Resource Optimization - LOBOL

Margaret Brunner - Longs Drug Stores

## LOBOL - INTRODUCTION

SECTION I.

### A. WHAT LOBOL IS

Lobol is a high-level business-oriented interactive programming language designed for data entry. It is similar to COBOL, but takes advantage of the processor architecture to provide simplified programming and fast-running programs. It maximizes the power of the HP2645 or HP2649 terminals. Instead of using the resources of the HP3000 computer, the program runs within the terminal and builds its files in the terminal. It allows interactive manipulation and editing of both records and batches. Once the batch is clean and has been thoroughly edited, it is transmitted to the HP3000 for further processing.

## A. SIMILARITIES

1. Divisions - LOBOL has five divisions:
   See SECTION V.

2. Structure - LOBOL code consists of sections executed by PERFORM statements.
   See SECTION VI.

3. Reserved Word List - the similarity in the Reserved Word Lists of COBOL and LOBOL allows the programmer to master LOBOL very quickly.
   See SECTION V.

4. Techniques - all structured analysis, design, and programming techniques that are used for COBOL lend themselves to LOBOL.

5. Tables - LOBOL allows use of internal tables.
   See SECTION V.

6. Editing and Validation Capabilities - there is powerful string and numeric editing with pattern matching and range checking. See SECTION V.

7. Data Manipulation - LOBOL has extensive data manipulation capabilities similar to COBOL. See SECTION V.

B. DIFFERENCES

1. Operating Software - LOBOL requires an HP3000 COBOL interface program for downloading and file transfer. See SECTION IV.

2. Bootstrap Compilation - LOBOL is a high level language which requires processing into assembler code prior to assembling into actual object code. See SECTION IV.

3. Internal Files - LOBOL manipulates its files within the terminal memory. See SECTION V.

4. Screen Formatting - Simultaneous scrolling and fixed screen formats allowed. See SECTION V.

5. Field Lengths - All references to data fields or literals in code statements must include field lengths as parameters. See SECTION VI.

## A. IT IS EASY ON PROGRAMMERS

Working on the assumption that a majority of shops are COBOL oriented, LOBOL was designed to be as similar to COBOL as possible. Many of the statements are identical. Thus, it is quick and easy to learn.

LOBOL lends itself to modular programming practices:

1. Structured Programming
2. Top-down Development
3. HIPO (Hierarchy Input Process Output) documentation
4. Structured Walkthroughs

LOBOL is modular, simple, readable, and inexpensive to maintain.

B. IT IS VERY FRIENDLY TO INPUT OPERATORS

Because of the speed, the operators can never get ahead of the machine. Terminal response is immediate.

Features of LOBOL such as:

1. the ability to have a mask on the screen and a scrolling section on the screen at the same time,

2. the ability to manipulate a file of information in whatever way desired and as often as desired before releasing the file to the mainframe,

3. the ability to accept any data and redisplay it fully edited,

4. the ability to maintain a complete and automatic dialogue with the operator until the data is correct,

give the operator full control over the data entered. LOBOL pays attention to the needs of the people who input and make use of the LOBOL program.

C. LOBOL is easy on the HP3000

It allows the HP3000 to operate more efficiently because the program resides in the HP2645 or HP2649 terminal. The data file is kept in the terminal until the operator feels satisfied that all the data is clean. Only at that time is the file sent to the HP3000. Therefore, CPU utilization on the HP3000 is remarkably low.

The terminal does all the data entry handling and editing, thus providing very fast response time and very high throughput. Each terminal handles its' own program while the HP3000 acts only as a host to the data files sent to it. The HP3000 resources are barely tapped, thus expanding other usage of the HP3000.

LOBOL is a powerful, easy to use, flexible tool which will save you time, effort, and expense.

