188 MVS update

May 2002

In this issue

MVS Update

Published by

Xephon
27-35 London Road
Newbury
Berkshire RG14 1JL
England
Telephone: 01635 38342
From USA: 01144 1635 38342
E-mail: trevore@xephon.com

North American office

Xephon
PO Box 350100
Westminster, CO 80035-0100
USA
Telephone: 303 410 9344

Subscriptions and back-issues

A year's subscription to *MVS Update*, comprising twelve monthly issues, costs £340.00 in the UK; $505.00 in the USA and Canada; £346.00 in Europe; £352.00 in Australasia and Japan; and £350.00 elsewhere. In all cases the price includes postage. Individual issues, starting with the January 1998 issue, are available separately to subscribers for £29.00 ($43.50) each including postage.

MVS Update on-line

Code from *MVS Update*, and complete issues in Acrobat PDF format, can be downloaded from our Web site at http://www.xephon.com/mvs; you will need to supply a word from the printed issue.

Editor

Trevor Eddolls

Disclaimer

Readers are cautioned that, although the information in this journal is presented in good faith, neither Xephon nor the organizations or individuals that supplied information in this journal give any warranty or make any representations as to the accuracy of the material it contains. Neither Xephon nor the contributing organizations or individuals accept any liability of any kind howsoever arising out of the use of such material. Readers should satisfy themselves as to the correctness and relevance to their circumstances of all advice, information, code, JCL, EXECs, and other contents of this journal before making any use of it.

Contributions

When Xephon is given copyright, articles published in *CICS Update* are paid for at the rate of £170 ($260) per 1000 words and £100 ($160) per 100 lines of code for the first 200 lines of original material. The remaining code is paid for at the rate of £50 ($80) per 100 lines. In addition, there is a flat fee of £30 ($50) per article. To find out more about contributing an article, without any obligation, please download a copy of our *Notes for Contributors* from www.xephon.com/nfc.

Printed in England.

Test REXX EXECs while editing them

Many environments containing an integrated editor allow the program being edited to run without leaving the editor and without saving it. The program presented here is an attempt to do the same thing while editing a REXX program. All you have to do is to type ‘RTEST’ (REXX test) in the editor prompt, optionally followed by the parameters that the edited program expects.

This way, the program is run, and you can quickly see whether it behaves the way you want, whether it contains syntax errors, etc. What RTEST does is to create a temporary member in the current PDS with a name that doesn’t already exist, save the contents of the editor into it, run it with the specified parameters, and delete the temporary member afterwards.

When the test run finishes, you are back in the editor as you were before. The real member is not saved at all.

I have never tried RTEST with CLISTs, because I hardly use them, but it probably works just as well.

```
/*== REXX * ISPF EDITOR ==========================================*/
/*                                                                */
/* RTEST - Test REXX programs while being edited. Invoke RTEST at */
/* the editor prompt, passing it the arguments for the            */
/* EXEC to run.                                                   */
/* RTEST creates a temporary member in the current PDS with the   */
/* content of the editor, executes it, and then deletes that member. */
/* The actual member being edited is not saved and remains as it is */
/* Temporary member names are based on the userid, in order to be */
/* unique, and are checked for existence before being created.    */
/*                                                                */
/*================================================================*/

address ispexec
'isredit macro (ARG)' /* get args for test */
'isredit (PDS) = dataset' /* get current pdsname*/
call generate_tempname
'isredit create' tempname .zfirst .zlast /* create temp member */
address tso exec "'"pds"("tempname")' '"arg"'" /* exec REXX with args*/

'lminit dataid('ddd') dataset('pds') enq(shrw)' /* create a dataid */
```

```
'lmopen dataid('ddd') option(output)' /* in order to del */
'lmmdel dataid('ddd') member('tempname')' /* the temp member */
'lmclose dataid('ddd')'
exit


generate_tempname: /* generate tempname */
tempname = 'ABC'right(userid(),3) /* based on userid() */
do x = 10 to 20 /* and suffix number */
tempname = left(tempname,6)||x /* that is increased */
fulltemp = "'"pds"("tempname")'" /* until an unused one is found */
xx = sysdsn(fulltemp) /* name is achieved */
if xx = 'MEMBER NOT FOUND' then return
end
say 'Member temporary name not generated - RTEST cancelled'
exit
```

Systems Programmer (Portugal)

Interface for ISPF application testing

This is a very simple and easy interface to test new versions of ISPF applications that already exist in a TSO concatenation without involving the existing versions or the connected users currently running them.

INTRODUCTION

On our site I developed a particular form of configuration management product for ISPF and all related objects, such as CLIST/REXX EXECs, panels, skeletons, tables, and messages.

In this way I created a repository, with wide-ranging functions set to deal with the objects. If the object is a REXX EXEC, it is possible (the default) to compile it with the IBM REXX/370 compiler to speed up run-time execution and optimize the source code. This option is available only for the final phase because the other steps all use the language processor with interpreted source code.

I decided to implement in my object's life-cycle two different layers of testing – a personal TEST and a global TEST. Before testing there is the development phase, consisting of object definitions in the repository

(if new) and writing the necessary code (using whatever management interface is provided at your site).

After testing is completed, there will be a consolidation of the changes into the production environment.

PERSONAL TEST

In order to write this article, I extracted the TEST component from the configuration management product and made it able to run stand-alone, without the entire suite around it.

The test component is simple to describe. It's a user interface for creating, editing, managing (deleting too), and above all executing complete ISPF applications.

Let me explain quickly using an example.

You have a CLIST displaying a panel and a skeleton to be filled; these objects are present on your system and many users are probably accessing them. Normally, if you want to test them before deployment, you are compelled to create an object (panel or skeleton) with a different name in a PDS in the logon concatenation, and refer to it after modifying your script (CLIST/REXX EXEC), inserting some IF statements to your userID, eg:

```
.........
ADDRESS ISPEXEC
IF ZUser = MyUser THEN 'DISPLAY PANEL(NEWPAN1) '
                  ELSE 'DISPLAY PANEL(PAN1) '
.........
```

You do the same thing if you need to read a different table (but with the same name) or if you want to display a new set of messages.

There are other possible ways to achieve the same result – for example, we could work on a personal logon concatenation using logon scripts that we could provide directly to developers, etc.

However, panels and skeleton are often not so easy to update because they may be stored in user address spaces for future reference, and sometimes it can be difficult to quickly restore a previous version, if required.

These are good reasons for using the TEST interface.

HOW DOES IT WORK?

Note that I assume two particular conditions:

- No distinction is made between CLISTs and REXX EXECs. From now on I will use 'EXEC' for both kinds of objects.
- Developers must have been created a specifically-named set of libraries. These are PDSs (or PDSEs) called *USERID.SOURCE.ISRxLIB*.

A description of the contents can be found directly on the panel itself.

First of all a panel is displayed to allow users to enter input parameters.

There are only two input fields – UserID (prefix) and ExecName.

UserID

UserID is the first-level qualifier for ISPF user datasets (libraries). You can run tests even with a dataset prefix that's different from your own. Of course, in that case, all the source libraries with that prefix will be allocated for you to execute, while yours will not be available.

You are free to change these defaults, adding personal user libraries (in cases where the prefix is not equal to the userID) before or after the library specified in the panel field. I also provided a check-point to test whether a userID is valid or not, according to your site's standards. You can use it or not, it's not important.

The TEST application will search for all the datasets and will allocate them based on the DDName, if present, just before running the EXEC required. If incorrect, a message will be issued to the user.

Only the SOURCE.ISRCLIB is needed. Others are optional.

ExecName

ExecName is the name of the CLIST/REXX EXEC you want to be executed in TEST.

You can enter it in two ways – the full-name or a partial-name.

If you use the full-name, this means that no special characters are used. The TEST application will search for the EXEC and will run it. If it is not found, you will be asked whether you want to create it. If you choose 'Y' (YES), after completing the editing of it, you will be prompted whether to run the EXEC just created.

If you use a partial-name, you can enter special characters ('%' or '*'), or if you leave the field 'blank' the application builds a list of matching EXECs in your library with the associated information in a secondary panel. From that panel you are allowed to issue the following commands:

- V – allows you to view the EXEC in a customized panel (optional).
- E – the same as 'V' for EDIT.
- B – browse the EXEC in a customized panel.

  If you like, you may implement this solution. In that case you will have to install the TESTBROW EXEC and a related panel (optional). Since the BROWSE process in a POPUP is not allowed in full screen mode, you need to issue BROWSE as an external command. SUSPEND mode must be active (for exiting POPUP). For VIEWing and EDITing I found no problems with ISPF 2.10. Maybe you need to do the same with older versions.
- D – deletes the EXEC from a library following a confirm request.
- S – run the EXEC selected.

The EDIT panel is a customized one derived from ISREFR04. It's interesting, if you've never seen one, to learn how they work. I suggest that you read the chapter about *Customizing Edit and Browse Panels* in the *ISPF Planning and Customizing* book.

Note: the system is based on a sequence of ALTLIB and LIBDEF commands. Use of these commands in an EXEC scheduled by TEST may result in errors. I suggest you avoid library definitions in EXECs except main ones. So you will be able to easily comment them when running in TEST.

REQUIREMENTS

The application is entirely written in REXX and may run under ISPF. We have ISPF for OS/390 02.10.00 installed on our system, but I suppose it can run without problems on earlier versions, too. No other requirements exist.

PANEL TESTCONF

```
)ATTR DEFAULT(@{})
 ¦ TYPE(INPUT) INTENS(HIGH)
 % TYPE(INPUT) INTENS(HIGH) COLOR(RED) PAD('_') caps(on)
 | TYPE(INPUT) INTENS(HIGH) COLOR(pink) PAD('_') caps(on)
 + TYPE(TEXT) INTENS(HIGH) COLOR(YELLOW) SKIP(ON)
 $ TYPE(TEXT) INTENS(LOW) COLOR(red)
 { TYPE(TEXT) INTENS(HIGH) COLOR(blue)
 # TYPE(TEXT) INTENS(HIGH) COLOR(turq)
 ^ TYPE(OUTPUT) INTENS(low) COLOR(blue) caps(off) just(asis)
 } TYPE(OUTPUT) COLOR(turq) caps(off) just(asis)
)BODY WINDOW(54,6)
+
}Z                                                    +
^Z                                                    +
^Z                                                    +
+
+                 +Confirm?+==>|Z+
)INIT
  .ZVARS = '(YNMsg1,YNMsg2,YNMsg3,ConfInp)'
  &ZWinTtl = 'Warning !!'
)PROC
)END
```

PANEL TESTEDIT

```
)PANEL KEYLIST(ISRSPEC ISR)
)ATTR DEFAULT(%+_) FORMAT(MIX)
^ TYPE(text) INTENS(low) COLOR(blue) SKIP(on)
? TYPE(text) INTENS(high) COLOR(yellow) SKIP(on)
] TYPE(output) INTENS(high) CAPS(off) JUST(left) COLOR(turq) skip(on)
[ TYPE(output) INTENS(high) CAPS(off) JUST(right) COLOR(blue) skip(on)
¢ TYPE(output) INTENS(low) CAPS(off) COLOR(yellow)
£ TYPE(output) INTENS(low) CAPS(off) COLOR(red)
 ~ TYPE(AB)
 ' TYPE(ABSL)
 1B TYPE(PT)
 ! TYPE(FP)
 @ TYPE(NT)
```

```
 # TYPE(NEF) PADC(USER)
 * TYPE(VOI) PADC(USER)
 _ AREA(DYNAMIC) EXTEND(ON) SCROLL(ON) USERMOD('2Ø')
 Ø1 TYPE(DATAOUT) INTENS(LOW)
 Ø2 TYPE(DATAOUT)
 Ø3 TYPE(DATAOUT) SKIP(ON)
 Ø4 TYPE(DATAIN) INTENS(LOW) CAPS(OFF) FORMAT(&MIXED)
 Ø5 TYPE(DATAIN) CAPS(OFF) FORMAT(&MIXED)
 Ø6 TYPE(DATAIN) INTENS(LOW) CAPS(IN) COLOR(&ECOLOR) FORMAT(&MIXED)
 Ø7 TYPE(DATAIN) CAPS(IN) FORMAT(&MIXED)
 Ø8 TYPE(DATAIN) INTENS(LOW) FORMAT(DBCS)
 Ø9 TYPE(DATAIN) INTENS(LOW) FORMAT(EBCDIC)
 ØA TYPE(DATAIN) INTENS(LOW) FORMAT(&MIXED)
 ØB TYPE(DATAIN) INTENS(LOW) CAPS(IN)  COLOR(&ZPLEXCLR) FORMAT(&MIXED)
 ØC TYPE(DATAIN) INTENS(LOW) CAPS(OFF) COLOR(&ZPLEXCLR) FORMAT(&MIXED)
 ØD TYPE(DATAIN) INTENS(LOW) CAPS(IN)  COLOR(BLUE)      FORMAT(&MIXED)
 2Ø TYPE(DATAIN) INTENS(LOW) CAPS(IN) FORMAT(&MIXED)
 Z  TYPE(CHAR) COLOR(PINK) HILITE(REVERSE)
 R  TYPE(CHAR) COLOR(RED)
 G  TYPE(CHAR) COLOR(GREEN)
 B  TYPE(CHAR) COLOR(BLUE)
 W  TYPE(CHAR) COLOR(WHITE)
 P  TYPE(CHAR) COLOR(PINK)
 Y  TYPE(CHAR) COLOR(YELLOW)
 T  TYPE(CHAR) COLOR(TURQ)
 L  TYPE(CHAR) COLOR(RED)
 K  TYPE(CHAR) COLOR(&ZCK) HILITE(&ZHK)
 O  TYPE(CHAR) COLOR(&ZCO) HILITE(&ZHO)
 Q  TYPE(CHAR) COLOR(&ZCQ) HILITE(&ZHQ)
 C  TYPE(CHAR) COLOR(&ZCC) HILITE(&ZHC)
 V  TYPE(CHAR) COLOR(&ZCV) HILITE(&ZHV)
 D  TYPE(CHAR) COLOR(&ZCD) HILITE(&ZHD)
 F  TYPE(CHAR) COLOR(&ZCF) HILITE(&ZHF)
 S  TYPE(CHAR) COLOR(&ZCS) HILITE(&ZHS)
)BODY EXPAND(//) WIDTH(&ZWIDTH) CMD(ZCMD) LMSG(LMsg)
%Command >_Z                              %Cols >£Z    £Z    %Scroll >_Z
^-------------------------------------------------------------------------
[TestLabl          ]ZMemb                                ?PF3^Back to TEST
^Related Source DS:]ZDsn                    ¢LMsg?CAN^Cancel (no SAVE)
^-------------------------------------------------------------------------
_ZData,ZShadow                             / /
_
_                                          / /
_
)INIT
.ZVARS = '(ZCmd ZCL ZCR ZScEd)'
 &ZPm3 = Ø
 IF (&ZVModeT = 'VIEW') .HELP = ISR10000       /* HELP default panel */
 ELSE                   .HELP = ISR20000
 IF (&ZVModeT = 'VIEW') &EColor = 'BLUE'    /* Colour of row numbers */
```

```
 ELSE                    &EColor = 'GREEN'
 IF (&ZPdMix = N) &Mixed = EBCDIC    /*IF required SET format EBCDIC */
 ELSE             &Mixed = MIX                    /* SET format MIX */
 .ATTR(ZScEd) = 'CAPS(ON)'
 VGET (ZScEd) PROFILE
 IF (&ZScEd = ' ') &ZScEd = 'PAGE'
)REINIT
 REFRESH(*)
 IF (&ZVModeT = 'VIEW') .HELP = ISR10000
 ELSE                    .HELP = ISR20000
)PROC
 REFRESH(*)
 &ZCursor = .Cursor
 &ZCsrOff = .CsrPos
 VPUT (ZScEd) PROFILE
 &ZLvLine = LvLine(ZData)
)END
```

PANEL TESTMAIN

```
)ATTR DEFAULT(@?+)
 ? TYPE(INPUT) INTENS(HIGH) COLOR(WHITE)
 ] TYPE(TEXT) INTENS(HIGH) COLOR(BLUE)
 } TYPE(OUTPUT) INTENS(low) COLOR(yellow) caps(off)
 ^ TYPE(OUTPUT) INTENS(low) COLOR(white) caps(off)
 % TYPE(TEXT) INTENS(HIGH) COLOR(YELLOW)
 £ TYPE(TEXT) INTENS(HIGH) COLOR(TURQ)
 @ TYPE(TEXT) INTENS(HIGH) COLOR(WHITE)
 # TYPE(TEXT) INTENS(low) COLOR(GREEN)
 " TYPE(INPUT) INTENS(HIGH) PAD('_')
 + TYPE(TEXT) INTENS(LOW) COLOR(RED)
)BODY SMSG(SMsg) LMSG(LMsg)
#   ______ ______ _____ ______
^ZSYSID
#  /_  __// ____// ___//_  __/
^ZTIME
#   / /  / __/   ç__ ç  / /    Interface
^ZJ4DATE
#  / /  / /___  ___/ / / /     for CLIST or REXX EXEC management
^ZDATE
# /_/  /_____/ /____/ /_/      in libraries not defined in ISPF logon
^ZUSER
]_____________________________________________________________________________
@Command ===>?ZCMD
}Lmsg}SMsg
}TestMsg
+

   %-]Specify the name of the£EXEC](CLIST or REXX EXEC) to be executed
```

```
   %-£TEST]application itself will provide you the needed allocations

       %-£Userid.SOURCE.ISRCLIB]- Clist / REXX Exec
       %-£Userid.SOURCE.ISRPLIB]- ISPF Panels
       %-£Userid.SOURCE.ISRSLIB]- Skeletons for file tailoring actions
       %-£Userid.SOURCE.ISRTLIB]- ISPF Tables
       %-£Userid.SOURCE.ISRMLIB]- Messages

                                                    ]Userid:"Z      +
                                          ]CLIST/REXX EXEC name:"Z
+

]_____________________________________________________________________
]Specify EXEC and press@ENTER]- Press@PF3]to go back to TSO main menu
)INIT
  .ZVars = 'TestUser TestExec'
  .Cursor = TestExec
)PROC
)END
```

TESTBROW

```
/*  Compiled REXX - Use MISP application to apply changes  */
/* ------------------------------------------------------------- */
/*        Description: CLIST to recall external BROWSE process       */
/*  Output parameters: Issue BROWSE cmd against specified dataset    */
/*          Called by: Online TSO - 'TEST' Option - 'B' Command      */
/* ------------------------------------------------------------- */
/*             Author: Luciano Lorini              Date: Ø6.Ø3.2ØØ1  */
/* ------------------------------------------------------------- */
ADDRESS ISPEXEC

BroPanel:
  PARSE ARG TestDsn TestPanl
  "BROWSE Dataset('"TestDsn"') PANEL("TestPanl") "

EXIT Ø
```

