COBOL簡介

http://www.csis.ul.ie/cobol/Course/COBOLIntro.htm#part3

資科系

林偉川

1

COBOL歷史

- COBOL (Common Business Oriented Language) was one of the earliest high-level programming languages.
- It was developed in 1959 by a group of computer professionals called the Conference on Data Systems Languages (CODASYL). Since 1959 it has undergone several modifications and improvements.

2

COBOL歷史

- In an attempt to overcome the problem of incompatibility between different versions of COBOL, the American National Standards Institute (ANSI) developed a standard form of the language in 1968. This version was known as ANSI COBOL.
- **COBOL is self-documenting**

3

COBOL歷史

- In 1974, ANSI published a revised version of ANSI COBOL, containing a number of features that were not in the 1968 version. In 1985, ANSI published still another revised version that had new features not in the 1974 standard.

4

COBOL歷史

- The language continues to evolve today. Object-oriented COBOL is a subset of COBOL 97, which is the fourth edition in the continuing evolution of ANSI/ISO standard COBOL.
- COBOL 97 includes conventional improvements as well as object-oriented features. Like the C++ programming language, object-oriented COBOL compilers are available even as the language moves toward standardization.

5

COBOL語言特性

- **The first language that automated business**
- **Allows names to be truly connotative** - permits both long names (up to 30 characters) and word-connector characters (dashes)
- **Every variable is defined in detail** - this includes number of decimal digits and the location of the implied decimal point
- **File records are also described with great detail, as are lines to be output to a printer** - ideal for printing accounting reports

6

COBOL語言特性

- **Offers object, visual programming environments**
- **Class Libraries**
- **Rapid Application Capabilities**
- **Integration with the World Wide Web**

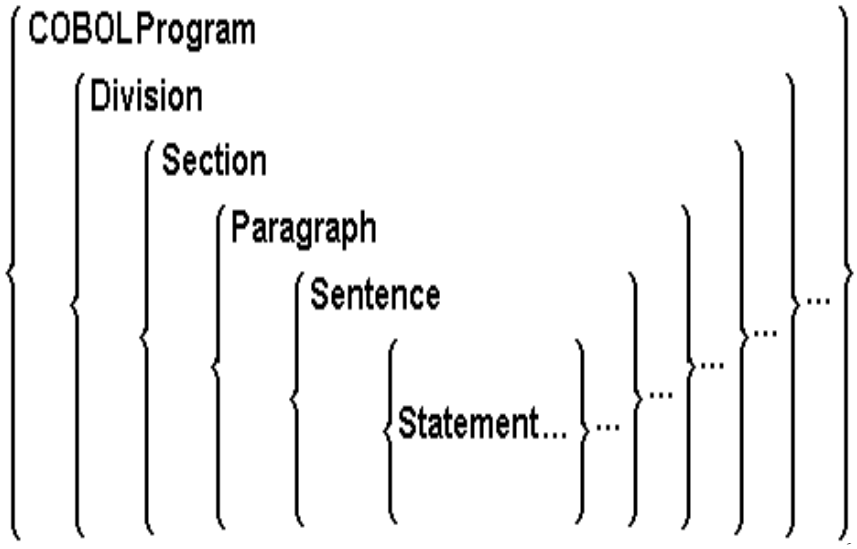
7

COBOL語言特性

- The hierarchy consists of Divisions, Sections, Paragraphs, Sentences and Statements.
- A Division may contain one or more Sections, a Section one or more Paragraphs, a Paragraph one or more Sentences and a Sentence one or more Statements

8

COBOL語言特性



COBOL語言特性

- **The actual program text starts in column 8. The four positions from 8 to 11 are known as Area A, and positions from 12 to 72 are Area B.**
- **Although many COBOL compilers ignore some of these formatting restrictions, most still retain the distinction between Area A and Area B.**

10

COBOL語言特性

- **When a COBOL compiler recognizes the two areas, all division names, section names, paragraph names, FD entries and 01 level numbers must start in Area A. All other sentences must start in Area B.**
- **Divisions**
  A division is a block of code, usually containing one or more sections, that starts where the division name is encountered and ends with the beginning of the next division or with the end of the program text.

11

COBOL語言特性

- **Sections**
  A section is a block of code usually containing one or more paragraphs. A section begins with the section name and ends where the next section name is encountered or where the program text ends.
- Section names are devised by the programmer, or defined by the language. A section name is followed by the word SECTION and a period.
  SelectUnpaidBills SECTION.
  FILE SECTION.