A. On the HP3000

1. Type in the English language style LOBOL instructions.

2. Run the LOBOL compiler which processes these instructions into 8080 assembler code.

3. Run the 8080 compiler which assembles the assembler code into machine readable code.

4. Create a COBOL program that calls LOBOL intrinsics to download the LOBOL program into the terminal and start the execution of the LOBOL program. (This COBOL program will also receive the clean files from the terminal.)

B. On the HP2645 or HP2649 terminal

5. The terminal operator initiates a session and types in the run command for the COBOL program.

6. The COBOL program downloads the LOBOL program and execution of the LOBOL program starts.

A. DESCRIPTION DIVISION (the description of the program is included in this division)

1. NAME
2. VERSION
3. COMPILED

B. -EXTERNALS DIVISION- (the linkage to other code which must be loaded with the program (such as operating software) is defined in this division)

1. ENTRY
2. INCLUDE

C. FILES DIVISION (definitions of all memory files in the program are specified in this division)

1. FILE
2. SPACE
3. RECORD

D. DATA DIVISION (all constants, work areas, display lines are specified in this division)

1. VALUE
2. DATA
3. FIELD
4. REDEFINES

E. PROCEDURES DIVISION (all executable code is present in this division)

Structure Statements
1. SECTION
2. EXIT
3. END

Control Statements
1. PERFORM
2. GOTO
3. COMP (equal, unequal, greater, or less)
4. STOP

Keyboard and Display Statements
1. KEYIN
2. DISPLAY
3. CLEARLINE
4. CLRSCREEN
5. ROLLSCREEN

Data Manipulation Statements
1. MOVE
2. PACK
3. UNPACK
4. ADD
5. SUBTRACT
6. LEFTZERO
7. SPREAD
8. LEFTSPACE
9. EDIT
10. SEARCH

Validation Statements
1. VALNUMERIC
2. VALCKDIGIT

File Management Statements
1. OPEN
2. CLEAR
3. RESET
4. READ
5. WRITE
6. REPLACE
7. DELETE