TESTMAIN

```
/*  Compiled REXX - Use MISP application to apply changes  */
/* ------------------------------------------------------------- */
/*        Description: MAIN program to join external test EXEC       */
/*                     Dynamically provides alloc for user datasets  */
/*  Output parameters: Verifies and schedules required EXEC          */
/*          Called by: Online TSO - 'TEST' Option                    */
/*                               - 'TEST' stage of MISP product     */
/* ------------------------------------------------------------- */
```

```
/*             Author: Luciano Lorini              Date: 06.03.2001  */
/* ---------------------------------------------------------------- */
ADDRESS ISPEXEC

'VGET (ZUser ZPrefix) '
ZipPref = LEFT(ZPrefix,2)

TestUser = ZPrefix
TestExec = ''
ExecArg = ''
TestMsg = ''

Start:
  TestTry = 0                   /* INIT ExecName_Verify hits_counter */
  ADDRESS ISPEXEC 'DISPLAY PANEL(TESTMAIN) '
  IF RC = 8 THEN EXIT 0

  IF TestUser = '' THEN DO                   /* Check User_insertion */
    TestMsg = 'No USER specified - ' ||,
              'Declare a valid UserID (with existing libraries)'
    SIGNAL Start
  END

CheckLib:
  RCLib = ComputeLibs()                      /* Check User_libraries */
  PARSE VAR RCLib RCLib';'TestMsg
  IF RCLib ¬= 0 THEN SIGNAL Start

       /* Extract member-list in user.ISRCLIB and show it in TBDISPL */
CheckMbr:
  IF TestExec = '' |,                 /* Check insertion of ExecName */
     POS('*',TestExec) ¬= 0 |,
     POS('%',TestExec) ¬= 0 THEN DO
    TestPatt = STRIP(LEFT(TestExec'*',8))
    RCList = ListExec()
    PARSE VAR RCList RCList';'TestMsg
    IF RCList ¬= 0 THEN SIGNAL Start
    ELSE DO
      TestExec = STRIP(TestMsg)
      IF TestExec = '' THEN DO
        TestMsg = 'No input - ' ||,
                  'Declare a valid EXEC from '''TestCLib'''
        SIGNAL Start
      END
    END
  END

                      /* Allocate necessary datasets (if they exist) */
ExecMbr:
  TestDS = "'"TestCLib"("STRIP(TestExec)")'"
```

```
IF SYSDSN(TestDS) = 'OK' THEN DO
  ADDRESS TSO
  "ALTLIB ACT APPL(CLIST) DA('"TestCLib"')"
  ADDRESS ISPEXEC
  IF PLib THEN "LIBDEF ISPPLIB DATASET ID('"TestPLib"')"
  IF SLib THEN "LIBDEF ISPSLIB DATASET ID('"TestSLib"')"
  IF TLib THEN DO
    "LIBDEF ISPTLIB DATASET ID('"TestTLib"')"
    "LIBDEF ISPTABL DATASET ID('"TestTLib"')"
  END
  IF MLib THEN "LIBDEF ISPTLIB DATASET ID('"TestMLib"')"
  ADDRESS TSO
  INTERPRET TestExec
  'ALTLIB RESET'
  ADDRESS ISPEXEC
  IF PLib THEN 'LIBDEF ISPPLIB'
  IF SLib THEN 'LIBDEF ISPSLIB'
  IF TLib THEN DO
    'LIBDEF ISPTLIB'
    'LIBDEF ISPTABL'
  END
  IF MLib THEN 'LIBDEF ISPMLIB'
END
ELSE DO          * ExecName (fullname) with no pattern - not found */
  IF TestTry THEN DO                 /* Creation already attempted */
    TestMsg = ''''TestExec''' not found - ' ||,
              'Declare a valid EXEC from '''TestCLib''''
    SIGNAL Start
  END
  ELSE DO                  /* First passage with ExecName not found */
    TestTry = 1             /* RE-INIT ExecName_Verify hits_counter */
    ExecName = TestExec           /* Assign ExecName for EDIT Panel */

    'CONTROL ERRORS RETURN '
    TestLabl = ' EDIT of TEST-EXEC:'
    TestDsn = TestCLib'('STRIP(TestExec)')'
    ZEdSMsg = 'EXEC 'TestExec' not found'
    ZEdLMsg = 'You may create it now - ' ||,
              'After edit is complete you will be able to run it'
    'SETMSG MSG(ISRZØØ1) '
    "EDIT Dataset('"TestDsn"') PANEL(TESTEDIT) "
    EditRc = Rc
    'CONTROL ERRORS '

    IF EditRc > 4 THEN DO            /* eg Dataset already in use */
     TestMsg = 'Creation of '''TestExec''' KO - ' ||,
               'Problems encountered on '''TestCLib''' (RC='EditRc')'
      SIGNAL Start
    END
    IF EditRc = 4 THEN DO         /* NO changes made on PDS member */
```

```
        TestMsg = ''''TestExec''' not found ' ||,
                  'and member creation skipped (CAN pressed) - ' ||,
                  'Please retry'
        SIGNAL Start
      END
      ELSE DO
        TestDsn = TestCLib'('STRIP(ExecName)')'
        YNMsg1 = TestDsn
        YNMsg2 = 'Would you like to run the EXEC you just created?'
        YNMsg3 = ''
        YNMsg1 = CENTER(YNMsg1,52)
        YNMsg2 = CENTER(YNMsg2,52)
        YNMsg3 = CENTER(YNMsg3,52)
        ConfInp = 'N'
        'ADDPOP Row(5) Column(9) '
        'DISPLAY PANEL(TESTCONF) '
        OptRC = RC
        'REMPOP '
        IF ConfInp = 'Y' &,
           OptRC = Ø THEN DO
          SIGNAL ExecMbr         /* Back to routine for EXEC-running */
        END
        ELSE DO
          TestMsg = ''''TestExec''' created on '''TestCLib''' - ' ||,
                    'EXEC skipped by user request'
          SIGNAL Start
        END
      END
    END
  END

  TestMsg = 'Execution of '''TestExec''' scheduled successfully '
  SIGNAL Start

EXIT(Ø)

/* ---------------------------------------------------------- */
/* ── End of EXEC ─ End of EXEC ─ End of EXEC ─ End of EXEC ── */
/* ---------------------------------------------------------- */

/* ---------------------------------------------------- *
 *  Check existence of libraries and load "Exist-flags"  *
 */
ComputeLibs:
                    /* Check-point to test if UserID is valid or not */
  IF ZipPref ¬= 'X1' & ZipPref ¬= 'Y1' &,         /* Group1 of users */
     ZipPref ¬= 'X2' & ZipPref ¬= 'Y2' &,         /* Group2 of users */
     ZPrefix ¬= 'X111111' &,             /* User1 - FullName - infos */
     ZPrefix ¬= 'X111112' &,             /* User2 - FullName - infos */
     ZPrefix ¬= 'X111113' THEN DO        /* ........................ */
```

```
    TestMsg = '12;User 'ZPrefix' not authorized '
    RETURN TestMsg
  END

  PLib = Ø                          /* "Exist-flag" for Panel library */
  SLib = Ø                        /* "Exist-flag" for Skeleton library */
  TLib = Ø                          /* "Exist-flag" for Table library */
  MLib = Ø                         /* "Exist-flag" for Message library */

  TestCLib = TestUser'.SOURCE.ISRCLIB'
  IF SYSDSN(''''TestCLib'''') = 'OK' THEN CLib = 1
  ELSE DO
    TestMsg = '16;Library '''TestCLib''' not found - Impossible to run
EXEC'
    RETURN TestMsg
  END

  TestPLib = TestUser'.SOURCE.ISRPLIB'
  TestSLib = TestUser'.SOURCE.ISRSLIB'
  TestTLib = TestUser'.SOURCE.ISRTLIB'
  TestMLib = TestUser'.SOURCE.ISRMLIB'

  IF SYSDSN(''''TestPLib'''') = 'OK' THEN PLib = 1
  IF SYSDSN(''''TestSLib'''') = 'OK' THEN SLib = 1
  IF SYSDSN(''''TestTLib'''') = 'OK' THEN TLib = 1
  IF SYSDSN(''''TestMLib'''') = 'OK' THEN MLib = 1

RETURN 'Ø;'

/* ------------------------------------------------ *
 *  Load matching member-list from EXEC user library  *
 */
ListExec:
  PdsDs = TestCLib
  "LMINIT DATAID(Data1) DATASET('"PdsDs"') ENQ(SHR) "
  'LMOPEN DATAID('Data1') OPTION(INPUT) '
  PdsMem = ' '
  'LMMLIST DATAID('Data1') OPTION(SAVE) MEMBER(PdsMem) ,
           PATTERN('TestPatt') STATS(YES) GROUP(TEST) '
  RCLmList = RC
  'LMCLOSE DATAID('Data1') '
  'LMFREE DATAID('Data1') '

  IF RCLmList ¬= Ø THEN DO
    TestMsg = 'Ø8;EXEC suspended - No module 'TestExec' found'
    RETURN TestMsg
  END

  ELSE DO
    ADDRESS TSO
```

```
    x = MSG('OFF')
    IF ZPrefix ¬= ZUser THEN AppoPref = ZUser'.'
                        ELSE AppoPref = ''
    "ALLOCATE DA("AppoPref"TEST.MEMBERS) F(ListaMem) SHR "
    'EXECIO * DISKR ListaMem(Stem MemLib. FINIS) '
    "DELETE ("AppoPref"TEST.MEMBERS)"
    x = MSG('ON')
    NumMem = Memlib.0
    ADDRESS ISPEXEC
    'TBCREATE ExecList KEYS(ExecName) ' ||,
                      'NAMES(ExecDate ExecTime) Nowrite Replace '

    'VGET ZOS390R1 '
    PARSE VAR ZOS390R1 . VV'.'RR'.'MM .  /*Format: 'OS/390 02.xx.00' */
     /* OS/390 earlier than 2.10 use a different PDS dir-info format */
    DO PdsI = 1 TO NumMem
      ExecName = STRIP(SUBSTR(MemLib.PdsI,4,8))
      IF RR >= 10 THEN DO
        ExecDate = STRIP(SUBSTR(MemLib.PdsI,41,8))
        ExecTime = STRIP(SUBSTR(MemLib.PdsI,50,5))
      END
      ELSE DO                                    /* IF OS/390 < 2.10 */
        ExecDate = STRIP(SUBSTR(MemLib.PdsI,40,8))
        ExecTime = STRIP(SUBSTR(MemLib.PdsI,49,5))
      END
      'TBADD ExecList '
    END
  END

/* ------------------------------------------------------------ *
 *  Display member-list in popup panel for further processing  *
 */
DisplayExec:
  'ADDPOP ROW(00) COLUMN(40) '

  DO WHILE RC < 7
    'TBTOP ExecList '
    'TBVCLEAR ExecList '
    IF POS('*',ExecArg) = 0 THEN ExecArg = ExecArg'*'
    ExecName = ExecArg
    'TBSARG ExecList '
    'TBSCAN ExecList '
    'TBDISPL ExecList PANEL(TESTPOP0) '

    DO UNTIL ZTdSels = 0
      SELECT
/*    ---------------------------------------------------------------
      To be uncommented if you want the ability to browse members.
      Since the BROWSE process in a POPUP is not allowed in full screen
      mode, you need to do the following: issue BROWSE as an external
```

```
command. SUSPEND mode MUST be active (for exiting POPUP).
For VIEWing and EDITing I found no problems with ISPF 2.1Ø .
Maybe you need to do the same with earlier versions.
-------------------------------------------------------------
WHEN A = 'B' THEN DO
  TestPanl = 'TESTBROW'
  TestDsn = TestCLib'('STRIP(ExecName)')'
  'SELECT CMD(%TESTBROW 'TestDsn' 'TestPanl') SUSPEND'
END
------------------------------------------------------------ */
WHEN A = 'V' THEN DO
  TestLabl = ' VIEW of TEST-EXEC:'
  TestDsn = TestCLib'('STRIP(ExecName)')'
  "VIEW Dataset('"TestDsn"') PANEL(TESTEDIT) "
END
WHEN A = 'E' THEN DO
  TestLabl = ' EDIT of TEST-EXEC:'
  TestDsn = TestCLib'('STRIP(ExecName)')'
  "EDIT Dataset('"TestDsn"') PANEL(TESTEDIT) "
END
WHEN A = 'D' THEN DO
  TestDsn = TestCLib'('STRIP(ExecName)')'
  YNMsg1 = TestDsn
  YNMsg2 = 'Delete request pending for member'
  YNMsg3 = ''
  YNMsg1 = CENTER(YNMsg1,52)
  YNMsg2 = CENTER(YNMsg2,52)
  YNMsg3 = CENTER(YNMsg3,52)
  ConfInp = 'N'
  'ADDPOP Row(5) Column(-5) '
  'DISPLAY PANEL(TESTCONF) '
  OptRC = RC
  'REMPOP '
  IF ConfInp = 'Y' &,
     OptRC = Ø THEN DO
    "LMINIT DATAID(DDDel) DATASET('"TestCLib"') ENQ(SHRW) "
    "LMOPEN DATAID("DDDel") OPTION(OUTPUT) "
    "LMMDEL DATAID("DDDel") MEMBER("ExecName") "
    "LMCLOSE DATAID("DDDel") "
    "LMFREE DATAID("DDDel") "
    'TBDELETE ExecList '
  END
END
WHEN A = 'S' THEN DO
  TestMsg = 'Ø;'ExecName
  'REMPOP '
  'TBCLOSE ExecList '
  RETURN TestMsg
END
OTHERWISE NOP
```

```
      END
      IF ZTdSels = 1 THEN LEAVE
      ELSE DO
        IF POS('*',ExecArg) = 0 THEN ExecArg = ExecArg'*'
        ExecName = ExecArg
        'TBSARG ExecList '
        'TBSCAN ExecList '
        'TBDISPL ExecList '
      END
    END
  END
  'REMPOP '

RETURN 0
```

Luciano Lorini
System Programmmer
Cariverona Banca SpA – SESI (Italy)

A powerful multistring search EXEC

How many times have you wished that ISPF/EDIT had a complex search string capability? On numerous occasions I wanted to see all the lines that satisfied a certain search criterion while in ISPF/EDIT or ISPF/VIEW mode. Even though ISPF/EDIT allows you to issue single Find and Exclude commands, you do not have a way to issue complex search commands. Sometimes you can issue multiple Find and Exclude commands to achieve what you want, and sometimes it is not possible to get the desired result at all, especially if you have negative search strings.

In such cases you may have to be content with a super set and try to work with it. However, with this EXEC, you can issue complex search strings and get the desired results.

The format of the EXEC is:

```
MS {SearchString} {X|NX}
```

SearchString can be a simple or complex string. You can use | (or), & (and), and ¬(not) with your search strings to narrow your results. After issuing the MS command, for subsequent MS commands, if your last

word of the search string is X, the EXEC will search all the hidden lines for the string and, if it is found, will append the line to the displayed lines. If your last word of the search string is NX, the EXEC will search in the displayed lines for the string and, if it is found, will hide the line from the displayed lines.

Some examples:

```
MS UPDATE & ^VM & ^VSE
```

will display all lines that have ‘UPDATE’ but do not have ‘VM’ or ‘VSE’.

```
MS PERFORM & ^'-'
```

will display all in-line PERFORMs (assuming that you have a hyphen in your paragraph names and that PERFORM and paragraphname are in the same line).

```
MS (Cheese & ^Cheddar) | Yoghurt | ( Milk & ^Fatfree)
```

will display all lines that:

- Have Cheese but do not have Cheddar, or
- Have Yoghurt, or
- Have Milk but do not have Fatfree.

```
MS ( If | When ) & ^'Do'
```

will display all lines that have If or When but do not have Do.

```
MS 'row <' | 'row >'
```

will display all lines that have ‘row >’ or ‘row <’.

For case sensitive search:

```
MS c'if' | c'else'
```

will display all lines that have ‘if’ or ‘else’ (exact match).

Your search strings can be in quotes – single or double. If your string has any of these special characters, space |)(& ^ \, you must use quotes around the string. If you want to have an exact match of words, prefix your search string with ‘C’ (like you do in ISPF/EDIT).

After issuing the first MS command, you can issue further MS commands for your next search to further refine your search by appending X or NX to your MS command. If the last word of the search string is X, the EXEC will search all the hidden lines for the string and, if it is found, will append the line to the displayed lines. If your last word of the search string is NX, the EXEC will search in the displayed lines for the string and if found, will hide it from the displayed lines. If the last word is neither X nor NX, it assumes it's a new MS command and will search the entire file.

Once you see the display, press PF1 to see the REXX command the EXEC generated in producing those results.

To reset the display, simply issue an MS command without any parameters. If you place the cursor on a specific line and issue an MS command without any parameters, the display will be reset and that line will become the current line on the screen.

