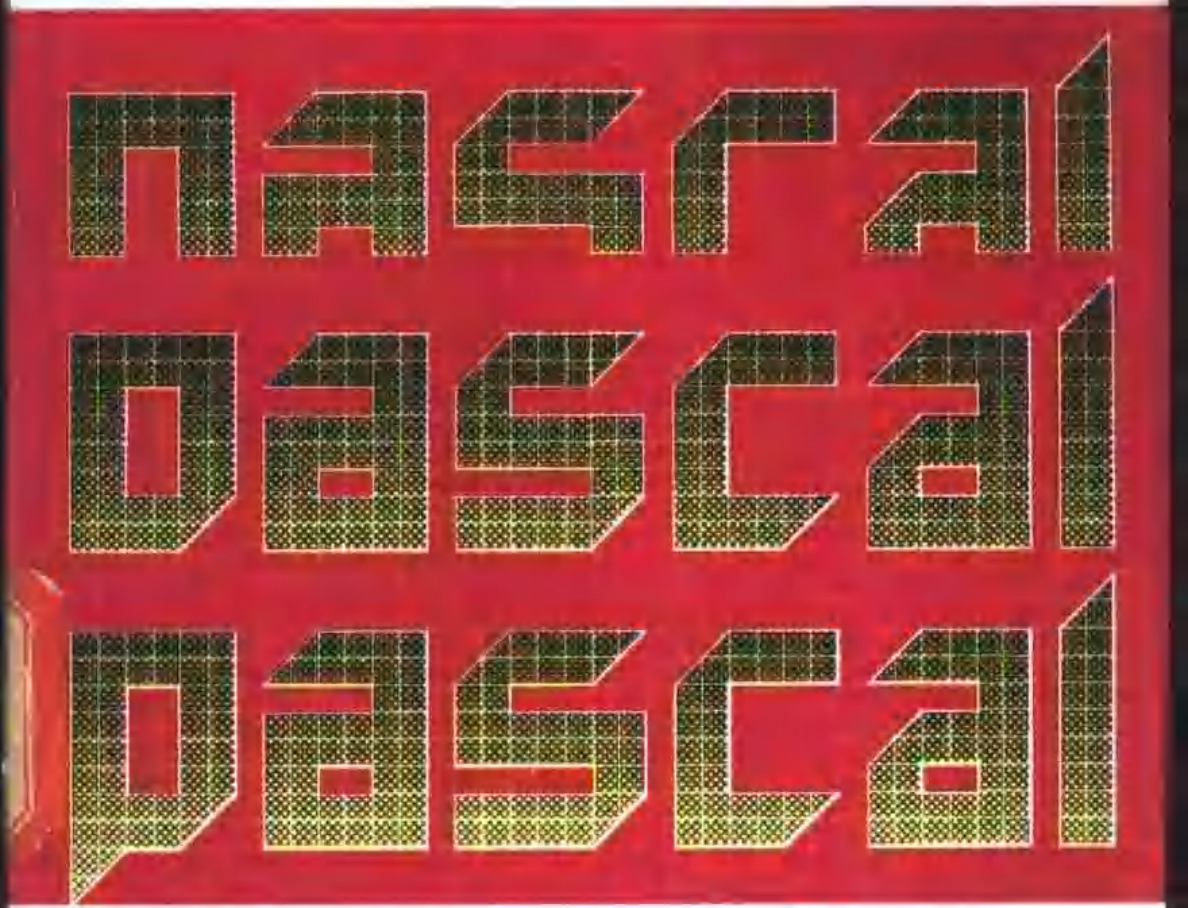
結構化程式語言
—PASCAL—

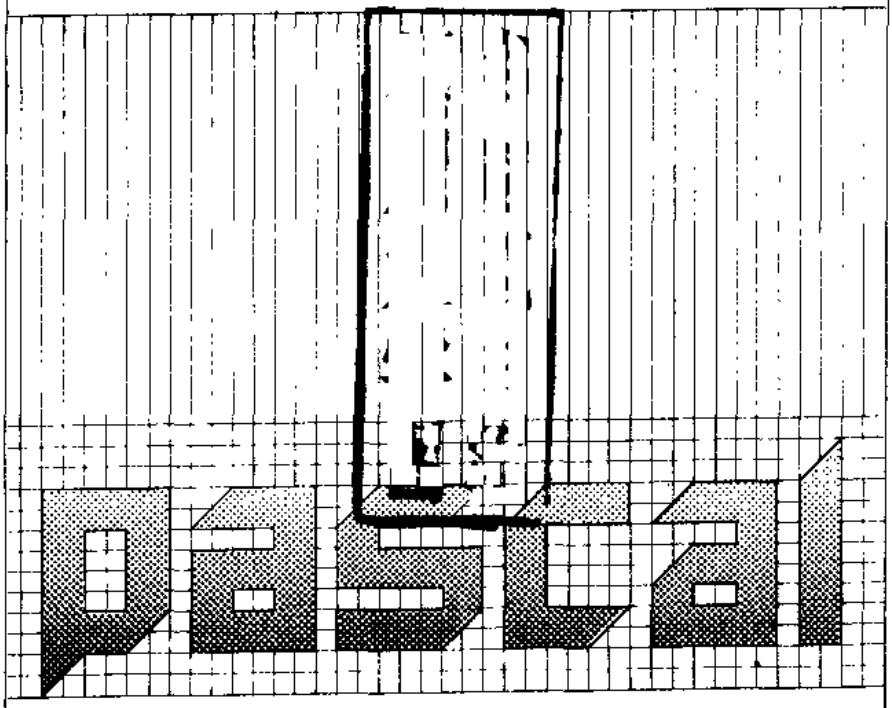
葉垂奇　編著



全華科技圖書股份有限公司

結構化程式語言
—PASCAL—

葉垂奇　編著

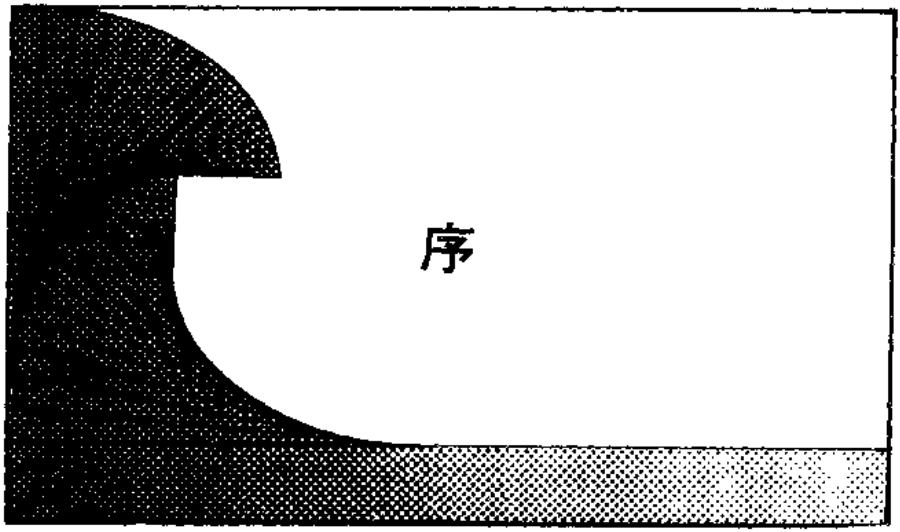


全華科技圖書股份有限公司

結構化程式語言
——PASCAL

葉垂奇　編著

出版者　全華科技圖書股份有限公司
北市龍江路76巷20-2號
電話：581-1300・541-5342
581-1362・581-1347
郵撥帳號：1 0 0 8 3 6
發行人　陳　本　源
印刷者　佳怡彩色印刷廠
定　價　新臺幣 180 元
再　版　中華民國73年3月