8. READBACK
9. INSERTB (insert a record before another)
10. INSERTA (insert a record after another)

Data Transmittal Statements

1. SEND
2. SENDFILE

Miscellaneous Statements

1. TITLE
2. PAGE
3. BEEP

COBOL intrinsics for LOBOL

1. LOADPROG
2. RECEIVE
3. CLOSETERM

```
DESCRIPTION DIVISION

        NAME SAMPLE

        COMPILED

        VERSION 02/14/80

@REMARKS:
@
@       TITLE = SAMPLE
@
@       AUTHOR = MARGARET BRUNNER
@
@       USER DEPARTMENT = 1980 HP CONVENTION
@
@       PROGRAM NARRATIVE =
@            THIS PROGRAM IS A SAMPLE PROGRAM TO DEMONSTRATE
@       LOBOL CODE.  THE PROGRAM HAS BOTH A FIXED FORMAT SCREEN
@       (TOP 5 LINES OF THE SCREEN) AND A SCROLLING FORMAT
@       (SCROLL FROM LINE 24 OF THE SCREEN UP TO LINE 5 OF THE
@       SCREEN).  THE PROGRAM ACCEPTS INPUT OF 3 FIELDS,
@       VALIDATES THEM AND REDISPLAYS THEM IN EDITED FORMAT.
@       AFTER ACCEPTING INPUT OF THE 3 FIELDS, IT ACCEPTS INPUT
@       OF AN OPTION CODE THAT SPECIFIES WHAT TYPE OF MANIPULA-
@       TION OF THE MEMORY FILE IS TO BE DONE.  THE OPTIONS IN
@       THIS PROGRAM ARE TO WRITE THE 3 INPUT FIELDS TO THE
@       MEMORY FILE, TO RESET THE MEMORY FILE TO THE BEGINNING
@       OF THE FILE, TO READ A RECORD FROM THE MEMORY FILE, TO
@       SEND THE ENTIRE MEMORY FILE TO THE HP3000.  THE OPTION
@       CODE ALSO ALLOWS INPUT OF A CODE TO RETURN FOR MORE
@       INPUT OF DATA OR THE END THE PROGRAM.
@
```

```
 TITLE @@@@@@@@@@@@@@     EXTERNALS DIVISION     @@@@@

EXTERNALS DIVISION

        ENTRY START  @DENOTES BEGINNING OF EXECUTABLE CODE
        INCLUDE LOB49EPT.UTLS.SYS @FILE W/OPERATING SYSTEM ENTRY POINTS

 TITLE @@@@@@@@@@@@@@      FILES DIVISION     @@@@@@@

FILES DIVISION

DATAFILE FILE                       @ OUTPUT/INPUT FILE
         SPACE 1900 @RESERVES 1900 BYTES IN MEMORY FOR THE FILE
         RECORD 19

 TITLE @@@@@     DATA DIVISION - RECORD DESCRIPTION     @@@@@@

DATA DIVISION

REC      DATA @THIS DEFINES THE RECORD AREA (COMPARABLE TO 01 IN COBOL)
VENDOR     FIELD 6
DATE       FIELD 6
GROSS      FIELD 7
```

```
 TITLE @@@@@     DATA DIVISION - INPUT AREAS     @@@@@@@@@

OPTION   FIELD 1          @INPUT FIELD FOR OPTION CODE

 TITLE @@@@@     DATA DIVISION - SAVE AREAS     @@@@@@@@@@

MMDDYY   DATA 6                  @ HOLD AREA FOR DATE TO BE VALIDATED
MM       FIELD 2
DD       FIELD 2
YY       FIELD 2

DUMMY  FIELD 2                 @WORK AREA FOR DATE VALIDATION

MDYDISP  FIELD 8                    @ DISPLAY FIELD FOR MDYMASK
GROSDISP FIELD 9                 @DISPLAY FIELD FOR GROSS AMOUNT

SVENDOR  FIELD 6                @SAVE AREA FOR VENDOR NUMBER
SDATE    FIELD 6                @SAVE AREA FOR DATE
SGROSS   FIELD 7                @SAVE AREA FOR GROSS AMOUNT

 TITLE @@@@@     DATA DIVISION - CONSTANTS     @@@@@@@@@@

KREAD    VALUE '1'               @TO CHECK IF OPTION=READ
KWRITE   VALUE '2'               @TO CHECK IF OPTION=WRITE
KRETURN  VALUE '3'               @TO CHECK IF OPTION=RETURN
KRESET   VALUE '4'               @TO CHECK IF OPTION=RESET
KSEND    VALUE '5'               @TO CHECK IF OPTION=SEND
KEND     VALUE '6'               @TO CHECK IF OPTION=END

 TITLE @@@@@     DATA DIVISION - SCREEN DISPLAYS     @@@@@@@

HTITLE   VALUE 'SAMPLE LOBOL PROGRAM'   @HEADING WITH PROGRAM TITLE
