The COBOL Language

Cobol 85, 2002, 2014 Revision 1

Fixed format:

- 1..6: Sequence number area (unused)
- 7: indicator area

`*`	comment line
`-`	continuation line
`D`	debugging line
`/`	print character

- 8..11: Area A (div./sec./par. names, etc.)
- 12..72: Area B (other sentences)
- 73..80: System generated number area (unused)

Free format: No indent, no indicator, `*>` for comment

```
123456 >>SOURCE FORMAT IS FREE
>>SOURCE FORMAT IS FIXED
```

Words contains `1..9`, `A..Z` and `-_`, with at least one letter and 30 chars max. They are case insensitive.

Syntax Terms

Fixed Point Numeric	`123`, `-12.3`
Non-numeric	`"Hello"`, `'Hello'` `"x""x"`, `'x''x'` (quote-escapes) `#"2A"` (hexa notation)
Floating Point Numeric	`123E12`, `-12.3E-7`
Booleans	`B"0"`, `B"1"`

Pre-processing

```
COPY  Nom-Module-Copy
  REPLACING ==:PREFIX:== BY ==Préfixe==
```

Program Structure

```
IDENTIFICATION DIVISION.
[...]
ENVIRONMENT DIVISION.
[...]
DATA DIVISION.
[...]
PROCEDURE DIVISION.
[...]
```

Identification Division

```
IDENTIFICATION DIVISION.
PROGRAM-ID. MY-PROGRAM [INITIAL].
AUTHOR. OCAMLPRO SAS.
```

Environment Division

```
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER IBM-370.
OBJECT-COMPUTER IBM-370.
SPECIAL-NAMES
  DECIMAL-POINT IS COMMA.
  CURRENCY SIGN X"9F" PICTURE SYMBOL "F".
[see Files sections for more]
```

Data Division

```
DATA DIVISION.
WORKING-STORAGE SECTION.
01 st.
         05 hh PIC 99.
         05 mm PIC 99.
         05 ss PIC 99.
         88 cond-ss-equals-0 VALUE 00. (conditional)
         66 ws-hhmm RENAMES hh THRU mm. (renaming)
01 x REDEFINES st PIC X(6). (renaming with cast)
01 STR X(50) JUST RIGHT VALUE "Just Left on INIT, right on MOVE".
01 STR X(50) JUST RIGHT VALUE Z"Terminated by 0".
01 FILLER PIC X.
01 ps PIC XXX VALUE "123".
01 ps PIC X(30) VALUE "1""3".
01 dec PIC 99V99 VALUE 11,22.
  77 ws-max-sub-variable PIC 99.
01 ws-decimal PIC 999V99 VALUE 123,45.
01 ws-signed-decimal S999V99 VALUE -123,45.
```

Picture clauses

`BINARY`, `COMP` or `COMP-4` (same with `S`, `V`)

9(1..4)	2 bytes	0..9,999
9(5..9)	4 bytes	0..999,999,999
9(10..18)	8 bytes	0..999,999,999,999,999,999

Native Binary `COMP-5`

9(1..4)	2 bytes	0..65,535
9(5..9)	4 bytes	0..4,294,967,295
9(10..18)	8 bytes	0..18,446,744,073,709,551,615
S9(1..4)	2 bytes	-32,768..+32,767
S9(5..9)	4 bytes	-2,147,483,648..+2,147,483,647
S9(10..18)	8 bytes	-9,223,372,036,854,775,808.. +9,223,372,036,854,775,807

Procedure Division

```
PROCEDURE DIVISION.
sec1 SECTION.
  [...]
sec1-paragraph1.
  STOP RUN.
sec2 SECTION.
  [...]
sec2-paragraph1.
  [...]
sec2-paragraph2.
  *> EXIT ends section, not program, use STOP RUN instead
  EXIT.
sec3-paragraph3.
  GO TO sec1-paragraph1.
```