MS ? will display help in how to use this EXEC.

```
/* ------------------------- Rexx ------------------------*/
/* If this character is not | logical OR, please make a    */
/* a global change to modify this character to logical OR. */
/* ------------------------- Rexx ------------------------*/
/* Rexx Exec To Display Lines With complex Search String.  */
/* | = Or   & = and   \ = Not     ^ = Not                  */
/*   MS {SearchString} {X|NX}                              */
/* Search String Can have | & or \ (Negative)              */
/* Ex                                                      */
/* MS (Cheese & ^Cheddar) | Yoghurt | ( Milk & ^Fatfree)   */
/* will display all lines that                             */
/* - contain Cheese but do not contain Cheddar OR          */
/* - contain Yoghurt OR                                    */
/* - contain Milk but do not contain Fatfree.              */
/* For X, the search will be in hidden lines and results   */
/* are appended to the displayed list                      */
/* For NX,the search will be in displayed list and will be */
/* removed from the list                                   */
/* ------------------------------------------------------- */
/* Author: Moyeen Ahmed Khan                               */
/* ------------------------------------------------------- */
Trace O

Address ISREDIT
'MACRO (xMySrch) PROCESS'
'ISREDIT (xUsrStat) = USER_STATE'
```

```
If xMySrch='?' Then Signal How2Use
Call BuiltEditCmds
Call Initialize
If xMySrch=''  Then Signal ResetDisplay
Call TranslateSrch
Call BuildLogic
Call HideAndSeek
If xType='' Then Call BuildLastShow
Call DispEojMsgs
Exit nRc

/*                        Start Of Subroutines                    */
/*                        --------------------                    */

Initialize:
/*********/
I=Ø
J=Ø
nColumn=1
nCount=Ø
nRc=Ø
nRow=1
sFirst='Y'
sHiddenLines='N'
sPlaceLbl='N'
xBuildCmd=''
xHide='X'
xShow='NX'
xStatus=''
xType=''

cmdGetCrsrPos
cmdResetLbls
cmdWhereAmI
cmdLastLine
cmdPlaceCrsr
cmdSeekHidden
If Rc=Ø Then sHiddenLines='Y'
Return

ResetDisplay:
/***********/
nRc=Ø
If sHiddenLines='Y' Then Do
  cmdReset
  If nColumn>Ø Then cmdPlaceCrsr
   Else nRc=1
  End
 Else Do
  zedsmsg='Required String Missing'
```

```
  zedlmsg='The Command Requires a Specific Search String'
  Address ISPEXEC 'SETMSG MSG(ISRZØØ1)'
  nRc=16
  End
Exit nRc

TranslateSrch:
/************/
xMyNegSearch='^'
xMySrch=Translate(xMySrch,'\',xMyNegSearch)
xMySrch=Strip(xMySrch,'B',' ')
If Words(xMySrch)=1 & Pos("'",xMySrch)=Ø & Pos('"',xMySrch)=Ø Then Do
  If Left(xMySrch,1)='\' Then Do
    xRest=Substr(xMySrch,2)
    If \DataType(xRest,'A') Then xMySrch="\'"xRest"'"
    End
   Else Do
    If \DataType(xMySrch,'A') Then xMySrch="'"xMySrch"'"
    End
 End
xMySrchØ=xMySrch
If Pos("'",xMySrchØ)>Ø Then Do
  nPos1=Pos("'",xMySrchØ)
  Do While nPos1>Ø
   nPos2=Pos("'",xMySrchØ,nPos1+1)
   xMySrchØ=Overlay('`',xMySrchØ,nPos1,nPos2-nPos1+1,'`')
   nPos1=Pos("'",xMySrchØ)
   End
  End
If Pos('"',xMySrchØ)>Ø Then Do
  nPos1=Pos('"',xMySrchØ)
  Do While nPos1>Ø
   nPos2=Pos('"',xMySrchØ,nPos1+1)
   xMySrchØ=Overlay('`',xMySrchØ,nPos1,nPos2-nPos1+1,'`')
   nPos1=Pos('"',xMySrchØ)
   End
  End
xMySrch1=Translate(xMySrchØ,' ','|&()')
If xMySrch1='' Then Do
  Call DispEojMsgs
  Exit nRc
  End
xMySrch2=Translate(xMySrchØ,'~~~~','|&()')
xMySrch3=Translate(xMySrchØ,'~~','|&')
If Words(xMySrch2)>1 Then Do
  nWords=Words(xMySrch2)
  xCrntWord=Word(xMySrch2,nWords)
  xCrntWord=Translate(xCrntWord)
  Select
   When xCrntWord='X' Then xType='Append'
```



```
   When xCrntWord='NX' Then xType='SubSelect'
   Otherwise Do
    xType=''
    cmdReset
    End
   End
  If xType<>'' Then Do
    xMySrch=Subword(xMySrch,1,Words(xMySrch)-1)
    xMySrch1=Substr(xMySrch1,1,Length(xMySrch))
    End
  End
Return

BuildLogic:
/*********/
Do I=1 By 1 Until I=Words(xMySrch1)
 xCrntWord=Word(xMySrch1,I)
 nCnt=0
 Select
  When Length(xCrntWord)=1 Then xSymbol=">"
  When Left(xCrntWord,1)='\' Then Do
   nCnt=1
   xSymbol="="
   End
  Otherwise xSymbol=">"
  End
 Select
  When Translate(Left(xCrntWord,2))="C'" Then Do
   sExact='Y'
   nCnt=1
   End
  When Translate(Left(xCrntWord,3))="\C'" Then Do
   sExact='Y'
   nCnt=2
   End
  Otherwise Do
   sExact='N'
   End
  End
 If nCnt>0 Then xCrntWord=Substr(xCrntWord,nCnt+1)
 If Verify(xCrntWord,'')=0 Then Do
   nWordPos=WordIndex(xMySrch1,I)+nCnt
   nLen=Length(xCrntWord)
   xCrntWord=Substr(xMySrch,nWordPos,nLen)
   End
 Select
  When Left(xCrntWord,1)="'" Then xDat=xCrntWord
  When Left(xCrntWord,1)='"' Then xDat="'"||xCrntWord||"'"
  Otherwise xDat="'"||xCrntWord||"'"
  End
```

```
 If sExact='N' Then Do
   xDat.I="Pos(Translate("xDat"),Translate(xData))"||xSymbol||"Ø"
   End
  Else Do
   xDat.I="Pos("xDat",xData)"||xSymbol||"Ø"
   End
 End
If I=1 Then Do
  xBuildCmd=xDat.I
  Return
  End
Do I=1 By 1 Until I=Length(xMySrch2)
 xFirstCh=Substr(xMySrch2,I,1)
 Select
  When xFirstCh='~' | xFirstCh=' ' Then Do
   sFirst='Y'
   xBuildCmd=xBuildCmd||Substr(xMySrch,I,1)
   End
  When sFirst='Y' Then Do
   sFirst='N'
   J=J+1
   xBuildCmd=xBuildCmd||xDat.J
   End
  Otherwise Nop
  End
 End
Return

HideAndSeek:
/**********/
Do nLine=1 By 1 Until nLine=nLast
 cmdLineRead
 If sHiddenLines='Y' Then cmdLineStatus
 If xType<>'' Then Do
   Select
    When xType='Append' & xStatus='NX' Then Iterate nLine
    When xType='Append' & xStatus='X' Then Nop
    When xType='SubSelect' & xStatus='X' Then Iterate nLine
    When xType='SubSelect' & xStatus='NX' Then Nop
    Otherwise Nop
    End
   End
 xData=Space(xData)
 sRet='N'
 If xData<>'' Then Interpret "Rc="xBuildCmd
 If Rc>Ø Then Do
   Select
    When xStatus='X' & xType='Append' Then Do
     cmdLineShow
     nCount=nCount+1
```

```
     End
    When xStatus='NX' & xType='SubSelect' Then Do
     nStart=nLine
     cmdLineHide
     nCount=nCount+1
     End
    Otherwise Do
     If xStatus='X' Then cmdLineShow
     sRet='Y'
     End
   End
  End
 If xType<>'' Then Iterate nLine
 If sRet='N' Then Do
   If sPlaceLbl='N' Then Do
     nStart=nLine
     sPlaceLbl='Y'
     End
   End
  Else Do
   If sPlaceLbl='Y' Then Do
     If nStart+1=nLine Then cmdLineHide
      Else Do
       cmdPlaceLbl1
       nEnd=nLine-1
       cmdPlaceLbl2
       cmdHideLines
       End
     sPlaceLbl='N'
     End
   nCount=nCount+1
   End
 End
Return

BuildLastShow:
/************/
If sPlaceLbl='Y' Then Do
  If nStart=nLine Then cmdLineHide
   Else Do
    cmdPlaceLbl1
    nEnd=nLine
    cmdPlaceLbl2
    cmdHideLines
    End
  End
Return

DispEojMsgs:
/**********/
```

```
Select
 When xMySrch1='' Then Do
  nRow=nDispRow
  nColumn=1
  cmdReset
  cmdPlaceCrsr
  zedsmsg='*Put String In Quotes*'
  zedlmsg='Put String In Quotes Appropriately'
  nRc=16
  End
 When nCount=Ø & xType='' Then Do
  nRow=nDispRow
  nColumn=1
  cmdReset
  cmdPlaceCrsr
  zedsmsg='*No Hits*'
  zedlmsg='Program Found No Lines That Satisfy:-' xBuildCmd
  nRc=16
  End
 When nCount=Ø Then Do
  nRow=nDispRow
  nColumn=1
  cmdPlaceCrsr
  zedsmsg='*No Hits*'
  zedlmsg='Program Found No Lines That Satisfy:-' xBuildCmd
  nRc=16
  End
 When xType='SubSelect' Then Do
  cmdUpMax
  zedsmsg=nCount 'Lines Excluded'
  zedlmsg=nCount 'Lines Excluded Due To:-' xBuildCmd
  nRc=Ø
  End
 When xType='Append' Then Do
  cmdUpMax
  zedsmsg=nCount 'Lines Added'
  zedlmsg=nCount 'Lines Have Been Added Due To:-' xBuildCmd
  nRc=Ø
  End
 When xType='' Then Do
  cmdUpMax
  zedsmsg=nCount 'Lines Found'
  zedlmsg='Program Found' nCount 'Lines Due To:-' xBuildCmd
  nRc=Ø
  End
 Otherwise Nop
 End
'USER_STATE = (xUsrStat)'
Address ISPEXEC 'SETMSG MSG(ISRZØØ1)'
Return
```

```
BuiltEditCmds:
/************/
cmdWhereAmI='(nDispRow,nDispCol) = DISPLAY_LINES'
cmdGetCrsrPos='(nRow,nColumn) = CURSOR'
cmdHideLines="EXCLUDE p'=' .AAAFMAK .SSAHSF ALL"
cmdLastLine='(nLast) = LINENUM .ZLAST'
cmdLineHide="XSTATUS (nStart) = (xHide)"
cmdLineRead='(xData) = LINE (nLine)'
cmdLineShow='XSTATUS (nLine) = (xShow)'
cmdLineStatus='(xStatus) = XSTATUS (nLine)'
cmdPlaceCrsr='CURSOR=(nRow,nColumn)'
cmdPlaceLbl1='LABEL (nStart) = .AAAFMAK'
cmdPlaceLbl2='LABEL (nEnd) = .SSAHSF'
cmdReset='RESET'
cmdResetLbls='RESET LABEL'
cmdSeekHidden="SEEK P'=' X FIRST .ZFIRST .ZLAST"
cmdUpMax='UP MAX'
Return

How2Use:
Parse Source . . xExecName .
Say 'Use' xExecName 'for complex searches while in ISPF/EDIT'
Say 'Format is'
Say '     ' xExecName 'SearchString {<X>|<NX>}'
Say 'Your Search String Can have Or(|) and (&) Or Not(^) operand'
Say 'Search strings can be in quotes to search for multi words.'
Say "To search for exact matches, presfix the string with C'"
Say 'Example:-'
Say xExecName '(Cheese & \Cheddar) | Yoghurt | ( Milk & ^Fatfree)'
Say 'will display all lines that'
Say '- contain Cheese but do not contain Cheddar OR'
Say '- contain Yoghurt OR'
Say '- contain Milk but do not contain Fatfree.'
Say 'Once in this subset, you can further issue' xExecName 'commands'
Say 'to either filter out from what is shown Or'
Say 'to add more lines that are hidden'
Say xExecName 'Asiago NX'
Say 'will hide all the lines that contain Asiago from the above'
Say 'display'
Say xExecName 'Cream X'
Say 'will append all the lines that have Cream from the excluded lines'
Say 'to the displayed list'
Say 'Of course, you can issue any EDIT command any time'
Say xExecName 'without any parameters will reset to full display'
Exit
```

Moyeen A Khan
Decision Consultants Inc (USA)

A PROC finder utility

PROCFIND is an ISPF dialog used to locate which PDS(s) in the JES2 PROC00 concatenation contains a specified PROC. It is made up of three parts: PROCFIND is a REXX EXEC that initiates a COBOL program named OSJS0001, and the program displays an ISPF panel named PROCFND1 with the results.

Often multiple copies of a PROC may exist, and this utility is helpful when you need to quickly see everywhere the PROC exists. After the panel is displayed, the user may EDIT or BROWSE the PROC from the PROCFIND panel.

To install the system, copy the EXEC to a SYSPROC dataset, compile and link the COBOL program, and enter the panels into an ISPPLIB dataset. There is little customization required. In the EXEC, edit the PDS name where your JES2 PROC is located, and edit the PDS LOADLIB name where the program is linked. Then, on a command line, enter **TSO PROCFIND *procname***. If you leave off the PROC name, the EXEC will prompt you for it.

PROCFIND REXX EXEC

```
/* REXX */
   CALL MSG 'OFF'
   PARSE ARG LN
   IF LENGTH(LN) = Ø THEN DO
      SAY 'NO PROC WAS ENTERED, PLEASE ENTER A PROC OR F3 TO QUIT'
      PULL LN
      IF LENGTH(LN) = Ø THEN DO
      SAY 'NO PROC WAS ENTERED, PROCFIND CANCELLED'
      SIGNAL QUIT
      END
   END

   FILENAME  = "SYS1.PROCLIB(JES2)"

   "FREE DDNAME(INPUT1)"
   "ALLOC DDNAME(INPUT1) DSNAME("FILENAME") SHR"
   "CALL 'TECH.SUPPORT.LOADLIB(OSJSØØØ1)'" "'"LN FILENAME"'"
   "FREE DDNAME(INPUT1)"
```

```
QUIT:
EXIT Ø
```