序

自從 1972 年瑞士蘇黎士大學的 Niklaus Wirth 教授發表第一個 Pascal 語言的編譯器以來，在這短短十年之間，Pascal 語言已經受到了舉世的肯定與接受，而成爲目前最重要的高級語言之一。Pascal 語言之所以能這麼快就被計算機界所接受，最主要的原因就是使用這種語言所設計出來的程式，本身就具有結構化（Structured）的特性。這種特性使得程式很容易被人了解，也很容易被人修改。這一點在程式設計師嚴重缺乏，而且薪水日益高漲的今天，正可以保障計算機界與企業界在軟體方面的投資，使之不受程式師流動的影響。除了上述這個因素之外，又由於 Pascal 語言所提供之功能並不過份地複雜，因而使得 Pascal 語言的編譯器可以很容易地裝在現今最熱門的微電腦上執行。這就更助長了 Pascal 語言的聲勢。在政府極力提倡資訊工業的今天，我們希望也能藉此書使更多的人能利用 Pascal 語言來設計程式，而更有效地擴大計算機在各層面的使用範圍。

本書共分十二章，可以概分成三個階段。第一階段包括第一章到第七章，其最主要內容在於介紹 Pascal 語言中的基本陳述與資料類別（Data

Types），以期讀者在看完該階段之後，就能吸取 Pascal 語言在程式設計方面的技巧與結構化的精神。第二段是八、九兩章，其重點在於介紹 Pascal 語言中的結構化資料類別（Structured Data Types）。經由這些資料類別，可以大大地提高 Pascal 語言在解決問題上的能力。第三階段就是由第十章到第十二章。在這一段之中，我們所强調的是 Pascal 語言的一些應用例題，希望能由這些應用例子，啓發讀者利用 Pascal 語言去解決各種不同的問題。最後，在第十二章之中，我們也討論了一些 Pascal 語言的限制與特性。

本書在編寫之初，就以教科書或在職人員的自修用書作爲目標；因此全書的進行，盡量以一種口語化的方式來完成，以期能爲更多的讀者所接受。在本書付梓之前，筆者已經在所開的十個不同的 Pascal 語言班級中，試用了本書的內容。根據所得到的經驗與反應，我們認爲本書適於一學期三學分或兩學期四學分的課堂使用。如果以三學分的課來進行的話，至少可以涵蓋第一章到第九章；而第十章與第十一章則必須由課堂時間來取捨講授的範圍。如果本書被使用於兩學期四學分的課程，則將有充份的時間涵蓋第一章到第十一章。至於本書的第十二章，較不適於在初學程式語言的課堂中使用；它的目的是給在職人員或有較多程式經驗的讀者來參考。另外，如果讀者已經有程式語言的經驗，則第一章可以省略不看。最後，如果在課堂及學分有限制的情形之下，本書亦可用於一學期兩學分的語言課程之中，條件是學生必須已有程式語言的經驗。

雖然本書從開始動筆到付印爲止，總共花了將近兩年的時間；但是難免有不盡人意之處，但願讀者先進能多予賜評指教。本書的完成，首先得歸功於內子李芬園的鼓勵與幫助才能順利交稿。同時也感謝家父葉進鳳先生對第一章的修改及所提供之意見。還有得感謝台北工專電子科的王主任瑞材先生所提供之機會與機器設備。最後，但不是最少，得謝謝家母在這段日子中對小犬的照顧，才得使內子與筆者有時間來完成這件工作。沒有以上這些人的幫忙就沒有此書，我衷心感謝他們。

