三民電腦叢書

PASCAL程式語言

陳振三 著

三民書局 印行

PASCAL 程式語言

陳振三著

學歷：國立交通大學計算機工程系畢業
經歷：曾任助教一年，並參與系統分析
及程式設計多年

三民書局印行

行政院新聞局局版臺業字第〇二〇〇號

中華民國七十五年九月初版

基本定價伍元伍角陸分

著作者 陳 振 三

發行人 劉 振 強

出版者 三民書局股份有限公司

印刷所 三民書局股份有限公司

臺北市重慶南路一段六十一號

郵撥：〇〇〇九九八一五號

序

計算機使得今日科技進入了太空的領域，其發展之快，若非親眼目睹，身置其中，實難相信於萬一，計算機已逐漸和吾人的生活結合在一起，依目前的預估，不出十年，不了解計算機者，將成爲新一代的“計算機盲”，今人或有感於此，遂起學習之念，然多不得其門而入，或一知半解、或中途放棄，實屬遺憾，筆者有鑑於此，振筆疾書，將十餘年的研習經驗，幾番琢磨，咀嚼復出，期對欲入門計算機的新進，有所助益。

帕斯卡語言，是近年來，發展比較圓熟的一種程式語言，它嚴謹的設計概念，使初習者能迅速入門，易學、易懂的特性及結構化的觀念，兼而有之，實是初習計算機者的最好入門課程，惟在研習過程中，須配合實作，才能獲事半功倍之效。筆者編著此書，希初習者早日跨進計算機大門，共爲資訊世紀，貢獻一己之力。

這本書的編作，初稿是兩年前完成，年來經再三修正，才付梓，其間有賴於同儕何秋風多方協助及先進蔡文能先生鼎力相助，才得順利出版，於此特致感謝之意。

編者謹識

PASCAL程式語言
目　次

第一章

計算機與帕斯卡
(Computer & PASCAL)

1.1 計算機的沿革及特性

(Computer history & characteristics)

自從計算機的發明以來，即掀起又一次的工業革命，在這個資訊爆炸的世紀，計算機的確將工業、科技提昇到另一個境界，這是由於計算機具有算得快、記得多以及高精確度的特性。

計算機的沿革，要追溯到1833年，美國數學家卡羅貝基(Charles Babbage)設計的分析器（Analytical Engine）。一直到1940年，哈佛大學發展了"馬克一號"(Mark I)，往後計算機的研究便有了啓蒙，從1946年；第一部計算機ENIAC（Electronic Numerical Integrator And Calculator）問世起，迄今短短四十年，已由第一代，進入第五代，其演進分述如下：

1. 眞空管時代（第一代計算機）：1958年以前，計算機的組成另件，多爲眞空管，計算的速度以千分之一秒（Millisecond）爲單位。
2. 電晶體時代（第二代計算機）：1958至1964年期間，電晶體的問世，使計算機的體積大大地縮小，同時計算的速度也提高到以百萬分之一秒（Microsecond）爲單位，往昔眞空管

產生大量熱而導致的高故障率，也大幅地降低。

3. 積體電路（I.C.)時代（第三代計算機）：1964年以來，由於積體電路的發展成功，計算機的體積較第一代計算機小了四十倍左右。計算速度更是以十億分之一秒（Nanosecond）爲單位。穩定度也增加了許多，此外，其週邊裝置（peripheral equipment）也隨之多樣化，精密化陸續成功地發展，使得計算機又跨入另一個新紀元。
4. 大規模積體電路（LSI）時代（第四代計算機）：1970年開始，科學家一直致力於計算機體積的縮小，於是各式大規模的積體電路（LSI）相繼誕生，計算機的體積，更形縮小，速度也更快，穩定性更高。
5. 智慧型及超大規模積體電路（Intelligent & VLSI)時代（第五代計算機）：八〇年代，超大規模積體電路使得計算機在體積縮小方面以及速度方面，達到最顚峯，加上智慧的賦予，使得計算機逐漸有思考的能力，計算機的科幻世界也逐一被實現。

