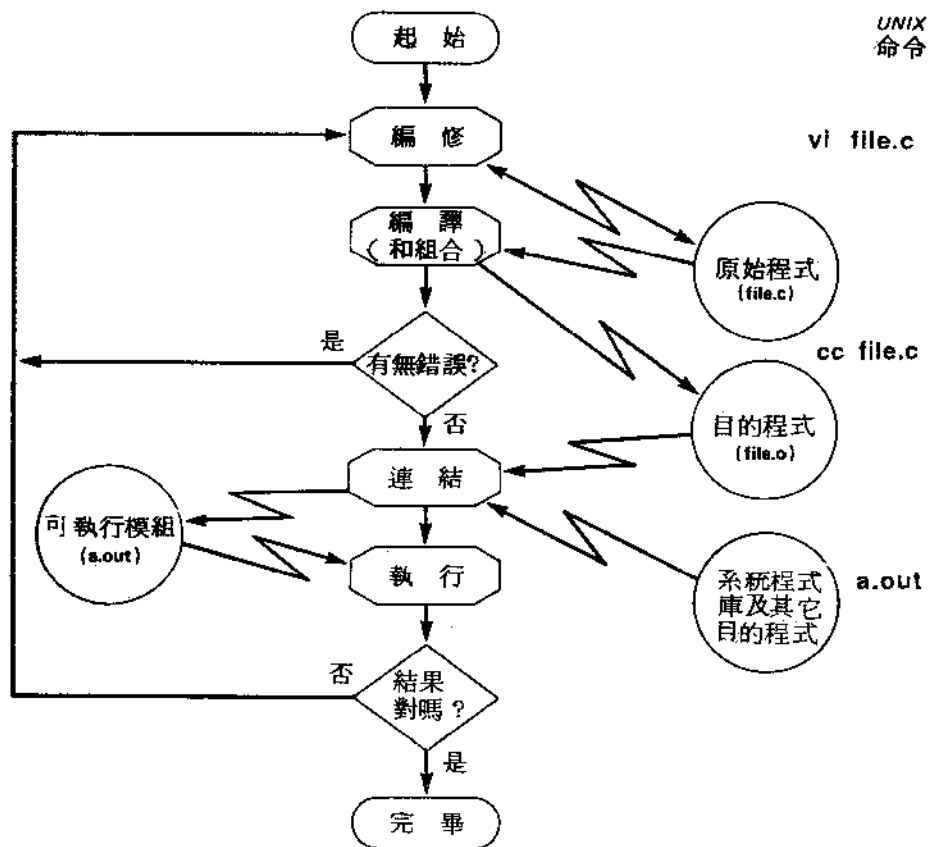


圖 2-1 鍵入、編譯及執行 C 程式的過程(顯示典型的UNIX 命令)

首先將你要編譯的程式鍵入存於檔案 (file) 中，每個電腦系統對於檔案名稱的格式有不同的規定，一般而言是取決於你。在 UNIX 之下，C 程式檔案的名稱只要最後兩個字元是“ . C ”即可