

SMART TEAM

Pascal

程式設計與問題解析

林維甫 編譯

本書包含：

Turbo Pascal 4.0 版之操作說明

MS-DOS 3.2 版之操作說明

書內範例磁片請洽松崗購買

松崗電腦圖書資料股份有限公司

Pascal

程式設計與問題解析

本書包含：

Turbo Pascal 4.0 版之操作說明

MS-DOS 3.2 版之操作說明

書內範例磁片請洽松崗購買

林維甫 編譯

松崗電腦圖書資料有限公司 印行

松崗電腦圖書資料股份有限公司
已聘任本律師為常年法律顧問，
如有侵害其著作權或其他權益者
，本律師當依法保障之。

長立國際法律事務所

陳 長 律 師



Pascal

程式設計與問題解析

編譯者：林 維 甫

發行人：朱 小 珍

發行所：松崗電腦圖書資料股份有限公司

台北市敦化南路五九三號五樓

電 話：(02) 7082125 (代表號)

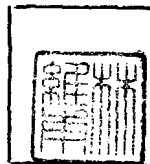
郵政劃撥：0109030-8

印刷者：建發印刷設計公司

中華民國七十七年八月初版

中華民國七十八年五月第四版

版權所有



翻印必究

每本定價 320 元整

書號：2101300

本出版社經行政院新聞局核准登記，登記號碼為局版台業字第三一九六號

譯 序

本書是密西根中央大學名教授Douglas W. Nance所著"Pascal Understanding Programming and Problem Solving"的翻譯本，其主要內容是介紹Pascal語言的應用與程式設計的理念。因內容充實，而且說明與範例相互對映，故廣用於國內外各大學，例如台大資訊系即以此書作為Pascal之教科書。

全書共分十五章，第一章在簡介電腦基本結構，其餘各章都在介紹Pascal語言與其應用。因係針對初學者所設計之Pascal教材，故其具有下列特點：

1. 以循序誘導方式（Step by step）配合圖形與範例說明，不致讓初學者產生恐懼之心理。
2. 每個章節在研討前，都預先闡述該章節所欲達成之目標，以作為讀者自行驗證學習效果之依據。
3. 每介紹一個新觀念後，都附有簡短的自我測驗題目，以驗證讀者是否熟習之。
4. 每個章節都附有豐富的練習題，每章最後是以一個完整的程式來整合全章之觀念。
5. 在全書內容中，一再強調程式的撰寫風格與設計理念，此點對程式設計者而言是非常重要的。

本書在編譯過程中，備極艱辛，數度幾近停筆；有時在驚歎他人完成著作之速，實在感慨自己之低能；但是，每當望見坊間低品質書籍的充斥，以及莘莘學子的無奈，即興起繼續寫作之強烈慾望。

最後，我衷心感謝資策會戴治先生、台灣大學朱撼寰先生、交通大學姚鴻州先生、慧鼎科技季陳基先生，假如沒有他們的幫助，這本書將無法完成。本書的編纂儘管全力以赴，相信疏漏之處在所難免，仍望各界師長、前輩不吝指教。

林 維 甫 謹識

1988年7月於台北

目錄

第 1 章 計算機結構與程式語言	1-1
1.1 今日電腦	1-2
1.2 電腦硬體	1-3
1.3 電腦語言	1-7
第 2 章 練習撰寫程式	2-1
2.1 程式的發展	2-2
2.2 撰寫程式	2-9
2.3 資料型態與輸出	2-23
2.4 Pascal 的算術運算	2-38
第 3 章 變數、輸入、常數與標準函數	3-1
3.1 變數的使用	3-1
3.2 輸入	3-10
3.3 常數的使用	3-19
3.4 標準函數	3-23
第 4 章 撰寫完整的程式	4-1
4.1 撰寫程式	4-1
4.2 利用程序來輸出	4-17
4.3 初學者容易發生的錯誤	4-37
4.4 執行程式	4-47
本章綜程式範例	4-57
程式設計綜合問題	4-63

