COBOL· Common business oriented language

1958 codasyl – pentagon

Dr· Grace hopper

Application programming language

English like language·

Add a to b giving c·

Divide a by b giving d remainder e·

Create a pds

HLQ·URNAME·REVAT·COBOL·PDS

LRECL=80,RECFM=FB,,DSORG=PDS

PQ=10,SQ=10,DB=10,

Create a member → COMPJCL→ type the job mentioned in the chat·

Create another → RUNJCL → type the next job mentioned in the chat·

CREATE A LIBRARY

HLQ·URNAME·REVAT·COBOL·LOADLIB

LRECL=80,RECFM=U,DSORG=LIBRARY

PQ=10,SQ=10,DB=10,

1· Write the cobol pgm
2· Submit a compile job on behalf of the cobol pgm·
   a· Step1 → check syntax errors in cobol stmnts
   b· Step2→ convert the cobol codes in loda modules/object code

3· Submit another the job that will run the load module·

PROGRAM STRUCTURE:

4 DIVISIONS·

1·IDENTIFICATION DIVISION·(mandatory)

Note: identifies the program·

PROGRAM-ID· Entry· mandatory
[AUTHOR· Entry]·
[INSTALLATION Entry]·
[DATE-WRITTEN· Entry]·
[DATA-COMPILED· Entry]·
[SECURITY· Entry·
[REMARKS· Entry·]

2· ENVIRONMENT DIVISION·(optional)

Note: Files and other external resources are linked here·

CONFIGURATION SECTION·
    SOURCE-COMPUTER· ZOS·
    OBJECT COMPUTER· 390·
INPUT-OUTPUT SECTION·
FILE CONTROL· → where the files are linked·

3·DATA DIVISION· (optional)

Note: all the variables are declared

FILE SECTION·
    FD – FILE'S FIELD DISCRIPTION (FILE LAYOUT/FILE VARIABLES)
WORKING-STORAGE SECTION·
    ALL THE COMMONLY USED VARIABLES
LOCAL-STORAGE SECTION·
    REPORTS
LINKAGE SECTION·
    SUB PROGRAMS AND to which the values can be received from the user dynamically

PROCEDURE DIVISION·(Mandatory)

Has no predefined sections or paragraphs.
Executable statements are written/verbs
Must be ended by 'stop run'.

COLUMN DISCRIPTION OF COBOL PROGRAM
1-6 – SEQNUM
7 –INDICATOR. SPACE → executable sentence
'*' → comment
'-' → continuation of string from previous sentence.
8-11 AREA/MARIGIN A
DIVISIONS, SECTIONS,PARAGRAPHS, FD,SD 01,77 ENTERIES.
12-72 –AREA/MARIGIN B
02-49,66,88 LEVEL NUMBERS. ALL EXECUTABLE SENTANCE IN PROCEDURE DIVISION(VERBS).
73-80 – IDENTIFIER – USERS DISCRETION.

Data types:
1. Char → A
2. Numeric → 9
3. Alphanumeric →X

DECLARE A VARIABLE

SYNTAX:

LEVEL-NUMBER SPACE NAME-VAR SPACES PICTURE-CLAUSE DATATYPE(SIZE) VALUE-CLAUSE

LEVEL NUMBER → 01,77,02-49,66,88
VAR-NAME (MAX 36). ALPHANUMERIC,-,FIRST CHAR MUST BE ALPHABET.
(let the name tell the story)
PIC MANDATORY
DATATYPE(SIZE)
A(05) → left justified with auto spaces on the suffix
9(05) → right justified with auto zeros prefixed
X(10) → left justified with auto spaces on the suffix
VALUE(OPTIONAL) → the user can assign the variable with some initial values.

Date: 29:09-2023

Topics to be covered:

- Singed variables
- Figurative constants
- Decimal variables
- Move statement
- Reference Modification
- Group items
- Arithmetic operations
- Conditions
- Conditional statements
- Loops.

Task of the day:

Max digits. Max chars → alphabetic, max chars → alphanumeric

Singed variables. Numeric.

Pic 9(03) → no sign

```
01 WS01-A       PIC S9(03) VALUE +123.   A23.
```

Note: Signed variable, by default is sign leading. The sign will be combined with the leading digit with the below convention.

