COBOL 入門

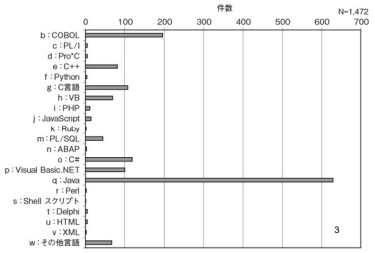
プログラム言語論第五回

まずは

- COBOL – COmmon Business Oriented Language
  - バッチ処理(それしかなかった)用に設計
  - データ書式(ファイル書式と処理途中のデータの書式)がまず念頭にある、プログラム論理は次.
  - データフローに基づいてプログラムロジックを組み立てる.

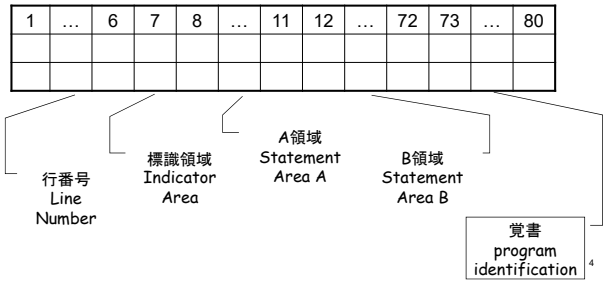
今でも?

- ソフトウエア開発プロジェクト(新規・改修他)の13.3%の開発言語がCOBOL
  - 情報処理推進機構「ソフトウエア開発データ白書2018-2019」
  - ただし、金融・保険に偏っている

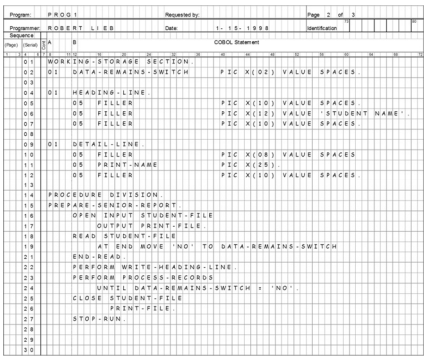


文の書式

- 書式は 固定 です, また.



COBOL コーディング フォーム



全体的な注意事項

- 大文字小文字の区別なし(Case insensitive)
  - 通常は大文字を使う(歴史的理由)
- コメント
  - '*'を標識領域に書く
- 文の終端を示す記号はなし
  - 一文が複数行に跨ってよい.
- '.'は有効範囲の終端を示す
- たくさんの 予約語に たくさんの 類義語!
- Noise words: 省略可能な語

Hello World!

```
000010 IDENTIFICATION DIVISION.
000020 PROGRAM-ID.   HELLO-WORLD-PROG.
000030 AUTHOR.       TIMOTHY R P BROWN.
000040*The standard Hello world program
000050
000060 ENVIRONMENT DIVISION.
000070
000080 DATA DIVISION.
000090 WORKING-STORAGE SECTION.
000100 01 TEXT-OUT    PIC X(12) VALUE 'Hello World!'.
000110
000120 PROCEDURE DIVISION.
000130 MAIN-PARAGRAPH.
000140         DISPLAY TEXT-OUT
000150         STOP RUN.
```


プログラムの構造

- 4つの division:
  - 見出し部 Identification Division (required)
  - 環境部 Environment Division (optional)
  - データ部 Data Division (optional)
  - 手続き部 Procedure Division (optional)

Identification Division

- プログラム名等, 実行への直接的影響ないもの

```
000100 INDENTIFICATION DIVISION.
000110 PROGRAM-ID. EXAMPLE-1-PROG.
000120 AUTHOR. HOPPER.
000130 INSTALLATION. XYZ GROUP.
000140 DATE-WRITTEN. 16/1/04.
000150 DATE-COMPILED.
000160 SECURITY. LOCAL GROUP.
```

Optional

Area A　　Area B

Environment Division

- 環境節と入出力節 Configuration & I/O section

```
000260 ENVIRONMENT DIVISION.
000270 CONFIGURATION SECTION.
000280 SOURCE-COMPUTER. IBM PC.
000290 OBJECT-COMPUTER. IBM PC.
000300 INPUT-OUTPUT SECTION.
000310 FILE-CONTROL.
000320     SELECT INPUT-FILE ASSIGN TO 'input.dat'
000330       ORGANIZATION IS LINE SEQUENTIAL.
000340     SELECT PRINT-FILE ASSIGN TO PRINTER.
```

optional

Area A

Area B

Environment Division

- 入出力節 I/O section
  - 識別名をファイルやプリンタに割り当てる
  - 構文

```
000320     SELECT identifiers ASSIGN TO filename
000330       ORGANIZATION IS LINE SEQUENTIAL.
```