12

COBOL語言特性

- **Paragraphs**
  A paragraph is a block of code made up of one or more sentences. A paragraph begins with the paragraph name and ends with the next paragraph or section name or the end of the program text.
- A paragraph name is devised by the programmer or defined by the language, and is followed by a period.
  PrintFinalTotals.
  PROGRAM-ID.

13

COBOL語言特性

- **Sentences and statements**
  A sentence consists of one or more statements and is terminated by a period.
  For example:
  MOVE .21 TO VatRate
  MOVE 1235.76 TO ProductCost
  COMPUTE VatAmount = ProductCost * VatRate.
  SUBTRACT Tax FROM GrossPay GIVING NetPay

14

COBOL之4大**DIVISION**

- **IDENTIFICATION DIVISION.**
  Contains program information
- **ENVIRONMENT DIVISION.**
  Contains environment information
- **DATA DIVISION.**
  Contains data descriptions
- **PROCEDURE DIVISION.**
  Contains the program algorithms

15

IDENTIFICATION DIVISION

- Supplies information about the program to the programmer and the compiler. Most entries in the IDENTIFICATION DIVISION are directed at the programmer. The compiler treats them as comments.

16

IDENTIFICATION DIVISION

- The PROGRAM-ID clause is an exception to this rule. Every COBOL program must have a PROGRAM-ID because the name specified after this clause is used by the linker when linking a number of subprograms into one run unit, and by the CALL statement when transferring control to a subprogram.
- The IDENTIFICATION DIVISION has the following structure:
  IDENTIFICATION DIVISION
  PROGRAM-ID. NameOfProgram.
  [AUTHOR. YourName.]
  other entries here

17

ENVIRONMENT DIVISION

- Describe the environment in which the program will run. The purpose of the ENVIRONMENT DIVISION is to isolate in one place all aspects of the program that are dependant upon a specific computer, device or encoding sequence.
- The idea behind this is to make it easy to change the program when it has to run on a different computer or one with different peripheral devices.
- In the ENVIRONMENT DIVISION, aliases are assigned to external devices, files or command sequences. Other environment details, such as the collating sequence, the currency symbol and the decimal point symbol may also be defined here.

18

DATA DIVISION

- Provides descriptions of the data-items processed by the program.
- The DATA DIVISION has two main sections: the FILE SECTION and the WORKING-STORAGE SECTION. Additional sections, such as the LINKAGE SECTION (used in subprograms) and the REPORT SECTION (used in Report Writer based programs) may also be required.

19

DATA DIVISION

- The FILE SECTION is used to describe most of the data that is sent to, or comes from, the computer's peripherals.
- The WORKING-STORAGE SECTION is used to describe the general variables used in the program.

20

DATA DIVISION

DATA DIVISION.

[FILE SECTION.
File Section entries.]

[WORKING – STORAGE SECTION.
WS entries.]

21

COBOL變數宣告

- All user-defined names, such as data names, paragraph names, section names condition names and mnemonic names, must adhere to the following rules:
- They must contain at least one character, but not more than 30 characters.
- They must contain at least one alphabetic character.

22

COBOL變數宣告

- They must not begin or end with a hyphen.
- They must be constructed from the characters A to Z, the numbers 0 to 9, and the hyphen.
- They must not contain spaces.
- Names are not case-sensitive: TotalPay is the same as totalpay, Totalpay or TOTALPAY.

23

DATA DIVISION - Sample

- IDENTIFICATION DIVISION.
  PROGRAM-ID. SequenceProgram.
  AUTHOR. Michael Coughlan.
  DATA DIVISION.
  WORKING-STORAGE SECTION.
  01 Num1 PIC 9 VALUE ZEROS.
  01 Num2 PIC 9 VALUE ZEROS.
  01 Result PIC 99 VALUE ZEROS.

24

PROCEDURE DIVISION

- Contains the code used to manipulate the data described in the DATA DIVISION. It is here that the programmer describes his algorithm.
- The PROCEDURE DIVISION is hierarchical in structure and consists of sections, paragraphs, sentences and statements.
- Only the section is optional. There must be at least one paragraph, sentence and statement in the PROCEDURE DIVISION.
- Paragraph and section names in the PROCEDURE DIVISION are chosen by the programmer and must conform to the rules for user-defined names.