HLIST1 VALUE 'OPTIONS:    1=READ    2=WRITE    3=RETURN'
HLIST2 VALUE '4=RESET    5=SEND    6=END'
HINPUT   VALUE 'VENDOR     DATE       GROSS'  @HEADING FOR INPUT FIELDS
HOPTION  VALUE 'OPTION'            @OPTION FIELD HEADING
HFILE    VALUE 'MEMORY FILE RECORD' @HEADING FOR DISPLAY OF FILE RECORD

 TITLE @@@@@     DATA DIVISION - EDIT MASKS     @@@@@@@@@@

MDYMASK  VALUE 'XX/XX/XX'          @ EDITED FORMAT FOR DATE
GROSMASK VALUE 'ZZ,ZZZ.99'   @EDITED FORMAT FOR GROSS AMOUNT
```

```
 PAGE
 TITLE @@@@     PROCEDURES DIVISION - START SECTION     @@@@@@

PROCEDURES DIVISION

START SECTION

        PERFORM INIT
        PERFORM MAINLINE
        PERFORM ENDJOB

        STOP

        EXIT

 TITLE @@@@     PROCEDURES DIVISION - INIT SECTION      @@@@@@

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

@  INIT
@       THIS SECTION WILL OPEN THE MEMORY FILE AND
@       PERFORM THE SECTION TO DISPLAY THE HEADINGS ON THE SCREEN
@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

INIT    SECTION

        OPEN DATAFILE
        PERFORM HEADING

        EXIT
```

```
 TITLE @@@@     PROCEDURES DIVISION - MAINLINE SECTION    @@@@@

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

@  MAINLINE
@           THIS SECTION WILL PERFORM ALL THE SECTIONS THAT
@           WILL ACCEPT INPUT OF DATA AND THE SECTION
@           THAT WILL ACCEPT INSTRUCTIONS AS TO THE FILE
@           MANIPULATION.
@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

MAINLINE SECTION

$1         PERFORM INVENDR
           PERFORM INDATE
           PERFORM INGROSS
           PERFORM INOPTON
           COMPE OPTION,KRETURN,1,$1

           EXIT

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

@  ENDJOB
@          THIS SECTION CLEARS THE SCREEN PRIOR TO ENDING THE
@          PROGRAM.
@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

 TITLE @@@@     PROCEDURES DIVISION - ENDJOB SECTION     @@@@@@

ENDJOB     SECTION

          CLRSCREEN 1,1

          EXIT
```

```
 TITLE @@@@     PROCEDURES DIVISION - DATA ENTRY ROUTINES  @@@@

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

@  INVENDR
@         THIS SECTION ACCEPTS INPUT OF A VENDOR NUMBER AND
@         WILL VALIDATE IT AS CONTAINING ONLY NUMBERS. THE
@         SECTION LOOPS UNTIL A VALID VENDOR NUMBER IS INPUT.
@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

INVENDR SECTION

$1       KEYIN 24,5,SVENDOR,6
         LEFTZERO SVENDOR,SVENDOR,6
         VALNUMERIC SVENDOR,6,$2
         GOTO EXIT
$2       BEEP
         GOTO $1

         EXIT

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

@  INDATE
@         THIS SECTION WILL ACCEPT A DATE IN MMDDYY FORMAT AND
@         WILL PERFORM A SECTION TO VALIDATE IT. IF IT IS A
@         VALID DATE, IT IS REDISPLAYED IN EDITED FORMAT OF
@         MM/DD/YY. THE SECTION LOOPS UNTIL A VALID DATE IS
@         INPUT.
@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

INDATE  SECTION

$1        KEYIN 24,14,SDATE,6
          VALNUMERIC SDATE,6,$2
          MOVE SDATE,MMDDYY,6
          PERFORM DATVAL
          COMPE MMDDYY,'000000',6,$1
          EDIT SDATE,MDYMASK,MDYDISP,8
          DISPLAY 24,14,MDYDISP,8
          GOTO EXIT
$2        BEEP
          GOTO $1

          EXIT
```

