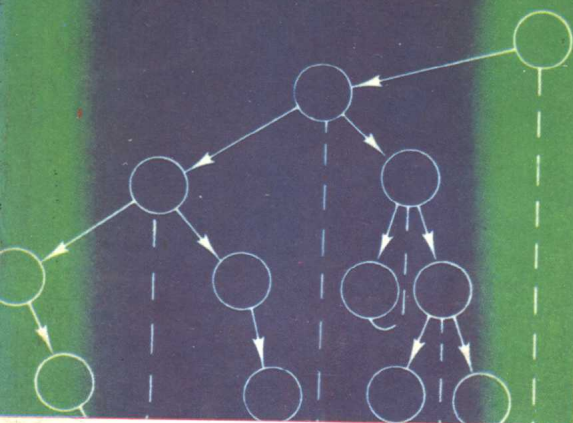


資料結構-PASCAL

Data Structures Using Pascal

Aaron M. Tenenbaum
Moshe J. Augenstein

葉耀明譯



```
procedure intrav4 (tree: nodeptr);  
var p, q: nodeptr;  
begin  
    p := tree;  
    repeat  
        q := nil;  
        while p <> nil { traverse left branch }  
            do begin  
                q := p;  
                p := p↑.left;  
            end;  
        if q <= nil  
            then begin  
                writeln(q↑.info);  
                p := q↑.right;  
                while q↑.rthread < back up  
                    do begin  
                        writeln(p↑.info);  
                        q := p;  
                        p := q↑.right;  
                    end; while do begin  
                        end { then begin }  
    until q = nil
```

資料結構—PASCAL

葉耀明譯

香港科技出版社

資料結構 — PASCAL

編譯者：葉 耀 明

出版者：香港科技出版社

發行者：香港科技出版社

九龍彩虹道810號六樓

印刷者：永達印刷公司

香港黃竹坑建明工業大廈九樓D座

定 價：H. K. \$ 35.00

原 序

本書專為兩學期的資料結構及程式設計課程所編定的。累積數年的資料結構授課經驗，我們深深感受到，雖同學們已修過高階程式語言與組合語言，修資料結構的課時，對程式設計往往沒有足夠的概念，無法將理論上的抽象結構適當地應用到程式設計。因此導致我們在授課之時必須投注相當心血於程式設計技巧之講解，反應快的同學尚能抓住重點，但反應較差的學生往往仍是一頭霧水，摸不著邊。有感於此，我們認為上完一學期資料結構之後，必須接著再上一學期程式設計，此書即為此種信念下的產物。

本書的特性為先介紹抽象的觀念，指出這些觀念在解決實際問題之效用，然後說明如何以程式語言將這些觀念具體化。其特色為理論的抽象概念與具體的程式製作都同等重視。因此同學可同時學到理論觀念，其製作技巧及應用方向。

本書特別選用 PASCAL 語言，因為 PASCAL 語言本身的程式流程控制結構可讀性高，且其提供的資料型態允許基本的資料結構，像堆疊，連索串列，樹等，可用多種方式來製作。此點可使同學在面對實際問題時，能夠權衡輕重，作最佳的抉擇。使用本書的唯一要求，就是要修過一個學期的高階程式語言的課。若以前只學過 FORTRAN 或 PL/1，可將此書與書末所列之參考書目合用，第一章及附錄也提供些許 PASCAL 語言的概念，使這些同學能跟上進度。

第一章為資料結構導論。§ 1.1 節闡述抽象的資料結構概念及實際製作之概念。§ 1.2 節與 § 1.3 節介紹 PASCAL 語言的陣列

及錄，亦討論了此二種資料結構的製作技巧及應用。

第二章討論堆疊及其 PASCAL 的製作方式，開始進入資料結構的主題，因此我們以極大篇幅來討論製作此結構可能遭遇的困難。§ 2.3 節介紹後序 (postfix)，前序 (prefix)，中序 (infix) 表示法。