01 WS01-A   PIC S9(03) **sign leading** VALUE +123.   A23

The sign will be combined with the leading digit with the below convention.

01 WS01-B   PIC S9(03) **sign trailing** VALUE +123.   12C.

Convention→ +1 → A, +2 →B... +9→ I, -1 → J, -2→ K...
-9 → R. +0 → {, -0 → }

01 WS01-STS   PIC S9(03) **sign trailing Separate** VALUE +123.
➔ 123+

01 WS01-SLS   PIC S9(03) **sign leading Separate** VALUE +123.
➔ +123

Figurative constants

Space, spaces, 0, zero,zeros,zeroes

Decimal variables.

Virtual → numeric → arithmetic operation. The decimal point will not be displayed. 123.456 → 123456

01 ws01-vd   PIC 9(03)V(02) value 123.45

But when you display → 12345

- Physical (alphanumeric) → no arithmetic operation allowed on this Variable.
- no value clause is allowed.
- Move statement is allowed

01 WS01-PD   PIC 9(03).9(02)

MOVE

Allocation of a value to a variable can be done 2 ways.

1. Value clause- data division.
2. Move statement in Procedure division.

   Syntax:

   Move source to dest-variables

   Move 10 to ws01-var1

   Move 'ca7' to ws05-name

   Move ws01-a to ws05-b (provided their data type are same)

   Move 10 to ws01-a, ws05-b,ws05-c

   Move ws05-d to ws01-a, ws05-b,ws05-c

Source → single. Value/literal or variables

Destination → single or multiple variables.

Reference Modification.

A technique by which parts of a variable can be handled individually.

Syntax : var(stpos:length)

01 ws01-a pic a(05) value 12345.

01 ws01-b pic 9(05) value 00000.

1	2	3	4	5

Move ws01-a(2:3) to ws01-b(3:3)

0	0	2	3	4

Group items.

01 - level → 1page-4kb

01 ws01-A pic x(03)

77 level number - elementary data item.

The number bytes allocated is the number bytes mentioned in the size clause.

Suggest: Declare one variable at 01 level number and declare all the other variables in the same page/under that 01 level number variable.

A group item is a variable which has member variables defined.

```
01 ws01-vars.
        02 ws02-var1          pic 9(03).
        02 ws02-var2          pic X(04).
        02 ws02-var3          pic A(15).
```

Note: level numbers 02 - 49 must be mentioned in marigin b

Rules of group item.

- It cannot have picture clause.
- It CAN HAVE value clause. Value are considered to alphanumeric.

- A group item is identified as a group item, if it has variables with higher level number.
- The member items can have their own different pic clauses and sizes.
- The size of a group item is calculated by adding the sizes of its members.

Rules of a member item.

- Must have a level number higher than its group item.
- Any higher level number is the member of its immediate previous lower level number.
- A member item can be a SUB GROUP item.

```
01 ws01-vars VALUE “TOMMY007USANJK991111’.
      05 WS05-NAME        PIC A(05).
      05 WS05-ID          PIC X(03).
      05 WS05-ADDRESS.
            10   WS10-COUNRTY       PIC A(3).
            10   WS10-STATE         PIC A(3).
            10   WS10-PCODE         PIC 9(02).
      05 WS05-PHONE       PIC 9(4).
```

```
PD
      DISPLAY → WS01-VARS.
```

Arithmetic operations:

Numeric data type.

1. ADD.

```
ADD 10 TO WS05-A → WS05-A = WS05-A + 10
ADD WS05-A TO WS05-B →  WS05-B = WS05-A + WS05-B
ADD WS05-A TO WS05-B WS05-C. →
            WS05-B = WS05-B + WS05-A
            WS05-C = WS05-C + WS05-A
ADD 10 TO WS05-B GIVING WS05-C
            WS05-C = WS05-B + 10.
```

2. SUBTRACT.

```
SUBTRACT 10 FROM WS05-A→ WS05-A = WS05-A - 10
```

SUBTRACT WS05-A FROM WS05-B → WS05-B = WS05-B - WS05-A

SUBTRACT WS05-A FROM WS05-B GIVING WS05-C

3. MULTIPLY.

MULTIPLY WS05-A BY WS05-B

MULTIPLY 10 BY WS05-B

MUTLIPLY WS05-A BY WS05-B GIVING WS05-C

4. DIVIDE

DIVIDE WS05-A BY WS05-B → WS05-B(QUO)= WS05-A / WS05-B

DIVIDE WS05-A BY WS05-B GIVING WS05-C REMAINDER WS05-D

5. COMPUTE. (NATURAL ARITHMATIC OPERATORS.)

C = A * B / F (- 23 + 2) **4

BODMAS RULE IS FOLLOWED

COMPUTE C = A * B / F (- 23 + 2) **4

COMPUTE C ROUNDED = (A * B) / 2 * (- 23 + 2) **4

Note: a Space must be given before and after the operator.