第5章 條件敘述	5-1
5.1 布林表示式	5-1
5.2 IF... THEN 敘述	5-15
5.3 IF... THEN... ELSE 敘述	5-25
5.4 多層式 IF 敘述 (Nested IF statement)	5-31
5.5 CASE 敘述	5-44
本章綜合程式範例	5-54
程式設計綜合問題	5-59
第6章 迴路敘述(重複性敘述)	6-1
6.1 重複執行固定次數的迴路	6-2
6.2 前端測試的迴路	6-13
6.3 底端測試的迴路	6-29
6.4 迴路之間的轉換	6-36
6.5 多層式迴路	6-41
6.6 迴路與條件敘述	6-52
本章綜合程式範例	6-59
程式設計綜合問題	6-63
第7章 函數與程序	7-1
7.1 函數	7-1
7.2 程序	7-11
7.3 識別字的使用範圍	7-26
7.4 副程式的應用	7-38
7.5 向前參考、套疊和遞迴	7-46
本章綜合程式範例	7-59
程式設計綜合問題	7-61

第8章 使用者自定資料型態	8-1
8.1 Pascal 的型態定義	8-1
8.2 子區間可以定義為資料型態	8-7
8.3 有序資料型態的應用	8-14
本章綜合程式範例	8-19
程式設計綜合問題	8-23
第9章 一維陣列	9-1
9.1 基本概念和符號	9-1
9.2 陣列的應用	9-13
9.3 選擇式排序	9-25
9.4 陣列和副程式	9-33
9.5 聚集式陣列	9-41
本章綜合程式範例	9-53
程式設計綜合問題	9-57
第10章 多維陣列	10-1
10.1 二維陣列	10-1
10.2 字串陣列	10-17
10.3 並行陣列	10-22
10.4 多維陣列	10-28
本章綜合程式設計	10-34
程式設計綜合問題	10-37
第11章 記錄	11-1
11.1 記錄的定義	11-1
11.2 記錄的使用	11-2

11.3	包含記錄的資料結構	11-23
11.4	變動記錄	11-35
	本章綜合程式設計	11-41
	程式設計綜合問題	11-47
第12章	檔案	12-1
12.1	檔案的定義	12-2
12.2	檔案的處理	12-6
12.3	本文檔案	12-18
12.4	具有結構化元素的檔案	12-23
	本章綜合程式範例	12-32
	程式設計綜合問題	12-37
第13章	排序、合併和搜尋	13-1
13.1	排序演算法	13-2
13.2	合併演算法	13-9
13.3	搜尋演算法	13-15
	研討本章主題之相關書籍	13-23
	程式設計綜合問題	13-24
第14章	集合	14-1
14.1	集合的宣告方式與其應用術語	14-2
14.2	集合運算和關係運算子	14-8
14.3	集合的應用	14-13
	本章綜合程式範例	14-18
	程式設計綜合問題	14-21

第15章 動態變數和資料結構

15.1	指標變數	15-3
15.2	鏈結串列	15-10
15.3	鏈結串列的處理	15-19
15.4	動態資料結構——堆疊、佇列、二元樹	15-35
	研討本章主題之相關書籍	15-48
	程式設計綜合問題	15-49
附錄 A	Pascal的保留字	A-1
附錄 B	Pascal的標準識別字(常數、型態、檔案、 函數與程序)	B-1
附錄 C	Pascal的語法圖	C-1
附錄 D	字元集	D-1
附錄 E	Pascal 編譯程式的錯誤訊息	E-1
附錄 F	GOTO敘述	F-1
附錄 G	MS-DOS的操作說明	G-1
G.0	本附錄要旨	G-1
G.1	作業系統簡介	G-2
G.2	Microsoft 公司簡介	G-4
G.3	MS-DOS 簡介	G-6
G.4	MS-DOS的相容產品	G-7
G.5	MS-DOS的載入	G-7
G.6	檔案名稱	G-11
G.7	內在命令的使用	G-13
G.8	外在命令的使用	G-18

G.9	作業系統下的控制鍵和功能鍵.....	G- 21
G.10	作業系統的批次執行.....	G- 22
G.11	樹狀結構的檔案管理.....	G- 30
G.12	PC/AT 的 Setup.....	G- 36
G.13	硬式磁碟機的備份 (Back up)	G- 41

附錄 H Turbo Pascal 4.0 版的操作說明 H- 1

H.0	Turbo Pascal 簡介	H-1
H.1	Turbo Pascal 系統的安裝	H-2
H.2	Turbo Pascal 的編譯程式	H-3
H.3	整合環境的操作說明	H-3
H.4	熱鍵的功能介紹.....	H-6
H.5	Turbo Pascal 的功能表結構.....	H-7
H.6	Turbo Pascal 的載入 (Load)	H-9
H.7	程式的撰寫 (Coding)	H-9
H.8	程式的編修 (Editing)	H-10
H.9	程式的儲存.....	H-14
H.10	程式的編譯.....	H-14
H.11	程式的執行	H-15
H.12	查詢檔案目錄.....	H-15