ピリオドなし

  - プリンタの構成を記す必要はない

Data Division

ファイル節、作業場所節、連絡節

- File, working-storage と linkage section

```
000400 DATA DIVISION.
000410 FILE SECTION.
000420 FD INPUT-FILE.
000440 01 CUSTOMER-DATA.
000450     03 NAME PIC X(12).
000460     03 ADDRESS.
000470         05 HOUSE-NUMBER PIC 99.
000480         05 STREET PIC X(19).
000500 WORKING-STORAGE SECTION.
000510 01 RECORD-COUNTER PIC 9(5) VALUE ZERO.
```

Area A

Data Division – File Section

- File は record からなる
  - Business-oriented language

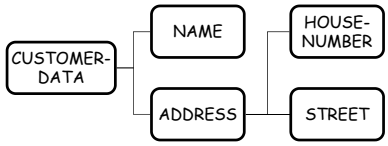
```
000420 FD INPUT-FILE.
000440 01 CUSTOMER-DATA.
000450     03 NAME PIC X(12).
000460     03 ADDRESS.
000470         05 HOUSE-NUMBER PIC 99.
000480         05 STREET PIC X(19).
```

定義するファイルに関する記述; Area A

レコード名; Area A

フィールド名; Area B

下位フィールド名; Area B



Data Division – Working Storage Section

- 変数代わりの領域の定義である

```
000500 WORKING-STORAGE SECTION.
000510 01 RECORD-COUNT PIC 9(5) VALUE ZERO.
000520 01 OUTPUT-LINE  PIC X(8) VALUE SPACES.
000530 01 INITIAL-NUM  PIC 9999 VALUE 10.
```

Area A

- レベル番号 Level number
  - 01 がレコード相当 (Area A)
  - それより上はフィールド相当 (Area B)
    - 大きい番号は深いレベルに
  - 66, 77, 88 予約

Data Division – Variable Declaration

- 識別名(一意名としているが) identifier
  - 最大 30 文字
  - 英数字かハイフン
- データ型
  - 文字
    - `01 INPUT-STR PIC XXXX VALUES 'ABCD'.`
    - `01 INPUT-STR PIC X(4) VALUES SPACES.`
  - 数
    - `01 INPUT-NUM PIC 9999 VALUES 1000.`
    - `01 INPUT-NUM PIC 9(4) VALUES ZERO.`

Procedure Division

- 実行制御はここに

```
000820 PROCEDURE DIVISION.
000830 MAIN-PARAGRAPH.
000840     DISPLAY TEXT-OUT
000850     STOP RUN.
```

Area A

Main paragraph (何でも OK); Area A

文; Area B

End 文; Area B

- 注意: “.”

代入文

- 構文

  MOVE *value* TO *variable*

- 例

```
01 DATE-IN.
    03 W-DAY PIC 99.
    03 W-MONTH PIC 99.
...
    MOVE 31 TO W-DAY
    MOVE 12 TO W-MONTH IN DATE-IN
```


算術式

- 単純な算術式

  ADD *value1* TO *value2* GIVING *variable*
  SUBTRACT *value2* FROM *value1* GIVING *variable*
  MULTIPLY *value1* BY *value2* GIVING *variable*
  DIVIDE *value1* BY *value2* GIVING *variable*

- 普通の算術式

  COMPUTE *variable* = *arithmetic expression*

  - 演算子: +, -, *, /, **

IF文

- 構文

```
IF condition THEN
  statement 1
ELSE
  statement 2
END-IF
```

- 論理演算子 Logical operators
  - NOT, AND, OR
- 関係演算子 Relational operators
  - =, >, <, <=, >=, NOT =

条件変数(conditional names)

- 例

```
000140 DATA DIVISION.
000150 WORKING-STORAGE SECTION.
000160 01 NUMBER-SIZE PIC X.
000170      88 BIG-NUMBER VALUE 'Y'.
```

- レベル 88 は条件変数. 上記の例では、 NUMBER-SIZEの値が
  - 'Y' ⇒ true
  - 'Y' 以外 ⇒ false

  と、BIG-NUMBER に値が自動設定されると考えればよい

条件変数(conditional names)

- NUMBER-SIZE に代入する
  - MOVE 'Y' TO NUMBER-SIZE
  - MOVE 'N' TO NUMBER-SIZE
- そうすると次のように書ける
  - IF BIG-NUMBER THEN …
- 多水準に分けることもできる

```
01 GRADES-CHECK PIC 999.
     88 A-GRADE VALUE 70 THRU 100.
     88 B-GRADE VALUE 60 THRU 69.
     88 C-GRADE VALUE 50 THRU 59.
     88 FAIL-GRADE VALUE 0 THRU 49.
```

- 対応するIF文は
  - IF B-GRADE THEN …

繰り返し文 – Until Loop

- Until ループ

```
PERFORM UNTIL condition
  ...
END-PERFORM
```

- Do-Until ループ

```
PERFORM WITH TEST AFTER
  UNTIL condition
  ...
END-PERFORM
```


