

修 訂 版

電子計算機程式語言

—COBOL

編 譯

吳 建 平

中央圖書出版社出版

電子計算機程式語言

—COBOL

編 譯

吳 建 平

國立台灣大學電機工程學系 教授

中央圖書出版社出版

143

TP312

835

行政院新聞局出版事業登記證，
局版台業字第〇九二〇號

修訂版
電子計算機程式語言
—COBOL—

每本定價新台幣貳佰貳拾元整
版權所有·翻印必究

編譯者：吳

發行人：林

建 高 平

出版者：中

中央圖書出版社
台北市新陽路二二號

總經銷：中

中央圖書供應社
台北市新陽路二二號

電話：三三一五七二
郵政劃撥帳戶：九一四三六

印刷所：聯和印製廠有限公司

台北市寶興街二十一號

中華民國六十年十月二十日初版
中華民國七十年三月第十二版

編號：0602

序 言

隨着科學進步，工商業發展，電子計算機之應用，在我國已日趨廣泛。然有關之中文書籍，却極貧乏，於商業資料處理之程式語言方面為尤甚。無論教學及應用，各方面均感不便。編者有鑑於此，爰將授課講義予以整理，並依據 Nancy B. Stern 著 COBOL Programming 及日本電氣情報處理教育部編之 EDPS 入門シリーズ 3 —— COBOL プログラミング入門兩書，更參考 McCracken 之 A Guide to COBOL Programming 及 Mario V. Farina 之 COBOL Simplified 等書編譯，其要旨如下：

- (一) 務求講述清楚，易於瞭解。
- (二) 不在於敘述 COBOL 程式語言之格式，而在使讀者能於寫出完整之程式。
- (三) 以教科書之循序漸進方式編排，而含有一般實用所必需之資料。故既可作為完整之教材，亦可為自修及應用之參考書。
- (四) 取材力求廣泛，範例與習題亦多。使學者能由各完整程式範例中，對 COBOL 之應用，有更確切之認識。

編者利用授課之暇，編著本書；時間有限，漏誤在所難免，尚冀海內方家，不吝指正。

吳建平 民國六十年國慶於台北
國立台灣大學

目 錄

第一章	電子資料處理系統和 COBOL	
1	電子資料處理系統	1
2	COBOL 簡史	11
3	COBOL 程式簡介	14
第二章	資料的組織	22
1	檔、錄、和欄	22
2	資料的型式	29
第三章	識別部	40
1	COBOL 程式的基本結構	41
2	識別部的寫碼要求	47
第四章	設備部	52
1	組態節	53
2	輸入輸出節	54
第五章	資料部—檔部	64
1	檔描述	64
2	錄的描述	80
第六章	程序部	99
1	OPEN 敘述	102
2	READ 敘述	106
3	WRITE 敘述	108
4	簡化的 MOVE 和 GO TO 敘述	111
5	結束工作作用	115
第七章	移動敘述	125
1	基本方法	125
2	MOVE 敘述的格式	128
3	數值移動	133

4	文數字移動	141
第八章	算術運算	151
1	ADD 敘述	151
2	SUBTRACT 敘述	156
3	MULTIPLY 和 DIVIDE 敘述	160
4	ROUNDED 取捨子句	166
5	ON SIZE ERROR 取捨子句	169
第九章	條件敘述	177
1	簡單條件	177
2	正負號、類別測試、和否定條件句	189
3	複合條件句	192
第十章	工作儲存節	208
1	獨立項目和VALUE (初值)子句	208
2	集體項目	217
第十一章	其他的資料表示法	230
1	名稱的限定	230
2	右靠子句	233
3	SYNCHRONIZED 子句	236
4	重定 (REDEFINES) 子句	240
5	數值欄的其他 PICTURE 說明	249
6	條件名稱	253
第十二章	印出結果的編排	263
1	編排職責	264
2	編排所用字元之解釋	266
3	浮串符號及零則空白子句	279
第十三章	印刷輸出的特殊情況	290
1	表格的間格	290
2	每頁印刷紙用完的測試及跳換頁	293

3	資料編排及標題印製	298
第十四章	程式中程序的改變	315
1	簡單的GO TO 複習	315
2	GO TO.....DEPENDING ON.....敘述	317
3	STOP 敘述	320
4	ALTER 敘述	324
5	PERFORM 敘述	330
6	簡單PERFORM 敘述在更新程式方面的應用	335
第十五章	PERFORM 敘述	346
1	基本格式	346
2	PERFORM 敘述的其他形式	352
第十六章	OCCURS 子句—單等級	373
第十七章	OCCURS 子句—雙重及三重等級	397
1	雙重等級OCCURS 子句	397
2	三重等級OCCURS 子句	406
第十八章	其他的輸入—輸出敘述	418
1	DISPLAY敘述	418
2	ACCEPT 敘述	422
第十九章	其他程序述句	430
1	COMPUTE 敘述	430
2	MOVE CORRESPONDING 敘述	438
3	巢狀條件敘述	446
4	EXAMINE 敘述	451
5	庫存敘述	456
第二十章	磁盤操作	463
1	磁盤檔的組織	465
2	出入法	467
3	關 鍵	468