第三章包括遞迴 (recursion) 的應用與作法。第四章討論佇列 (queue) 與連索串列 (link list) 及其利用陣列節點的製作方式。

第五章討論 PASCAL 語言的不定存量 (dynamic storage)，以及利用不定存量製作佇列，堆疊和連索串列的方法。

第六章討論樹狀結構，第七章介紹圖形結構，第八章介紹分類技術，第九章討論搜尋方法。

附錄討論 PASCAL 語言與資料結構相關的課題。可能有些同學曾在 PASCAL 的課程上接觸過，有些同學卻沒有。因此教師可將附錄指定為自修的教材，或將其溶入課程之中。

在本書末尾有許多參考書目，每一項目皆以標題，章節加以標明。

若是一學期的資料結構課程，可包括 §1.1 節，第二～六章，第八、九章，但如果時間有限，可略去第三，六，八，九章。

本書教材適合於 ACM 於 1978 所提倡的大學資訊教育課程計劃的教材之用。

書內的演算法以 PASCAL 程式加入英文描述來表示。使讀者閱讀演算法之時，可專心致力於解決問題的方法上，而不用理會變數的宣告，或程式語言文法上的怪異特性。在將這些演算過程轉變為程式時，我們會特別指出其來龍去脈與該小心的要點。

PASCAL 程式的鋸齒狀格式與演算流程的表達是根據 Peterson 於 1977 年所提出的概念 (SIGPLAN Notices)。我們使用關鍵字 end 作為指令群的結束，配合鋸齒狀格式，使程式的可讀性大為提高。並以斜體字表示演算法，正楷字表示程式，來區分出二者的不同。

書中大部分的觀念都以舉例方式加以說明，其中某些例子本身亦自成一重要主題（譬如，後序表示法，長整數算術等等）。有些例子則

提出不同的製作方法，（例如，樹的循序存法與不定純量），教師可從其中挑取教材，這些例子也可指定為學生的自修讀物。預料教師不可能在短短一、二學期內將所有的例子詳細剖析，因此站在教育的立場來看，詳細剖析其中幾個重要例子，要比草率地瀏覽所有例子，來得有效得多。

書中所有程式皆經過上機測試與除錯，應特別感謝Kai Ming Lee與Allen J. Schreier 對此工作的貢獻，他們所投入的工作熱忱與卓絕建議，已遠超過其份內的工作。當然，若書中尚有任何錯誤，都應是作者的責任。

習題的型態與難易程度涵蓋極廣，有些可讓你深一層了解課文的概念，有些則將書中的程式或演法過程略加變化，有些則是相當富有挑戰性的新觀念。而且有些一系列的連鎖習題涵蓋了整個論題的發展過程，可當作學術演講或是整個學期的研究計劃的題目。教師指定作業時，應配合學生程度，我們建議每一學期可規定學生5到12個程式設計（視難易度而定）。教師可在我們先前的著作——Data Structure and PL/1 Programming——中發現許多額外的習題及研究計劃，雖然其中的程式是以PL/1語言撰寫的，但其很容易轉換成PASCAL形式。

我們特別引用PASCAL Manual and Report與BSI/ISO 第三版二者所提倡的標準PASCAL來設計程式。儘量避免此二者的相異特性，若迫不得已無法避免，則明顯標明並在嚴密的假設下使用此特性。（二者最明顯的差別為WRITE程序所使用的字串包裝型陣列的註標參數使用）

當然，教師於課程上，應先告訴同學們上機所使用的PASCAL版本。在本書編輯之時，又有新的PASCAL標準提出，其中額外涵蓋了在執行之時才決定陣列之上下限的特性。我們尚無法預測何時或是否會被採用於標準PASCAL中，或是何時可產生包含此種特性的編譯器。基於此點，我們將目前提出的特性置於附錄第5節中討論，並標明適用此特性的程式範例。

感謝 Itchy Goldbrenner, Simon Krischer 和 Carl Markowitz 在原稿的打字及校對上的貢獻。並特別感激 Carl Markowitz 和 Allen J. Schreier 對本書的出版所貫注的熱忱，嘉許 Kai Ming Lee 的有始有終與奉獻，並感謝 J. Barone, D. Bieber, A. Ciappina, J. Davis, R. Gerard, J. Goldfiuger, W. Lee, G. Markowitz, D. Rybstein, G. Schechter, R. Teich, D. Zaslowsky 和 B. Zusin 等人極有價值的協助。