Manipulating Control flow

```
  PERFORM one-section.
  PERFORM one-paragraph.
  PERFORM first-section THROUGH last-section.
  PERFORM section UNTIL condition.
  PERFORM [statements...] END-PERFORM.
  PERFORM UNTIL condition [...].
  PERFORM VARYING var FROM nbr1 BY incr UNTIL cond [...].
  PERFORM
    VARYING var1 FROM nbr1 BY incr1 UNTIL cond1
    AFTER   var2 FROM nbr2 BY incr2 UNTIL cond2 [...].
  PERFORM nbr TIMES [...].
  EVALUATE variable
    WHEN expr1 [statements1...]
    WHEN expr2 [statements2...]
    WHEN OTHER [statements3...] END-EVALUATE.
  IF condition THEN action ELSE action END-IF
  CONTINUE.
```

Modifying variables

```
  SET variable TO [TRUE/FALSE].
  MOVE expr TO variable.
  MOVE LOW-VALUE TO Variable.
  MOVE LOW-VALUE TO field OF variable.
  MOVE "XXX" TO string.
  MOVE "XXX" TO string(pos:len).
  MOVE "XXX" TO string(1:).
  MOVE WHEN-COMPILED TO x. ( x PIC X(16) )
  MOVE LENGTH OF String TO x.
  MOVE ADDRESS OF x TO y. ( y PIC USAGE POINTER )
  MOVE CORR Var TO Var. (every field separately)
```

Manipulating strings

```
  MOVE str1 (pos1:len1) TO str2 (pos2:len2)
  INSPECT String TALLYING ws-inspect-counter FOR ALL "X".
  INSPECT String TALLYING ws-inspect-counter FOR LEADING " ".
  INSPECT String TALLYING ws-inspect-counter FOR ALL "X".
  INSPECT String REPLACING ALL "X" BY "Y" AFTER "A" BEFORE "Z".
  INSPECT String REPLACING FIRST "X" BY "Y".
  INSPECT String REPLACING LEADING "X" BY "Y".
  INSPECT String CONVERTING "abcde" BY "ABCDE".
  STRING str1 str2 DELIMITED BY SIZE INTO dst-str END-
STRING.
  STRING str1 str2 DELIMITED BY delim-str INTO dst END-
STRING.
  UNSTRING String
    DELIMITED BY ";" OR "," OR ALL " "
    INTO
      Variable1 DELIMITER IN ws-delim-1
      Variable2 COUNT IN ws-count-2
        *> last COUNT contains final len
    TALLYING IN ws-inspect-counter
    ON OVERFLOW Instructions
  END-UNSTRING.
```

Computations on numbers

```
  SUBTRACT expr FROM var ROUNDED ON SIZE ERROR Actions END-SUBTRACT
    *> replace SUBTRACT FROM by ADD TO, MULTIPLY BY, DIVIDE BY
  SUBTRACT expr FROM expr GIVING var ROUNDED
    ON SIZE ERROR Actions
  END-SUBTRACT
  DIVIDE expr BY expr GIVING var REMAINDER var END-DIVIDE
  COMPUTE var [ROUNDED] = Expression [ON SIZE ERROR Actions]
  END-COMPUTE
```

Calling sub-programs

```
 WORKING-STORAGE SECTION.
   01 ws-val PIC 9(2) VALUE 95.
   01 ws-subprog PIC X(5) VALUE "xx".
   01 ws-code PIC S9(4) BINARY.
 LINKAGE SECTION.
* variables received from other prog
   01 ls-arg1 PIC 9(2).
   01 ls-arg2 PIC X(3).
   01 pt-param POINTER.
   01 ls-param PIC 9(2).
* main entry can also take arguments:
 PROCEDURE DIVISION [USING ...].
   CALL "f" USING [BY REFERENCE] ws-val OMITTED .
* set exit code of program:
   MOVE 8 TO RETURN-CODE.
* equiv to EXIT PROGRAM in sub-prog, STOP RUN in main:
   GOBACK.
* multiple entries can be defined in a sub-prog:
 ENTRY "f" USING ls-arg1 ls-arg2.
   IF ADDRESS OF ls-arg2 = NULL THEN
*    second argument has been omitted
   END-IF
* some compilers cannot pass args from linkage section:
   CALL "g" USING ADDRESS OF ls-arg1
   GOBACK.
 ENTRY "g" USING pt-param.
* recover typed param from received pointer:
   SET ADDRESS OF ls-param TO pt-param.
   MOVE "h" TO ws-subprog.
* re-init storage of called sub-program
   CANCEL ws-subprog.
   CALL ws-subprog USING BY CONTENT 1 2 3 RETURNING ws-code
     ON OVERFLOW DISPLAY "missing " ws-subprog
   END-CALL.
   DISPLAY "95 = " ls-param.
   GOBACK.
```