4	磁盤處理之程序部述句.....	472
附錄 A		
1	COBOL 中所用的字元.....	486
2	COBOL 保留字.....	487
附錄 B	流程圖的基本元件.....	491
附錄 C	各種計算機之設備部述句特例.....	504
附錄 D	CDC-3150 計算機 COBOL 語言格式	512
附錄 E	提高程式效率的規則.....	522
附錄 F	部份問題之答案.....	525
索引	596

第一章

電子資料處理系統和 COBOL

1.1 電子資料處理系統

電子資料處理系統 (Electronic Data Processing System) 乃是使用電子計算機處理資料的系統，簡稱爲 EDPS。

本節就電子計算機的構造及其功能加以說明。本書的主要目的在於學習 COBOL，因此只是說明電子資料處理系統的必要之基本知識。讀者若想進一步瞭解詳細的情形，請參考計算機原理方面的資料。

電子計算機的構造

電子計算機係由

算術和邏輯單位

記憶單位

控制單位

輸入輸出單位

四個單位組成。前三個單位合稱中央處理單位 (Central Processing Unit, CPU)，相當於人腦的功用。輸入輸出單位相當於手脚的功用。以下使用算盤計算 12×132 的情形爲例，將各種功能加以說明。

1. 算術和邏輯單位

此單位進行資料 (此例爲 12 和 132) 之計算，以及大小關係之比較和決定。這相當於算盤。不過，算盤本身不能運算，必須由人撥動算珠才能進行

計算。計算機的算術和邏輯單位則依照指令（此例中為乘法）就所給的資料自動地計算乘積。

2. 記憶單位

我們使用算盤計算的時候，暗記計算所用的資料（此例為 12 和 132）以及指令（此例中為乘法），就此資料和指令用手指撥動算珠計算。然後在拂落算盤的算珠之前，讀取計算的結果。這種儲存資料、指令、以及計算結果的設置稱為記憶單位。

以下概要地說明計算機的記憶方式。

記憶單位係由稱為磁心 (Magnetic core) 的小圓環（外徑 0.5 到 2mm 左右）組成。是即記憶單位的細胞。這種磁心係用一種所謂陶鐵磁體 (Ferrite) 的合金製成的。這種合金容易磁化，一旦磁化就保持磁化的狀態。

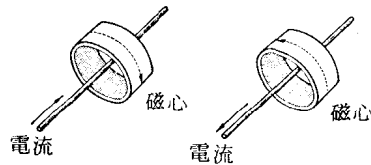


圖 1-1 磁心

如圖 1-1 所示，導線通以電流，磁心就帶有磁性。如此，磁心可取兩種不同的狀態，而且具有保持這種狀態的性質。圖 1-2 中顯示使用許多磁心和導線排成陣列形式的記憶器。

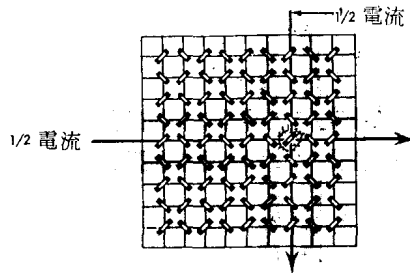


圖 1-2 磁心記憶面

這兩種相異的狀態可以方便地規定其中一種狀態為 0 而另一種為 1，因

此能够表示數值。記憶單位乃是由許多 0 和 1 構成。這種由 0 和 1 構成的數字稱之爲二進數。記憶單位通常使用二進數表達資訊。

直接深入瞭解計算機頗爲麻煩。但是學習 COBOL 並無必要直接瞭解計算機的構造原理，再者經由學習 COBOL 也能瞭解 EDPS。

3. 控制單位

控制單位比較難於瞭解，它乃是記憶單位和算術邏輯單位間之橋樑，解釋儲存在記憶單位內的指令，用這些指令指示取出資料到算術邏輯單位。然後從算術邏輯單位取出運算的結果再行存入記憶單位。因此控制單位乃是控制計算機的所有動作之部分。

