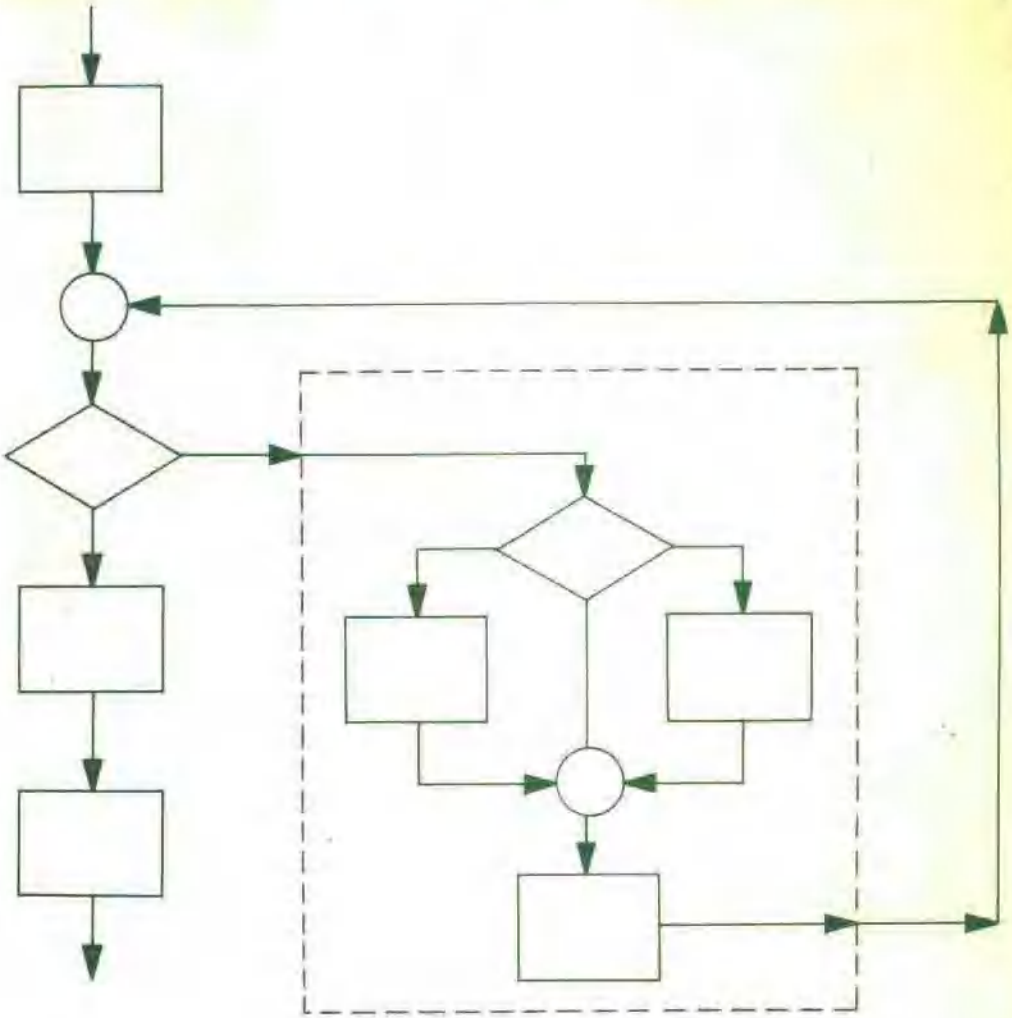
福傳五程式設計

ANSI FORTRAN PROGRAMMING



鄭守成 編著

松崗電腦圖書資料有限公司

福傳五程式設計

ANSI FORTRAN PROGRAMMING

第 二 版

鄭 守 成 編著

松崗電腦圖書資料有限公司 印行

福傳五程式設計

ANSI FORTRAN PROGRAMMING

每本定價 300 元整

書號：210124

編著者：鄭 守 成

發行人：吳 守 信

發行所：道明出版社

台北市仁愛路二段一一〇號三樓

總經銷：松崗電腦圖書資料有限公司

台北市仁愛路二段一一〇號三樓

電話：3930255、3930249

郵政劃撥：10930

印刷者：鵬森印刷有限公司

台北市民生東路一一一二號

電話：7608528

中華民國六十九年九月 初 版
中華民國七十年八月 第一版第二次印行
中華民國七十一年十月 第一版第三次印行
中華民國七十二年九月 第二版

本出版社經行政院新聞局核准登記，

登記證號為局版台業字第一七二九號

第一版序

爲什麼要寫這一本福傳課本呢？其理由有三：

1. 1977 年美國國家標準福傳（ANSI FORTRAN）使得現有的福傳課本變成過時的教材。
2. 最近幾年來在程式的型態和程式設計方法上的進步使得大部份現有的課本不適用於教學。
3. 過去福傳的教學偏重於個別陳述（statement）的介紹。我們認爲教學生用規律的、清楚的型式撰寫完整的程式是一種較好的教學方法。

本書是根據1977 年美國標準福傳編寫的。大部份新的編譯程式將會符合這個標準，但是很多小型電腦將僅只使用1977 年標準福傳的簡單版叫做福傳子集合。本書將介紹 1977 福傳的全部，但是不包括在福傳子集合裏的特性將會加以註明，並在附錄三裏加以彙總。