葉垂奇　謹識

71年5月22日于三重自宅

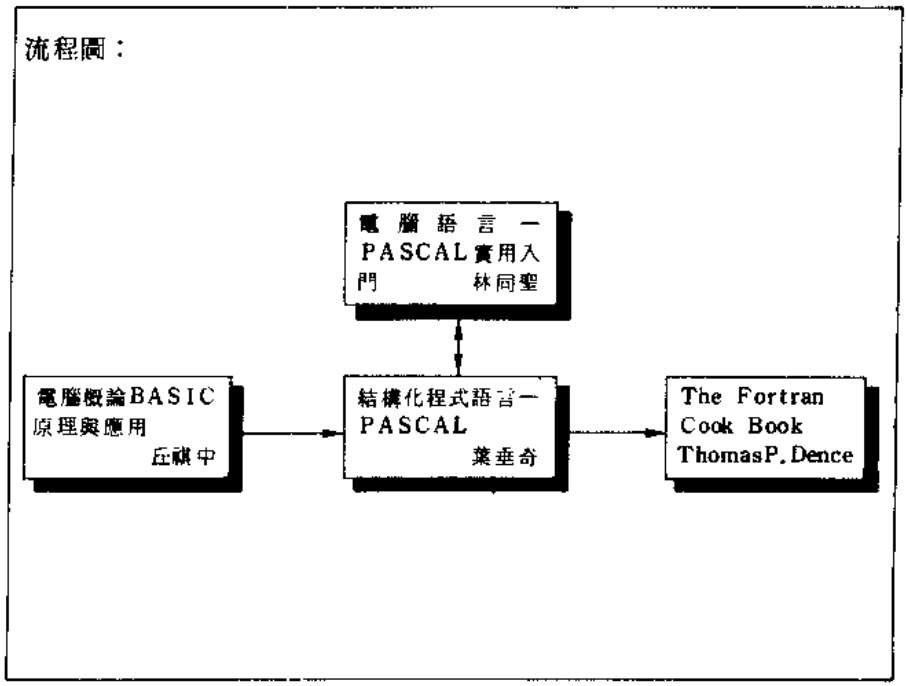
編輯部序

「系統編輯」是我們的編輯方針，我們所提供給您的，絕不只是一本書，而是關於這門學問的所有知識，它們由淺入深，循序漸進。

自從1972年出現PASCAL編譯器後，短短十年間，PASCAL已成爲目前最重要的高階語言之一。本書作者任教於台北工專，累積其教學經驗編寫而成，作者以口語化的方式編寫，以期更多的讀者能夠接受。首先介紹PASCAL語言中的基本陳述、資料類別，再介紹結構化資料類別，最後並舉出一些應用例題，啟發讀者利用 PASCAL 語言去解決各種不同的問題。

爲提供您對電腦方面更完整的知識，我們特地以流程圖方式列出各關圖書的閱讀順序，由淺入深，循序漸進地引導您得到有系統的知識，同時也可以減少您獨自摸索的時間。若您還有任何問題，歡迎來函連繫，我們將竭誠爲您服務。

流程圖：


電腦語言－PASCAL實用入門
林同聖
電腦概論BASIC原理與應用
丘祺中
結構化程式語言－PASCAL
葉垂奇
The Fortran Cook Book
ThomasP. Dence

目　錄

1 計算機與程式設計

計算機被譽爲廿世紀三大發明之一，今天的計算機已經從十幾二十年前的大型機器，也就是大機關單位才買得起的一項設備，漸漸地普遍到我們每個人的日常生活領域，例如在新的汽車上你可能找到很小型的計算機（微電腦）在自動控制油門，使汽車照你的定速奔馳；你也可能在新型的洗衣機、微波烹調器上找到計算機的踪影。根據預測，在1980年代末期，每個家庭中就可能有電腦來幫忙管理家務，同時和外界的大電腦連接。你只要按幾個按鍵便可收看當天的各種新聞、氣象、班機時刻、股票行情等等。在這種日子來臨之前，你得趁早學學如何來使喚這個明日最忠實的僕人，事實上也是今日最重要的工具。

1.1 程式設計

我打賭在你未看過本書之前，你早就聽過“程式設計”這個名詞。也許你早已知道它的含意；但是爲了給第一次接觸計算機課程的讀者一個適當的了解，我們還是解釋一次的好。如果我們把計算機比做人的話，那麼他就是心算速度非常快的人。依計算機種類的不同，它每秒鐘能夠心算十

萬次或者數百萬次以上；但是它却毫無思考的能力。因此除非人們告訴他應該做什麼事，他只會不知所措地呆在那兒。好了，所謂“程式”就是人們要計算機工作的詳細步驟。所謂“程式設計”就是如何把這些步驟，以一種計算機看得懂的語言，正確無誤地寫下來。

1.2 Pascal語言

前節我們給程式設計下定義時，曾提到必須用一種計算機看得懂的語言，寫下你想要計算機做的工作。人跟人之間交談需要語言，人與計算機之間也靠一種特殊的語言來聯繫。

在人的世界裡有無數種的語言存在，在計算機的世界也一樣。我們的語言能力跟環境以及所受的教育有密切的關係。有些人精通很多國家的語言。可是多半的人只會一種語言。同樣地，有些計算機能懂十來種語言，有些只懂得一種。這完全是由於人們對計算機所做的栽培工夫不同所致。