以上三個要素的關係顯示於圖 1-3。

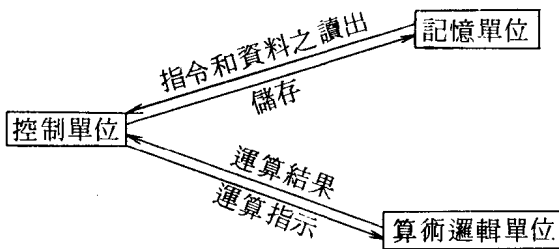


圖 1-3 三個要素之關係

4. 輸入輸出 (Input-Output) 單位

輸入輸出單位有各式各樣的裝置（稱之爲輸入輸出設置）。圖 1-4 示計算機的系統。

輸入輸出單位雖然有各式各樣的裝置，以下僅就最常使用的記錄方式加以說明。

卡片

圖 1-5 中顯示卡片的穿孔組合，一行代表一個符號。假如一行中都沒有穿孔即表示空行。一張卡片最多能够代表 80 個符號。



圖 1-4 計算機系統

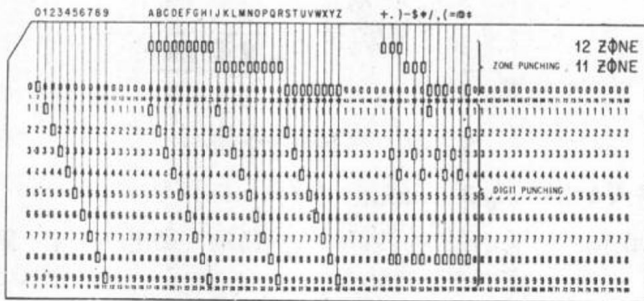


圖 1-5 卡片

印字用紙

普通的打字機每一次打擊只印出一個字元，因此速度太慢。計算機係使用幾乎能夠同時印出一列資料（在數十分之一秒內）的高速印字機（Printer）。普通一列可印 120 個字元，有的可多至 140 個字元，視所用的印字機而定。紙卷係連續使用。

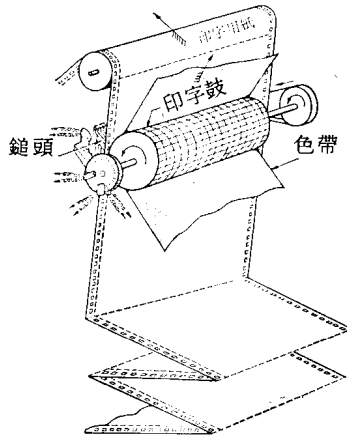


圖 1-6 印字機略圖

磁帶 (Tape)

磁帶乃是在合成塑膠帶上塗一層容易被磁化的氧化物。其原理和日常使用的錄音帶相同，係使用從記憶單位送出的脈波磁化磁帶的表面來記錄資料。讀出的情形則按完全相反的方法進行。通常磁帶約 700 米長，捲在捲盤 (Reel) 上。

最後說明記憶單位的性質。記憶單位中的資料使用多少次都不會消失。相反地，資料移入記憶單位中時，以前的資料就消失，而記憶著 (儲存著) 新的資料。

以下圖中顯示資料從記憶單位的某一位置轉移到另一位置的情形。

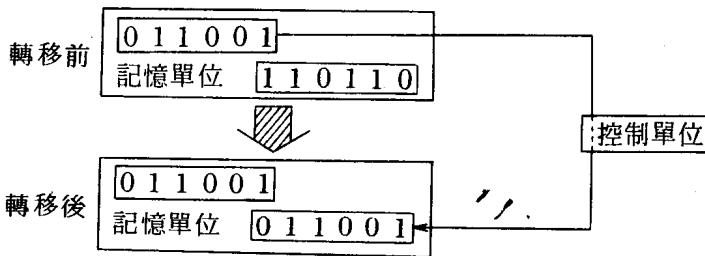


圖 1-7 記憶單位轉移資料的情形

程式與計算機

指示計算機處理次序的一組指令稱為**程式 (Program)**，製作程式的人稱為**程式計劃師 (Programmer)**。製作程式的過程稱為**程式計劃 (Programming)**。使用計算機執行業務，則非製作程式不可。程式是儲存在記憶器中。控制單位則依次解釋所存的程式中各個指令的意義，並指示如何執行。

程式寫就之後，假如處理的程序不再改變，則可永久使用。從以上的說明，就可瞭解計算機使用程式能夠解決各種問題。

程式語言

計算機能夠直接瞭解的就只是二進數。因此，程式非使用計算機特有的二進數製作不可。例如，數值 X 和數值 Y 相加而將結果儲存在 Z，則非用類似以下的指令不可：

```
00110000000001100100000110100
```

```
011110000001100000100000110100
```

像這種製作程式所使用的語言稱為**計算機語言 (Computer language)**，像上述使用二進數表示的計算機語言稱為**機器語言 (Machine language)**。在計算機應市的初期乃是使用機器語言。這種語言既難瞭解，用之寫程式又需要很大的精力。因此改進為使用符號表示而比較容易的**組合語言 (Assembly language)**。前例使用組合語言表示如下：

```
MCW    X, Z (將 X 轉移到 Z)
```

```
A      Y, Z (將 Y 和 Z 相加)
```