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

@  INGROSS
@          THIS SECTION ACCEPTS IN UT OF A GROSS AMOUNT IN
@          9999999 FORMAT. THE DAT  IS VALIDATED FOR NUMBERS.
@          THE ASSUMPTION IS MADE HAT THE LAST TWO DIGITS ARE
@          DECIMALS AND THE DATA I  EDITED INTO A ZZ,ZZZ.99
@          FORMAT AND REDISPLAYED  N THE SCREEN.  THE SECTION LOOPS
@          UNTIL A VALID AMOUNT IS INPUT.
@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

INGROSS   SECTION

$1        KEYIN 24,25,SGROSS,7
          LEFTZERO SGROSS,SGROSS,7
          VALNUMERIC SGROSS,7,$2
          EDIT SGROSS,GROSMASK,GROSDISP,7,9
          DISPLAY 24,25,GROSDISP,9
          GOTO EXIT
$2        BEEP
          GOTO $1

          EXIT
```

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

@  INOPTON
@          THIS SECTION ACCEPTS INPUT OF THE OPTION CODES TO
@          MANIPULATE THE MEMORY FILE. THESE CODES ARE:
@          1 TO READ A RECORD FROM THE MEMORY FILE
@          2 TO WRITE THE INPUT DATA TO THE MEMORY FILE
@          3 TO RETURN THE CURSOR TO ACCEPT MORE INPUT DATA
@          4 TO RESET THE MEMORY FILE TO THE FIRST RECORD IN IT
@          5 TO SEND THE MEMORY FILE TO THE HP 3000 COMPUTER
@          6 TO END THE PROGRAM
@          THE SECTION LOOPS UNTIL A VALID CODE IS ENTERED.
@          ALL THE INPUT DATA IS ACCEPTED ON THE BOTTOM LINE
@          OF THE SCREEN, THEN THE SCREEN FROM ROWS 6 TO 24
@          IS SCROLLED UP 1 LINE.
@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

INOPTON  SECTION

$1       KEYIN 24,41,OPTION,1
         COMPE OPTION,KREAD,1,$2
         COMPE OPTION,KWRITE,1,$3
         COMPE OPTION,KRETURN,1,$4
         COMPE OPTION,KRESET,1,$5
         COMPE OPTION,KSEND,1,$6
         COMPE OPTION,KEND,1,EXIT
         BEEP
         GOTO $1
$2       PERFORM OREAD
         GOTO $1
$3       PERFORM OWRITE
         GOTO $1
$4       ROLLSCREEN 5
         GOTO EXIT
$5       PERFORM ORESET
         GOTO $1
$6       PERFORM OSEND
         GOTO $1

         EXIT
```

```
 TITLE @@@@     PROCEDURES DIVISION - MEMORY FILE MANIPULATION  @@

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

@  OREAD
@       THIS SECTION READS A RECORD FROM THE MEMORY FILE, EDITS THE
@       FIELDS IT READ, AND DISPLAYS THEM IN AN EDITED FORMAT ON
@       THE SCREEN. IF THE END OF THE MEMORY FILE IS ENCOUNTERED
@       IN THE READ, AN ERROR MESSAGE IS DISPLAYED ON THE SCREEN.
@       IT ROLLS THE SCREEN UP ONE LINE.
@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

OREAD    SECTION

         READ DATAFILE,REC,$1
         EDIT DATE,MDYMASK,MDYDISP,6,8
         EDIT GROSS,GROSMASK,GROSDISP,7,9
         DISPLAY 24,48,VENDOR,6
         DISPLAY 24,57,MDYDISP,8
         DISPLAY 24,68,GROSDISP,9
         ROLLSCREEN 5
         GOTO EXIT
$1       BEEP
         DISPLAY 24,48,'<<< END OF FILE >>>',19

         EXIT

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

@  OWRITE
@       THIS SECTION MOVES THE INPUT FIELDS TO THE OUTPUT
@       FIELDS AND WRITES OUT ONE RECORD AT THE END OF THE
@       MEMORY FILE. IT ROLLS THE SCREEN UP ONE LINE.
@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

OWRITE   SECTION

         MOVE SVENDOR,VENDOR,6
         MOVE SDATE,DATE,6
         MOVE SGROSS,GROSS,7
         WRITE DATAFILE,REC
         ROLLSCREEN 5

         EXIT
```

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

@  ORESET
@         THIS SECTION CLOSES THE FILE AND REOPENS IT AS INPUT
@         TO ALLOW IT TO BE READ. IT ROLLS THE SCREEN UP ONE LINE
@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

ORESET    SECTION

          RESET DATAFILE
          ROLLSCREEN 5

          EXIT

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

@  OSEND
@         THIS SECTION SENDS THE ENTIRE MEMORY FILE TO THE
@         HP3000 COMPUTER.  IT THEN CLOSES THE FILE AND OPENS
@         IT FOR OUTPUT AGAIN.  THE SCREEN ROLLS UP ONE LINE
@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