特別感謝紐約市立大學電腦中心與布魯克林學院電腦中心所給予的支援。

Aaron Tenenbaum

Moshe Augenstein

目 錄

第一章 資料結構導論

1.1 資訊及其意義	1-1
1.1.1 二進位和十進位整數	1-2
1.1.2 實 數	1-5
1.1.3 字元字串	1-6
1.1.4 硬體和軟體	1-7
1.1.5 製作的概念	1-8
1.1.6 實 例	1-9
1.1.7 PASCAL 之資料型態	1-14
1.2 PASCAL 之陣列與集合	1-16
1.2.1 製作一維陣列	1-19
1.2.2 字 串	1-22
1.2.3 字串運算	1-24
1.2.4 不定大小的陣列與 PASCAL	1-27
1.2.5 二維陣列與多維陣列	1-28
1.2.6 PASCAL 之集合	1-35
1.2.7 集合之運算	1-36
1.2.8 集合之製作	1-38
1.2.9 從集合中選取元素	1-40
1.3 錄	1-46
1.3.1 字串與錄	1-47
1.3.2 錄之製作	1-50

1.3.3	不定錄	1-53
1.3.4	不定錄之製作	1-56

第二章 堆疊

2.1	定義及實例	2-1
2.1.1	基本運算	2-5
2.1.2	實例	2-6
2.2	PASCAL 之堆疊表示法	2-11
2.2.1	例外情形的測試	2-15
2.2.2	PUSH 運算的製作	2-16
2.2.3	不同型態的堆疊	2-19
2.3	實例：前序，中序，後序	2-22
2.3.1	基本定義與實例	2-22
2.3.2	後序運算式的計算求值	2-25
2.3.3	後序運算式的求值程式	2-26
2.3.4	程式的限制	2-30
2.3.5	中序至後序的運算式轉換	2-31
2.3.6	從中序轉換成後序的程式	2-36

第三章 遞迴

3.1	遞迴的定義與過程	3-1
3.1.1	階乘函數	3-1
3.1.2	自然數的乘法	3-5
3.1.3	費伯那西序列	3-5
3.1.4	二元搜尋法	3-7
3.1.5	遞迴定義和演算法的性質	3-10
3.2	PASCAL 中的遞迴表示式	3-12
3.2.1	PASCAL 中的階乘運算	3-12

3.2.2	PASCAL 與費伯那西數	3-17
3.2.3	PASCAL 與二元搜尋法	3-19
3.2.4	遞迴鏈	3-22
3.2.5	代數式的遞迴定義	3-23
3.3	設計遞迴程式	3-30
3.3.1	漢諾塔問題 (Tower of Hanoi)	3-32
3.3.2	以遞迴方式將前序式轉換成後序式	3-37
3.4	模擬遞迴	3-44
3.4.1	由副程式回轉	3-46
3.4.2	製作遞迴常式	3-47
3.4.3	階乘的模擬	3-48
3.4.4	改善模擬常式	3-53
3.4.5	消除 Go To	3-56
3.4.6	模擬漢諾塔	3-58
3.5	遞迴的效率問題	3-66

第四章 佇列和串列

4.1	佇列與其順序表法	4-1
4.1.1	插入運算	4-8
4.2	連索串列	4-11
4.2.1	由串列插入，除去節點	4-12
4.2.2	堆疊的連索製作	4-16
4.2.3	getrode 和 freenode 運算	4-17
4.2.4	佇列的連索製作	4-20
4.2.5	以連索串列當資料結構	4-21
4.2.6	串列運算的例子	4-24
4.2.7	PASCAL 中的串列	4-25
4.2.8	PASCAL 中以串列表示佇列	4-29

