高等 Pascal

程式設計技巧

楊洪生 編譯

高等Pascal

程式設計技巧

楊洪生 編譯

五南圖書出版公司 印行

高等 Pascal 程式設計技巧

中華民國74年 7 月初版

編譯者 楊 洪 生
發行人 楊 榮 川
發行所 五南圖書出版公司
局版臺業字第0598號
臺北市銅山街1號
電 話:3916542
郵政劃撥:0106895-3
印刷所 茂榮印刷事業有限公司
臺北縣三重市重新路五段632號
電 話:9711628・9713227

售價 250 元

簡　介

本書寫作的目的，在於增強業餘或職業程式設計師的PASCAL 語言方面的知識，並用來解決問題。本書主要強調完整和可用的程式，在建立程式時注意到語言特性、演算法、和資料結構。讀者須先讀過PASCAL 語言簡介的教科書，也能看得懂PASCAL 程式表列，並且能寫簡單的PASCAL 程式。

為什麼本書書名冠上「高等」一詞呢？因為，典型的介紹 PASCAL的教科書，包括許多小程式，每一個設計指示其所用到的特殊語言元素，其程式都是簡單、無趣的。此外，小程式只是教導語言特性方面的正確使用方式，對於如何利用這些程式特性組成大程式則付之闕如。所以學生僅能把教科書當作工具，並不能從中獲得設計一個真正實用程式的概念。

本書的程式都是真正實用的，每個設計都講求實用性和趣味性，我們也希望這些程式能在讀者的電子計算機上運轉（ RUN ）。

本書的程式平均比教科書的程式長。較長的程式允許我們將一些較大的任務分成若干小的任務（ task ），每一個任務實現一相當小而且可了解的程式模組。

有興趣的讀者可任意修飾程式，適用其任意的環境、增加或刪除程式的特性、改進效率或彈性化、等等。事實上，本書的程式非常容易修訂，部分

改進建議附於各章之後，並鼓勵讀者創作及改進。

最後，希望本書可作為技巧智囊。本書的程式反覆的使用成打的模組來解決普通的程式問題。當讀者運轉他們自己寫的這類程式，就可充分利用這些常式而不必再費心思去設計了。

觀察全書

本書最好的部分是寫程式之前，先了解設計決策：為什麼選擇某程式的結構而捨另外一個？在某一固定任務中最佳演算法是什麼？我們如何決定那種方法最好，並能代表現實世界的量、和抽象的資料結構？第一章探討這類設計決策。

本書另外一個主要的理論，是設計一般目的的「工具」常式，這些常式在不同的程式中使用。雖然在寫本書的大部分程式之中發展工具常式，前面若干章集中注意發展工具。第二章設計CRT 螢幕輸出常式，在後面各章中均使用到；在示範程式 theseus 就利用它在螢幕上建立並解決迷宮的問題。

第三章的常式，是用於交作或由使用人在不同的型態自鍵盤輸入資料，並將某一資料型態轉換成另一種型態。本章的示範程式是 gaslog，它自汽油消費者處接受資料，並提供列印資料的報表。第五章是另一工具內建設計、教導讀者閱讀或設計具彈性而有效率的常式。這些工具都在 print 中使用之（排印本文檔的公用程式，輸出形式是分頁輸出）。

其他各章各自建立一個或二個有趣而且有用的程式：第四章的 calc 由使用人處接收算術公式、並輸出其值。第六章設計 reversi 是一種與電子計算機對抗的遊戲。有二個翻棋（reversi）的版本；其一是在正常本文螢幕上使用，另外一個是利用APPLE Ⅱ的圖形能力。

第七章介紹電子計算機的模擬（simulation），對於了解現實世界系統

而言，是一個強有力的工具。本章內典型代表是 bouncer ，即球在盒內反跳的模子：另一程式 isaac 是模擬行星在萬有引力下的移動。第八章致力於 Pascalc的設計，該程式是深具彈性、有力的「電子工作表報」，對於商界而言甚有價值。

PASCAL語言

PASCAL 於60 年代晚期由Niklaus Wirth 發展出來，他的目的是產生一個適於教學的程式，能夠有很清楚而系統化的概念，並可在大型電腦上使用。很顯然的他獲得相當成就。無論是大型電子計算機、小型電子計算機，甚且個人計算機都相當風行。