索引

第1章

Pascal

計算機結構與 程式語言

本章將簡單說明電腦定義及其應用概念。雖然，世界上各種不同電腦的價位，分別從幾佰美元的個人電腦到上仟萬美元的大型電腦，然而這些電腦的基本運作原理都是相同的。本章也將介紹電腦的各個部門以及程式語言的概念。同時，在不涉及任何特定的程式語言下，我們也將討論如何應用電腦來解決問題。

讀者閱讀本章時，不必考慮太過深入，因為所有的專用術語都將再說明於後面章節中。而研讀本書的好方法，乃是每間隔一個段落就重新閱讀一次，因其將幫助您融會貫通正確的觀念與技巧，並能擴展應用電腦的技能。最後，我們願意強調：

學習設計程式是令人興奮的事情，因其將帶給您相當的成就感與滿足感。

1.1 今日電腦

人們尋找輔助計算方法的歷史，就如同數字系統（number system）的出現歷史同樣久遠。早期的算盤、Napier's bones 與計算尺等輔助計算工具，以及近代計算器（calculator）的發明，使得計算工作在方便、快速與低成本的優點下，而有更進一步的轉變。

歷經二十年的努力，現代電腦已有長足進步。在 60 年代，一部電腦必須置於數個房間之內；但由於矽晶片的發明，使得現在電腦的體積大為減小，而利用價值也隨之提高。目前，大部分的父母已有能力購買個人電腦作為孩子的耶誕禮物，而個人電腦的功能却遠超過以前的大型電腦。

何謂電腦？依據第二版 New World Dictionary of the American Language 的定義，電腦是一種電子機器，它能藉助所儲存的指令與資訊，來達成快速而複雜的計算以及資料處理。基本上，電腦可以視為一種能夠處理數字與文字的機器，而這些數字與文字就稱為資料（data）。電腦之身價不凡，就是因其能夠快速而準確的處理資料。

電腦的型式，可依功能簡單區分為大型電腦（mainframe）、迷你電腦（minicomputer）與微電腦（microcomputer）三種。大型電腦是用在規模龐大的學校或公司，它可以提供 100 人以上同時使用，其價位在百萬美元以上。而迷你電腦之記憶容量雖然較小，但亦可同時提供數十人使用。微電腦常被稱為個人電腦，如與前二種電腦相較，其記憶容量相當有限，通常只提供單人使用，而價格只要數佰美元左右。

剛開始接觸電腦，我們常會聽到硬體（hardware）與軟體（software）這兩個術語。所謂硬體就是實體的機器與其支援設備的通稱，而軟體則是指揮機器工作的程式（program）。目前，電腦通用的套裝軟體（software package）包括文書處理系統、資料庫系統、操作系統與編譯程式等。當然，我們也能經由學習而設計屬於自己專用的軟體，實際上這也是本書的目的。

電腦程式就是指令（instruction）的集合，這些指令告訴電腦該執行何種任務。例如：

```

PROGRAM ComputeAv (input, output);

VAR
  A,B,C : integer;
  Average : real;

BEGIN
  read (A,B,C);
  Average := (A + B + C) / 3;
  writeln (Average:20:3)
END,

```

此程式將命令電腦從輸入檔 (input file) 中讀取三個整數，並經運算求其平均值後輸出之。讀者在此處不必瞭解程式中的特定用語，因為上述範例只在說明程式是指令的集合。在本書的引導下，相信您不久就能設計更複雜而且更有意義的程式。

學習程式設計的兩個必備條件：

- 1 欲與電腦溝通，則必須使用特定符號與術語。亦即，必須使用特定的程式語言 (programming language)。
- 2 欲藉助電腦解決問題，則必須能夠提出解決問題的計劃，此種計劃在電腦專業用語則稱為解題方案 (algorithm)。解題方案包含解決問題的一連串步驟，電腦依循這些步驟的處理方式就能解決問題。

最初，初學者所碰到的問題可能比較容易解決，所以很容易誤認為學習電腦語言遠比解決問題困難。事實上，對於學習電腦科學 (computer science) 的初學者而言，最重要的一件事情，就是如何加強解決問題的能力；一旦具備這種能力，則可以輕易的利用各種不同的語言來設計程式。

1.2 電腦硬體

依照我們先前的說法，電腦乃是一種機器，但現在讓我們以另外一個角度來談論電腦的定義。雖然電腦的種類繁多，但它們都是由中央單元 (central unit) 與輸出 / 入裝置所組合而成 (如圖 1.1)。中央單元包括中央處理機 (central processing unit, 簡稱 CPU) 與記憶體 (memory) 兩部份。而中央處理機相當於電腦的 " 大腦 "，其包含算術邏輯單元 (arithmetic / logic