這種組合語言目前使用得很多。但是組合語言仍然不適合於初學者。因此有例如 $Z \leftarrow X + Y$ 等等接近日常所用語言之形式的計算機語言出現。這種語言稱為**編譯程式語言 (Compiler language)**。這種編譯程式語言依照適用的應用性質可區分如下：

1. 科學技術計算用的計算機語言
 ALGOL (ALGO^rithmic Language)
 FORTRAN (FOR^mula TRAN^slator)
2. 商業處理用的計算機語言

COBOL (CO mmon Business Oriented Language)

3. 高級計算機語言

PL/1 (Programming Language 1)

以下係就上例使用各種編譯程式寫成的：

Z := X + Y; (ALGOL)

Z = X + Y (FORTRAN)

COMPUTE Z = X + Y. (COBOL)

Z = X + Y; (PL/1)

這種程式語言的出現使得程式計劃非常容易。一般的編譯程式語言有以上的通用語言，也有適用於各種特殊處理的語言。編譯程式語言仍在不斷地發展中。

編譯程式

編譯程式語言使用起來非常方便。但是計算機能夠直接瞭解的却只是機器語言。假若使用編譯程式語言寫成的程式能夠在計算機直接執行，那就好了。這好像不同民族之間不能直接表達意思一樣。我們知道在這種情況下經過通譯員的翻譯即能夠解決。

同樣地，在 EDPS 的情形，使用計算機能夠直接了解的機器語言來翻譯編譯程式語言。此時非注意不可的是使用編譯程式語言寫成的程式，與翻譯這種程式所得的機器語言程式在語言上雖然不同，但是其意義却完全相同。

爲了區別起見，程式計劃師用編譯程式語言寫成的程式稱爲原始程式

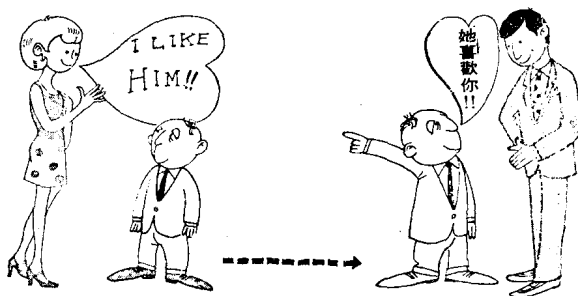


圖 1-8

(Source program)，而翻譯成的機器語言程式稱為目的程式 (Object program)。

編譯程式非常之大 (通常含有10萬個指令)，計算機使用者 (User) 製作非常困難。因此，計算機的製造廠家設計各種編譯程式語言的編譯程式以供使用者之用。

本書乃是學習使用 COBOL 以設計原始程式。本書中所謂程式係指原始程式而言。

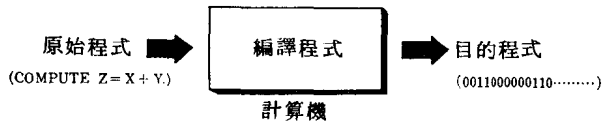


圖 1-9 翻譯

程式的流程

程式乃是用以表示執行指令的次序。實際上計劃程式時要先分析問題。

例如

處理 $X \leftarrow A + B \times C$ 的四則運算

1. B 和 C 相乘
2. 將 A 加到第 1 步所得的結果
3. 將第 2 步所得結果存入 X

但是，如此 1.，2.，……逐項寫出不但麻煩，而且容易出錯誤，因此用圖 1-10 表示。

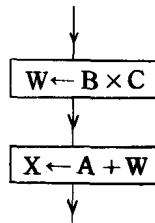


圖 1-10

這種圖形稱為**流程圖** (Flow chart)。通常使用各種形狀的圖形來表達處理的內含。繪製流程圖的原則是由上而下，由左而右。流程圖有助於檢查處理的次序是否有錯，而且假如有錯誤的話容易**改錯** (Debug)。

一般使用的圖形示於附錄 B。

當流程圖一繪出來，即可使用適當的計算機語言寫成程式。這種過程稱為**寫碼** (Coding)，寫碼所用的紙稱為**寫碼紙** (Coding sheet)。COBOL 的寫碼紙之格式及用法將在第三章說明。

練 習

每題下面均印有星號，表示其後即為答案。請先用紙遮蓋答案再試行作答。

1. 電子計算機係由 _____、_____、_____ 和 _____ 四個單位組成。

算術和邏輯

記憶

控制

輸入輸出

2. 計算機的程式計劃師 _____ 計算機的程式。

寫出並測試

3. 就輸入資料加以運算而換出輸出資料的一組指令稱為 _____。

程式

4. 指令必須使用 _____ 語言才能被計算機執行。

機器