COBOL PROGRAM OSJS0001

```
       IDENTIFICATION DIVISION.
       PROGRAM-ID.  OSJSØØØ1.

       ENVIRONMENT DIVISION.
       INPUT-OUTPUT SECTION.
       FILE-CONTROL.
           SELECT JESPROC-FILE        ASSIGN TO INPUT1.

      ****************************************************************
      *    D A T A   D I V I S I O N                                 *
      ****************************************************************

       DATA DIVISION.
       FILE SECTION.

       FD  JESPROC-FILE
           RECORDING MODE IS F
           RECORD CONTAINS 8Ø CHARACTERS
           DATA RECORD IS JESPROC-RECORD.
       Ø1  JESPROC-RECORD       PIC X(8Ø).

       WORKING-STORAGE SECTION.
      *============================================================*
      *    DISPLAY PANEL TO SHOW MEMBERS IN EACH JES DATASET       *
      *============================================================*

       Ø1  TABLE1-NAME       PIC X(8) VALUE 'TPROCFND'.
       Ø1  TABLE1-ROWID      PIC 9(6) COMP VALUE Ø.
       Ø1  TABLE1-VARS.
          Ø3  FILLER  PIC X(1) VALUE '('.
          Ø3  FILLER  PIC X(8) VALUE 'EDITOPT '.
          Ø3  FILLER  PIC X(9) VALUE 'LIBR'.
          Ø3  FILLER  PIC X(9) VALUE 'MBER'.
          Ø3  FILLER  PIC X(8) VALUE 'CHGD'.
          Ø3  FILLER  PIC X(1) VALUE ')'.

      *============================================================*
      *    LMM VARIABLES NEEDED TO RUN                             *
      *============================================================*
       Ø1  ISPEXEC-BUFLEN      PIC 9(6) COMP VALUE 132.
       Ø1  ISPEXEC-BUF         PIC X(132) VALUE SPACES.
       Ø1  LM-ACTIVE           PIC XX.
       Ø1  SAVE-ROWID          PIC S9(Ø6) COMP VALUE +Ø.
```

```
*=============================================================*
*    ISPF ERROR MESSAGE AREA                                  *
*=============================================================*
 Ø1  FILE-STATUS          PIC XX.

 Ø1  WS-JESPROC-RECORD.
     Ø5 FILLER            PIC XX.
     Ø5 DDNAME            PIC X(8).
     Ø5 DSN-INFO          PIC X(61).

 Ø1  SUB-NUMBERS.
     Ø5 SUB-1             PIC 9(2).
     Ø5 SUB-2             PIC 9(2).

 Ø1  SEARCH-POINTERS.
     Ø5 START-PNT         PIC 9(2).
     Ø5 STOP-PNT          PIC 9(2).

 Ø1  DSN-NAMES.
     Ø5 DSN               PIC X(44) VALUE SPACES.
     Ø5 DSNNAME           PIC X(44) VALUE SPACES.
     Ø5 WS-DSN            PIC X(44) VALUE SPACES.
     Ø5 WS-DSN-FILL       PIC X(1) VALUE SPACES.
     Ø5 WS-DSN-FILL2      PIC X(1) VALUE SPACES.
     Ø5 WS-DSN2           PIC X(44) VALUE SPACES.

 Ø1  SEARCH-INFO2.
     Ø5 SRMBR             PIC X(8).
     Ø5 NOT-FOUND         PIC X(3) VALUE 'N/A'.
     Ø5 WS-FILENAME       PIC X(6Ø).

 Ø1  WS-PROC-SRCH-INFO    PIC X(68).

 Ø1  ISPF-MSG1.
     Ø3  FILLER           PIC X(29) VALUE
         'ISPLINK FUNCTION ERROR. RC = '.
     Ø3  ISPF-RC          PIC 99    VALUE Ø.

 Ø1  ISPF-MSG2.
     Ø3  FILLER           PIC X(1Ø) VALUE 'FUNCTION: '.
     Ø3  ISPF-FUNC        PIC X(5Ø) VALUE SPACES.
*************************************************************
*                                                           *
*    ISPF "ISPLINK" CONTROL VARIABLES.                      *
*                                                           *
*************************************************************
 Ø1  ISPF-CONTROL.
     Ø3  VCOPY          PIC X(8) VALUE 'VCOPY   '.
     Ø3  VPUT           PIC X(8) VALUE 'VPUT    '.
     Ø3  VGET           PIC X(8) VALUE 'VGET    '.
```

```
    Ø3  VDEFINE        PIC X(8) VALUE 'VDEFINE '.
    Ø3  VDELETE        PIC X(8) VALUE 'VDELETE '.
    Ø3  VREPLACE       PIC X(8) VALUE 'VREPLACE'.
    Ø3  SHAREDX        PIC X(8) VALUE 'SHARED  '.
    Ø3  MOVE-MODE      PIC X(8) VALUE 'MOVE    '.
    Ø3  DISPLAYE       PIC X(8) VALUE 'DISPLAY '.
    Ø3  CHAR           PIC X(8) VALUE 'CHAR    '.
    Ø3  FIXED          PIC X(8) VALUE 'FIXED   '.
    Ø3  SETMSG         PIC X(8) VALUE 'SETMSG  '.
    Ø3  TBCREATE       PIC X(8) VALUE 'TBCREATE'.
    Ø3  TBERASE        PIC X(8) VALUE 'TBERASE '.
    Ø3  TBEND          PIC X(8) VALUE 'TBEND   '.
    Ø3  TBADD          PIC X(8) VALUE 'TBADD   '.
    Ø3  TBSKIP         PIC X(8) VALUE 'TBSKIP  '.
    Ø3  TBQUERY        PIC X(8) VALUE 'TBQUERY '.
    Ø3  TBTOP          PIC X(8) VALUE 'TBTOP   '.
    Ø3  TBPUT          PIC X(8) VALUE 'TBPUT   '.
    Ø3  TBDISPL        PIC X(8) VALUE 'TBDISPL '.
    Ø3  TBSORT         PIC X(8) VALUE 'TBSORT  '.
    Ø3  TBDELETE       PIC X(8) VALUE 'TBDELETE'.
    Ø3  TBGET          PIC X(8) VALUE 'TBGET   '.
    Ø3  REPLACEX       PIC X(8) VALUE 'REPLACE '.
    Ø3  NOWRITE        PIC X(8) VALUE 'NOWRITE '.
    Ø3  CONTROLX       PIC X(8) VALUE 'CONTROL '.
    Ø3  ERRORS         PIC X(8) VALUE 'ERRORS  '.
    Ø3  CANCELX        PIC X(8) VALUE 'CANCEL  '.
    Ø3  RETURNX        PIC X(8) VALUE 'RETURN  '.
    Ø3  LINEX          PIC X(8) VALUE 'LINE    '.
    Ø3  REFRESHX       PIC X(8) VALUE 'REFRESH '.
    Ø3  SUBTASKX       PIC X(8) VALUE 'SUBTASK '.
    Ø3  ORDERX         PIC X(8) VALUE 'ORDER   '.
    Ø3  SAVEX          PIC X(8) VALUE 'SAVE    '.
    Ø3  RESTOREX       PIC X(8) VALUE 'RESTORE '.
    Ø3  PROFILE        PIC X(8) VALUE 'PROFILE '.
    Ø3  ASTERIK        PIC X(8) VALUE '*       '.
    Ø3  FTOPEN         PIC X(8) VALUE 'FTOPEN  '.
    Ø3  FTCLOSE        PIC X(8) VALUE 'FTCLOSE '.
    Ø3  FTINCL         PIC X(8) VALUE 'FTINCL  '.
    Ø3  FTTEMP         PIC X(8) VALUE 'TEMP    '.
    Ø3  ADDPOP         PIC X(8) VALUE 'ADDPOP  '.
    Ø3  POPLOC         PIC X(8) VALUE 'POPLOC  '.
    Ø3  REMPOP         PIC X(8) VALUE 'REMPOP  '.
    Ø3  ALLX           PIC X(8) VALUE 'ALL     '.
    Ø3  SELECTX        PIC X(8) VALUE 'SELECT  '.
    Ø3  ASIS           PIC X(8) VALUE 'ASIS    '.
    Ø3  LOCKX          PIC X(8) VALUE 'LOCK    '.
Ø1  FILLER.
    Ø2  MSGNO        PIC X(8) VALUE SPACES.
    Ø2  CURSOR-FIELD PIC X(8) VALUE SPACES.
```

```
*-------------------------------------------------------------*
*--FOLLOWING VARIABLES ARE FOR THE ISPF DEFINE AND ARE BY SIZE-*
*-------------------------------------------------------------*
 Ø1 FILLER.
     Ø2  L6Ø-ERR-VAR.
         Ø5  FILLER      PIC X(2Ø) VALUE '(ERRMSG ERMØ1 ERMØ2 '.
         Ø5  FILLER      PIC X(2Ø) VALUE 'ERMØ2 ERMØ3 ERMØ4 ER'.
         Ø5  FILLER      PIC X(2Ø) VALUE 'MØ5 ERMØ6)          '.
     Ø2  L6Ø-ERR-VAL.
         Ø5  ERRMSG      PIC X(6Ø) VALUE SPACES.
         Ø5  ERMØ1       PIC X(6Ø) VALUE SPACES.
         Ø5  ERMØ2       PIC X(6Ø) VALUE SPACES.
         Ø5  ERMØ3       PIC X(6Ø) VALUE SPACES.
         Ø5  ERMØ4       PIC X(6Ø) VALUE SPACES.
         Ø5  ERMØ5       PIC X(6Ø) VALUE SPACES.
         Ø5  ERMØ6       PIC X(6Ø) VALUE SPACES.
     Ø2  L44-PAN-VAR.
         Ø5  FILLER      PIC X(2Ø) VALUE '(JESC LIBR)         '.
     Ø2  L44-PAN-VAL.
         Ø5  JESC        PIC X(44) VALUE SPACES.
         Ø5  LIBR        PIC X(44) VALUE SPACES.
     Ø2  L37-PAN-VAR.
         Ø5  FILLER      PIC X(2Ø) VALUE '(CMND)              '.
     Ø2  L37-PAN-VAL.
         Ø5  CMND        PIC X(37) VALUE SPACES.
     Ø2  L8-PAN-VAR.
         Ø5  FILLER      PIC X(2Ø) VALUE '(MBER CHGD ZLMDATE) '.
     Ø2  L8-PAN-VAL.
         Ø5  MBER        PIC X(8) VALUE SPACES.
         Ø5  CHGD        PIC X(8) VALUE SPACES.
         Ø5  ZLMDATE     PIC X(8) VALUE SPACES.
     Ø2  L6-CHAR-NAMES.
         Ø3  FILLER      PIC X(1Ø) VALUE '(ZTDTOP )'.
     Ø2  L6-CHAR-VALUES.
         Ø3  ZTDTOP      PIC 9(Ø6) VALUE ZEROES.
     Ø2  L4-FIXED-NAMES.
         Ø3  FILLER      PIC X(1Ø) VALUE '(ROWID)'.
     Ø2  L1-PAN-VAR.
         Ø3  FILLER      PIC X(1Ø) VALUE '(EDITOPT)'.
     Ø2  L1-PAN-VAL.
         Ø3  EDITOPT     PIC X(Ø1) VALUE SPACES.
     Ø2  L4-FIXED-VALUES.
         Ø3  ROWID       PIC 9(Ø6) COMP VALUE Ø.
*-------------------------------------------------------------*
 Ø1  PANEL-NAME        PIC X(8)  VALUE SPACES.
 Ø1  NULLARG           PIC X(8) VALUE SPACES.
 Ø1  L6Ø               PIC 9(6) COMP VALUE 6Ø.
 Ø1  L44               PIC 9(8) COMP VALUE 44.
 Ø1  L37               PIC 9(8) COMP VALUE 37.
 Ø1  L8                PIC 9(8)  COMP VALUE 8.
```

```
 Ø1  L1                PIC 9(8)  COMP VALUE 1.
 Ø1  MAX-RC            PIC S9(8)  COMP VALUE +Ø.

 LINKAGE SECTION.
 Ø1  SEARCH-INFO.
     Ø5 LINK-LENGTH      PIC S9(4) COMP.
     Ø5 PROC-SRCH-INFO   PIC X(68).

*****************************************************************
*    P R O C E D U R E    D I V I S I O N                       *
*****************************************************************
 PROCEDURE DIVISION USING SEARCH-INFO.
*===============================================================*
*    VDEFINE THE PANEL/TABLE VARIABLES USED.                    *
*    THEY ARE DEFINED IN GROUPS OF VARIABLES WITH COMMON        *
*    LENGTHS TO REDUCE THE NUMBER OF 'ISPLINK' CALLS.           *
*===============================================================*

     CALL 'ISPLINK' USING VDEFINE L6Ø-ERR-VAR L6Ø-ERR-VAL
                                  CHAR L6Ø.
     IF RETURN-CODE NOT = Ø
        MOVE 'VDEFINE ON L6Ø-CHARS' TO ISPF-FUNC
        GO TO ISPF-ERROR.

     CALL 'ISPLINK' USING VDEFINE L44-PAN-VAR L44-PAN-VAL
                                  CHAR L44.
     IF RETURN-CODE NOT = Ø
        MOVE 'VDEFINE ON L44-CHARS' TO ISPF-FUNC
        GO TO ISPF-ERROR.


     CALL 'ISPLINK' USING VDEFINE L37-PAN-VAR L37-PAN-VAL
                                  CHAR   L37.
     IF RETURN-CODE NOT = Ø
        MOVE 'VDEFINE ON L37-CHARS' TO ISPF-FUNC
        GO TO ISPF-ERROR.


     CALL 'ISPLINK' USING VDEFINE L8-PAN-VAR L8-PAN-VAL
                                  CHAR   L8.
     IF RETURN-CODE NOT = Ø
        MOVE 'VDEFINE ON L8-CHARS' TO ISPF-FUNC
        GO TO ISPF-ERROR.

     CALL 'ISPLINK' USING VDEFINE L1-PAN-VAR L1-PAN-VAL
                                  CHAR   L1.
     IF RETURN-CODE NOT = Ø
        MOVE 'VDEFINE ON L1-CHARS' TO ISPF-FUNC
        GO TO ISPF-ERROR.
```

```
Ø1Ø-TBCREATE.
    CALL 'ISPLINK' USING TBCREATE
                         TABLE1-NAME NULLARG TABLE1-VARS
                         NOWRITE REPLACEX.
    IF RETURN-CODE > 4
       MOVE SPACES TO ISPF-FUNC
       STRING 'TBCREATE ON ' TABLE1-NAME DELIMITED BY '+'
          INTO ISPF-FUNC
       GO TO ISPF-ERROR
    END-IF.
Ø1Ø-EXIT. EXIT.

Ø5Ø-MOVE-SEARCH-INFO.
    MOVE PROC-SRCH-INFO TO WS-PROC-SRCH-INFO.
    UNSTRING PROC-SRCH-INFO
    DELIMITED BY ' ' INTO SRMBR WS-FILENAME.
    MOVE WS-FILENAME TO JESC.
Ø5Ø-EXIT. EXIT.

    OPEN INPUT JESPROC-FILE.
1ØØ-READ-FILE.
    READ JESPROC-FILE INTO WS-JESPROC-RECORD
    PERFORM 2ØØ-PROCØØ-SEARCH THRU 2ØØ-EXIT.
1ØØ-EXIT. EXIT.

2ØØ-PROCØØ-SEARCH.
    IF DDNAME = 'PROCØØ'
     PERFORM 4ØØ-PULL-DSN-START THRU 675-EXIT
    ELSE GO TO 1ØØ-READ-FILE
    END-IF.
2ØØ-EXIT. EXIT.

3ØØ-BLANK-PROC-SEARCH.
    READ JESPROC-FILE INTO WS-JESPROC-RECORD
    IF DDNAME NOT = '        '
     GO TO 75Ø-REPOS-TABLE
    ELSE
     PERFORM 4ØØ-PULL-DSN-START THRU 675-EXIT
     PERFORM 3ØØ-BLANK-PROC-SEARCH
    END-IF.
3ØØ-EXIT. EXIT.

4ØØ-PULL-DSN-START.
    MOVE SPACES TO WS-DSN.
    UNSTRING DSN-INFO
       DELIMITED BY 'DSN=' INTO WS-DSN-FILL WS-DSN.

    MOVE SPACES TO WS-DSN2.
    UNSTRING WS-DSN
       DELIMITED BY ',' OR ' ' INTO WS-DSN2 WS-DSN-FILL2.
```

```
400-EXIT. EXIT.

500-MOVE-DSNNAME.
    MOVE WS-DSN2 TO DSNNAME
    IF DSNNAME = '                                                '
     GO TO 300-BLANK-PROC-SEARCH
    ELSE
     PERFORM 600-LMM-INIT-OPEN THRU 650-EXIT.
500-EXIT. EXIT.

600-LMM-INIT-OPEN.
    MOVE SPACES TO ISPEXEC-BUF.
    STRING
       'LMINIT+DATAID(DSNCHK)+DATASET(' DSNNAME ')'
         DELIMITED BY SPACE
       INTO ISPEXEC-BUF.
    INSPECT ISPEXEC-BUF REPLACING ALL '+' BY SPACE.

    CALL 'ISPEXEC' USING ISPEXEC-BUFLEN ISPEXEC-BUF.
    IF RETURN-CODE NOT = 0
       MOVE 'ISPEXEC LMINIT ON DSNCHK' TO ISPF-FUNC
        GO TO ISPF-ERROR
    END-IF.

    MOVE 'LMOPEN DATAID(&DSNCHK) OPTION(INPUT)'
       TO ISPEXEC-BUF.
    CALL 'ISPEXEC' USING ISPEXEC-BUFLEN ISPEXEC-BUF.

600-EXIT. EXIT.

650-LMM-FIND.
     MOVE SPACES TO ISPEXEC-BUF.
     STRING
     'LMMFIND+DATAID(&DSNCHK)+MEMBER(' SRMBR ')+STATS(YES)'
        DELIMITED BY SPACE INTO ISPEXEC-BUF.
     INSPECT ISPEXEC-BUF REPLACING ALL '+' BY SPACE
     CALL 'ISPEXEC' USING ISPEXEC-BUFLEN ISPEXEC-BUF.
     IF RETURN-CODE = 8
       MOVE SPACES TO EDITOPT
       MOVE DSNNAME TO LIBR
       MOVE NOT-FOUND TO MBER
       MOVE SPACES TO CHGD
       PERFORM 675-LMFREE THRU 675-EXIT
       PERFORM 700-TABLE-ADD THRU 700-EXIT
       GO TO 300-BLANK-PROC-SEARCH
     END-IF.

     IF RETURN-CODE NOT = 0 OR NOT = 4
       MOVE SPACES TO EDITOPT
       MOVE DSNNAME TO LIBR
```

```
        MOVE SRMBR TO MBER
        MOVE ZLMDATE TO CHGD
        PERFORM 675-LMFREE THRU 675-EXIT
        PERFORM 700-TABLE-ADD THRU 700-EXIT
        GO TO 300-BLANK-PROC-SEARCH
     ELSE
      PERFORM ISPF-ERROR
     END-IF.
650-EXIT. EXIT.

675-LMFREE.
      IF LM-ACTIVE = 'Y'
         MOVE 'LMCLOSE DATAID(&DSNCHK)' TO ISPEXEC-BUF
         CALL 'ISPEXEC' USING ISPEXEC-BUFLEN ISPEXEC-BUF
         IF RETURN-CODE NOT = 0
            MOVE 'ISPEXEC LCLOSE ON DSNCHK' TO ISPF-FUNC
            GO TO ISPF-ERROR
         END-IF
         MOVE 'LMFREE DATAID(&DSNCHK)' TO ISPEXEC-BUF
         CALL 'ISPEXEC' USING ISPEXEC-BUFLEN ISPEXEC-BUF
         IF RETURN-CODE NOT = 0
            MOVE 'ISPEXEC LMFREE ON DSNCHK' TO ISPF-FUNC
            GO TO ISPF-ERROR
         END-IF
      END-IF
      MOVE 'N' TO LM-ACTIVE.
675-EXIT. EXIT.

700-TABLE-ADD.
      CALL 'ISPLINK' USING TBADD TABLE1-NAME.
      IF RETURN-CODE NOT = 0
         MOVE SPACES TO ISPF-FUNC
         STRING 'TBADD ON ' TABLE1-NAME DELIMITED BY '+'
            INTO ISPF-FUNC
         GO TO ISPF-ERROR
      END-IF.
700-EXIT. EXIT.


750-REPOS-TABLE.
      CALL 'ISPLINK' USING TBTOP TABLE1-NAME.
      CALL 'ISPLINK' USING TBSKIP TABLE1-NAME TABLE1-ROWID.
      IF RETURN-CODE NOT = 0 AND NOT = 8
         STRING 'TBSKIP ON ' TABLE1-NAME
            DELIMITED BY '+' INTO ISPF-FUNC
         GO TO ISPF-ERROR
      END-IF.
     MOVE 'PROCFND1' TO PANEL-NAME.
     CALL 'ISPLINK' USING TBDISPL TABLE1-NAME PANEL-NAME.
     IF RETURN-CODE = 8
```

```
         GOBACK
     END-IF.
 750-EXIT. EXIT.

*****************************************************************
* IF EDITOPT = Ø, NO LINE COMMAND WAS ENTER, JUST REPOS TABLE. *
*****************************************************************

     IF EDITOPT = 'E'
                 MOVE SAVE-ROWID      TO ROWID
                 PERFORM 8ØØ-PROCESS-ROW THRU 8ØØ-EXIT
     END-IF.
     GO TO 75Ø-REPOS-TABLE.


*****************************************************************
* A LINE COMMAND WAS ENTERED, PERFORM NECCESSARY ACTION.       *
*****************************************************************
 8ØØ-PROCESS-ROW.
     IF EDITOPT = 'E'
           MOVE SAVE-ROWID            TO ROWID
           MOVE '*' TO EDITOPT
     CALL 'ISPLINK'                    USING TBPUT
                                      TABLE1-NAME

           PERFORM 85Ø-EDIT-MBER THRU 85Ø-EXIT
     END-IF.
 8ØØ-EXIT. EXIT.

*****************************************************************
* PROCESS LINE COMMAND ENTERED FROM BROWSE DISPLAY.            *
*****************************************************************
 85Ø-EDIT-MBER.
     MOVE SPACES     TO ISPEXEC-BUF.
     STRING 'EDIT+DATASET(' LIBR '(' MBER ')' ')'
         DELIMITED BY SPACES INTO ISPEXEC-BUF
     END-STRING.
     INSPECT ISPEXEC-BUF REPLACING ALL '+' BY SPACES
     CALL 'ISPEXEC' USING ISPEXEC-BUFLEN ISPEXEC-BUF.
 85Ø-EXIT. EXIT.

*****************************************************************
*    AN ISPF ERROR HAS OCCURRED.                                *
*****************************************************************
 ISPF-ERROR.
     MOVE RETURN-CODE TO ISPF-RC.
     MOVE ISPF-MSG1 TO ERMØ1.
     MOVE ISPF-MSG2 TO ERMØ2.
 FATAL-ERROR.
     CALL 'ISPLINK' USING VDEFINE
```

```
                        L6Ø-ERR-VAR L6Ø-ERR-VAL CHAR L6Ø.
           CALL 'ISPLINK' USING SETMSG MSGNO.
           MOVE 'IERRORP' TO PANEL-NAME.
           CALL 'ISPLINK' USING DISPLAYE PANEL-NAME.
           MOVE 99 TO MAX-RC.
           GO TO EOJ-ROUTINE.
      ****************************************************************
      *    NORMAL END OF FUNCTION
      ****************************************************************
       END-FUNCTION.
           MOVE Ø TO MAX-RC.

       EOJ-ROUTINE.
           MOVE MAX-RC TO RETURN-CODE.
           GOBACK.
```

ISPF PANEL PROCFND1

```
)ATTR
  % TYPE(TEXT) COLOR(RED)
  + TYPE(TEXT) COLOR(WHITE)
  @ TYPE(TEXT) COLOR(RED)
  _ TYPE(INPUT) COLOR(BLUE) CAPS(ON)
  ? TYPE(INPUT) COLOR(TURQ) HILITE(USCORE) PAD(' ') DEPTH(4)
  # TYPE(OUTPUT) COLOR(YELLOW)
  * TYPE(OUTPUT) COLOR(BLUE)
  < TYPE(OUTPUT) INTENS(LOW) JUST(LEFT) PAD(' ') COLOR(YELLOW)
)BODY
%--------------------JES CONCATENATION PROC SEARCH---------------------
%COMMAND ====>_CMND                                    %SCROLL ===>?SCRL+
%F3-EXIT     F1-ISPF HELP
%
%
%   <ERRMSG                                                          %
%JES SOURCE LIBRARY:#JESC
%
@E=EDIT
+
+ PROCØØ LIBRARIES
+ S SEARCHED IN SAME ORDER AS SYSTEM SEARCH      MEMBER   CHANGED
+ - ------------------------------------------- -------- --------
)MODEL
 _Z*LIBR                                      #MBER    #CHGD    +
)INIT
.ZVARS=EDITOPT
   .CURSOR = CMND
)PROC
   .ALARM=YES
)END
```

ISPF PANEL IERRORP

This panel is just an error display in case of ISPF errors.

```
)ATTR
  < TYPE(OUTPUT) INTENS(LOW) JUST(LEFT) PAD(' ') COLOR(YELLOW)

)BODY

_ZCMD
%


%   <ERM01                                                    %
%                                                             %
%   <ERM02                                                    %
%                                                             %
%   <ERM03                                                    %
%                                                             %
%   <ERM04                                                    %
%                                                             %
%   <ERM05                                                    %
%                                                             %
%   <ERM06                                                    %
)END
```


A VSAM browse routine

This routine was developed because of my Assembler trace program (*MVS Update*, January 1999 to May 1999, Issues 148-152). At the time it was originally developed, it was not possible to do QSAM I/O in 31-bit addressing mode. To get around the problem, I used an ACB, which forced the use of either JES spool or a VSAM ESDS. I would normally use the spool, but a very big trace could easily fill the spool, forcing the use of the ESDS. Hence, a VSAM browse routine.