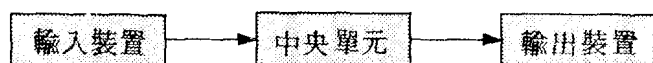


圖 1.1 電腦的組成元素

unit，簡稱 ALU）與控制單元（control unit）。算術邏輯單元是在執行算術與邏輯運算；而控制單元則用來控制電腦的各個部門，以正確執行使用者之程式。

記憶體可以想像為郵局的信箱，其是由一連串的位置（location）所組成，可以儲存各種不同的指令、數字與文字等資料。電腦主機中所含的記憶體稱為主記憶體（main memory 或 primary memory），主記憶體的儲存位置有一定的限制，如果需要額外的儲存空間，則必須使用副記憶體裝置（secondary memory device）。對個人電腦而言，副記憶體大都為軟式磁碟片（floppy disk）或磁帶；而大型電腦則為硬式磁碟（hard disk）或磁帶。主記憶體是由磁圈或積體電路所構成，主要是用來暫時儲存正在或立刻會被 CPU 處理的程式或資料，如果程式已經執行完畢，則程式本身及其相關資料，電腦都將自動改存於價格較便宜而適合長期儲存的磁碟、磁帶等副記憶體中。

至於「記憶體如何儲存資料？」這個問題雖然與學習 Pascal 無關，但可能有助於您獵取相關的電腦知識。記憶體的每個位置都是儲存二進位數字（binary digits），並有其特定的位址（address）。任何指令、符號、文字、數字等都被轉譯（translate）為適當的二進位數字，然後分別儲存於不同的記憶位置，而我們可以任意使用和更改這些資料。實際上，程式也是經由轉譯而儲存於主記憶體中。圖 1.2 與圖 1.3 分表為主記憶體與中央單元之透視圖。

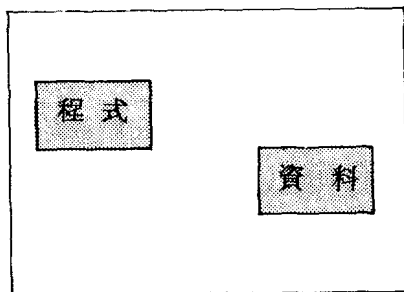


圖 1.2 主記憶體

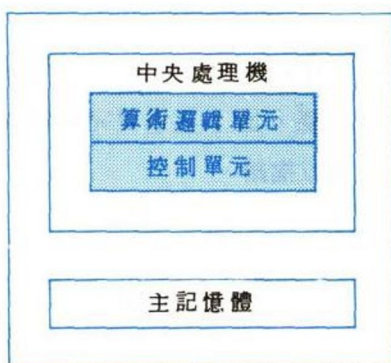
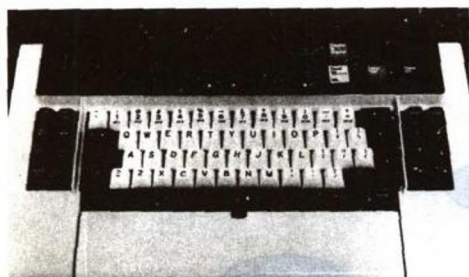
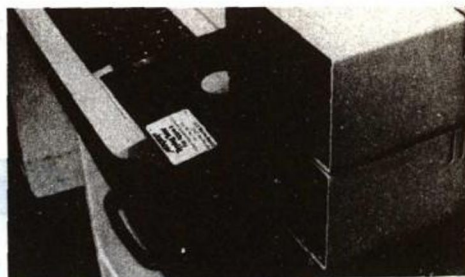


圖 1.3 中央單元

為了傳送程式與資訊，電腦必須具有**輸入裝置**（input device），例如圖 1.4 的鍵盤（keyboard）與磁碟機（disk drive）。如欲獲得程式的執行結果，則電腦必須具有**輸出裝置**（output device），常見的輸出裝置（如圖 1.5）有終端機的螢幕（terminal screen）與列印機（printer）等。通常，輸入裝置與輸出裝置合稱為**輸出／入裝置**，而電腦主機以外的設備則通稱為**週邊設備**（peripheral device）。



(a)