1.2 計算機的分類 (Computer categories)

計算機的分類，實在沒有一定的規範，不過通常都是以計算機本身的結構、特性、大小，甚至於價格來作分類，比較常見的分類是依其本身結構及運作原理來分類，分爲數位計算機（Digital Computer）、類比計算機（Analog Computer）及混式計算機（Hybrid Computer）等三種，其主要分別乃在於訊號傳遞原理的不同。另外，還有一種爲我們所知的分類，是依照記憶、容量大小、價格高低及功能的多寡來分，可將計算機分爲：微型計算機（micro-computer），如蘋果二號（Apple-II）、迷你計算機（mini-computer），如VAX-11/780，以

及巨無霸計算機（mainframe），如CYBER 170等三種分類。其差別除了上述的差異之外，當然最直覺的是外觀上，體積的差異。

再從研究、設計的觀點來看，一套計算機系統，可分爲軟體（software）、硬體（Hard ware）及韌體三個方面。硬體是指機器本身的實體，諸如線路板上的積體電路。而軟體是指使機器實體運作的邏輯，如系統軟體及應用軟體。至於韌體（Firm ware）則是介乎軟體及硬體之間的一種產品；因爲其呈現在外觀是屬於硬體的狀態，而眞正運作的原理，卻屬軟體的範疇，雖然韌體的發展較晚，但其兼具有軟、硬體之優點的特性，使得其發展的潛力大爲增加。

1.3 程式的概念（Concepts of Program）

上一節我們所說的軟體，可以說就是程式（Program），誠如前面所說的，軟體是無形的，是一連串的觀念組合，在這裏，我們要爲“程式”下一個簡單的定義：一個程式就是一連串的指令（Instructions）所組成的一個工作規劃。廣泛地說，像廚師的食譜，音樂家的樂譜，以及我們處理事情的方式、步驟，都可以稱爲程式。而實際上，我們所應用的計算機程式，比較起來，是複雜多了；雖然，本質上，強調的都是“方式”、“步驟”，由於計算機上的所有元素及組件，都要有明確的定義，所以自然是複雜一點，因此在學習計算機程式之時，必須格外地謹愼。首先，我們來認識一些常用的術語以及一些基本的概念。

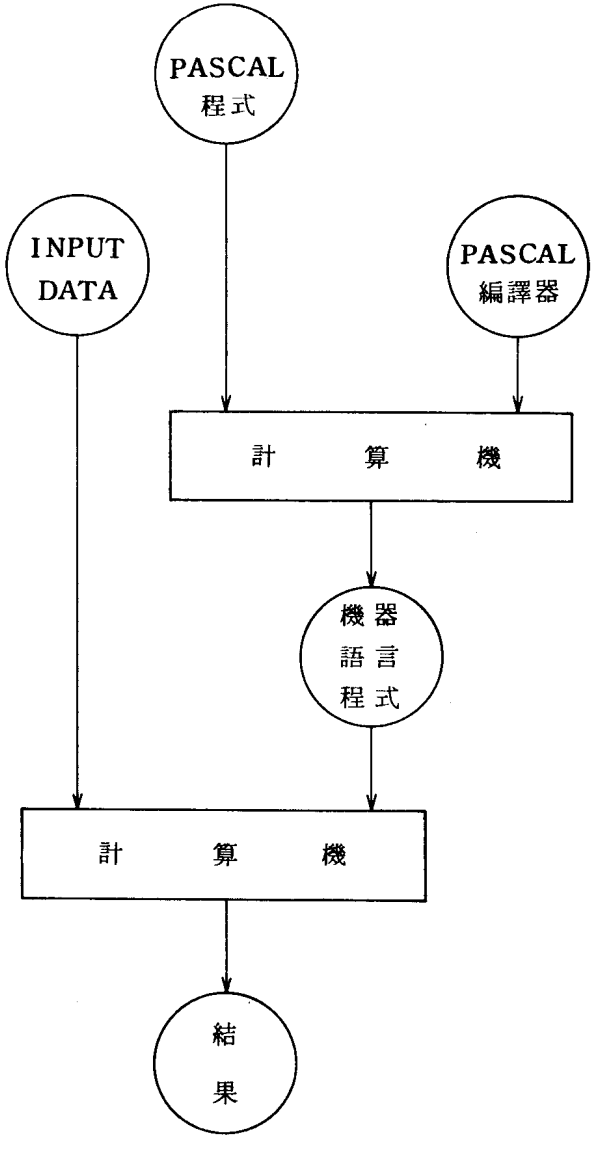
計算機之所以會運作，是我們下命令操縱的，我們下一個命令，計算機才做一個動作，而程式就是我們的命令集合。程式是我們用以命令計算機的一群陳述（statements）。也許你會開始覺得計算機並不聰明，不錯“它”本身並沒有智慧，需要“我們”賦予工作的能力及運作流程，才能發揮功能，當你體會到這一點時，你已經可以準備撰寫程式