Conditions

- Relation Condition

  <, LESS THAN, >,GREATER THAN, EQUALS TO, =, <=,>=
- Sign Condition – POSITIVE NEGATIVE,ZERO
- Class Condition –ALPHABETIC, NUMERIC, ALPHABETIC-LOWER, ALPHABETIC-UPPER
- Condition-Name 88 level number declaration.
- Negated Condition - NOT
- Combined Condition – LOGICAL OPERATORS. AND & OR

Conditional statements

```
If ( CONDITION)   THEN
        IMP STMNT
ELSE IF(CONDITION)  THEN
        IMP STMNTS
ELSE
        IMP STMNTS
END-IF
END-IF.
```

Note: The number of end-if must be equal to the number of IF in the structure.

There MUST NOT BE A PERIOD ANYWHERE INBETWEEN IF AND END-IF

Evaluate

```
EVALUATE TRUE/FALSE/VARIABLE
WHEN CONDITION
        IMP
WHEN CONDITON
       IMP
WHEN OTHER
        IMP
END-EVALUATE.
```

EXAMPLE:

```
IF ( WS05-A > WS05-B AND WS05-A > WS05-C ) THEN
        DISPLAY 'A IS THE GREATEST'
ELSE IF ( WS05-B > WS05-A AND WS05-B > WS05-C ) THEN
        DISPLAY ' B IS THE GREATEST'
ELSE
        DISPLAY ' C IS THE GREATEST'
END-IF
END-IF.
```

```
EVALUATE TRUE
WHEN (WS05-A > WS05-B AND WS05-A > WS05-C )
      DISPLAY ' A IS THE GREATEST'
WHEN ( WS05-B > WS05-A AND WS05-B > WS05-C )
      DISPLAY ' B IS THE GREATEST'
WHEN OTHER
      DISPLAY ' C IS THE GFREATEST'
END-EVALUATE.
```

LOOPS. ITERATION STATEMENTS.

PERFORM

1. CONDITIONAL

   Until a condition is satisfied the loop run.

   a. Inline perform

      When the group of statements to be iterated lies between "perform' and 'end-perform.

```
WS05-A PIC 9(02) VALUE 5.
PERFORM UNTIL WS05-A > 10
      DISPLAY 'HI'
      Add 1 TO ws05-a
END-PERFORM.
```

Ans: 6 timed.

```
WS05-A PIC 9(02) VALUE 5.
PERFORM UNTIL WS05-A > 10
      DISPLAY 'HI'
END-PERFORM.
```

Ans: infinite

a. Infinity 722 abend

```
Ws05-a pic 9(02) value 5.
```

```
PERFORM VARYING WS05-A FROM 1 BY 1 UNTIL WS05-A > 10
     DISPLAY 'HI'
     DISPLAY WS05-A
END-PERFORM.
ANS : 10
```

b. Out-of-the-line

2. UNCONDITIONAL

a. Inline perform

```
PERFORM 5 TIMES
      DISPLAY ' HI'
END-PERFORM.
PERFORM
      DISPLAY ' HI'
END-PERFORM.
PERFORM WS05-A TIMES
      DISPLAY 'HI'
END-PERFORM
ANS:ws05-a times
PERFORM WS05-A TIMES
      DISPLAY 'HI'
      ADD 1 TO WS05-A
      DISPLAY WS05-A
END-PERFORM
```

b. Out-of-the-line

Dosn't check any condition to be satisfied.

CONTROL STATEMENTS.

CONTNIUE

Transfers the control to the next COBOL statement which come next in the program flow.

NEXT SENTENCE

Transfers control to the next COBOL statement, which is immediately after the sentence ending with period.