第二版

pascal

使用手冊與報告

賴新民　譯

USER MANUAL AND REPORT

SECOND EDITION

Kathleen Jensen
Niklaus Wirth

使用手册與報告

賴新民　譯

雲陽出版社印行

*PASCAL*使用手冊與報告

0001B

211

作　者　賴新民
出版者　雲陽出版社
台北市光復南路17巷46號
台北郵政信箱36-60
7629705　7610482
登記證局版台業字第0908號
發行人　陳文惠
台北市光復南路17巷46號
7629705　7610482
印刷者　進利印刷廠
台北市東園街260巷25號
3071088
基　價　貳圓捌角
版　次　中華民國71年1月初版
中華民國72年8月版

序言

「科技中文化」是國人日暮企求的願望。惟限於中西文字、作法的差異，譯著之書往往無法善盡達意。譯者有鑑於此，乃將文中生澀之處，略加潤飾，裨儘量符合辭通意達之要求。但基於「信、雅、達」起見，不便增減其內容。

計算機應用之廣與作業之效，自不待言。究其因除得力於週邊介面裝置的完善外，軟體程式是主要的功臣。*PASCAL*語言，以其結構化的型態易於觀測檢視，同時富於變化的資料結構更是處理任何問題的重要利器。在美國已普及至教育工商機構，國內方值提倡之初，爲啟發科技新知計實有大力推展之必要。

研習程式之道，不外乎多思考、多練習和多上機。本書是*PASCAL*之創始人——魏斯教授的手稿，更是一種公認的標準。其它類家皆以此爲依歸，欲考其緣由、統一差異，自應以此書爲準。文中以*BNF*（*Backus - Naur Form*）公式描述*PASCAL*的語法，層層解說，抽絲剝繭似的剖析，並附以圖示說明（語法圖），未嘗不是一種特色。

「工欲善其事，必先利其器」。程式已發展成爲工程的實筏，應用的工具。*PASCAL*以其使用廣範之利，勢將成爲今後程式語言的主流。

本書譯著，初次嘗試，對文下筆，每至戰戰競競。幸而摯友張君龍翔、李君文欽與林君振興，在旁指導，多方鼓勵協助，始得以脫稿。同時，雲陽出版社胡社長秉其溝通中西文學的善意，予以出版，在此一併致謝之。譯者才疏學淺，錯誤謬漏之處在所難免，尙祈先進賢達者，不吝指正，甚感欣獲。

譯者　賴新民　謹識

民國七十年九月於彰化

前言

程式語言 *Pascal* 的頭一篇譯本於1968年開始起草，它承襲著 *Algol*-60 的風格。經過多方面的發展階段，第一部可運算的編譯器誕生於 1970年， 並於一年後發表（參閱參考文獻1 和8 ）。由於參與其它類型計算機編譯器的發展，日漸成長的興趣和兩年使用此種語言的經驗，口述一些修訂之處，於是乃有 1973 年修訂報告與以 *ISO* 文字組爲據之語言表示法定義的出版。

本小册由兩部份組成：使用手册和修訂報告。手册方面提供給熟悉計算機程式者，進一步通曉 *Pascal* 語言，由是手册使用教授式的格式，並輔以許多例子說明 *Pascal* 的各種特徵，且附錄綜合表和語法規則。報告部份則給與程式計劃師和執行者、簡明與基本的參考，它定義了對於此種語言，由各種執行情況彼此間共同基礎所組成的標準 *Pascal* 。

對於介紹一種語言，線性結構的一本書並非是理想的作法。但它基於教授式，當瀏覽此手册的組織時，我們建議讀者謹愼小心於程式例，並對於疑問之處再加詳讀，特別是第十二章的輸入和輸出共同協議。

手册中的 0 — 12 章與整個報告係敍述標準 *Pascal* ，一位執行者應將認可標準 *Pascal* 的工作視爲其系統的基本需求，而程式計劃師欲將其程式自一部計算機轉送至另外一部，僅需以標準 *Pascal* 描述特徵，即可達成。當然個別的執行情況可能提供額外的設備，他們均視爲標準 *Pascal* 的進一步延伸。 13 章和 14 章是 *Pascal* 於 *Control Data* 6000 型計算機上的執行說明，其中 13章描述了此語言的額外特徵，稱爲 *Pascal* 6000 — 3.4 。而 14 章是於作業系統 *SCOPE* 3.4 中編譯器的使用。

本手册的確已花了不少的心血，在此我要特別感謝蘇黎士*ETH. Institut fuer Informatik* 的同仁和 *John Larmouth*, *Rudy Schild*, *Olivier Lecarme*, 與 *Pierre Desjardins* 的審定、建議和鼓勵。同時*Helmut Sandmayr* 協助*Urs Ammann*，使程式得以執行，無他兩贊助本手册無以付梓，特此致謝。

卡斯林　强生
尼可斯　魏斯
瑞士　蘇黎士
ETH　1974年十一月

目　　錄

第 13 章 *PASCAL* 6000-3.4

第 14 章 如何使用 *PASCAL* 6000-3.4 系統

第 二 篇 報 告

第 0 章

緒論
(INTRODUCTION)