從 1966 年福傳第一次標準化到 1977 年新福傳標準被接納之間，在教學用的福傳編譯程式有很重大的改變。最有意義的進展是學生導向的三個版本叫做 WATFOR 、WATFIV 和 WATFIV-S ，是在 IBM 電腦上開發出來的。這些新福傳增加了好幾種很有用的特性。MNF 亦是一種學生導向的福傳使用於 CDC 的大型電腦上。 1977 年的標準福傳幾乎包括了WATFOR 、WATFIV和 MNF全部的新特性。 1977 年的福傳亦增加了一些 1966 年福傳所沒有的特性，但是幾乎完全沒有改變 1966 年的特性，換句話說，用 1966 年福傳所寫的程式在新標準下仍可使用。

雖然本書是根據 1977 年美國國家標準福傳編寫的，使用舊版福傳的人亦能夠很有效地使用本書。因爲本書第一章到第十章介紹福傳的基本特性，第十一章介紹檔案處理，第十三章介紹字符使用及其他特性。因爲 1977 年標準的主要改變或增加的項目是檔案處理和字符使用，影響新標準運用最多的地方只限於該兩章。使用舊版福傳的人，可以依下列方式使用本書。

WATFOR／WATFIV版—— 本書的第一到第十章和它沒有什麼差異，少數的例外列於附錄三。循序檔案的處理（第十一章）和第十三章大部份的特性，WATFOR／WATFIV 和 1977 年的標準福傳大都相同。

WATFIV-S 版——WATFIV-S 所介紹的六種結構化程式指令中，1977 年標準福傳只備有區段條件陳述。其他特性方面 WATFIV-S和WATFIV 相同。

MNF 版—— 本書前十章全部的特性都是最近版的 MNF 所允許的。

1966 年版標準福傳——使用本書前四章最大的問題是這一版的標準福傳缺少自由格式輸出入陳述，一定要使用格式陳述。第一章後面的附錄裏說明怎樣利用所需的格式陳述來解決這個問題。和新版標準福傳的其他差異請參閱附錄三。

其他版次的福傳——主要的困難是缺少自由格式輸出入陳述。有些福傳備有這種陳述，有的不具備這種陳述。如果不具備這種陳述可以參閱第一章末附錄的說明。

第十一、十二兩章的循序檔案處理方法和舊版福傳沒有什麼顯著的差異，而直接存取檔案却有很大的改變。使用舊版福傳的學生就必須參閱該電腦的福傳手册。第十三章新的字符指令是大部份舊版福傳所沒有的，所以必須詳細研讀該電腦的手册，怎樣處理字符資料。