PASCAL 為什麼會流行？最主要的原因是PASCAL 使得工件易於寫、讀，而且修飾程式時遠較許多其他程式語言來的容易。

- PASCAL 有控制結構的敍述（ while repeat for case 和 if-then-else ）、程式非常清楚、由上到下的流程設計。如果程式語言沒有控制敍述，則經常要用 if 來測試，強迫使用 goto 敍述，使得程式難以了解。
- PASCAL 讓設計師把一大型程式打破，分成小的，各自獨立的程序（ procedure ）和函數（ function），每一個程序或函數都有一固定的任務（ task）。每一個模組都有其各自的變數，僅在該模組執行時才有用。每一模組與其呼叫常式之間的通信都靠事先定義好的輸入和輸出參數傳遞訊息。這種模組化的目的在使程式易於閱讀而且簡明。
- PASCAL 允許設計師自行定義資料型態（ data type ）和資料結構（ data structure ），充分使用這些特性，使程式更具相容性與易於瞭解。

- PASCAL 允許使用長的變數識別符號（ identifiers ）、程序、和函數。（標準PASCAL 根據前八個字元區分識別符號。）這使得設計師可用助憶符號命名，也有助於了解程式。
- PASCAL 提供遞迴（ recursive ）程序和函數，換句話說該程序和函數可以自行呼叫本身。這種常式有時為誤用，然而其往往較無遞迴常式的程式易於了解、更簡明。

儘管 PASCAL 是一很好的語言，但不完美。沒有一種廣泛使用的語言是完美的。 PASCAL 的檔案能力是最受批評的設計。另外一種說法是，PASCAL 的陣列（array）、大小（ size ）為其型態（ type ）的一部分，不容易處理任意長的陣列。（新 ISO 標準 PASCAL 已彌補這個問題，但這新版本的 PASCAL 尚未廣泛的有效使用。）此外，PASCAL 申明一變數是當局變數（ local variable ），當控制權離開該常式時，即被遺忘；所以為了保留該值，必須在呼叫它的常式宣告一全盤（ global ）變數，在必要時可以接達該變數。這些是不完美的缺點，餘容後敘。

無論是好、是壞，程式設計師必須使用一些語言。PASCAL 仍然是一易於發展，而且流行甚廣的語言。

易傳性的特性

一個易傳性的程式，是稍加修改或根本不必改，即能在許多電子計算機上運轉。易傳性的優點很多，後面將費點篇幅討論之。大家都了解， PASCAL 並未明確的定義、教導程式設計師去寫一個可攜帶的程式。

有一些障礙物要易傳，其中有一套包含內建資料型態（ built-in data types ）的限制：最大整數值、實數的精確度和大小，集合內的元素個數等等。每有 PASCAL 的版本都有這些限制，在他們之間就產生了易傳的問題

；例如，為電子計算機寫一個程式，其實數型態假定為8數位，當在另一精度大於8的電子計算機上運轉，該程式即失敗。

其次，許多嚴重的障礙來自語言的實行程式（實行器）。PASCAL 在一特別的電子計算機或作業系統實行，實行器上可以增加一些標準PASCAL上所沒有的特性，使得程式設計師覺得工作更方便、簡單，也可能忽略掉一些比較難實行的特性。這些可能導致標準語言特性，再也說不上標準了。因此，在不同版本之下傳動一個程式，就只能利用語言的共同的特性。如此就產生嚴重的易傳問題，尤其兩個版本相差很大的時候。

第三種障礙來自程式設計師。往往由於電子計算機的能力或限制，使得他們誤入歧途。例如，他們可能用該機器的組合語言寫部分程式，也可能利用他們對電子計算機的記憶體的知識使用變數。他們可能寫一些碼來控制列表機、終端機的特性。

有了這些障礙，程式設計師如何能保證避免易傳性的問題呢？為了使程式達成最大的易傳性，就要注意實行下列技巧：

1.不管非標準語言的擴充，堅持使用標準版本。

2.避免使用標準 PASCAL 語言的某些不能於其他機器上使用的特性。

3.關於電子計算機系統，只有在最基本的前提之下，程式可以運轉（RUN）。

4.使用「最普通的名稱」，例如，實數的精確度，集合內最大的元素個數，劃底線的字元集之內容與先後次序等等。

這是吸引人的，許多介紹語言的教本以廣大的篇幅討論這些。對於大部分的程式而言，儘可能使其具易傳性是不切實際的。大多數的程式設計師也都發現有使用非標準 PASCAL 語言的必要。就如前面所提的，標準PASCAL 語言擁有一些不當的概念；許多擴充版本就是用來解決這些問題的。如果程式設計師拒絕使用擴充版本，堅持走易傳性的路子，則不但浪費時間

，而且浪費記憶空間，其效率比原始版本還低得多呢！

同理，真正執行時間的限制、記憶體的使用，和程式的審美觀念來看，經常迫使程式設計師寫不可攜帶的程式。

APPLE PASCAL與標準PASCAL

本書的程式使用APPLE PASCAL 1.1版的系統，書中提到的APPLE PASCAL都是這個版本，因此它可以在APPLE Ⅱ PLUS和APPLE Ⅱe 上運轉。大部分的程式設計師要運轉APPLE PASCAL 系統有一個以上的磁碟機，與正常的40行螢幕顯示。然而pascalc建議使用80一行介面卡。列表機則由print 和 pascalc 控制產生輸出排印。