以下諸文假設讀者對計算機術語已有基本上的了解且對程式的結構有所認識。此段之旨乃在於啟發讀者直覺上的興趣。

{ 程式　0.1

假設每年的通貨膨脹率是每分 7、8 和 10，找出在 1，2，…… *n* 年內某些貨幣如法朗、銀元、磅、*stg*、馬克或基爾德等貶值後的係數。 }

```
program inflation(outout);
const n=1Ø;
var i : integer; w1,w2,w3 : real;
begin i := Ø; w1 := 1.Ø; w2 := 1.Ø; w3 := 1.Ø;
   repeat i := i+1;
      w1 := w1 * 1.Ø7;
      w2 := w2 * 1.Ø8;
      w3 := w3 * 1.1Ø;
      writeln(i,w1,w2,w3)
   until i=n
end.
```

```
 1 1.Ø7ØØØØØØØØØØØe+ØØ 1.Ø8ØØØØØØØØØØØe+ØØ 1.1ØØØØØØØØØØØe+ØØ
 2 1.1449ØØØØØØØØØe+ØØ 1.1664ØØØØØØØØØe+ØØ 1.21ØØØØØØØØØØe+ØØ
 3 1.225Ø43ØØØØØØe+ØØ 1.259712ØØØØØØe+ØØ 1.331ØØØØØØØØØe+ØØ
 4 1.31Ø796Ø1ØØØØe+ØØ 1.36Ø48896ØØØØe+ØØ 1.4641ØØØØØØØØe+ØØ
 5 1.4Ø255173Ø7ØØe+ØØ 1.469328Ø768ØØe+ØØ 1.61Ø51ØØØØØØØe+ØØ
 6 1.5ØØ73Ø351849e+ØØ 1.586874322944e+ØØ 1.771561ØØØØØe+ØØ
 7 1.6Ø5781476478e+ØØ 1.713824268779e+ØØ 1.9487171ØØØØe+ØØ
 8 1.718186179832e+ØØ 1.85Ø93Ø21Ø282e+ØØ 2.14358881ØØØe+ØØ
 9 1.83845921242Øe+ØØ 1.999ØØ46271Ø4e+ØØ 2.357947691ØØe+ØØ
1Ø 1.967151357290e+ØØ 2.158924997273e+ØØ 2.59374246Ø1ØØe+ØØ
```

一份計算機程式或演算法（*algorithm*）是由兩個基本的部份所組成，其一爲執行動作的描述，而另一部份爲動作中所使用到的資料。描述動作的語句稱爲指述（*statements*）；相對地，描述資料的爲宣稱（*declarations*）和定義（*definitions*）。

PASCAL 程式整文稱爲一個程式域（*block*），可分爲標題區（*heading*）和主體。標題係給程式一名稱並列出它的參數（如檔案變數，代表的幅數（*argument*）和計算的結果，詳細情形請參閱十三章）。另外檔案"*output*"是一必須的參數。程式域由六段所組成，其中除最後一段外其餘皆可空白。他們之間的次序是：

＜標號宣稱部＞
＜常數定義部＞
＜型態定義部＞
＜變數宣稱部＞
＜程序程式和函數宣稱部＞
＜指述部＞

第一段列出程式域中定義的所有標號，其次一段定義常數同義字（*synonyms*）。註：同義字是以一識別字（*identifier*）來取代某些常數，而爲程式往後所用者。再其次分別爲型態定義、變數定義和副常規程式部（註：程序程式與函數），最後指述部則說明了須要的動作。

以上程式的描繪我們現在可以語法圖（*syntax diagram*）將它更明白的表示出。語法圖的起點是寫好名稱的程式，隨後有一條穿梭圖中的路徑它是用來定義正確的語句程式。圖中吾人藉著名稱的自定義來參考其中的某一盒（*box*），終結符號（實際在程式內所用到的）亦在程式域所圍繞的範圍內（請參閱附錄*D*全部的語法圖）。

表示程式語法另有一種方式是使用傳統的 *Backus-Naur Form*，在 *BNF* 符號中語句結構是由封閉在角括號＜和＞內的英文字所表示，這些字通常是選擇一些較接近被描述結構的含意和性質者。除此以外爲大括號｛和｝所封閉的一串結構係意味著它可重覆零次

或許多次。（*PASCAL* 的 *BNF* 符號請參閱附錄 *D* 。）例如圖 0.(a) 的＜程式＞結構，此處以下列的式子表示稱爲 *BNF* 的結果（*productions*）：