本書的說明都是指程式和資料錄製在卡片上（附錄一說明如何使用製卡機）。不過也有很多學生使用終端機（terminal）來進行程式設計工作。附錄二說明怎樣在終端機上設計福傳程式。

本書要強調的是學生應該一開始就教他們使用良好的程式結構，有規則地撰寫完整的程式。利用結構化的、規則化的方式撰寫程式，將可以提高生產力和降低錯誤率，這種程式比較容易除錯、容易維護（修改）。本書不僅只教你福傳語言，而且使用範例程式解釋怎樣應用福傳陳述來撰寫清楚的、結構化的程式。

很多學生稍加指點就能夠學會程式語言，有的學生却需要教師加強指導，爲了兼顧這兩種學習方法，本書的文字力求淺顯，希望可以當作自修的教材，也很適合在職訓練之用。在需要講解的情況下本書也很適用。每兩章是一個單元，單數章說明語言的特性，教師可以依序說明並舉例示範該陳述的特性；偶數章裏有兩個範例程式和程式設計練習題，教師可以選擇一題指定爲程式設計練習題，撰寫完畢後上機實習。教師可以利用範例來講解程式設計的方法。

不同的福傳課程有不同的教學目標，從對這種重要語言的初步認識到訓練一個優秀的福傳程式人員。選用本書的一部份或全部可以達成相當廣泛的教學目標。要介紹福傳語言的基本構件以及福傳程式的結構，可以選用前八章（或前十章）。如果想要瞭解這種語言的全部能量就必須包括九到十四章。本書可以用做專門學習福傳語言的教材，也可以當做介紹電腦科學或電腦資料處理的電腦語言教科書。後一種目的有一本可與之配合的教科書是戴維斯·戈登所著的電腦與資訊處理（Computers and information processing, Mcgraw-Hill, 1978 by Gordon Davis）。

我們認爲在教導與學習福傳方面，本書特別有價值的地方是

1. 把每一個單元分成兩章，單數章說明語言的特性，偶數章是範例程式和完整的程式文件，用來示範前一章剛學到的新特性。
2. 特別強調程式設計的型態，並且前後一貫地利用範例程式示範本書所建議的型態，

希望學生一開始就養成良好的程式設計習慣。

3. 參閱資料很容易——內函數表是在封面裏頁，各種語言特性可以查索引。

4. 備有個別福傳編譯程式的特殊限制的彙總（封底裏頁）。

5. 前四章使用自由格式輸出入陳述，使得學生不必學會複雜的格式敍述就能夠撰寫程式，提早上機實習，增加實作的機會，增強學習效果。

6. 遇到相類似的特性，特別強調能夠避免錯誤的寫法，像利用單引號列印說明文字和邏輯條件陳述等。

7. 備有衆多的程式設計練習題。每一個偶數章都備有十五個練習題分成五類。使得本書適用於各種不同的科系和興趣。

8. 介紹製卡機的用法（附錄一）和終端機的用法（附錄二）。

希望本書的出版對福傳的教學有所裨益。不過編者才疏學淺，錯誤難免，希望讀者多予指正。

中華民國六十九年八月於台北

第二版序

本書出版以來，看到採用本書的學生大都能夠依照本書所倡導的方式，撰寫出井然有序、可讀性很高的程式來，覺得非常高興。承蒙諸位教授及各界人士採用本書、廣爲推介、並提供改進的建議，謹此致最大的謝意。

第二版參照讀者及教授們的建議，並鑒於大部份的電腦系統都已經具備了福傳 77 的編譯程式，所以作如下的修改：

1. 舉例及範例程式裏儘量使用區段條件陳述而避免使用跳越陳述。
2. 提早在第三章介紹資料類別的宣告及字符資料，以便利字符資料的處理，並且在第五章介紹字符資料的輸出入。
3. 在第十一章增加內檔案的介紹，並且在第十二章的範例程式裏示範直接存取檔案的使用方法。