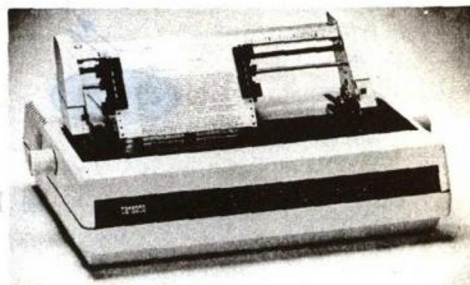


(b)

圖 1.4 (a) 鍵盤 (b) 磁碟機



(a)



(b)

圖 1.5 (a) 終端機螢幕 (b) 列印機

1-6 Pascal 程式設計與問題解析

簡而言之，電腦的硬體是由輸入裝置、中央單元與輸出裝置三者所組成，而圖1.6可說明這三者之間的關係。圖1.7為更完整的說明，其不但列出各種週邊設備，並對資料傳遞之路徑加以描繪。

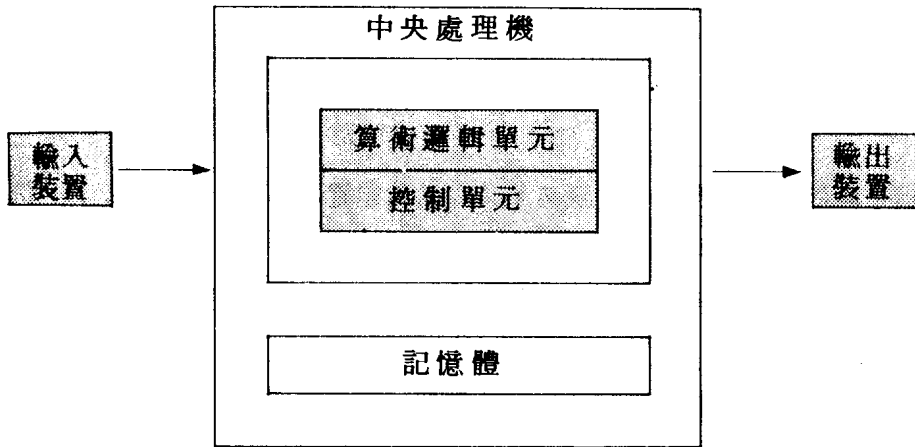


圖 1.6

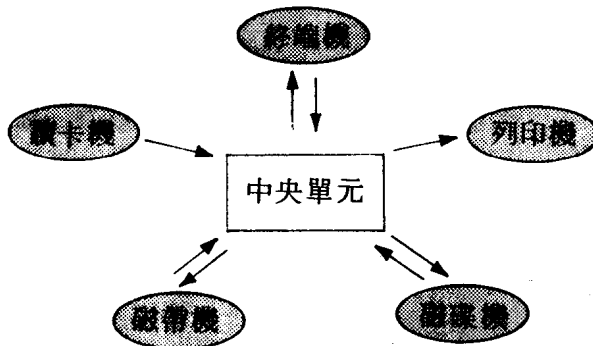


圖 1.7

1.3 電腦語言

所有資料的傳遞、處理、存取都是由一連串的二進位數字指揮電腦來達成。因此，如果希望兩個整數相加，則所需的指令可能為：

```
10100100100000000000000100100110100101000000000000001001010
011110110001010101001110000000000000001001011100110000000000
```

以這種二進位方式撰寫的指令就稱為機器語言 (machine language)。如以機器語言撰寫程式，則不但花費時間，而且不易閱讀與瞭解。

而進一步發展的電腦語言允許使用文字與符號來簡化指令，例如上面的機器語言可改寫為：

```
LOAD A
ADD B
STORE C
```

此即先將A值加至B中，其結果再儲存於C中，以備取用，此種語言就稱為組合語言 (assembly language) 或低階語言 (low-level language)。實際上，這些文字與符號都將被轉譯為適當的二進位數字以供機器執行。

雖然組合語言已使程式較容易被閱讀與設計，但以方便而言仍嫌不足，所以幾種高階語言 (high-level language) 被發展完成，例如Pascal、PL/I、FORTRAN、BASIC、COBOL等，這些語言更進一步簡化指揮機器的特定術語與符號系統。例如，兩整數相加之高階語言為：

C := A + B;	(Pascal)
C = A + B;	(PL/I)
C = A + B	(FORTRAN)
C = A + B	(BASIC)
ADD A,B GIVING C	(COBOL)

每種高階語言都具易讀、易懂、易寫等優點。本書將詳細說明有關Pascal的應用觀念、術語與符號系統。當您精通Pascal後，將可輕而易舉的學習其它高階語言。