This routine can be used in place of the standard ISRBRO, since it will call the standard ISPF browse services for PS or PO files. For VSAM, I use the BRIF interface. This interface allows the programmer to browse any type of dataset he wishes. At the very least, the call to BRIF

must contain the address of the READ routine. In this program, it also contains the address of a command interpreter. If a command is entered, ISPF will call this routine before processing the command. If the command interpreter returns with a return code of 4, ISPF will then try to interpret is as a standard command.

As an aid to random access (eg DOWN *n* lines), I read the entire file sequentially, storing the RBA of each record in an ISPF table. Thereafter, I close the ACB and re-open for random access. The read routine specified to BRIF then reads each required record by RBA.

BRIF allows one to pass the address of a communications area between the BRIF caller and the read routine. However, one must be careful here, as the address specified in the call to BRIF is the address of a pointer to the actual parameter. In this case, I pass the address of my working storage, making all fields in the area addressable.

Apart from standard MVS (OS/390) macros, I have some of my own:

1 PIFISPC0 (copybook) – the macro ISPEXEC and some sub-macros were developed because I can never remember a specific set of positional parameters for each ISPF service call. As far as I am concerned, it is easier to specify named parameters, without having to remember their positions in the list. I tried to use the CLIST syntax as far as possible. Some of the parameter specifications I copied from the VSAM macro interfaces. Parameters can normally be specified as a literal, an address, or the contents of a variable. Examples:

```
ISPEXEC SETMSG,MSG=ISRB1Ø7
ISPEXEC SETMSG,MSG=(*,ERROR_MESSAGE)
```

In the first example, message ISRB107 will be issued. In the second, field ERROR_MESSAGE is defined as an eight-character field, and will contain an ISPF message number.

2 PPFC14M0 and PPFGBLC0 – these are a set of structured programming macros (IF-ELSE-ENDIF, loop structures, and SELECT-WHEN-ENDSEL structures). These macros were distributed on SHARE tapes more than 10 years ago. I consolidated them into one copybook for easier distribution, and added the SELECT-WHEN structures, which they lacked at the time.

PPFGBLC0 contains global variables used by PPFC14M0. Most of these macro descriptions can be found in the Assembler toolkit manual, except for the SELECT-WHEN structure, which approximates to the REXX version, rather than the one in the toolkit.

3 PIETPERF – these are a couple of very simple macros to ease the calling of sub-procedures. These are:

- PERF *procedure* – this generates a BAS 14,*procedure*.
- MODENTRY – generates code to push R14 onto the return stack.
- MODEXIT – pops R14 from the return stack, and BR 14.
- RETSTACK SIZE=*number* – defines a return stack of *number* fullwords.
- INIT_RETURN_STACK – sets the stack level pointer to the first word in the stack.

PIETPERF

```
         MACRO
&LBL     PERF  &MOD
.*---------------------------------------------------------------------*.
.* THIS MACRO IS USED TO CALL A SUB-PROCEDURE.                         *.
.* SINCE THIS PROGRAM IS SO BIG,  A SIMPLE BAS R14,&MOD WILL          *.
.* NOT ALWAYS WORK,  HENCE THE BASR CONSTRUCT                         *.
.*---------------------------------------------------------------------*.
         L     R15,=A(&MOD)
&LBL     BASR  R14,R15
         MEND
         SPACE 3
         MACRO
&LBL     MODENTRY  &NEWBASE=,&LIST=NO,&BAKR=NO,&INITBASE=
.*---------------------------------------------------------------------*.
.* THIS MACRO IS THE ENTRY TO A SUB-PROCEDURE.                         *.
.* PARAMETERS:                                                        *.
.*       BAKR:     IF NO, USE RETURN STACK TO SAVE GPR 14, ELSE USE  *.
.*                 BAKR                                             *.
.*       NEWBASE:  IF NON-BLANK, WILL PRIMT NEWBASE FROM GPR 15,     *.
.*                 THEN ISSUE 'USING &LBL,&NEWBASE'                 *.
.*       INITBASE: MUTUALLY EXCLUSIVE WITH NEWBASE. IF NON-BLANK,    *.
.*                 THE FOLLOWING CODE IS GENERATED:                 *.
.*                     BASR  &INITBASE,Ø                            *.
```

```
.*                    USING *,&INITBASE                            *.
.*                    L     &INITBASE,=A(&LBL)                     *.
.*                    USING &LBL,&INITBASE                         *.
.*       LIST:     IF YES, ISSUE 'PRINT ON,GEN', ELSE 'PRINT OFF'   *.
.*                                                                  *.
.* THIS MACRO CANNOT BE NESTED - I.E. YOU CANNOT HAVE TWO MODENTRY  *.
.* STATEMENTS WITHOUT AN INTERVENING MODEXIT - THIS IS ENFORCED.    *.
.*------------------------------------------------------------------*.
         GBLC  &CURMOD
         GBLB  &BAKROFF
         AIF   ('&NEWBASE.&INITBASE' EQ '').NOLTORG
         PUSH  PRINT
         AIF   ('&LIST' EQ 'YES').PRTALL
         PRINT OFF
.PRTALL  ANOP
         LTORG
         POP   PRINT
.NOLTORG ANOP
         AIF   ('&CURMOD' NE '').BUSY
&CURMOD  SETC  '&LBL'
         DS    ØH
         DC    C'  &LBL  '
&LBL     DS    ØH
&BAKROFF SETB  ('&BAKR'(1,1) NE 'Y')
         AIF   (&BAKROFF).NEWBASE
         BAKR  R14,Ø
.NEWBASE ANOP
         AIF   ('&NEWBASE' EQ '').INITBASE
         DROP  &NEWBASE
         LR    &NEWBASE,R15
         USING &LBL,&NEWBASE
         AGO   .NOBASE
.INITBASE ANOP
         AIF   ('&INITBASE' EQ '').NOBASE
         DROP  &INITBASE
         BASR  &INITBASE,Ø
         USING *,&INITBASE
         L     &INITBASE,=A(&LBL)
         USING &LBL,&INITBASE
         AIF   (&BAKROFF).NOBASE
         MEXIT
.NOBASE  ANOP
         L     R15,@NXTRET@
         ST    R14,Ø(,R15)
         LA    R15,4(,R15)
         ST    R15,@NXTRET@
         MEXIT
         MEXIT
.BUSY    ANOP
         MNOTE 8,'''MODEXIT'' REQUIRED TO CLOSE MODULE ''&CURMOD'''
```

```
         MEND
         SPACE 3
         MACRO
&LBL     MODEXIT
.*-------------------------------------------------------------------*.
.* THIS MACRO IS THE EXIT FROM A SUB-PROCEDURE.                      *.
.* IF THE PRECEDING MODENTRY HAD BEEN CALLED WITH 'BAKR=YES', THEN   *.
.* A 'PR' INSTRUCTION WILL BE GENERATED,  ELSE THE RETURN STACK IS   *.
.* USED.                                                             *.
.*-------------------------------------------------------------------*.
         GBLC  &CURMOD
         GBLB  &BAKROFF
         AIF   ('&CURMOD' EQ '').NOTBUSY
&CURMOD  SETC  ''
&LBL     DS    ØH
         AIF   (&BAKROFF).RETURN
         PR
         MEXIT
.RETURN  ANOP
         L     R15,@NXTRET@
         S     R15,=F'4'
         ST    R15,@NXTRET@
         L     R14,Ø(,R15)
         BR    R14
         MEXIT
.NOTBUSY ANOP
         MNOTE 8,'''MODEXIT'' NOT PRECEDED BY ''MODENTRY'''
         MEND
         EJECT
         MACRO
&LBL     INIT_RETURN_STACK
.*-------------------------------------------------------------------*.
.* THIS MACRO INITIALIZES THE RETURN STACK FOR USE BY MODENTRY AND   *.
.* MODEXIT.                                                          *.
.*-------------------------------------------------------------------*.
&LBL     LA    R15,RETSTACK
         ST    R15,@NXTRET@
         S     R15,=F'16'
         MVC   Ø(16,15),=CL16'RETURN STACK'
         MEND
         SPACE 3
         MACRO
         RETSTACK  &SIZE=64
.*-------------------------------------------------------------------*.
.* THIS MACRO DEFINES THE RETURN STACK                               *.
.* PARAMETER SIZE = SUMBER OF FULLWORDS TO DEFINE FOR THE STACK.     *.
.*-------------------------------------------------------------------*.
@NXTRET@ DS    F
         DS    2D
RETSTACK DS    &SIZE.A
         MEND
```