爲什麼需要計算機懂得多種語言呢？就像人們需要學習幾種語言一般。要跟英美人士交談就必須學英語；跟歐洲人你必須懂得法、德、義等等語言。就拿中國話來說，也有許許多多的方言。計算機也是一樣，它所要應付的使用者更是五花八門。它可能碰上工程師、科學家、醫生、公司經理、律師、家庭主婦，甚至兒童等等。人人對計算機期望的工作範圍不同，大家所講的行話、術語也不同。因此爲了使各行各業都能充份而且方便地應用計算機，就要有不同的語言給不同行業的人來使用計算機。

再看人類語言的一個事實。例如有一位大學外文系畢業的小姐，如果她是台籍客家人，那麼她可能懂國語、台語及客家話；又因爲她就讀外文系，她可能還懂英語、法語、西班牙語等等。算算她可能會六、七種語言。好，假設現在她碰上一位追她的男士；只要這位男士懂得任何一種這位小姐懂得的語言，那麼故事就能繼續發展下去了。當時機成熟的時候，不管這位男士說“妳願嫁給我嗎？”或“Will you marry me？”甚至其他語言；由於小姐懂得這些話，那麼她就會有所反應。這可能是含羞低頭，也可能是巴掌一個，掉頭就走。這就是說，只要用這位小姐所懂得的任何一種語言對她說話，所得到的反應只與話的內容有關，而與用那一種語言是

無關的。對計算機而言，這種情形也是存在的。當你要計算機替你做件工作時，你可以使用任何一種計算機語言來告訴他。只要你沒說錯，則不管用那一種語言，所得到的效果都是一樣的。所不同的是，某種語言可能比較容易表達你的意思，而其他語言就不那麼容易了。

Pascal 語言是一般計算機所懂得的高級語言（ High-level language ）中的一種。Pascal 這個名字是爲要紀念十七世紀的法國數學家 Blaise Pascal 而命名的。那麼什麼是高級語言呢？籠統地說，凡是比較接近人類書寫或口語的計算機語言就叫高級語言。高級語言的每句話，往往包含著很多個計算機的動作。相反地，低級語言（ Low-level language ）則每一個命令只能對應一個機器的動作，也就是說比較接近機器的“習慣”。當代最普遍被使用的高級語言有“ COBOL ”（商用），“ BASIC ”（初級適用），“ FORTRAN ”（工程，科學用）。“ Pascal ”是較新興而相當受重視的一種高級語言。它適用的範圍涵蓋了上述的三種語言，同時它還適用於設計系統程式（ System Programs ）。

1.3 計算機結構

雖然學“程式語言”－軟體（ Software ），可以完全不理會“計算機結構”－硬體（ Hardware ）；但是對計算機結構的概略認識，仍然有助於程式語言的學習。圖 1-1就是計算機最簡單的結構。

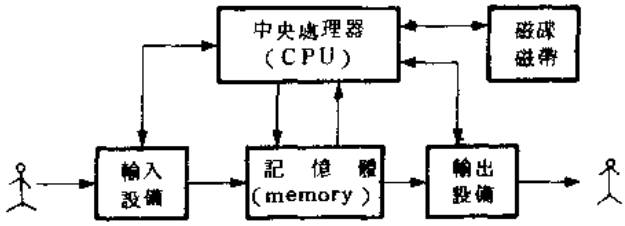


圖 1.1 計算機之組成

一部實實在在使用中的計算機並不一定如圖 1-1所示的組成。但是站在初學程式語言的立場，了解圖 1-1各部份的功能就足夠了。請看圖 1-1，一部計算機的組成包括中央處理器（ Centeral Processing Unit，

CPU），記憶體（ memory ），輸入設備（ Input devices ），輸出設備（ Output devices ）及磁碟或磁帶（ disk or magnetic tape）。下面各節中，我們將分別對這些設備加以簡單的介紹。

1.4 中央處理器(CPU)