4.2.9	PASCAL 中的串列運算例子	4-30
4.2.10	非整數串列	4-31
4.2.11	首節點	4-32
4.3	以連索串列模擬	4-37
4.4	其他串列結構	4-47
4.4.1	循環串列	4-47
4.4.2	以循環串列作堆疊	4-47
4.4.3	以循環串列製作佇列	4-49
4.4.4	循環串列的基本運算	4-50
4.4.5	約瑟夫 (Josephus) 問題	4-51
4.4.6	首節點	4-54
4.4.7	使用循環串列作長整數的加法	4-54
4.4.8	雙連索串列	4-57
4.4.9	使用雙連索串列作長整數加法	4-59

第五章 串列處理

5.1	不定存量處理串列資料	5-1
5.1.1	不定存量的必要性	5-1
5.1.2	PASCAL 的指標	5-2
5.1.3	設定及解除不定變數	5-3
5.1.4	不定變數處理連索串列	5-7
5.1.5	串列操作的不定處理	5-9
5.1.6	不定錄的取用及送回	5-11
5.1.7	陣列和不定處理串列資料的比較	5-12
5.2	實例：飛機訂位系統	5-13
5.3	實例：空曠矩陣	5-20

第六章 樹狀結構

6.1	二元樹	6-1
6.1.1	二元樹之應用	6-4
6.2	二元樹表示法	6-12
6.2.1	PASCAL 之二元樹巡行	6-15
6.2.2	綫穿二元樹	6-17
6.2.3	異質二元樹	6-21
6.3	其他應用—赫夫曼演算法	6-25
6.4	約瑟夫問題之再討論	6-35
6.5	樹及其應用	6-44
6.5.1	樹的 PASCAL 表示法	6-46
6.5.2	樹之巡行	6-48
6.5.3	陳式之樹表示法	6-51
6.5.4	樹的其他操作	6-56
6.6	實例：遊戲樹	6-60

第七章 圖形結構及其反應

7.1	圖 形	7-1
7.1.1	圖形應用實例	7-5
7.1.2	圖形之 PASCAL 表示	7-8
7.1.3	遮移封閉性	7-10
7.1.4	華雪爾演算法	7-14
7.2	匯流問題	7-19
7.2.1	改進匯流問題	7-22
7.2.2	例 子	7-27
7.2.3	演算法與程式	7-29
7.3	圖形的鏈結表示	7-39

7.3.1	規劃問題	7-44
7.3.2	PASCAL 程式	7-49
7.3.3	改進程式	7-53

第八章 分類

8.1	通論	8-1
8.1.1	效率考慮	8-4
8.2	更換分類技術	8-11
8.2.1	漂浮分類法	8-11
8.2.2	快速分類法	8-15
8.2.3	改進程式	8-21
8.3	選擇分類與樹狀分類法	8-26
8.3.1	直接選擇分類法	8-26
8.3.2	二元樹分類法	8-28
8.3.3	對抗分類法	8-30
8.3.4	錐形分類法	8-39
8.4	插入分類技術	8-48
8.4.1	簡單插入分類法	8-48
8.4.2	薛爾分類法	8-50
8.4.3	位址計算分類法	8-54
8.5	合併與基數分類技術	8-60
8.5.1	合併分類法	8-60
8.5.2	基數分類法	8-64

第九章 搜尋

9.1	基本搜尋技術	9-1
9.1.1	演算法表示形式	9-3
9.1.2	循序搜尋	9-4

9.1.3	循序搜尋的效率	9-6
9.1.4	重排串列以提高搜尋效率	9-7
9.1.5	搜尋有序列表	9-9
9.1.6	註標循序搜尋	9-10
9.1.7	二元搜尋	9-14
9.2	樹狀搜尋	9-19
9.2.1	二元搜尋樹的插入運算	9-20
9.2.2	二元搜尋樹的刪除運算	9-23
9.2.3	二元搜尋法的效率	9-26
9.2.4	平衡樹	9-29
9.2.5	數位搜尋樹	9-39
9.2.6	矩陣樹	9-43
9.3	雜湊法	9-53
9.3.1	以開放位址法解決雜湊碰撞	9-55
9.3.2	以鏈結法解決雜湊碰撞	9-59
9.3.3	雜湊函數的選擇	9-62
9.4	應用及實例	9-65
9.4.1	赫夫曼演算法	9-65
9.4.2	規劃問題	9-71
9.4.3	飛機訂位系統	9-77