編者才疏學淺，錯誤難免。敬請讀者把本書的優點告訴你的朋友，把本書的缺點告訴編者。來信請寄台北市南京東路三段一三一號六樓，郵遞區號一〇四。歡迎您提供寶貴的意見，謝謝。

鄭守成　　民國七十二年八月　於台北

目　　錄

第一章　程式設計規則、福傳語言、與撰寫簡單程式的福傳陳述

第二章　讀入、計算、與列印的範例程式

第三章　內函數、整數資料、選擇結構、資料檢核、與程式測試

第四章 使用內函數、選擇結構、和資料檢核的範例程式

第五章 帶格式的輸入與輸出陳述

第六章 帶格式輸出入的範例程式

第七章　重複結構、註標變數、及複作循環

第八章　註標變數和複作循環的範例程式

第九章　副程式和個案程式結構

第十章　副程式和個案結構的範例程式

第十一章　外檔案的用法

第十二章　使用外檔案的範例程式

第十三章　其他的資料類別、字符串、及其他特性

第十四章　其他特性的範例程式

第一章　程式設計規則、福傳語言、與撰寫簡單程式的福傳陳述

第一章

本章介紹程式語言的觀念，同時解釋程式設計規則的價值。說明福傳（FORTRAN）語言，準備一個福傳程式的一般程序，並介紹撰寫一個簡單的福傳程式所需要的五種福傳陳述（ Statements ）。在第二章裏有兩個完整的福傳程式，利用這兩個程式例子做示範，你將能夠撰寫一個簡單的福傳程式，將它製成卡片然後上機處理。在第一章裏提供瞭解福傳程式設計所需要的觀念，而在第二章裏提供準備一個福傳程式的初步經驗。第二章裏的程式同時提供把一個福傳程式提交給電腦中心並加以處理的經驗。

§1-1　指令一部電腦

在開始說明福傳語言之前，讓我們先介紹如何指揮命令一部電腦。一部電腦系統係由硬體及軟體所組成，硬體係指看得見摸得着的設備，軟體則包括指揮命令電腦設備運作的程式群。

(一)・硬體與軟體

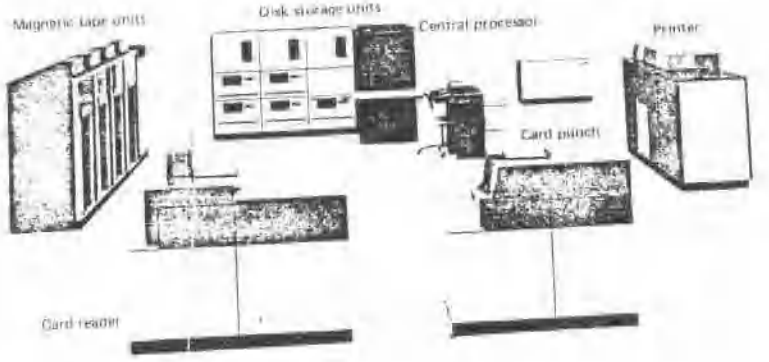


圖 1-1　一個電腦系統的硬體

在圖 1-1 裏的電腦系統有輸入設備（如讀卡機），中央處理機（CPU）以及輸出設備（如列表機）。還包括外部儲存設備如磁帶機或磁碟機。在一個典型的資料處理工作，資料係經由輸入設備（或自外部儲存設備）輸入中央處理機作計算或其他處理。在處理完畢之後的結果則經由輸出設備輸送出來，或者存入外部儲存設備供爾後的處理或輸出。