中央處理器（以下以 CPU 簡稱之）之對於計算機的地位，猶如人體中的頭腦一樣。它是整體的一個號令中樞；所有的機器動作均需由 CPU 發出命令方可達成。那麼 CPU 又是什麼所組成的呢？這個問題暫不作詳細的說明，讀者若是有興趣的話，可以自行參考坊間很多介紹計算機結構的書籍，相信它們能給你需要的答覆，本書無法多作說明。

一般而言，當我們在設計高級語言的程式時，我們並不在意 CPU 的功能，甚至可以忽略它的存在。因爲只要我們照著語言文法的規定來設計程式，機器本身自然會依據你的程式，算出你所想要的結果。只要結果是正確的，也正是你所需要的，至於機器是如何得到結果的，我們似乎可以不必太關心它。雖然從“硬體”的立場而言，CPU 是最重要的一部份，但是在本書的範圍內，我們可以暫且忽視它的存在。

1.5 記憶體(Momory)

記憶體是計算機中僅次於 CPU 的重要硬體。顧名思義，記憶體就是記憶資料的物體；更恰當地說，記憶體的功用是用來儲存 CPU 工作的步驟，及其運算過程中所需要的數據，以及所產生的數據。

記憶體構成之最小單位是位元（以後以 bit 簡稱之）。一個 bit 可以視爲二進制的一個位數；亦即只可以是 0 或 1 。通常 8 個 bit 組成一個位元組（以後以 byte 簡稱之），而由幾個 byte 才能組成一個字（ word ）。因此一個字的位元數常爲 8 之整倍數。一般來講，計算機中一個字的位元數之多寡，通常可能決定這部計算機的能力與價格。現今流行的微電腦，其一字之位元數常爲 8 或 16 bit 。而俗稱之迷你電腦（ minicomputer ）多半是 16 bit 至 32 bit 。至於大型電腦（ main-frame computer）則常爲 28 bit 以上。

記憶體中的每個byte 或word 都有其地址（address）。這種地址就猶如郵局出租的信箱號碼，可用來區分不同的位置；同時每個位置的容量大小都相同。試想，如果要你到郵局去，把3473，2589，1051及4402等信箱中的信件拿回來；請問你，除非你拿張紙抄下這些號碼，否則恐怕你是不容易記住的。但是若郵局的信箱是以人的名字來區分的話，則要你去拿回張三、李四、王五及趙六等的信件時，大概你就容易記些了！同樣的道理，當我們設計一個程式要計算機執行時，若程式中必須寫出每個數據想要佔的記憶體位置時；尤其當數據的數量又很龐大時，恐怕在你尚未完成一個程式之前你就先放棄了。因此在高級語言的程式中，我們都是用名字來代表記憶體中的位置，而不使用眞正的地址。

往後我們討論程式設計時，我們最應注意的硬體就是記憶體。譬如，我們要計算機做個簡單的加法；工作是把X與Y值加在一起，將它的結果放在Z裡頭。也許X值在記憶體中的地址是1000，Y值的地址是2000，而Z的地址是500。機器需要做的動作是把1000及2000位置裡的數據拿出來加在一起，然後放在500的位置中，我們如果用Pascal 語言將上述動作寫出來的話，是：

```
Z：＝X＋Y；
```

像這種寫法，機器怎會知道X，Y，Z各別的地址呢？稍後在1-8節將有詳細一點的解釋。本節所要強調的是，在程式中我們只關心到記憶體中的資料的操作與結果。至於CPU如何操作我們不管，同時資料究竟放在記憶體的那個地址我們也不管；反正答案對了就好。

1.6 磁碟與磁帶

在計算機裡頭，記憶體所存的數據是CPU每一個運算步驟中能直接取用的。因此記憶體有時又被稱爲“第一級儲存”（Primary Storge）。雖然計算機直接取用的資料都來自記憶體，但是由於記憶體的價格昂貴，也由於記憶體的地址有限，因此一部計算機的記憶體容量是不會太大的。隨著機器大小的變化，概略從1k bytes（1千位元組）到10M bytes（1千萬位元組）左右。當計算機所需的資料大於記憶體的容量時，我們就