來操作“它”了！

1.4 程式的編譯與執行

(Program compilation & execution)

前面提到命令計算機作事，我們可以想像成僱用僕人作事，要命令僕人作事的第一個要素，就是語言上的溝通，也就是說你必須用僕人了解的語言和他溝通，如果你們之間語言上有障礙，那就得請人翻譯了！同樣地，在計算機的國度裏，每種機器，有它自己原始的語言，稱爲機器碼（machine code），所謂的低階的語言(Low level Language)，就好比人類的家鄉話，而我們用以命令它的語言，所謂的高階語言（High level Language），這類的語言，比較前者更具通用性，好比人類目前通用的英語，國語…等等，通常也都有標準化（standarize）過。所以不會因機器不同，而有太大的改變。談到這裏，比較謹愼的讀者，會立刻想到這兩者之間必須有一個部門做翻譯的工作，是的，那就是編譯器（Compiler），程式經由我們用高階語言設計完成，交給特定的編譯器，編譯成低階的機器碼，然後計算機才能照程式的規劃，一步一步去作，那就是所謂的執行了。下圖就是說明程式和計算機間的關係。



程式的執行（program Execution）也稱作跑程式（Run a program）

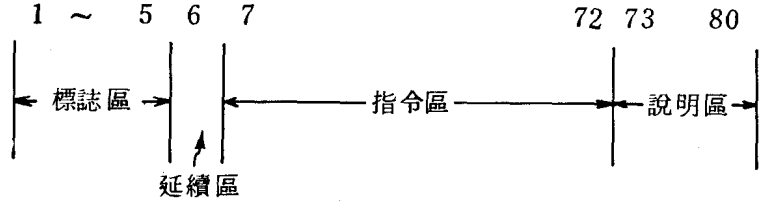
1.5 帕斯卡程式語言(PASCAL)簡介

(Introduction to PASCAL)

帕斯卡語言的基本概念是源自ALGOL 60，不過，就整體結構而言，帕斯卡(PASCAL)比較上，是謹嚴多了！它是瑞士籍的尼可斯微斯教授(Professor Niklaus Wirth)於1971年正式發表的，經過三年的數度修正，到了1974年才眞正有了帕斯卡程式語言的成品出現。

帕斯卡語言適用的範圍很廣，不但具有計算數值的能力，同時更具有結構組織的能力，這是其它程式語言，如福傳(FORTRAN)、科博(COBOL)所不能與之匹比的。

早期發展的計算機程式，撰寫方式，必須依照一定的格式安排，例如：福傳(FORTRAN)語言規定指令必須由第七格開始寫，一直到第七十二格都是指令區，七十三格以後是說明區，至於一至五格則是標誌區(Label area)，第六格是延續區(Continue area)如圖：



這就是所謂的固定格式(Fixed Formatted)，這類的語言雖然有其存在及方便的地方，但在程式的撰寫、維護方面，却不如自由格式(Free Formatted)來得應用自如，這類的語言，撰寫時，並沒有限制區域，所以，比較上，是“自由”得多，帕斯卡語言，便是屬於“自由”格式的一種語言。