Reading a File

```
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
  SELECT F-IN ASSIGN TO SYSPRINT
    ORGANIZATION SEQUENTIAL
    ACCESS MODE IS SEQUENTIAL
    RECORDING MODE IS [F/V]
    FILE STATUS IS STATUS-F-IN.
DATA DIVISION.
FILE SECTION.
FD F-IN.
01 ININ.
  05 IN01 PIC X OCCURS 1 TO 121
     DEPENDING ON       DATA-IN-LEN.
WORKING-STORAGE SECTION.
77  DATA-IN-LEN PIC 9(4)  COMP.
01  STATUS-F-IN PIC X(2)  VALUE "00".
01              PIC X     VALUE X"00".
   88  EOF-F-IN           VALUE X"01" THRU X"FF".
PROCEDURE DIVISION.
READER SECTION.
  OPEN INPUT F-IN
  PERFORM UNTIL EOF-F-IN
    READ F-IN
      AT END
        SET EOF-F-IN      TO TRUE
        CLOSE F-IN
      NOT AT END
         DISPLAY ININ
    END-READ
  END-PERFORM
```

Output and Input on Console

```
  DISPLAY "Hello," "var=" var.
  ACCEPT var
```

Getting current date

```
  ACCEPT yymmdd FROM DATE
  ACCEPT yyyymmdd FROM DATE YYYYMMDD
  ACCEPT yyddd FROM DAY
  ACCEPT d FROM DAY-OF-WEEK
  ACCEPT hhmmssss FROM TIME
```

Writing to a file

```
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
  SELECT F-OUT    ASSIGN TO SYSIN
    ORGANIZATION SEQUENTIAL
    ACCESS MODE IS SEQUENTIAL
    RECORDING MODE IS [F/V]
    FILE STATUS IS STATUS-F-OUT.
DATA DIVISION.
FILE SECTION.
FD  F-OUT.
01  OUT01.
  05  DATA-OUT PIC X(80) VALUE "Hello world".
WORKING-STORAGE SECTION
01  STATUS-F-OUT     PIC X(2)  VALUE "00".
PROCEDURE DIVISION.
  OPEN OUTPUT F-OUT
  IF STATUS-F-OUT NOT = "00"
    DISPLAY "Pb opening file,status:" STATUS-F-OUT
  ELSE
    WRITE OUT01
    IF STATUS-F-OUT NOT = "00" THEN
      DISPLAY "Pb writing to file,status:" STATUS-F-OUT
    ELSE
      CLOSE F-OUT
    END-IF
  END-IF
```

Intrinsic Functions

```
  COMPUTE y = FUNCTION f(x)
```

Maths: ACOS, ASIN, ATAN, COS, FACTORIAL, INTEGER, INTEGER-PART, LOG, LOG10, MOD, REM, SIN, SQRT, SUM, TAN
Stats: MEAN, MEDIAN, MIDRANGE, RANDOM, RANGE, STANDARD-DEVIATION, VARIANCE
Date/time: CURRENT-DATE, DATE-OF-INTEGER, DATE-TO-YYYYMMDD, DATEVAL, DAY-OF-INTEGER, DAY-OF-YYYYDD, INTEGER-OF-DATE, INTEGER-OF-DAY, UNDATE, WHEN-COMPILED, YEAR-TO-YYYY, YEARWINDOW, CURRENT-DATE
Chars: LENGTH, MAX, MIN, NUMVAL, NUMVAL-C, ORD-MAX, ORD-MIN, UPPER-CASE, LOWER-CASE, REVERSE