PIFBROI0

```
PUNCH ' MODE AMODE(31),RMODE(ANY) '
         PUNCH ' SETOPT PARM(REUS(RENT),LIST,XREF) '
         PUNCH ' ENTRY PIFBROIØ '
         PRINT OFF,NOPRINT
         COPY  PPFC14MØ                .IF-ELSE-ENDIF, LOOPS, etc.
         COPY  PIFISPCØ                .ISPEXEC MACRO
         COPY  PIETPERF                .PERFORM STRUCTURES
         PRINT ON,NOGEN,NOPRINT
PIFBROIØ CSECT
PIFBROIØ AMODE 31
PIFBROIØ RMODE ANY
         B     CODE_START-*(,R15)
         DC    AL1(L'EP_LITERAL)
EP_LITERAL     DC  C'PIFBROIØ..DATE=&SYSDATC..TIME=&SYSTIME..BROWSE
REP+
               LACEMENT WHICH WILL ALSO BROWSE KSDS AND ESDS'
CODE_START     DS  ØH
         BAKR  R14,Ø                  .ESA save with hardware stack
         LR    R12,R15
         USING PIFBROIØ,R12,R11
         LA    R11,2Ø48(,R12)
         LA    R11,2Ø48(,R11)
         STORAGE OBTAIN,LENGTH=WSLEN,LOC=RES,COND=NO,BNDRY=PAGE
         LR    R13,R1
         LR    R1,RØ                  .
         LR    RØ,R13                 ..
         LR    R14,RØ                 ...Clear obtained area
         XR    R15,R15                ..
         MVCL  RØ,R14                 .
         MVC   4(4,R13),=C'F1SA'      .Indicate use of linkage stack
         USING MYSAVE,R13
         MVC   READSAVE+4(4),=C'F1SA'
         MVC   CMDSAVE+4(4),=C'F1SA'
         INIT_RETURN_STACK             .set up return stack
         ISPFPARM CALLPARM             .SET ISPF MACS TO USE CALLPARM
         PERF  INIT                    .get parms, set up rent DCB's
         PERF  ISPINIT                 .set up ISPF variables, etc.
*        ISPEXEC VGET,VARS=(ZMLVEXIT,ZMLVDATA)
*        CSVQUERY INADDR=ZMLVEXIT,OUTEPNM=DUB,MF=(E,CSVQUERY,COMPLETE)
*        L     R1,ZMLVEXIT
*        IF    LTR,R15,R15,NZ,OR,CLC,=C'ISRMLVDS',NE,5(R1)
*        MVC   ZMLVEXIT,=A(BR14)
*        ENDIF
*        ISPEXEC CONTROL,PARM=(DISPLAY,SAVE)
         DO    INF
         XC    ZEDSMSG,ZEDSMSG
         ISPEXEC DISPLAY,PANEL=(*,PANELNM)
         DOEXIT LTR,R15,R15,NZ
            PERF  BROWSE
```

```
         ENDDO
*        ISPEXEC CONTROL,PARM=(DISPLAY,RESTORE)
         XR    R4,R4
         PERF  CLEANUP                 .Clear vars, erase table
DONE     DS    ØH
         LR    R1,R13
         STORAGE RELEASE,LENGTH=WSLEN,ADDR=(1)
         LR    R15,R4                  .Set R15 = Return Code
RETURN   DS    ØH
         PR
         EJECT
*************************************************************************
* Initialize certain control blocks                                     *
*************************************************************************
INIT     MODENTRY
         LA    RØ,CAMNAME
         L     R14,=A(MASTERS)
         L     R15,=A(MASTLEN)
         LR    R1,R15
         MVCL  RØ,R14
         EREG  R1,R1
         L     R1,Ø(,R1)
         IF    CLC,=H'Ø',EQ,Ø(R1)
         MVC   PANELNM,=C'ISRBROØ1'
         ELSE
         LH    R2,Ø(,R1)
         BCTR  R2,Ø
         MVC   PANELNM,=CL8' '
         EX    R2,MOVPNL
         ENDIF
         MODEXIT
MOVPNL   MVC   PANELNM(Ø),2(R1)
         EJECT
*************************************************************************
* Define all fields used by ISPF                                        *
*************************************************************************
ISPINIT  MODENTRY       .SET UP ISPF VARS
*        ISPEXEC VDEFINE,FLD=ZMLVEXIT,FMT=FIXED
*        ISPEXEC VDEFINE,FLD=ZMLVDATA,FMT=FIXED
         ISPEXEC VDEFINE,FLD=MAXRECNO,FMT=FIXED
         ISPEXEC VDEFINE,FLD=CRP,FMT=FIXED
         ISPEXEC VDEFINE,FLD=CAMDSN
         ISPEXEC VDEFINE,FLD=BRIFTITL
         ISPEXEC VDEFINE,FLD=ZCMD
         ISPEXEC VDEFINE,FLD=ZBCPRJ1
         ISPEXEC VDEFINE,FLD=ZBCLIB1
         ISPEXEC VDEFINE,FLD=ZBCLIB2
         ISPEXEC VDEFINE,FLD=ZBCLIB3
         ISPEXEC VDEFINE,FLD=ZBCLIB4
         ISPEXEC VDEFINE,FLD=ZBCTYP1
         ISPEXEC VDEFINE,FLD=ZMEM
```

```
         ISPEXEC VDEFINE,FLD=ZBCODSN
         ISPEXEC VDEFINE,FLD=ZODSN
         ISPEXEC VDEFINE,FLD=ZVOL
         ISPEXEC VDEFINE,FLD=ZPSWD
         ISPEXEC VDEFINE,FLD=ZBCMIX
         ISPEXEC VDEFINE,FLD=ZBCFNAM
         ISPEXEC VDEFINE,FLD=DATAID
         ISPEXEC VDEFINE,FLD=ZPREFIX
         ISPEXEC VDEFINE,FLD=ZERRMSG
         ISPEXEC VDEFINE,FLD=ZEDSMSG
         ISPEXEC VDEFINE,FLD=ZEDLMSG
         ISPEXEC VDEFINE,FLD=P6
         ISPEXEC VDEFINE,FLD=P7
         MODEXIT
         EJECT
BROWSE   MODENTRY
         PERF  FINDDSN
         CLI   ZEDSMSG,C'A'
         BNL   BRWEXIT
         USING IECSDSL1,CAMDSN
         SELECT
         WHEN  TM,DS1DSORG+1,DS1ORGAM,O     .VSAM?
            PERF  BRWVSAM                   .YES
         WHEN  CLI,ZBCODSN,GE,C'A',OR,CLI,ZBCODSN,EQ,C''''
            ISPEXEC BROWSE,DSN=(*,ZBCODSN),VOL=(*,ZVOL)
         WHEN  NONE
            ISPEXEC LMINIT,DATAID=DATAID,PROJECT=(*,ZBCPRJ1),          +
               GROUP1=(*,ZBCLIB1),GROUP2=(*,ZBCLIB2),                  +
               GROUP3=(*,ZBCLIB3),GROUP4=(*,ZBCLIB4),                  +
               DISP=SHR,VOL=(*,ZVOL),PASSWORD=(*,ZPSWD),               +
               TYPE=(*,ZBCTYP1)
            IF    LTR,R15,R15,Z
               IF    CLI,ZMEM,GE,C'A'
                  ISPEXEC BROWSE,DATAID=DATAID,MBR=(*,ZMEM)
               ELSE
                  ISPEXEC BROWSE,DATAID=DATAID
               ENDIF
            ENDIF
         ENDSEL
BRWEXIT  DS    ØH
         MODEXIT
         EJECT
*********************************************************************
* Find the desired DSN, and read its format-1 DSCB                  *
*********************************************************************
FINDDSN  MODENTRY
         IF    CLI,ZBCODSN,GE,C'A',OR,CLI,ZBCODSN,EQ,C''''
            PERF  COPYDSN
         ELSE
            PERF  PRJ_GRP
         ENDIF
```

```
         LA    R1,CAMDSN
         XR    R2,R2
         LA    R3,CAMWRK
         STM   R1,R3,CAMNAME+4
         LA    R2,CAMVOL
         STM   R1,R3,CAMDSCB1+4
         IF    CLI,ZVOL,GE,C'A'
         MVC   CAMVOL,ZVOL
         ELSE
            LOCATE CAMNAME
            IF    LTR,R15,R15,NZ
               ISPEXEC SETMSG,MSG=ISRDØ28
               B     FINDEXIT
            ELSE
               MVC   CAMVOL,CAMWRK+6
            ENDIF
         ENDIF
         OBTAIN CAMDSCB1
         IF    LTR,R15,R15,NZ
            ISPEXEC SETMSG,MSG=ISRDØ32
         ENDIF
FINDEXIT DS    ØH
         MODEXIT
         EJECT
COPYDSN  MODENTRY
         LA    R3,CAMDSN
         MVC   CAMDSN,=CL44' '
         LA    R4,ZBCODSN
         IF    CLI,Ø(R4),EQ,C''''
            LA    R4,ZBCODSN+1
         ELSE
            ISPEXEC VGET,VARS=ZPREFIX
            IF    CLI,ZPREFIX,GE,C'A'
               MVC   CAMDSN(8),ZPREFIX
               DO    WHILE=(CLI,Ø(R3),NE,X'4Ø')
                  LA    R3,1(,R3)
               ENDDO
               MVI   Ø(R3),C'.'
               LA    R3,1(,R3)
            ENDIF
         ENDIF
         DO    WHILE=(CLI,Ø(R4),NE,X'4Ø')
         DOEXIT CLI,Ø(R4),EQ,C''''
            MVC   Ø(1,R3),Ø(R4)
            LA    R3,1(,R3)
            LA    R4,1(,R4)
         ENDDO
         MODEXIT
         EJECT
PRJ_GRP  MODENTRY
         MVC   CAMDSN,=CL44' '
```

```
         MVC   CAMDSN(8),ZBCPRJ1
         LA    R3,CAMDSN
         DO    WHILE=(CLI,Ø(R3),GE,C'A')
            LA    R3,1(,R3)
         ENDDO
         MVI   Ø(R3),C'.'
         LA    R3,1(,R3)
         MVC   Ø(8,R3),ZBCLIB1
         DO    WHILE=(CLI,Ø(R3),GE,C'A')
            LA    R3,1(,R3)
         ENDDO
         MVI   Ø(R3),C'.'
         LA    R3,1(,R3)
         MVC   Ø(8,R3),ZBCTYP1
         MODEXIT
         EJECT
*********************************************************************
* Browse ESDS, KSDS or RRDS by primary sequence. Sorry, no AIX support*
*********************************************************************
BRWVSAM  MODENTRY
         PERF  ALLOC
         CLI   ZEDSMSG,C'A'
         BNL   VSEXIT
         GENCB BLK=EXLST,EODAD=EODAD,MF=(G,CALLPARM),WAREA=(S,EXLST),  +
               LENGTH=EXLST_SIZE
*********************************************************************
* Generate an ACB, default access by RBA (MACRF=ADR)                *
*********************************************************************
         GENCB BLK=ACB,BUFND=2Ø,DDNAME=(*,DYNDDN),RMODE31=ALL,         +
               MACRF=(ADR,SEQ,IN),MF=(G,CALLPARM),EXLST=(S,EXLST),     +
               WAREA=(S,ACB),LENGTH=ACB_SIZE
*********************************************************************
* Generate an RPL for locate mode, so we can retrieve the RBA's     *
*********************************************************************
         GENCB BLK=RPL,ACB=(S,ACB),OPTCD=(ADR,SEQ,LOC),                +
               AREA=(S,DUB),AREALEN=4,MF=(G,CALLPARM),WAREA=(S,RPL),   +
               LENGTH=RPL_SIZE
         OPEN  ACB,MODE=31,MF=(E,OPENIN)
*********************************************************************
* Retrieve record count (NLOGR), Key length and record length       *
*********************************************************************
         SHOWCB ACB=(S,ACB),AREA=(S,MAXRECNO),LENGTH=12,               +
               FIELDS=(NLOGR,KEYLEN,LRECL),MF=(G,CALLPARM)
         IF    CLC,MAXRECNO,EQ,=F'Ø'
            ISPEXEC SETMSG,MSG=ISRB1Ø7
         ELSE
            IF    ICM,R1,15,KEYLEN,NZ       .Non-zero keylength?
               MVC   BRIFTITL(7),=C'KSDS—'
               CLOSE ACB,MODE=31,MF=(E,CLOSE)
               MODCB ACB=(S,ACB),MACRF=(KEY,SEQ,IN),MF=(G,CALLPARM)
               MODCB RPL=(S,RPL),OPTCD=(KEY,SEQ,LOC),MF=(G,CALLPARM)
```

```
               OPEN  ACB,MODE=31,MF=(E,OPENIN)
            ENDIF
            PERF  GETRBAS       .Read file, find each rec's RBA
            LTR   R4,R4
            BNZ   VSAM_DONE
            MVC   BRIFTITL+7(44),CAMDSN
            ST    R13,SA_PTRS
 PRINT GEN,MCALL
*           ISPEXEC CONTROL,PARM=(DISPLAY,SAVE)
            LINK  EP=ISPLINK,                                          +
               PARAM=(=CL8'BRIF',           .service name              +
               BRIFTITL,                    .browse display title      +
               =C'V ',                      .recfm                     +
               MAXREC,                      .max record len            +
               =A(BRIFREAD),                .a(read routine)           +
               =A(BRIFCMD),                 .a(command interpreter)    +
               SA_PTRS),                    .dialog data               +
               VL=1,MF=(E,CALLPARM)
*           ISPEXEC CONTROL,PARM=(DISPLAY,RESTORE)
         ENDIF
VSAM_DONE      DS  ØH
         CLOSE ACB,MODE=31,MF=(E,CLOSE)
         PERF  FREE
VSEXIT   DS    ØH
         MODEXIT
         EJECT
ALLOC    MODENTRY
         LA    R1,S99RB
         ST    R1,S99RBPTR
         OI    S99RBPTR,X'8Ø'
         MVI   S99RBLN,2Ø
         MVI   S99VERB,S99VRBAL
         LA    R1,S99TUPL
         ST    R1,S99TUPP
         LA    R1,DALDSNAM
         LA    R2,DALSTATS
         LA    R3,DALRTDDN
         STM   R1,R3,S99TUPL
         OI    S99TUPL+8,X'8Ø'
         MVC   DALDSNAM+6(44),CAMDSN
         LA    R1,S99RBPTR
         DYNALLOC
         IF    LTR,R15,R15,NZ
            MVC   ZEDSMSG,=CL3Ø'DYNALLOC ERROR.'
            MVC   EMRETCOD,S99ERROR
            LA    R1,S99RB
            ST    R1,EMS99RBP
            LA    R1,CAMWRK
            ST    R1,EMBUFP
            LINK  EP=IEFDB476,PARAM=EMPARMS,MF=(E,CALLPARM)
            IF    LTR,R15,R15,Z
```

```
               LH    R1,CAMWRK
               IF    CH,R1,GT,=H'100'
                  LA    R1,100
               ENDIF
               BCTR  R1,0
               EX    R1,MOVMSG
               ISPEXEC SETMSG,MSG=ISRZ001
            ELSE
               MVC   ZEDLMSG(100),=CL100'Unable to retrieve dynalloc er+
               ror text'
            ENDIF
         ENDIF
         MODEXIT
MOVMSG   MVC   ZEDLMSG(0),CAMWRK+2
         EJECT
GETRBAS  MODENTRY
*********************************************************************
* Read each record sequentially, storing its RBA in a table.       *
* This makes random access (as in 'DOWN n' or 'UP n') easier        *
* It is not possible to read a RRDS by RBA; a RPL FDBK of X'c4'     *
* indicates that you are using RBA access on a RRDS.                *
* If this occurs, we bypass the table processing, since the record  *
* number is the key of the file.                                    *
*********************************************************************
*        IF    LTR,R4,R15,NZ
*           MVC   ZEDSMSG,=CL30'TBCREATE error.'
*           MVC   ZEDLMSG,=CL100'TBCREATE return code='
*           CVD   R4,DUB
*           OI    DUB+7,X'0F'
*           UNPK  ZEDLMSG+21(3),DUB+6(2)
*           ISPEXEC SETMSG,MSG=ISRZ001
*           B     READ_DONE
*        ENDIF
         MVI   TYPE_RRDS,C'N'
         L     R0,MAXRECNO
         SLL   R0,2
         STORAGE OBTAIN,LENGTH=(R0),LOC=ANY
         ST    R1,@RBALIST
         LR    R9,R1
         DO    INF
            GET   RPL=RPL
            IF    LTR,R4,R15,NZ
               SHOWCB RPL=(S,RPL),AREA=(S,DUB),LENGTH=4,FIELDS=FDBK,   +
               MF=(G,CALLPARM)
               IF    CLC,=A(X'C4'),EQ,DUB
                  MVI   TYPE_RRDS,C'Y'
                  MVC   BRIFTITL(7),=C'RRDS—'
                  XR    R4,R4
                  B     EODAD
               ENDIF
               MVC   ZEDSMSG,=CL30'VSAM Read error.'
```

```
               MVC   ZEDLMSG,=CL1ØØ'Sequential read, FDBK=X''...._...''+
               '
               UNPK  ZEDLMSG+24(9),DUB(5)
               MVI   ZEDLMSG+24+8,C''''
               TR    ZEDLMSG+24(8),HEX-C'Ø'
               ISPEXEC SETMSG,MSG=ISRZØØ1
               CLOSE ACB,MODE=31,MF=(E,CLOSE)
               B     READ_DONE
            ENDIF
            SHOWCB RPL=(S,RPL),AREA=((R9)),LENGTH=4,FIELDS=RBA,        +
               MF=(G,CALLPARM)
            LA    R9,4(,R9)
         ENDDO
EODAD    DS    ØH
         IF    CLI,BRIFTITL,LT,C'A'
            MVC   BRIFTITL(7),=C'ESDS—'
         ENDIF
         CLOSE ACB,MODE=31,MF=(E,CLOSE)
         IF    CLI,TYPE_RRDS,NE,C'Y'
            MODCB ACB=(S,ACB),MACRF=(ADR,DIR,IN),EXLST=Ø,              +
               MF=(G,CALLPARM)
            MODCB RPL=(S,RPL),OPTCD=(ADR,DIR,LOC),ARG=(S,RBA),         +
               MF=(G,CALLPARM)
         ELSE
            MODCB ACB=(S,ACB),MACRF=(KEY,DIR,IN),EXLST=Ø,              +
               MF=(G,CALLPARM)
            MODCB RPL=(S,RPL),OPTCD=(KEY,DIR,LOC),ARG=(S,RBA),         +
               MF=(G,CALLPARM)
         ENDIF
         OPEN  ACB,MODE=31,MF=(E,OPENIN)
READ_DONE      DS  ØH
         IF    ICM,R1,15,@RBALIST,NZ
         L     RØ,MAXRECNO
         SLL   RØ,2
         STORAGE RELEASE,LENGTH=(RØ),ADDR=(1)
         ENDIF
         MODEXIT
         EJECT
FREE     MODENTRY
         MVI   S99VERB,S99VRBUN
         LA    R1,DALDDNAM
         ST    R1,S99TUPL
         OI    S99TUPL,X'8Ø'
         MVC   DALDDNAM+6(8),DYNDDN
         LA    R1,S99RBPTR
         DYNALLOC
         DLVRP MODE=31,MF=(E,DLVRP)
         MODEXIT
         EJECT
CLEANUP  MODENTRY
 WTO 'BEFORE VRESET'
```

```
         ISPEXEC VRESET
 WTO 'AFTER VRESET'
         MODEXIT
         EJECT
         LTORG
         EJECT
         DROP
BR14     DS    ØH
         XR    R15,R15
         BR    R14
*************************************************************************
* Read routine called by BRIF.                                          *
* Parameters are: 1) Record area                                        *
*                 2) Record length                                      *
*                 3) Record number required.                            *
*                 4) Program data. In this case, the address of the     *
*                                  dynamic (obtained) area              *
*************************************************************************
BRIFREAD DS    ØD
         B     BRIFSTART-*(,R15)
         DC    AL1(L'BRIF_NAME)
BRIF_NAME      DC  C'BRIFREAD..DATE=&SYSDATC..TIME=&SYSTIME'
BRIFSTART      DS  ØH
         BAKR  R14,Ø
         LR    R12,R15
         USING BRIFREAD,R12
         L     R13,12(,R1)
         L     R13,Ø(,R13)
         LA    R13,READSAVE-MYSAVE(,R13)
         MVC   4(4,R13),=C'F1SA'
         USING READSAVE,R13
         LM    R6,R8,Ø(R1)
*************************************************************************
* Locate the required record; if beyond EOF, return the last record     *
* with RC=8, indicating EOF. Also return the number of the last record*
*************************************************************************
         XR    R4,R4
         MVC   RBA,Ø(R8)
         IF    CLC,RBA,GT,MAXRECNO
            MVC   RBA,MAXRECNO
            MVC   Ø(4,R8),MAXRECNO
            LA    R4,8
         ENDIF
         IF    CLI,TYPE_RRDS,NE,C'Y'
            L     R9,RBA
            BCTR  R9,Ø
            SLL   R9,2
            A     R9,@RBALIST
            MVC   RBA,Ø(R9)
         ENDIF
         GET   RPL=RPL
```

```
         IF    LTR,R15,R15,Z
            MVC   Ø(4,R6),DUB
            SHOWCB RPL=(S,RPL),AREA=(R7),LENGTH=4,FIELDS=RECLEN,       +
               MF=(G,CALLPARM)
         ELSE
            SHOWCB RPL=(S,RPL),AREA=(S,DUB),LENGTH=4,FIELDS=FDBK,      +
               MF=(G,CALLPARM)
            IF    CLC,=A(X'1Ø'),EQ,DUB      .record not found
               LA    R1,EMPTY               .(most likely RRDS with
               ST    R1,Ø(R6)               .empty record)
               MVC   Ø(4,R7),=A(L'EMPTY)    .set zero length
            ELSE
               MVC   ZEDSMSG,=CL3Ø'VSAM READ ERROR.'
               MVC   ZEDLMSG,=CL1ØØ'READ BY RBA, FDBK=X''....+...'''
               UNPK  ZEDLMSG+2Ø(9),DUB(5)
               MVI   ZEDLMSG+2Ø+8,C''''
               TR    ZEDLMSG+2Ø(8),HEX-C'Ø'
               ISPEXEC SETMSG,MSG=ISRZØØ1
               LA    R4,16
            ENDIF
         ENDIF
         LR    R15,R4
         EREG  RØ,R1
READRET  DS    ØH
         PR
         EJECT
         DROP
*********************************************************************
* Command handler routine called by BRIF. At this stage does nothing, *
* so all commands are the processed by ISPF. This leaves the program  *
* a back door to have specific commands not defined to ISPF           *
* Parameters are: 1) Command                                          *
*                 2) Program data. In this case, the addrss if the    *
*                                  dynamic (obtained) area            *
*********************************************************************
BRIFCMD  DS    ØD
         B     CMD_START-*(,R15)
         DC    AL1(L'CMD_NAME)
CMD_NAME DC    C'BRIFCMD..DATE=&SYSDATC..TIME=&SYSTIME'
CMD_START      DS  ØH
         BAKR  R14,Ø
         LR    R12,R15
         USING BRIFCMD,R12
         L     R13,4(,R1)
         L     R13,Ø(,R13)
         LA    R13,CMDSAVE-MYSAVE(,R13)
         MVC   4(4,R13),=C'F1SA'
         USING CMDSAVE,R13
         L     R3,Ø(,R1)
         L     R4,Ø(,R4)
 wto 'brifcmd called'
```

```
         LA    R15,4
CMDRET   DS    ØH
         PR
         EJECT
ISRSUBS  DC    CL8'ISRSUBS'
HEX      DC    C'Ø123456789ABCDEF'
EMPTY    DC    C'*** EMPTY RECORD ***'
*********************************************************************
* CONSTANTS TO BE COPIED TO THE DYNAMIC WORKING STORAGE              *
*********************************************************************
MASTERS  DS    ØF
         CAMLST NAME,Ø,,Ø
         CAMLST SEARCH,Ø,Ø,Ø
         OPEN  Ø,MODE=31,MF=L
         CLOSE Ø,MODE=31,MF=L
         BLDVRP BUFFERS=(4Ø96(3)),MODE=31,MF=L
         LOAD  Ø,LOADPT=Ø,SF=L
* TEXT UNITS
         DC    AL2(1,1,8),CL8' '
         DC    AL2(2,1,44),CL44' '
         DC    XL2'55,1,8',CL8' '
         DC    AL2(4,1,1),AL1(8,Ø)
* EMPARMS
         DS    ØF'Ø',ØCL(EMSIZE)
         DC    AL1(EMRETURN)
         DC    AL1(EMSVC99)
         DS    X
         DS    X
         DS    F
         DS    F
         DS    F
         DS    F
         DS    2F
MASTLEN  EQU   *-MASTERS
*********************************************************************
* END OF RE-LOCATABLE CONSTANTS                                      *
*********************************************************************
         EJECT
WRKSTOR  DSECT
MYSAVE   DS    9D
READSAVE DS    9D
CMDSAVE  DS    9D
SA_PTRS  DS    2F
*********************************************************************
* CONTROL BLOCKS FOR CATALOG MANAGEMENT                              *
*********************************************************************
CAMNAME  CAMLST NAME,Ø,,Ø
CAMDSCB1 CAMLST SEARCH,Ø,Ø,Ø
*********************************************************************
* OPEN AND CLOSE LISTS                                               *
*********************************************************************
```

```
OPENIN   OPEN  Ø,MODE=31,MF=L
CLOSE    CLOSE Ø,MODE=31,MF=L
*********************************************************************
* PARAMETER BLOCK TO FREE BUFFERS THAT WERE AUTOMATICALLY OBTAINED   *
* DURING OPEN PROCESSING. IF THIS IS NOT DONE, S878 ABENDS WILL      *
* EVENTUALLY RESULT.                                                 *
*********************************************************************
DLVRP    BLDVRP BUFFERS=(4Ø96(3)),MODE=31,MF=L
*********************************************************************
* LOAD PARAMETER BLOCK                                              *
*********************************************************************
LOAD     LOAD  Ø,LOADPT=Ø,SF=L
*********************************************************************
* TEXT UNITS FOR DYNAMIC ALLOCATION                                 *
*********************************************************************
DALDDNAM DC    AL2(1,1,8),CL8' '      .DD NAME
DALDSNAM DC    AL2(2,1,44),CL44' '    .DSN
DALRTDDN DC    XL2'55,1,8',CL8' '     .DYNAMIC DD NAME TO BE RETURNED
DYNDDN   EQU   DALRTDDN+6,8
DALSTATS DC    AL2(4,1,1),AL1(8,Ø)    .STATUS (DISP SHR)
*********************************************************************
* PARAMETER BLOCK TO RETRIEVE DYNALLOC ERROR MESSAGES               *
*********************************************************************
EMPARMS  DS    ØF
EMFUNCT  DC    AL1(EMRETURN)
EMRETURN EQU   X'2Ø'
EMIDNUM  DC    AL1(EMSVC99)
EMSVC99  EQU   5Ø
EMNMSGBK DS    X
         DS    X
EMS99RBP DS    F
EMRETCOD DS    F
EMCPPLP  DS    F
EMBUFP   DS    F
         DS    2F
EMSIZE   EQU   *-EMPARMS
*********************************************************************
* DYNALLOC REQUEST BLOCK                                            *
*********************************************************************
S99RB    DS    ØCL2Ø
S99RBLN  DS    X
S99VERB  DS    X
S99VRBAL EQU   1
S99VRBUN EQU   2
         DS    H
S99ERROR DS    H
S99INFO  DS    H
S99TUPP  DS    F
S99RBXP  DS    F
         DS    F
S99TUPL  DS    1ØF
```

```
DUB      DS    D
         DS    D
MAXRECNO DS    F
KEYLEN   DS    F
MAXREC   DS    F
         DS    F
CALLPARM DS    2ØF
ZMLVEXIT DS    F
ZMLVDATA DS    F
         CSVQUERY MF=(L,CSVQUERY)
S99RBPTR DS    A
PANELNM  DS    CL8
@RBALIST DS    F
RBA      DS    F
CRP      DS    F
ZCMD     DS    CL8Ø
DATAID   DS    CL8
ZERRMSG  DS    CL8
ZEDSMSG  DS    CL3Ø
ZEDLMSG  DS    CL1ØØ
ZBCPRJ1  DS    CL8
ZBCLIB1  DS    CL8
ZBCLIB2  DS    CL8
ZBCLIB3  DS    CL8
ZBCLIB4  DS    CL8
ZBCTYP1  DS    CL8
ZMEM     DS    CL8
ZODSN    DS    CL6Ø
ZBCODSN  DS    CL6Ø
ZVOL     DS    CL6
ZPSWD    DS    CL8
ZBCMIX   DS    CL3
ZBCFNAM  DS    CL8
ZPREFIX  DS    CL8
BRIFTITL DS    CL6Ø
RECFM    DS    CL8
TYPE_RRDS      DS  C
         CNOP  4,8
ACB      ACB   AM=VSAM
ACB_SIZE EQU   *-ACB
RPL      RPL   AM=VSAM
RPL_SIZE EQU   *-RPL
EXLST    EXLST AM=VSAM
EXLST_SIZE     EQU *-EXLST
CAMVOL   DS    CL6
P6       EQU   CAMVOL,6
         DS    ØD
         DS    F
*************************************************************************
* CAMDSN AND CAMWRK MUST BE CONTIGUOUS. THIS WILL ALLOW US TO MAP THE *
* AREA WITH IECSDSL1, SINCE THE CATALOG MANAGEMENT MACROS ONLY        *
```

```
* RETRIEVE THE NON-KEY (NON-DSN) PORTION OF THE FORMAT 1 DSCB         *
************************************************************************
CAMDSN   DS    CL44
P7       EQU   CAMDSN,44
CAMWRK   DS    ØD,CL512
         RETSTACK
WSLEN    EQU   *-WRKSTOR
         REGEQU
         DCBD  DSORG=PS
         PRINT GEN
DSCB1    DSECT
         IECSDSL1 1
         END
```