早期所發展的程式語言，還有另一個特點，那就是指令逐一寫下，

只要計算機能“跑”得出結果，就交差了，毫無結構的觀念，隨著工作的複雜要求，程式也愈趨複雜，由於缺乏結構的管理，顯得雜亂無章，雖然，程式完成之初，可以跑得出結果，可是一旦要修改、維護，就往往使得程式員束手無策，寧可重新設計，也不願從一個毫無章法的程式裏去琢磨，所以結構化的程式語言，也應運而生，其重要性，我們可以用一句話來形容－整個程式語言的發展史，就是程式結構的改進史。帕斯卡語言，便是結構化的要求下而發展的一種相當嚴謹的語言，不論是系統上或一般應用都很適合，近年來美國軍方發展的新語言ADA ，事實上，可以說是帕斯卡的化身，因此帕斯卡的學習，實在是身爲計算機從業者，勢在必行的。

1.6 語言的可讀性 (Readibility of program)

對於編譯器而言，程式只是一連串符合規則的文字符號，如：

```
PROGRAM SQUAREROOT (INPUT; OUTPUT); VAR X:
REAL; BEGIN REPEAT READ(X); IF X > = 0 THEN WRITE
(SQRT(X)) ELSE WRITE ('PARAMETER ERROR ') UNTIL
X = 0 END.
```

編譯器是能接受的。不過，基於可讀性的要求，這樣的程式設計並不可取，爲什麼呢？因爲這樣的程式，除了計算機不會抱怨之外，幾乎沒有一位程式設計員，願意去看個究竟，如果我們改成：

```
PROGRAM SQUAREROOT (INPUT; OUTPUT); VAR X: REAL
BEGIN
     REPEAT READ(X);
            IF X > = 0 THEN WRITE (SQRT(X))
                       ELSE WRITE ('PARAMETER ERROR')
     UNTIL X = 0
END.
```

是不是一目了然？所以，程式的撰寫必須考慮可讀性的要求，對於程式的維護，會有很大的幫助。

由上述的例子，不難看出，程式的撰寫要儘量依照程式結構加以層次化，這種方法，又稱凹入法，除此之外，還有其它提高可讀性的方法，將在後面的章節一一介紹，在這裏強調的是可讀性的重要。帕斯卡語言，基於這一項要求，在其發展之初，就著重區域（ block ）的概念，整個程式是由一層套一層的區域組成，站在可讀性的觀點，帕斯卡是一種有潛力的語言，這也是爲什麼近廿年來的系統規劃設計，多走向採用結構化語言的原因。

第二章

帕斯卡語言的基本概念
(PASCAL Concepts)

2.1 引　言 (Introduction)

在這個章節，我們要討論的是帕斯卡語言裏，基本的程式組成元件——數據，算式以及指述，讀者不必以強記的方式來閱讀，應以瀏覽的方式很快地讀一遍，有一簡單的概念即可，等到在後面章節眞正用到了，再回頭來詳細研究，印象可以較深刻，同時也可以加快入門的速度。

在計算機的領域裏，所有的東西，都必須有明確的定義（well-defined ），才能被認可。小由數據的安排，記憶體的容量……，大至系統間的協定，都有一定的規範，我們從最基本的程式本身組成元件——數據、算式及指述開始敍述。

任何程式裏，從符號（ Symbols)的角度來看，大致可分爲兩種符號，一種是語言本身所用的符號，另一種是使用者自己定義的符號，就編譯器理論（ Compiler theory ）而言，這些符號，都統稱爲識別元（ Token ），一個程式，說穿了，也只不過是一些規則化識別元的組合。在研究識別元的排列規則之前，我們必須先了解識別元的定義，一般說來，各種語言對識別元的定義大致是沒有多大的差異。因此，如果你曾熟悉過別的語言，這一章可以跳過，或很快的瀏覽一遍，而不影響學習進度。