電腦硬體能執行讀、寫、計算等等運作，但是這些運作執行的順序以及輸出入設備的使用等是由儲存在電腦記憶體裏的一組指令來指揮的。這些電腦處理指令一般就叫做軟體。這些指令組成許多集合叫做常規（Routines）及程式（Programs）。一個常規係指一組指令執行某一特定的任務，例如計算一個數值的平方根，或是當輸入資料錯誤時產生一項錯誤訊息。一個程式則包含數個常規用以解決一個完整的問題。硬體、軟體、和電腦使用人的關係如圖 1-2。

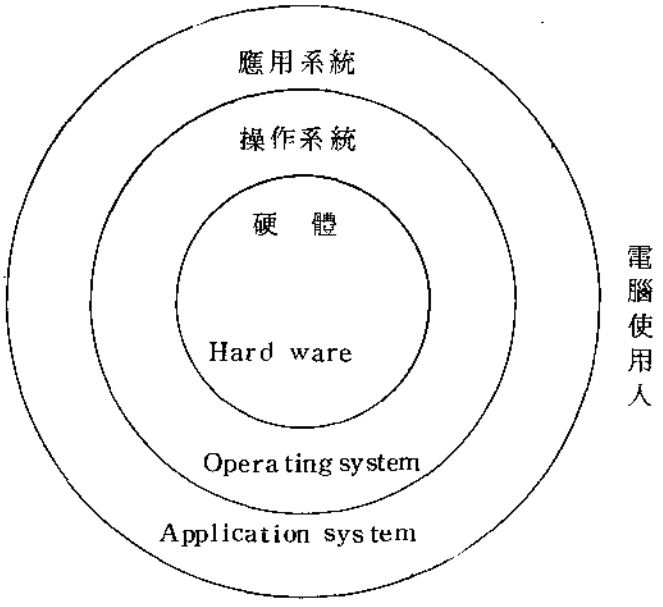


圖 1-2　電腦硬體、軟體與使用人的關係

軟體有數種類型。與福傳的學習特別有關的軟體有三種：應用程式、編譯程式、和操作系統。

1 **應用程式**（Application programs）：電腦使用人爲了利用電腦解決某一個問題所撰寫的程式叫做應用程式。例如爲某一專案計算其投資報酬率，爲一個建築結構計算應力等，以及本書的一些範例程式都是應用程式。應用程式可以由電腦中心的人員自己撰寫，或向軟體公司購買。軟體公司常能提供一些常用的應用程式組

（Application package）。

2. **編譯程式**（Compilers）：利用高等程式語言所撰寫的程式例如福傳程式，必先利用編譯程式把它翻譯成機器語言之後，電腦才能夠據以執行。這種編譯程式大都由電腦製造廠商所提供。

3. **操作系統**（Operating system 簡稱OS）：管理一個電腦系統運作的程式組叫做操作系統。應用程式就是在操作系統的指揮和支援之下運作的。例如有一個應用程式要自讀卡機讀入一張卡片，而讀卡機無法運作時，操作系統會送一個適當的資訊給電腦操作員。操作系統通常是由電腦製造廠家所提供的。

正在執行的程式是存放在內部儲存體（Internal storage）或稱爲記憶體（Memory）內，不是正在執行的程式則存放在外部儲存器裏。操作系統亦是存放在外部儲存器裏，而其中最常用的控制程式則常駐在記憶體內。這些控制程式根據工作控制陳述的要求把所需要的常規和程式自外部儲存器讀進來並安排他們的執行。這些執行的程式可能是編譯程式、應用程式、或其他軟體。

(二)．機器語言(Machine-level Language)

在記憶體裏待執行的程式必須是機器語言。一個機器語言指令是由一串二進位數字所組成，包括要執行的運作以及它所要用到的資料或資料的位址。例如一個典型的CDC CYBER 170 電腦加法指令如下：

0111100110010l0

用這種機器內部的表示方法來撰寫程式就相當困難而且容易導致錯誤。當我們把機器指令列印出來或顯示在螢光幕上讓操作員或程式人員閱讀時，是利用一種濃縮的格式，是用八進位或十六進位數字來表示。例如上述指令列印出來如下：　36312