PIFISPC0

```
        MACRO
         ISPFPARM &PARMLST
         GBLC  &PLST
         AIF   ('&PARMLST' EQ '').BADPARM
&PLST    SETC  '&PARMLST'
         MEXIT
.BADPARM ANOP
         MNOTE 8,'THE NAME OF A VARIABLE CONSISTING OF AT LEAST 6 FULLW+
               ORDS REQUIRED'
         MEND
************************************************************************
         MACRO
         TESTPLST
         GBLC  &PLST
         AIF   ('&PLST' EQ '').SETUP
         MEXIT
.SETUP   ANOP
&PLST    SETC  'ISPF_CALL_PARMS'
         CNOP  Ø,4
         B     &PLST+L'&PLST
&PLST    DS    1ØF
         MEND
************************************************************************
         MACRO
         FLDLIST &KEYS
         GBLC  &FLDLST
         AIF   ('&KEYS   '(1,3) EQ '(*,').REF
         AIF   ('&KEYS' NE '').NOTNULL
&FLDLST  SETC  '=C'' '''
         MEXIT
.NOTNULL ANOP
&FLDLST  SETC  '&KEYS'
         AIF   ('&FLDLST'(1,1) EQ '(').KEYSOK1
&FLDLST  SETC  '(&FLDLST)'
```

```
.KEYSOK1 ANOP
&FLDLST  SETC  '=C''&FLDLST'''
         MEXIT
.REF     ANOP
&FLDLST  SETC  '&KEYS(2)'
         MEND
*********************************************************************
         MACRO
         SETFLD &FIELD
         GBLC  &ISPFLD
         AIF   ('&FIELD   '(1,3) EQ '(*,').FLDREF
         AIF   ('&FIELD '(1,1) EQ '(').FLDREG
&ISPFLD  SETC  '=CL8''&FIELD '''
         MEXIT
.FLDREF  ANOP
&ISPFLD  SETC  '&FIELD(2)'
         MEXIT
.FLDREG  ANOP
&ISPFLD  SETC  '&FIELD'
         MEND
*********************************************************************
         MACRO
         SETTBWRT &WRITE
         GBLC  &TBWRT
         AIF   ('&WRITE' NE '').TESTWRT
         MNOTE Ø,'TBCREATE, DEFAULT=WRITE'
         AGO   .WRITE
.TESTWRT ANOP
         AIF   ('&WRITE   '(1,3) EQ '(*,').REF
         AIF   ('&WRITE '(1,1) EQ 'N').NOWRT
         AIF   ('&WRITE '(1,1) NE 'Y').BADWRT
.WRITE   ANOP
&TBWRT   SETC  '=CL8''WRITE'''
         MEXIT
.NOWRT   ANOP
&TBWRT   SETC  '=CL8''NOWRITE'''
         MEXIT
.REF     ANOP
&TBWRT   SETC  '&WRITE(2)'
         MEXIT
.BADWRT  MNOTE 8,'ERROR: FORMAT IS ''TBCREATE <TBL-NAME-VAR>,<WRITE/NOW+
               WRITE>,KEYS=<KEY FIELDS>,FIELDS=<OTHER TABLE FIELDS>,REP+
               LACE=<Y/N>'''
         MEND
*********************************************************************
         MACRO
         ISPEXEC &SERVICE,                                          +
               &PGM=,&CMD=,&PANEL=,&OPT=,&LANG=,&PARM=,&MODE=,&WRK=,   +
               &TBNAME=,&WRITE=,&KEYS=,&FIELDS=,&REPLACE=,   TBCREATE  +
               &CRP=,&NOREAD=,                               TBBOTTOM  +
               &NAMES=,&ROWNUM=,&KEYNUM=,&NAMENUM=,&POS=,    TBQUERY   +
```

```
              &COUNT=,&ROWID=,                                    TBSKIP    +
              &MSG=,&CURSOR=,&CSRROW=,&AUTOSEL=N,&MSGLOC=,        TBDISPL   +
              &ORDER=NO,               (YES/NO)                   TBPUT     +
              &DIR=,&READ=,&SAVENAME=,&COND=,                     TBSCAN    +
              &VARS=,&POOL=,                                      VGET      +
              &FLD=,&FMT=CHAR,&LEN=,                              VDEFINE   +
              &DATAID=,                                           LMXXXXXX  +
              &PROJECT=,&GROUP1=,&GROUP2=,&GROUP3=,               LMINIT    +
              &GROUP4=,&TYPE=,&DSN=,&DDN=,&VOL=,&PASSWORD=,       LMINIT    +
              &DISP=SHR,                                          LMINIT    +
              &VAR=,&LMMODE=INVAR,&LENVAR=,&MAXLEN=,              LMGET     +
              &MBR=,                                              LMMFIND   +
              &PIET
         GBLC  &PLST,&TBN,&FLDLST,&ISPFLD,&TBWRT
         TESTPLST
         AIF   ('&SERVICE' EQ 'SELECT').SELECT
         AIF   ('&SERVICE' EQ 'TBCREATE').TBCREAT
         AIF   ('&SERVICE' EQ 'TBBOTTOM').TBBOT
         AIF   ('&SERVICE' EQ 'TBQUERY').TBQUERY
         AIF   ('&SERVICE' EQ 'TBSKIP').TBSKIP
         AIF   ('&SERVICE' EQ 'TBSCAN').TBSCAN
         AIF   ('&SERVICE' EQ 'TBSORT').TBSORT
         AIF   ('&SERVICE' EQ 'TBPUT').TBPUT
         AIF   ('&SERVICE' EQ 'TBDISPL').TBDISPL
         AIF   ('&SERVICE' EQ 'TBTOP').TBFUNC
         AIF   ('&SERVICE' EQ 'TBADD').TBFUNC
         AIF   ('&SERVICE' EQ 'TBEND').TBFUNC
         AIF   ('&SERVICE' EQ 'TBDELETE').TBFUNC
         AIF   ('&SERVICE' EQ 'TBGET').TBFUNC
         AIF   ('&SERVICE' EQ 'TBERASE').TBFUNC
         AIF   ('&SERVICE' EQ 'VRESET').VRESET
         AIF   ('&SERVICE' EQ 'VGET').VGET
         AIF   ('&SERVICE' EQ 'VPUT').VPUT
         AIF   ('&SERVICE' EQ 'VERASE').VERASE
         AIF   ('&SERVICE' EQ 'VDEFINE').VDEFINE
         AIF   ('&SERVICE' EQ 'CONTROL').CONTROL
         AIF   ('&SERVICE' EQ 'DISPLAY').DISPLAY
         AIF   ('&SERVICE' EQ 'SETMSG').SETMSG
         AIF   ('&SERVICE' EQ 'LMOPEN').LMFUNC
         AIF   ('&SERVICE' EQ 'LMFREE').LMFUNC
         AIF   ('&SERVICE' EQ 'LMCLOSE').LMFUNC
         AIF   ('&SERVICE' EQ 'LMINIT').LMINIT
         AIF   ('&SERVICE' EQ 'LMGET').LMGET
         AIF   ('&SERVICE' EQ 'LMMFIND').LMMFIND
         AIF   ('&SERVICE' EQ 'BROWSE').BROWSE
         AIF   ('&SERVICE' EQ 'EDIT').EDIT
         MEXIT
*********************************************************************
.SELECT  ANOP
         LCLC  &VP
         AIF   ('&CMD' NE '').RUNCMD
```

```
         AIF   ('&PANEL' NE '').RUNPANL
         AIF   ('&PGM' EQ '').BADKW
.RUNPGM  ANOP
&SEL     SETC  'PGM(&PGM)'
         AIF   ('&PARM' EQ '').COMM1
         AIF   ('&PARM    '(1,3) EQ '(*,').VARPARM
         AIF   ('&PARM '(1,1) NE '''').STDPARM
&P       SETC  '&PARM'(2,K'&PARM-2)
&SEL     SETC  '&SEL PARM(&P)'
         AGO   .COMM1
.STDPARM ANOP
&SEL     SETC  '&SEL PARM(&PARM)'
         AGO   .COMM1
.VARPARM ANOP
&VP      SETC  '&PARM'(4,K'&PARM-4)
         AGO   .COMM1
.RUNCMD  ANOP
&SEL     SETC  'CMD(%&CMD)'
         AIF   ('&PARM' EQ '').NOCPARM
         AIF   ('&PARM   '(1,3) EQ '(*,').CPVAR
&L       SETA  K'&SEL-1
&SEL     SETC  '&SEL'(1,&L)
&SEL     SETC  '&SEL &PARM)'
         AGO   .NOCPARM
.CPVAR   ANOP
&VP      SETC  '&PARM'(4,K'&PARM-4)
         AIF   ('&VP' EQ '').CMDNEXT
.CMDNEXT ANOP
&L       SETA  L'&VP
&SEL     SETC  '&SEL'(1,K'&SEL-1)' '
.CMDLOOP ANOP
&SEL     SETC  '&SEL '
&L       SETA  &L-1
         AIF   (&L GT Ø).CMDLOOP
&SEL     SETC  '&SEL)'
         MNOTE *,'SEL="&SEL"'
.NOCPARM ANOP
         AIF   ('&LANG' EQ '').COMM1
         AIF   ('&LANG' NE 'APL').BADLANG
&SEL     SETC  '&SEL LANG(APL)'
.COMM1   ANOP
         AIF   ('&MODE' EQ 'LINE').MODEOK
         AIF   ('&MODE' EQ 'FSCR').MODEOK
         AIF   ('&MODE' NE '').BADMODE
         AGO   .COMMON
.MODEOK  ANOP
&SEL     SETC  '&SEL MODE(&MODE)'
         AGO   .COMMON
.RUNPANL ANOP
&SEL     SETC  'PANEL(&PANEL)'
         AIF   ('&OPT' EQ '').COMMON
```

```
&SEL     SETC  '&SEL OPT(&OPT)'
.COMMON  ANOP
&A       SETA  K'&SEL
&F       SETC  '&A'
         AIF   ('&VP' EQ '').NOVP1
&W       SETC  '&WRK'
         AIF   ('&W' EQ '').NOWRK
&B       SETA  L'&W
&G       SETC  '&B'
         MVC   &W.(&G.),=CL&G.'&SEL'
         AGO   .MOVEVP
.NOWRK   ANOP
&B       SETA  &A+L'&VP+6
&G       SETC  '&B'
&W       SETC  'SPFSEL_&SYSNDX'
         B     PAST_&W
&W       DC    CL&G.'&SEL'
PAST_&W  DS    ØH
.MOVEVP  ANOP
         AIF   ('&CMD' NE '').CMDPRM
         MVC   &W+&F.(6),=C' PARM('
         MVC   &W+&F+6(L'&VP),&VP
         MVI   &W+&F+6+L'&VP,C')'
&A       SETA  &A+L'&VP+7
         AGO   .SELISP
.CMDPRM  ANOP
&K       SETA  K'&CMD+6
         MVC   &W+&K.(L'&VP),&VP
.SELISP  ANOP
&F       SETC  '&A'
         LINK  EP=ISPLINK,PARAM=(=CL8'SELECT',=A(&F),&W),VL=1,         +
               MF=(E,&PLST)
         MEXIT
.NOVP1   ANOP
         LINK  EP=ISPLINK,PARAM=(=CL8'SELECT',=A(&F),=C'&SEL'),        +
               VL=1,MF=(E,&PLST)
         MEXIT
.BADKW   ANOP
         MNOTE 8,'ERROR: AT LEAST 1 KEYWORD REQUIRED'
         MEXIT
.BADLANG MNOTE 8,'LANG MUST BE ''APL'' OR BLANK'
         MEXIT
.BADMODE MNOTE 8,'MODE MUST BE ''LINE'', ''FSCR'' OR BLANK'
         MEXIT
*********************************************************************
.TBCREAT ANOP
         AIF   ('&TBNAME' EQ '').BADNAME
         SETFLD &TBNAME
&TBN     SETC  '&ISPFLD'
&REP     SETC  ' '
         AIF   ('&REPLACE '(1,1) NE 'Y').NOREP
```

```
&REP     SETC  'REPLACE'
.NOREP   ANOP
         SETTBWRT &WRITE
         FLDLIST &KEYS
&K       SETC  '&FLDLST'
         FLDLIST &FIELDS
&F       SETC  '&FLDLST'
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                          +
               PARAM=(=CL8'TBCREATE',&TBN,&K,&F,&TBWRT,=CL8'&REP')
         MEXIT
*********************************************************************
.TBBOT   ANOP
         AIF   ('&TBNAME' EQ '').BADNAME
         SETFLD &TBNAME
&TBN     SETC  '&ISPFLD'
         AIF   ('&NOREAD '(1,1) EQ 'Y').NOREAD
&NR      SETC  '=CL8'' '''
         AGO   .CRP
.NOREAD  ANOP
&NR      SETC  '=CL8''NOREAD'''
.CRP     ANOP
         SETFLD &CRP
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                          +
               PARAM=(=CL8'TBBOTTOM',&TBN,=CL8' ',=CL8' ',&NR,&ISPFLD)
         MEXIT
*********************************************************************
.TBQUERY ANOP
         AIF   ('&TBNAME' EQ '').BADNAME
         SETFLD &TBNAME
&TBN     SETC  '&ISPFLD'
         FLDLIST &KEYS
&K       SETC  '&FLDLST'
         FLDLIST &NAMES
&N       SETC  '&FLDLST'
         SETFLD &ROWNUM
&RN      SETC  '&ISPFLD'
         SETFLD &KEYNUM
&KN      SETC  '&ISPFLD'
         SETFLD &NAMENUM
&NN      SETC  '&ISPFLD'
         SETFLD &POS
&P       SETC  '&ISPFLD'
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                          +
               PARAM=(=CL8'TBQUERY',&TBN,&K,&N,&RN,&KN,&NN,&P)
         MEXIT
*********************************************************************
.TBSKIP  ANOP
         AIF   ('&TBNAME' EQ '').BADNAME
         SETFLD &TBNAME
&TBN     SETC  '&ISPFLD'
&CNT     SETC  '=F''1'''
```

```
         AIF   ('&COUNT' EQ '').ROWID
         AIF   ('&COUNT   '(1,3) EQ '(*,').CNTREF
         AIF   ('&COUNT   '(1,1) EQ '(').CNTREG
&CNT     SETC  '=A(&COUNT)'
         AGO   .ROWID
.CNTREF  ANOP
&CNT     SETC  '&COUNT(2)'
         AGO   .ROWID
.CNTREG  ANOP
&CNT     SETC  '&COUNT'
.ROWID   ANOP
         SETFLD &ROWID
&RID     SETC  '&ISPFLD'
         SETFLD &CRP
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                          +
               PARAM=(=CL8'TBSKIP',&TBN,&CNT,=CL8' ',&RID,=CL8' ',    +
               =CL8' ',&ISPFLD)
         MEXIT
*********************************************************************
.TBSCAN  ANOP
         AIF   ('&TBNAME' EQ '').BADNAME
         SETFLD &TBNAME
&TBN     SETC  '&ISPFLD'
         FLDLIST &FIELDS
&F       SETC  '&FLDLST'
         SETFLD &DIR
&D       SETC  '&ISPFLD'
&RD      SETC  ' '
         AIF   ('&READ' NE 'NO').RDOK
&RD      SETC  'NOREAD'
.RDOK    ANOP
         SETFLD &ROWID
&ROWNBR  SETC  '&ISPFLD'
         SETFLD &SAVENAME
&SAVE    SETC  '&ISPFLD'
         FLDLIST   &COND
         SETFLD &CRP
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                          +
               PARAM=(=CL8'TBSCAN',&TBN,&F,&SAVE,&ROWNBR,&D,          +
               =CL8'&RD',&ISPFLD,&FLDLST)
         MEXIT
*********************************************************************
.TBSORT  ANOP
         AIF   ('&TBNAME' EQ '').BADNAME
         SETFLD &TBNAME
&TBN     SETC  '&ISPFLD'
         AIF   ('&FIELDS' EQ '').BADFLDS
         FLDLIST &FIELDS
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                          +
               PARAM=(=CL8'TBSORT',&TBN,&FLDLST)
         MEXIT
```

```
.BADFLDS MNOTE 8,'AT LEAST 1 SORT FIELD REQUIRED'
         MEXIT
*********************************************************************
.TBPUT   ANOP
         AIF   ('&TBNAME' EQ '').BADNAME
         SETFLD &TBNAME
&TBN     SETC  '&ISPFLD'
         AIF   ('&ORDER '(1,1) EQ 'Y').ORDER
&O       SETC  '=CL8'' '''
         AGO   .PUTCALL
.ORDER   ANOP
&O       SETC  '=CL8''ORDER'''
.PUTCALL ANOP
         FLDLIST &FIELDS
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                          +
               PARAM=(=CL8'TBPUT',&TBN,&FLDLST,&O)
         MEXIT
*********************************************************************
.TBDISPL ANOP
         AIF   ('&TBNAME' EQ '').BADNAME
         SETFLD &TBNAME
&TBN     SETC  '&ISPFLD'
         SETFLD &PANEL
&P       SETC  '&ISPFLD'
         SETFLD &MSG
&M       SETC  '&ISPFLD'
         SETFLD &CURSOR
&C       SETC  '&ISPFLD'
         SETFLD &CRP
&RN      SETC  '&ISPFLD'
         SETFLD &ROWID
&RID     SETC  '&ISPFLD'
         SETFLD &MSGLOC
&ML      SETC  '&ISPFLD'
&AUTO    SETC  '=CL8''NO'''
         AIF   ('&AUTOSEL'(1,1) NE 'Y').NOSEL
&AUTO    SETC  '=CL8''YES'''
.NOSEL   ANOP
&CP      SETC  '=A(Ø)'
         AIF   ('&POS' EQ '').CSRROW
         AIF   ('&POS'(1,3) EQ '(*').CPREF
         AIF   ('&POS'(1,1) EQ '(').CPREG
&CP      SETC  '=A(&POS)'
         AGO   .CSRROW
.CPREF   ANOP
&CP      SETC  '&POS(2)'
         AGO   .CSRROW
.CPREG   ANOP
&CP      SETC  '&POS'
.CSRROW  ANOP
&CR      SETC  '=CL8'' '''
```

```
         AIF   ('&CSRROW' EQ '').CALLTBD
         AIF   ('&CSRROW   '(1,3) EQ '(*,').CRREF
         AIF   ('&CSRROW   '(1,1) EQ '(').CRREG
&CR      SETC  '=A(&CSRROW)'
         AGO   .CALLTBD
.CRREF   ANOP
&CR      SETC  '&CSRROW(2)'
         AGO   .CALLTBD
.CRREG   ANOP
&CR      SETC  '&CSRROW'
.CALLTBD ANOP
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                           +
               PARAM=(=CL8'TBDISPL',&TBN,&P,&M,&C,&CR,&CP,&AUTO,&RN,   +
               &RID,&ML)
         MEXIT
*********************************************************************
.TBFUNC  ANOP
         AIF   ('&TBNAME' EQ '').BADNAME
         SETFLD &TBNAME
&TBN     SETC  '&ISPFLD'
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                           +
               PARAM=(=CL8'&SERVICE',&TBN)
         MEXIT
*********************************************************************
.VRESET  ANOP
         LINK  EP=ISPLINK,PARAM=(=CL8'VRESET'),VL=1,MF=(E,&PLST)
         MEXIT
*********************************************************************
.VGET    ANOP
         AIF   ('&VARS' EQ '').BADVAR
         FLDLIST &VARS
         SETFLD &POOL
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                           +
               PARAM=(=CL8'VGET',&FLDLST,&ISPFLD)
         MEXIT
*********************************************************************
.VPUT    ANOP
         AIF   ('&VARS' EQ '').BADVARS
         FLDLIST &VARS
         SETFLD &POOL
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                           +
               PARAM=(=CL8'VPUT',&FLDLST,&ISPFLD)
         MEXIT
.BADVARS MNOTE 8,'ERROR: VARIABLE NAME REQUIRED'
         MEXIT
*********************************************************************
.VERASE  ANOP
         AIF   ('&VARS' EQ '').BADVARS
         FLDLIST &VARS
         SETFLD &POOL
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                           +
```

```
               PARAM=(=CL8'VERASE',&FLDLST,&ISPFLD)
         MEXIT
*********************************************************************
.VDEFINE ANOP
         AIF   ('&FLD' EQ '').BADFLD
         AIF   ('&LEN' EQ '').SETLN
&L       SETC  '&LEN'
         AGO   .TESTYPE
.SETLN   ANOP
&L       SETC  'L''&FLD'
.TESTYPE ANOP
         AIF   ('&FMT' EQ 'CHAR').TYPOK
         AIF   ('&FMT' EQ 'FIXED').TYPOK
         AIF   ('&FMT' EQ 'HEX').TYPOK
         AIF   ('&FMT' NE 'BIT').BADTYP
.TYPOK   ANOP
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                           +
               PARAM=(=CL8'VDEFINE',=CL8'&FLD',&FLD,=CL8'&FMT',=A(&L))
         MEXIT
.BADFLD  MNOTE 8,'ERROR: ''FLD'' PARAMETER REQUIRED'
         MEXIT
.BADTYP  MNOTE 8,'ERROR: ''FMT'' PARAMETER MUST BE ''CHAR'', ''FIXED'',+
                ''HEX'' OR ''BIT'''
         MEXIT
*********************************************************************
.CONTROL ANOP
         AIF   ('&PARM(4)' EQ '').TCTL3
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                           +
               PARAM=(=CL8'CONTROL',=CL8'&PARM(1)',=CL8'&PARM(2)',     +
               =CL8'PARM(3)',=CL8'&PARM(4)')
         MEXIT
.TCTL3   ANOP
         AIF   ('&PARM(3)' EQ '').TCTL2
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                           +
               PARAM=(=CL8'CONTROL',=CL8'&PARM(1)',=CL8'&PARM(2)',     +
               =CL8'PARM(3)')
         MEXIT
.TCTL2   ANOP
         AIF   ('&PARM(2)' EQ '').TCTL1
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                           +
               PARAM=(=CL8'CONTROL',=CL8'&PARM(1)',=CL8'&PARM(2)')
         MEXIT
.TCTL1   ANOP
         AIF   ('&PARM(1)' EQ '').BADPARM
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                           +
               PARAM=(=CL8'CONTROL',=CL8'&PARM(1)')
         MEXIT
.BADPARM MNOTE 8,'ERROR: AT LEAST ONE PARM REQUIRED'
         MEXIT
*********************************************************************
.DISPLAY ANOP
```

```
         AIF   ('&PANEL' EQ '').BADPNL
         SETFLD &PANEL
&P       SETC  '&ISPFLD'
         SETFLD &MSG
&M       SETC  '&ISPFLD'
         SETFLD &CURSOR
&C       SETC  '&ISPFLD'
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                           +
               PARAM=(=CL8'DISPLAY',&P,&M,&C)
         MEXIT
.BADPANL MNOTE 8,'ERROR: PANEL NAME REQUIRED'
         MEXIT
*********************************************************************
.SETMSG  ANOP
         AIF   ('&MSG' EQ '').BADMSG
         SETFLD &MSG
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                           +
               PARAM=(=CL8'SETMSG',&ISPFLD)
         MEXIT
.BADMSG  MNOTE 8,'ERROR: MESSAGE NUMBER REQUIRED'
         MEXIT
*********************************************************************
.LMFUNC  ANOP
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                           +
               PARAM=(=CL8'&SERVICE',&DATAID)
         MEXIT
*********************************************************************
.LMINIT  ANOP
         AIF   ('&DATAID' EQ '').BADID
         SETFLD &DATAID
&DID     SETC  '&ISPFLD'
         SETFLD &DISP
&DSP     SETC  '&ISPFLD'
         SETFLD &VOL
&V       SETC  '&ISPFLD'
         SETFLD &PASSWORD
&PW      SETC  '&ISPFLD'
         AIF   ('&DDN' NE '').INITDDN
         AIF   ('&DSN' NE '').INITDSN
         AIF   ('&PROJECT' EQ '').BADDSN
         AIF   ('&GROUP1' EQ '').BADGRP
         AIF   ('&TYPE' EQ '').BADTYPE
&DS      SETC  '=CL8'' '''
&DD      SETC  '=CL8'' '''
         SETFLD &TYPE
&T       SETC  '&ISPFLD'
         SETFLD &PROJECT
&PR      SETC  '&ISPFLD'
         SETFLD &GROUP1
&G1      SETC  '&ISPFLD'
         SETFLD &GROUP2
```

```
&G2      SETC  '&ISPFLD'
         SETFLD &GROUP3
&G3      SETC  '&ISPFLD'
         SETFLD &GROUP4
&G4      SETC  '&ISPFLD'
         AGO   .CALL
.INITDSN ANOP
&DS      SETC  '&DSN'
         AIF   ('&DS   '(1,3) EQ '(*,').DSNREF
         AIF   ('&DS '(1,1) EQ '(').DSNDONE
&DS      SETC  '=CL44''&DSN'''
         AGO   .DSNDONE
.DSNREF  ANOP
&DS      SETC  '&DSN(2)'
.DSNDONE ANOP
&PR      SETC  '=C'' '''
&G1      SETC  '=C'' '''
&G2      SETC  '=C'' '''
&G3      SETC  '=C'' '''
&G4      SETC  '=C'' '''
&T       SETC  '=C'' '''
&DD      SETC  '=C'' '''
         AGO   .CALL
.INITDDN ANOP
         SETFLD &DDN
&DD      SETC  '&ISPFLD'
&PR      SETC  '=C'' '''
&G1      SETC  '=C'' '''
&G2      SETC  '=C'' '''
&G3      SETC  '=C'' '''
&G4      SETC  '=C'' '''
&T       SETC  '=C'' '''
&DS      SETC  '=C'' '''
.CALL    ANOP
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                          +
               PARAM=(=CL8'LMINIT',&DID,&PR,&G1,&G2,&G3,&G4,&T,&DS,   +
               &DD,&V,&PW,&DSP)
         MEXIT
.BADDSN  MNOTE 8,'DSN, DDN OR PROJECT-GROUP-TYPE REQUIRED'
         MEXIT
.BADGRP  MNOTE 8,'AT LEAST GROUP1 REQUIRED WITH PROJECT (UP TO 4)'
         MEXIT
.BADTYPE MNOTE 8,'TYPE REQUIRED WITH PROJECT-GROUP'
         MEXIT
.BADID   MNOTE 8,'ERROR: DATAID VARIABLE REQUIRED'
         MEXIT
*********************************************************************
.LMGET   ANOP
         AIF   ('&VAR' EQ '').BADVAR
         AIF   ('&DATAID' EQ '').BADID
```

```
         AIF   ('&LENVAR' EQ '').BADLEN
&M       SETC  '&MAXLEN'
         AIF   ('&MAXLEN' NE '').MAXOK
&M       SETC  'L''&VAR'
         MNOTE Ø,'MAXLEN DEFAULTS TO LENGTH OF &VAR'
.MAXOK   ANOP
         SETFLD &LMMODE
&MDE     SETC  '&ISPFLD'
         SETFLD &VAR
&V       SETC  '&ISPFLD'
         SETFLD &LENVAR
&LV      SETC  '&ISPFLD'
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                           +
               PARAM=(=CL8'LMGET',&DATAID,&MDE,&V,&LV,=A(&M))
         MEXIT
.BADLEN  MNOTE 8,'ERROR: ''LENVAR'' VARIABLE REQUIRED'
         MEXIT
.BADVAR  MNOTE 8,'PLEASE SPECIFY VARIABLE TO RECEIVE DATA'
         MEXIT
*********************************************************************
.LMMFIND ANOP
         AIF   ('&DATAID' EQ '').BADID
         AIF   ('&MBR' EQ '').BADMBR
         SETFLD &MBR
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                           +
               PARAM=(=CL8'LMMFIND',&DATAID,&ISPFLD)
         MEXIT
.BADMBR  MNOTE 8,'ERROR: MEMBER NAME REQUIRED'
         MEXIT
.BROWSE  ANOP
         SETFLD &VOL
&V       SETC  '&ISPFLD'
         SETFLD &PASSWORD
&PW      SETC  '&ISPFLD'
         AIF   ('&DSN' EQ '').BRWID
         SETFLD &DSN
&D       SETC  '&ISPFLD'
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                           +
               PARAM=(=CL8'BROWSE',&D,&V,&PW)
         MEXIT
.BRWID   ANOP
         AIF   ('&DATAID' EQ '').BADID
&D       SETC  '=C'' '''
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                           +
               PARAM=(=CL8'BROWSE',&D,&V,&PW,&D,&DATAID)
         MEXIT
.EDIT    ANOP
         SETFLD &VOL
&V       SETC  '&ISPFLD'
         SETFLD &PASSWORD
```

```
&PW      SETC  '&ISPFLD'
         AIF   ('&DSN' EQ '').EDITID
         SETFLD &DSN
&D       SETC  '&ISPFLD'
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                           +
               PARAM=(=CL8'EDIT',&D,&V,&PW)
         MEXIT
.EDITID  ANOP
         AIF   ('&DATAID' EQ '').BADID
&D       SETC  '=C'' '''
         LINK  EP=ISPLINK,VL=1,MF=(E,&PLST),                           +
               PARAM=(=CL8'EDIT',&D,&V,&PW,&D,&DATAID)
         MEXIT
.BADNAME ANOP
         MNOTE 8,'TBNAME PARAMETER REQUIRED'
         MEND
```