至於其他版本的PASCAL如何使用呢？首先考慮UCSD PASCAL。APPLE PASCAL直接由UCSD Ⅱ.1發展出來，二者相似，UCSD Ⅱ.1的用戶運轉這些程式只需稍加修改，或者不要修改，最多是改那些針對APPLE的特殊硬體特性方面的常式。稍後會警告中止使用UCSD PASCAL IV.0版，雖然它增加了某些特性可以改進程式執行狀況。除非特別的申明，本書此後提到的UCSD PASCAL指其所有的版本。

APPLE Ⅲ的APPLE PASCAL與APPLE Ⅱ的PASCAL相似，大部分的程式不需改即能運轉，只改那些APPLE Ⅲ PASCAL 提供改進的特性部分。

使用其版本 PASCAL 的人，可能必須作大幅度的修改，雖然需要詳細的精確翻譯的指令集，却不是本書討論的範疇；修訂的程度端賴其PASCAL特性而定；如果特性相同或相似，則翻譯起來就簡便得多，實際上，許多PASCAL的實行，若與標準PASCAL 的擴充版不同，也是非常相近似。

APPLE PASCAL和標準PASCAL有若干不同之處，主要的是：

字串（ STRINGS ）APPLE和UCSD PASCAL除了標準PASCAL內的內建資料型態（ built-in data types ）之外，尚提供一內建字串（ string ）資料型態。所謂字串是任意個數字元的，其最大長度是固定的，通常最大長度定為80字元，但是最長可達255個字元。字串在記憶內儲存方式是一個字元占用一位元組的陣列；第一個位元組（編號0）包含該字串的長度。此外，尚有若干操作字串的功能。字串在其他的版本，也許有意義，但是實行的細節可能不同。

長整數（ LONG INTEGER ） APPLE和UCSD除了標準PASCAL 的正常整數資料型態之外，還提供一種長整數資料型態。APPLE PASCAL 的整數範圍是－32767到32767，長整數可擁有36個位數（ digits ）。例如，宣告

```
< type >
    longint = integer〔 10 〕;
```

longint型態的變數，其值可能是介於－9999999999到9999999999的整數值。長整數是因應程式設計師的需要而定的。許多版本都提供一些廣域的精度型態。

輸入和輸出（ INPUT AND OUTPUT ） APPLE和UCSD PASCAL在輸入和輸出領域較標準 PASCAL 更寬廣。它們有為讀或寫磁碟檔的內建常式（ routine ）。並有捕捉輸入／出錯誤的方法（例如，硬體失靈，或要讀一個根本不存在的檔案）。程式可能隨機接連檔案，可以讀或寫任意一個結論，不受干涉。有低階（ low-level ）I/O 常式直接接達週邊裝置。其他版本也有類似功能的常式。

動態變數（ DYNAMIC VARIABLES ） APPLE PASCAL 並無標準PASCAL內的dispose功能，却有mark和release內建程序（ procedure

）；它們需要一些不同的管理方法。APPLE和UCSD都提供 memavail 函數，它回轉動態變數的有效記憶空間。

圖形（GRAPHICS） APPLE PASCAL利用APPLE的繪圖能力。

分隔編譯（SEPARATE COMPILATION） APPLE和UCSD允許一群副常式當作單元編譯。而這些單元可直接讓其他程式使用，不必重新編譯。

段程序（SEGMENT PROCEDURE） APPLE和UCSD PASCAL允許程式內一些程序宣告成段（segment）程序。一段程序在程式運轉時不一定要在記憶中；當該程序被呼叫時，即自磁碟讀到記憶裡。程序執行完畢，即被遺忘，其在記憶內所占的空間可由程式的其他部分使用。如此有效的應用記憶體，在小電子計算機使用尤具價值。

雜項截取（MISCELLANEOUS） APPLE和UCSD PASCAL 提供下列常式，使程序設計師的工作簡便：

moveleft 和moveright 程序，允許自記憶的一部分抄若干位元組，到記憶體的另一地方。

fillchar 程序被填滿一陣列的字元（或記憶體的任意範圍。）。

sizeof 回轉一變數或型態大小的函數。

exit 自一程式模組或程式本身依序出口的一程序。

gotoxy 是一程序，它將 CRT 螢幕的游標，輸送到一特定的位置。

APPLE PASCAL 同時提供一群額外常式，這些常式常用到的有：

keypress 布耳函數，當按下電子計算機的一個鍵，該函數即回轉 **TRUE**，否則回轉 **FALSE**。

random 函數，回轉一介於 0 到 32767 之間的虛擬亂數。

randomize程序，將虛擬亂數產生器定初值。

讀者很容易了解，有這些特性的程式，很難在不具備這些特性的版本上運轉。我們並未忽視這問題；為了使程式儘量完美，必要時得犧牲一些携帶性。茲觀察下列規則以降低易傳性問題到最小。