附錄 PASCAL 的結構

1.	PASCAL 之純量資料型態	附-1
2.	一維陣列	附-4
3.	包裝型陣列	附-6
4.	陣列參數	附-8
5.	數值參數與變數參數	附-11
6.	PASCAL 的錄	附-15

資料結構使用 PASCAL 語言

7. 錄陣列	附 -20
8. 識別字的有效範圍	附 -22
9. WITH 敘述	附 -27
10. 取用錄之效率問題	附 -31
11. 包裝之錄	附 -34
12. 錄的檔案	附 -35

索 引

第一章 資料結構導論

計算機 (computer) 是一種處理資訊 (information) 的機器。計算機科學的研究包括三方面：(1) 計算機內的資訊如何組織，(2) 資訊如何被處理，(3) 資訊如何被利用。因此想在計算機科學領域繼續研究者，瞭解資訊結構及其運作的概念是很重要的。

1.1 資訊及其意義 (Information and Meaning)

假如計算機科學的基礎是資訊研究，那麼首先遇到的問題是：何謂資訊？不幸地，雖然資訊概念是整個領域的基礎，但這個問題却無法明確地被回答。計算機科學的資訊概念就像幾何學中的點、線、面的概念一樣——它們是無定義名詞，可略加說明，但不能用其他更基本的概念來定義它們。

在幾何學裡，線是無定義名詞，但要談線的長度是可以的，線的長度是一種數量的測度。同樣地，在計算機科學裡，我們也可以測度資訊的數量。資訊的基本單位是位元或稱數元 (bit)，它的值一定是兩個互斥可能情形之一面，例如電燈開關，不是“開”，便“關”，決不能同時是“開”又是“關”。若一個裝置能表示兩種以上的狀態 (states)，事實上，它就是用了超過一個位元的一種特定裝置，例如一個指針盤 (dial) 有 8 個可能狀態，事實上它用了 3 個位元。

這種現象可用下列想法：假設我們只有一種兩路（Two-way）開關，要用多少個就有多少個。很明顯地，一個開關能表示二種狀態（如圖 1.1.1a），兩個能表示 4 種狀態（圖 1.1.1b），三個就可表示我們所需要的 8 種狀態（圖 1.1.1c）。通常 n 個開關能表示 2^n 種不同的狀態。

二進位數字 0 和 1 是用來表示一個特殊位元的二種可能狀態。（其實，“bit”是“binary digit”的縮寫字）。 n 個位元就用 n 個 0 或 1 來表示。例如字串 101011 表示 6 個開關，第一個是 1 代表“開”，第二個是 0 代表“關”，第三個是“開”，第四個是“關”，第五和第六個是“開”。

因此我們可看出，3 個位元足以表示 8 個狀態（即 000、001、010、011、100、101、110、111）。這可代表整數 0 到 7。本來位元安排並不代表特殊的整數，只是任何一種位元安排只能代表一個整數不能同時代表兩個整數，但爲了統一起見，我們給予特定“指定法如下面幾節所示。

1.1.1 二進位和十進位整數 (Binary and Decimal Integers)

位元安排最廣泛的用法是用來表示“非負整數”（Nonnegative Integer）叫做“二進位數字系統”（binary number system）。在這個系統，每個位元代表 2 的次方，最右的位元位代表 2^0 （等於 1），再左一位代表 2^1 （相當於 2），再來是 2^2 （等於 4），依此類推。每一個整數是用 2 的次方的和來表示，全是 0 所成的位元串表示整數 0，若某一位元位出現 1 則其代表的 2 的次方加入總和，出現 0 則不加入總和。例如數元集合 00100110，其中 1 出現於第 1、2、5 位（從右到左計算，最右邊的稱第 0 位），則代表的整數爲 $2^1 + 2^2 + 2^5 = 2 + 4 + 32 = 38$ 。依照這種換算法，任何 n 個數元的數元串可代表唯一的非負整數，從 0 到 $2^n - 1$ 。反之，從 0 到 $2^n - 1$ 的任何非負整數可被唯一的 n 個數元的數元串來表示。