OSEND     SECTION

          SENDFILE DATAFILE
          CLEAR DATAFILE
          ROLLSCREEN 5

          EXIT
```

```
 TITLE @@@@     PROCEDURES DIVISION - MISCELLANEOUS ROUTINES       @

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

@  HEADING
@          THIS SECTION CLEARS THE SCREEN INITIALLY
@          AND DISPLAYS ON THE TOP FIVE LINES OF THE SCREEN
@          WHAT IS TO BE THE FIXED PORTION OF THE SCREEN:
@          LINE 1 WILL CONTAIN THE PROGRAM TITLE
@          LINE 3 WILL CONTAIN THE LIST OF OPTIONS FOR
@                 FILE MANIPULATION
@          LINE 5 WILL CONTAIN THE COLUMN HEADINGS
@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

HEADING SECTION

          CLRSCREEN 1,1
          DISPLAY 1,26,HTITLE,20
          DISPLAY 3,5,HLIST1,41
          DISPLAY 3,50,HLIST2,26
          DISPLAY 5,5,HINPUT,27
          DISPLAY 5,38,HOPTION,6
          DISPLAY 5,53,HFILE,18

          EXIT
```

```
 TITLE @@@@     PROCEDURES DIVISION - VALIDATION ROUTINES      @@@

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

@  DATVAL
@
@       THIS ROUTINE, TOGETHER WITH "MODFOUR" COMPLETELY
@       VALIDATES ANY DATE IN THE NEXT 100 YEARS, INCLUDING
@       AN ALLOWANCE FOR LEAP YEARS. IT EXPECTS TO GET THE DATE
@       IN 3 FIELDS "MM", "DD" AND "YY". IT ALSO USES
@       A FIELD CALLED "DUMMY" FOR CALCULATION. ALL
@       CONSTANTS ARE LITERALS IN THE CODE SO THAT THESE
@       TWO SECTIONS ARE AS SELF-CONTAINED AS POSSIBLE.
@       THE DATE FIELD IS LEFT UNTOUCHED IF THE DATE
@       IS VALID; OTHERWISE, A BEEP OCCURS AND ZEROES ARE
@       MOVED TO THE THREE DATE FIELDS.
@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

DATVAL SECTION

        COMPL MM,'01',2,$C
        COMPL DD,'01',2,$C
        COMPG MM,'12',2,$C
        COMPE MM,'02',2,$F      @ FEBRUARY SPECIAL ROUTINE
        COMPE MM,'04',2,$S      @ 30-DAY MONTH
        COMPE MM,'06',2,$S      @ 30-DAY MONTH
        COMPE MM,'09',2,$S      @ 30-DAY MONTH
        COMPE MM,'11',2,$S      @ 30-DAY MONTH
        COMPG DD,'31',2,$C
        GOTO EXIT
$S      COMPG DD,'30',2,$C
        GOTO EXIT
$F      MOVE YY,DUMMY,2
        PERFORM MODFOUR
        COMPE DUMMY,'00',2,$L
        COMPG DD,'28',2,$C
        GOTO EXIT
$L      COMPG DD,'29',2,$C
        GOTO EXIT
$C      MOVE '00',MM,2
        MOVE '00',DD,2
        MOVE '00',YY,2
        BEEP

        EXIT
```

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

@ MODFOUR

@       THIS SECTION DIVIDES DUMMY BY FOUR AND PLACES
@       "01" IN DUMMY IF THE REMAINDER IS NON-ZERO. A
@       ZERO REMAINDER INDICATES A LEAP-YEAR.
@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

MODFOUR SECTION

$A      COMPL DUMMY,'04',2,$E
        SUBTRACT '04',DUMMY,2
        GOTO $A
$E      COMPE DUMMY,'00',2,EXIT
        MOVE '01',DUMMY,2

        EXIT


        END START
```