繰り返し文 – For Loop

- 版 1

```
PERFORM n TIMES
   ...
END-PERFORM
```

- 版 2

```
PERFORM VARYING counter FROM
   initial BY step UNTIL condition
   ...
END-PERFORM
```


表

- 一次元

```
01 W-NAME PIC X(10) OCCURS 5 TIMES.
```

Optional

- 使い方

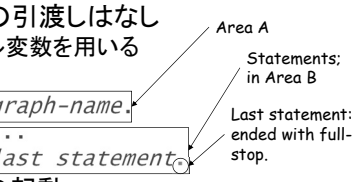
```
MOVE 'Hopper' TO W-NAME(1)
```

- 多次元

```
01 SALES-TABLE.
    03 BRANCH-NO OCCURS 4.
        05 SALES PIC 9(4) OCCURS 4.
```

Paragraph

- サブルーチンの COBOL 版
- パラメータの引渡しはなし
  - グローバル変数を用いる
- 構文



- Paragraphの起動:

```
PERFORM pname
PERFORM pname until condition
```


ファイル入出力 – Open と Close

- ファイル Open
  - OPEN *mode filename1 filename2 ...*
  - Mode: INPUT, OUTPUT, I-O, EXTEND
- ファイル Close
  - CLOSE *filename1 filename2 ...*

ファイル入出力 – 読み込み

- 仮に、ファイル構造は:

```
Grace Hopper   000001
William Selden 123456
...
```

- ファイルのレコード構造を下記で定義

```
FD IN-FILE.
01 CUSTOMER-DETAILS.
    03 CUS-NAME      PIC X(15).
    03 CUS-NUM       PIC 9(6).
```


ファイル入出力 – 読み込み

- ファイルからの1レコードの読み込み:

```
READ IN-FILE
```

  - CUS-NAME には顧客名が
  - CUS-NUM には顧客番号が、読み込まれる
  - 次のレコードを読むためには、同じ文を再実行する.

ファイル入出力 – 書き出し

- 仮に、ファイル構造が:

```
Grace Hopper   000001
William Selden 123456
...
```

- ファイルのレコード構造を下記で定義

```
FD OUT-FILE.
01 CUSTOMER-DETAILS.
    03 CUS-NAME      PIC X(15).
    03 CUS-NUM       PIC 9(6).
```


ファイル入出力 – 書き出し

- ファイルに1レコード書き込むには:

```
MOVE 'William Selden' TO CUS-NAME
MOVE 123456 TO CUS-NUM
WRITE CUSTOMER-DETAILS
```

英語らしく

```
$ SET SOURCEFORMAT"FREE"
IDENTIFICATION DIVISION.
PROGRAM-ID.  Iteration-If.
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.
01  Operator       PIC X  VALUE SPACE.

PROCEDURE DIVISION.
Calculator.
PERFORM 3 TIMES
   DISPLAY "Enter First Number      : " WITH NO ADVANCING
   ACCEPT Num1
   DISPLAY "Enter Second Number     : " WITH NO ADVANCING
   ACCEPT Num2
   DISPLAY "Enter operator (+ or *) : " WITH NO ADVANCING
   ACCEPT Operator
   IF Operator IS EQUAL TO "+" THEN
         ADD Num1 TO Num2 GIVING Result
   END-IF
   IF Operator IS EQUAL TO "*" THEN
        MULTIPLY Num1 BY Num2 GIVING Result
   END-IF
   DISPLAY "Result is = ", Result
END-PERFORM.
STOP RUN.
```


もう一つ

```
DATA DIVISION.
WORKING-STORAGE SECTION.
01  Char               PIC X.
    88 Vowel           VALUE "a", "e", "i", "o", "u".
    88 Consonant       VALUE "b", "c", "d", "f", "g", "h"
                             "j" THRU "n", "p" THRU "t", "v" THRU "z".
    88 Digit           VALUE "0" THRU "9".
    88 ValidCharacter  VALUE "a" THRU "z", "0" THRU "9".

PROCEDURE DIVISION.
Begin.
    DISPLAY "Enter lower case character or digit. No data ends.".
    ACCEPT Char.
    PERFORM UNTIL NOT ValidCharacter
        EVALUATE TRUE
           WHEN Vowel DISPLAY "The letter " Char " is a vowel."
           WHEN Consonant DISPLAY "The letter " Char " is a consonant."
           WHEN Digit DISPLAY Char " is a digit."
           WHEN OTHER DISPLAY "problems found"
        END-EVALUATE
    END-PERFORM
    STOP RUN.
```

全体は http://www.csis.ul.ie/cobol/examples/Conditn/Conditions.htm

学習用COBOLインタプリター

- YCOBOL というものがあります
  - http://akisakha.w7u.org/
- 別件ですが COBOL example programs
  - http://www.csis.ul.ie/cobol/examples/default.htm