＜程式＞::=＜程式標題區＞＜程式域＞

＜程式標題區＞::=*program* ＜識別字＞（＜檔案識別字＞{，＜檔案識別字＞}）；

＜檔案識別字＞::=＜識別字＞

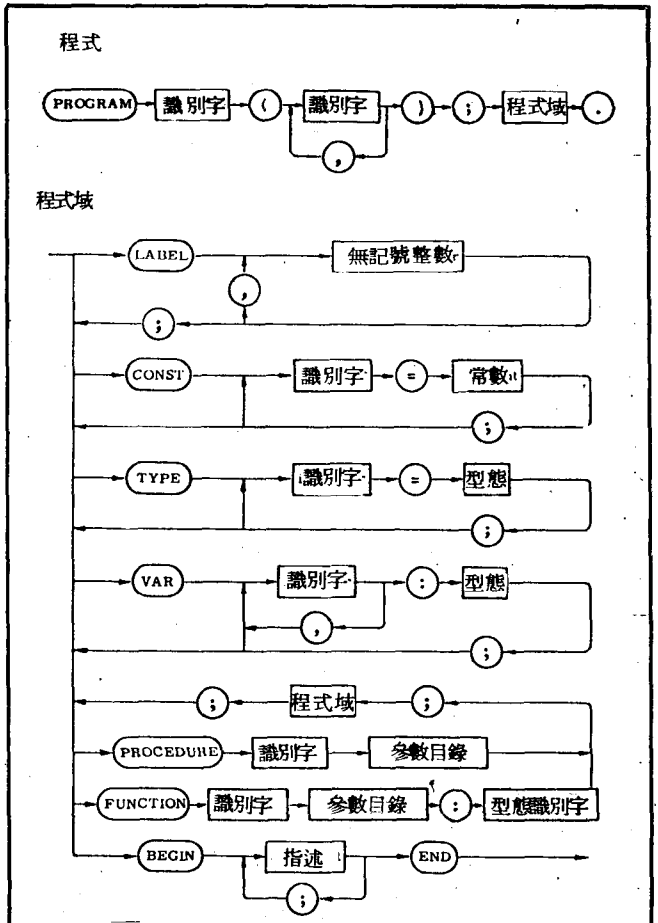
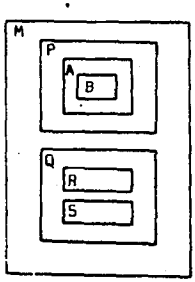


圖 0-*a*　通常程式結構定義的語法圖

程序程式（函數）有類似於主程式的結構，註：同樣的有一標題和主題。因此，程序程式可以在其他的程序程式中宣佈（巢狀結構）。標號、常數、同義字、型態、變數和程序程式宣稱對於宣稱它們的程序程式言皆是本地性的（*local*）。換句話說，組成程序程式宣稱部的識別字僅有在此主文內始有意義。此種現象稱爲識別字的領域範圍（*scope*）。因爲程序程式可聚成巢狀，觀測的問題可能就應運而生。另一類識別字係在主程式中所宣佈的，因它于整個程式內皆是有意義的稱爲整體的（*global*）。

因爲藉著程序程式和函數的宣稱，某一程式域可被聚集在其它的程式域內。因此吾人能以階層（*level*）來描述一巢狀結構。假設我們指定最外層主程式域的階層爲0，那麼在此程式域內定義的程式域，其階層將爲1。通常一個在階層爲 i 的程式域內所定義的程式域，其階層將變爲（$i+1$）。圖 0.(b) 舉例說明一程式域結構。



階層 $0 = M$
階層 $1 = P, Q$
階層 $2 = A, R, S$
階層 $3 = B$

圖 0-*b* 程式域

以公式化而言，一個識別字 X 合理的領域或範圍是位於定義 X 的整個程式域內，這些包含與 X 相同域內所定義的程式域。（對此例子而言，所有識別字必須不同。第三章中將討論識別字可相同之情況。）

在程式域中定義的主題	在程式域中可接觸者
M	*M*, *P*, *A*, *B*, *Q*, *R*, *S*
P	*P*, *A*, *B*
A	*A*, *B*
B	*B*
Q	*Q*, *R*, *S*
R	*R*
S	*S*

對熟悉 *ALGOL*、*PL/I* 或 *FORTRAN* 的程式計劃師，他可能藉著這些語言而更確信對 *PASCAL* 程式的瀏覽。爲此目的以下我們列出 *PASCAL* 的特性：

1. 變數的宣稱係强制性的。
2. 某些關鍵字（如 *begin*、*end*、*repeat*）是保留字，在程式段中不能當識別字用，本手册中將以底線區別之。
3. 半支點（*semicolon*）（；）係被視爲指述的分離符號，而非終結符號（如 *PL/I* 中是之）。
4. 標準的資料型態有整數、實數、邏輯數值和那些可印出的字元。*PASCAL* 提供的基本資料結構設備有數列（*array*）、錄（*record*）（相對於 *COBOL* 和 *PL/I* 的結構）、集合（*set*）和循序檔案（*sequential file*）。使用這些結構我們能將它組合聚集成集合的數列（*arrays of sets*）或錄的檔案（*files of records*）等。資料可以動態的型式配置也可經由指標（*pointer*）來接觸，即因此種指標而使得串列處理（*list processing*）更爲普遍。除此以外另有一種設備是以符號常數宣稱新的、基本的資料型態。
5. 集合資料結構所提供的設備近似於 *PL/I* 中的位元字串（*bit string*）。
6. 數列可能是任意界限的任意因次（*dimension*）且此界限需爲常數（註：無動態的數列）。
7. 一如 *FORTRAN*，*ALGOL* 和 *PL/I* 有一條 *GOTO* 指述。