- 在少數程式模組內的硬體相關有相當的獨立性。例如，只有二個程序被用來控制APPLE的**CRT**螢幕（其中之一是**gotoxy** 即為 APPLE PASCAL和UCSD PASCAL中的內建函數）；將程式移動到另外的電子計算機或終端機，程式設計師只需改本代碼。
- 當一携帶性解法良好，就不必須用APPLE PASCAL 的非標準特性。（例如，不使用非易傳的**exit**常式，自一程序離開，除非該結果使得代碼簡明或有效率。）
- 在APPLE中的6502組合語言寫的常式，在PASCAL 中也使用之，將之翻譯成其他處理機（ processor ）也較容易。

附錄A 提供解決携帶問題的詳細建議，但對於少量調整成本書的代碼並不固定。

推薦閱讀（Recommanded Reading）

K. Jensen和N. Wirth著的Pascal User Mannual and Report對於PASCAL 有全面介紹。本書對於已熟悉電子計算機語言，也作簡單介紹。新的ISO標準PASCAL設計的非常清晰，有關ISO PASCAL的書有：D. Cooper著的Standard Pascal User Reference Manual。當我們參考標準PASCAL時，上述兩書不 可或缺。大部分情況而言，其間並無差別。如有不同之處，只要我們使用到，就會作一交代。

好的PASCAL教科書有：Programming in Pascal（ P. Grogono

著）；Introduction to Pascal（J. Welsh and Elder 著）。

除了介紹階層的書籍之外，尚有二本傑作：一是N. Wirth 著的 Algorithms + Data Structure = Programs，另一是B. Kernighan 和 P. J. Plauger 合著的Software Tool in Pascal。

高等PASCAL程式設計技巧

目　次

1　什麼是好程式 …… *1*

使用者的準則…… 2
程式設計師的準則…… 7
程式語言…… 10
推薦閱讀…… 11

2　陰極射線管技術 …… *13*

theseus 常式…… 22
建立迷宮…… 25
解答迷宮…… 31
建　　議…… 35
推薦閱讀…… 36

3　交作輸入 …… *37*

標準的 PASCAL 本文輸入 …… 37
鍵盤輸入…… 38

字串輸入…… 41
字串操作常式…… 46
gaslog 常式 …… 47
資料結構…… 54
字串序連…… 56
布耳輸入…… 56
定點數值的輸入…… 58
接達字串字元…… 63
日期輸入…… 64
資料檔案初域…… 66
資料登錄…… 70
產生報表…… 73
建　　議…… 77
推薦閱讀…… 77

4　打碎數字：一般目的計算器 …… *79*

錯誤復原…… 81
資料結構…… 82
變數儲存…… 85
計算的主常式…… 87
全盤變數的初值化…… 90
三個簡易的模組…… 91
摘取格式指引…… 92
敘述語法…… 93
認識識別符號…… 96

表式求值…… 98
項目求值…… 100
因素求值…… 101
算　　術…… 102
一個字串轉變成一個廣域實數…… 107
將廣域實數轉換成一字串…… 109
變數的儲存和檢索…… 112
建　　議…… 115
推薦閱讀…… 117

5　本文檔案工具…… ***119***

選　　替…… 119
APPLE PASCAL 本文檔格式 …… 121
本文檔資料結構…… 121
打開輸入檔常式…… 123
建立新檔常式…… 125
關閉檔案…… 126
自本文檔讀一字串 tfread …… 127
組合語言…… 134
單元和館…… 141
度　　量…… 143
排　　印…… 145
全盤變數初值化…… 150
改變排印參數…… 151
取得檔案名稱…… 152

排印檔案…… 153
輸出初值和終止…… 154
解碼逸出順序…… 157
印各個檔案…… 159
字串和字元輸出…… 162
頁格式…… 163
找包含指引…… 164
建　　議…… 167
推薦閱讀…… 167

6　遊戲與戰略…… ***169***

翻棋的規則…… 170
運轉程式…… 172
資料結構…… 173
翻棋的主常式…… 176
設定全盤變數的初值…… 177
顯示棋盤…… 180
簡易棋賽…… 180
定比賽變數的初值…… 182
雜項的簡單常式…… 184
找合法移動…… 186
取得玩家的着手…… 189
下手棋…… 190
結束棋賽…… 192
比賽理論…… 193