25

PROCEDURE DIVISION - Sample

```
IDENTIFICATION DIVISION.
PROGRAM-ID. SequenceProgram.
AUTHOR. Michael Coughlan.
DATA DIVISION.
WORKING-STORAGE SECTION.
01 Num1 PIC 9 VALUE ZEROS.
01 Num2 PIC 9 VALUE ZEROS.
01 Result PIC 99 VALUE ZEROS.
PROCEDURE DIVISION.
CalculateResult.
    ACCEPT Num1.
    ACCEPT Num2.
    MULTIPLY Num1 BY Num2 GIVING Result.
    DISPLAY "Result is = ", Result.
    STOP RUN.
```

26

執行COBOL

- RMCOBOL XXX.cbl
- RUNCOBOL XXX

27

Program Statements

- 由螢幕輸入 **Num1**
  ACCEPT Num1.
- 由螢幕輸入 **Num2**
  ACCEPT Num2.

28

Program Statements

- Num1乘以 Num2 結果放於 Result
  MULTIPLY Num1 BY Num2 GIVING Result.
- 顯示**Result**於螢幕
  DISPLAY "Result is = ", Result.

29

COBOL syntax

- Words in uppercase are reserved words. When underlined they are mandatory. When not underlined they are "noise" words, used for readability only, and are optional. Because COBOL statements are supposed to read like English sentences there are a lot of these "noise" words.
- Words in mixed case represent names that must be devised by the programmer (like data item names).

30

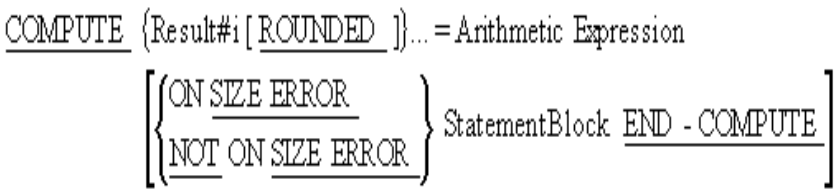
COBOL syntax

- When material is enclosed in curly braces **{ }**, a choice must be made from the options within the braces. If there is only one option then that item in mandatory.
- Material enclosed in square brackets **[ ]**, indicates that the material is optional, and may be included or omitted as required.

31

COBOL syntax

- In COBOL, evaluating an arithmetic expression and assigning the result to a data item is achieved by means of the COMPUTE statement. The syntax diagram for the COMPUTE is shown below.



32

COBOL開關檔

- 開檔

OPEN INPUT IN-FILE OUTPUT OUT-FILE.

- 讀檔

READ IN-FILE AT END MOVE "Y" TO EOF.

- 關檔

CLOSE IN-FILE OUT-FILE.

33

資料檔輸入螢幕輸出SIN.dat

0104A018550
0104A018580
0104A018595
0104A026580
0104A026545
0105A018555
0105A018580
0105A026590
0105A028590
0201A018580
0202A026576
0203A026569
0203A026540
0204A026590
0204A036890
0205A036790
0205A045690
0205A056710
0205A057820

34

資料檔輸入螢幕輸出

```
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT IN-FILE ASSIGN TO INPUT "SIN.DAT"
        ORGANIZATION IS LINE SEQUENTIAL.
    SELECT OUT-FILE ASSIGN TO OUTPUT "SOUT.DAT"
        ORGANIZATION IS LINE SEQUENTIAL.
FILE SECTION.
FD IN-FILE.
01 IN-REC.
        05 MON-I    PIC 99.
        05 DAY-I    PIC 99.
        05 GNO-I    PIC X(3).
        05 UP-I     PIC 99.
        05 QY-I     PIC 99.
```

35

資料檔輸入資料檔輸出

```
FD OUT-FILE.
01 OUT-REC      PIC X(80).
WORKING-STORAGE SECTION.
01 DL.
        05 MON-O    PIC Z9.
        05 FILLER   PIC X(4).
        05 DAY-O    PIC Z9.
        05 FILLER   PIC X(4).
        05 GNO-O    PIC X(4).
        05 FILLER   PIC X(4).
        05 AMOUNT-O PIC 99,999.
        05 FILLER   PIC X(4).
        05 DISCOUNT-O PIC 9,999.
        05 FILLER   PIC X(4).
        05 NET-O    PIC 9,999.
```

36

COBOL讀檔不只一筆資料

- PERFORM 200-MAIN-RTN UNTIL EOF = "Y".
- 語法：
  PERFORM 段名 ?? TIMES
  PERFORM 段名1 THROUGH 段名2 ?? TIMES
  PERFORM 段名1 VARING 變數 FROM ??
  BY ?? UNTIL 變數 > ??

37