PPFC14M

```
         MACRO
         GETCC &COND
         GBLA  &PF_CCVAL
         LCLC  &LWK1
         AIF   ('&COND'(1,1) LT 'Ø' OR '&COND'(1,1) GT '9').NOTNUM
&PF_CCVAL      SETA  &COND
         MEXIT
.NOTNUM  AIF   (K'&COND NE 1).TWOCHAR
&LWK1    SETC  '&COND'
         AGO   .CALCC
.TWOCHAR AIF   (K'&COND NE 2).INVCOND
         AIF   ('&COND'(1,1) NE 'N').OTHERMN
&LWK1    SETC  '&COND'(2,1)
         AGO   .CALCC
.OTHERMN AIF   ('&COND' EQ 'EQ').BC8
         AIF   ('&COND' EQ 'LT').BC4
         AIF   ('&COND' NE 'LE').TRYGT
&PF_CCVAL      SETA  13
         MEXIT
.TRYGT   AIF   ('&COND' EQ 'GT').BC2
         AIF   ('&COND' NE 'GE').INVCOND
&PF_CCVAL SETA 11
         MEXIT
.CALCC   AIF   ('&LWK1' NE 'O').TRYH
&PF_CCVAL      SETA  1
         AGO   .TSTN
.TRYH    AIF   ('&LWK1' EQ 'P' OR '&LWK1' EQ 'H').BC2
         AIF   ('&LWK1' EQ 'L' OR '&LWK1' EQ 'M').BC4
         AIF   ('&LWK1' EQ 'E' OR '&LWK1' EQ 'Z').BC8
         AGO   .INVCOND
```

```
.BC8     ANOP
&PF_CCVAL      SETA  8
         AGO   .TSTN
.BC4     ANOP
&PF_CCVAL      SETA  4
         AGO   .TSTN
.BC2     ANOP
&PF_CCVAL      SETA  2
.TSTN    AIF   ('&COND'(1,1) NE 'N').DONE
&PF_CCVAL      SETA  15-&PF_CCVAL
.DONE    MEXIT
.INVCOND ANOP
&PF_CCVAL      SETA  15
         MNOTE 8,'INVALID CONDITION MNEMONIC. NOP GENERATED'   @BA25155
         MEND
*********************************************************************
         MACRO
         POPINS &P
         COPY  PPFGBLCØ
         LCLA   &W
&W       SETA   &P
         AGO    .TEST
.UNSTACK ANOP
         AIF   ('&PF_IIND3(&W)' EQ '').ONEOP
         AIF   ('&PF_IIND4(&W)' NE '').THREEOP
&PF_IIND5(&W)  &PF_IIND1(&W) &PF_IIND2(&W),&PF_IIND3(&W)
         AGO   .INCTR
.THREEOP ANOP
&PF_IIND5(&W)  &PF_IIND1(&W) &PF_IIND2(&W),&PF_IIND3(&W),&PF_IIND4(&W)
         AGO   .INCTR
.ONEOP   ANOP
&PF_IIND5(&W)  &PF_IIND1(&W) &PF_IIND2(&W)
.INCTR   ANOP
&W       SETA  &W+1
.TEST    AIF    (&W LE &PF_II).UNSTACK
&PF_II   SETA  &P-1
         AIF   ('&PF_NEST(&PF_NI)'(3,1) NE ' ' OR '&PF_NEST(&PF_NI)'(4,+
               1) EQ ' ').NEQ
&PF_IIND5(&PF_II) &PF_IIND1(&PF_II) &PF_IIND2(&PF_II)
.NEQ     AIF   (&PF_II GT Ø OR (&PF_II EQ Ø AND '&PF_NEST(&PF_NI)'(5,4)+
                EQ 'IF')).END
         MNOTE 8,'NEGATIVE INSTRUCTION STACK PTR. EXPANSION INVALID.'
.END     MEND
*********************************************************************
```

Editor's note: this article will be concluded in next month's issue.

Pieter Wiid
Systems Engineer
OutSource (South Africa)

Computer Associates has announced Release 7.5 of its Unicenter CA-SYSVIEW real-time performance management tool for z/OS and OS/390, which enables system programmers to review short-term historical event data to determine the cause of degraded system performance.

The idea is to strengthen the ability of organizations to analyse IT processes to support the roll-out of Internet-based operational improvements.

It has a new Event Capture Option, along with extended performance monitoring capabilities for IMS and a better user interface that provides a consolidated view of system-wide activity.

In addition, the enhanced product provides a single point of performance monitoring and control of jobs processing across the entire sysplex.

For further information contact:
Computer Associates International, One Computer Associates Plaza, Islandia, NY 11749, USA
Tel: (631) 342 6000.
URL: http://ca.com.

* * *

Mainstar has upgraded its back-up and recovery software with new enhancements, which include a dynamic back-up/ reorganization facility, DFSMShsm incremental, and a FullRename feature.

There's disaster recovery support with RACF functions to support primary and line commands and a dynamic back-up/ reorganization facility allows the back-up and recovery manager inventory dataset to be backed up, or reorganized, without stopping the product or product programs. This only applies to documented VSAM datasets used by Mainstar products.

The dynamic back-up function ensures a clean back-up of the Backup and Recovery Manager inventory dataset while Backup and Recovery Manager programs are executing. It temporarily pauses access to the Backup and Recovery Manager inventory dataset by Backup and Recovery Manager programs, until the back-up is finished. And the Dynamic Reorganization function reorganizes the Backup and Recovery Manager inventory dataset.

For further information contact:
Mainstar Software, PO Box 4132, Bellevue, WA 98009-4132, USA.
Tel: (425) 455 3589.
URL: http://www.mainstar.com.

* * *

IBM has announced Tivoli NetView for z/OS Version 5 Release 1, with enhancements in TCP/IP management, NetView Web Application, and broader Linux support.

As a stand-alone management application, it manages TCP/IP resources and SNA resources and provides facilities to automate any network or system event.

V5R1 can forward collected z/OS management information to other management applications. It provides management services to Tivoli Business Systems Manager for its handling of OS/390 and z/OS subsystems such as CICS, DB2, and IMS.

For further information contact your local IBM representative.
URL: http://www.tivoli.com.