雖然它比二進位表示法容易閱讀些，這種濃縮的符號仍然很難使用。因此當程式人員必須撰寫機器層面的指令時，通常是使用符式組合語言（Symbolic assembly language）。這種語言是機器導向的，因爲一個符式組合指令是與一個機器指令相對應的。上述機器指令用符式組合語言來表示時可能是　I X 3　X 2 + X 1　意指把X 2 和X 1記錄器（Register）裏的整數相加，然後把加完的結果存放在X 3記錄器裏。電腦並不能直接執行一個符式指令，因此必先把利用符式語言撰寫的程式翻譯成機器語言程式。負責翻譯的程式就叫做符式組合系統，簡稱組合程式（Assembler），它把每一個符式指令翻譯成爲一個對等的機器指令。機器導向的程式設計對某些應用系統非常有用，因爲可以寫出效率非常高的程式來。但是用組合語言撰寫程式比較困難，邏輯錯誤難以發現，同時修改程式時不但很難而且非常耗時。而且這種機器導向語言撰寫的程式

在不同電腦間的移轉性很低。一旦換機器，程式必須重寫，耗費不貲。因此近年來不但應用程式很少用符式語言撰寫，就連操作系統等大型軟體系統亦多轉而使用高等語言撰寫，以提高生產力，降低維護成本。

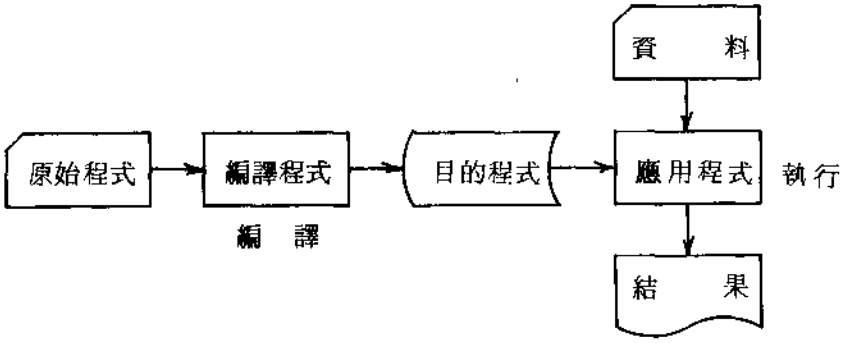


圖1-3　程式的編譯與執行

(三)·高等語言(High-level Language)

一種高等程式語言是問題解法導向或處理程序導向的，而不是某一種電腦的機器層面指令。高等語言的指令陳述所使用的語彙與符號和我們習用的解法描述或處理程序相類似。高等語言指令和符式組合指令的另外一項主要區別是一個高等語言陳述翻譯成好幾個機器語言指令。

爲了解決各種不同類型的問題，目前已有多種不同的高等語言。每一種語言各有它一套語法和特定的字來撰寫指令。用某種高等語言撰寫的程式叫做原始程式（Source program），必須用該語言的編譯程式把它翻譯成機器語言程式（目的程式Object program），然後電腦才能依照機器語言程式裏的指令執行我們所期望的計算或處理工作。如果該電腦沒有某種語言的編譯程式，那麼你用這種語言寫出來的程式，電腦是不認識的。

高等語言比符式組合語言具有兩項重要的優點：其一是高等語言是一種機器獨立（Machine independent）語言，也就是說用高等語言撰寫的程式只需要加以少量的修改或完全不用修改就可以在任何一個電腦上經過編譯之後執行。現在這些高等語言的能力和效率都相當高，除了少數特殊的應用之外，我們幾乎可以不需要撰寫組合語言程式。其二是使用高等語言撰寫程式的方法相當容易加以標準化。關心程式的正確性和希望建立程式設計規則的電腦中心都採用高等語言。