Solitaire

Contributed by Anon.
Saturday, 20 January 2007

The popular Solitaire game

```
PROGRAM ZMN00003 NO STANDARD PAGE HEADING.

  DATA: NUMBER LIKE DATATYPE-INTEGER2.

  DATA: DUMMY  LIKE DATATYPE-INTEGER2.

  DATA: BEGIN OF CARDS OCCURS 52 ,

      ENTRY TYPE I,

      ROW TYPE I,



COL

 TYPE I,



VIS
(1),

      END OF CARDS.

  DATA: CARD_NUM TYPE I, SUIT_NUM TYPE I, SUIT(1), CARD(2), ROWS TYPE I.

  DATA: BLANK(3) VALUE '***'.

  DATA: NEXT-CARD(4) VALUE 'Next'.

  DATA: RESTART(8) VALUE 'New game'.

  DATA: REVEAL(6) VALUE 'Reveal'.

  DATA: TEST_NUM TYPE I, TO_NUM TYPE I, MOVING_NUM TYPE I,

       PILE_NUM TYPE I.

  DATA: CURR_C TYPE I, CURR_H TYPE I, CURR_S TYPE I, CURR_D TYPE I.

  DATA: MOVING_SUIT(1), TO_SUIT(1), MOVING_CARD(2), TO_CARD(2),
```

```
  PILE_SUIT(1), PILE_CARD(2),MOVING_TYPE(1)    .

DATA: MOVING_ROW TYPE I, MOVING_COL TYPE I,

      TO_ROW TYPE I, TO_COL TYPE I, TEST_ROW TYPE I.

DATA: CURRENT_CARD TYPE I,

       MIN_CARD TYPE I , MAX_CARD TYPE I.

DATA: CURRENTFIELD(50).

DATA: DISC_C(3),DISC_H(3),DISC_S(3),DISC_D(3).

DATA: ROWS_OUT TYPE I, COLS_OUT TYPE I.




PERFORM SHUFFLE.

PERFORM SHOW_SCREEN.



AT LINE-SELECTION.

  GET CURSOR FIELD CURRENTFIELD.

  PERFORM PROCESS-INPUT.



*---------------------------------------------------------------------*

*       FORM SHUFFLE                                                  *

*---------------------------------------------------------------------*

*       ........                                                      *
```

```
*-----------------------------------------------------------------------*

FORM SHUFFLE.

  MIN_CARD = 3.MAX_CARD = 24.

  CURR_C = -1.CURR_D = -1.CURR_H = -1.CURR_S = -1.

  DISC_C = 'CCC'.DISC_H = 'HHH'.DISC_S = 'SSS'.DISC_D = 'DDD'.

  NUMBER = SY-UZEIT MOD 1000.

  DO NUMBER TIMES.

    CALL FUNCTION 'RANDOM_I2'

        IMPORTING

            RND_VALUE = DUMMY.

  ENDDO.

  REFRESH CARDS. CLEAR CARDS. SY-TABIX = 0.

  WHILE SY-TABIX < 52.

*do 40 times.

    CALL FUNCTION 'RANDOM_I2'

        EXPORTING

            RND_MIN   = 0

            RND_MAX   = 51

        IMPORTING

            RND_VALUE = NUMBER.

    READ TABLE CARDS WITH KEY NUMBER TRANSPORTING NO FIELDS.

    IF SY-SUBRC <> 0.
```

```
      CARDS-VIS = 'n'.

      DESCRIBE TABLE CARDS LINES ROWS.

      CASE ROWS.

        WHEN 0.CARDS-VIS = 'y'.CARDS-ROW = 0.CARDS-COL = 0.

        WHEN 7.CARDS-VIS = 'y'.CARDS-ROW = 1.CARDS-COL = 1.

        WHEN 13.CARDS-VIS = 'y'.CARDS-ROW = 2.CARDS-COL = 2.

        WHEN 18.CARDS-VIS = 'y'.CARDS-ROW = 3.CARDS-COL = 3.

        WHEN 22.CARDS-VIS = 'y'.CARDS-ROW = 4.CARDS-COL = 4.

        WHEN 25.CARDS-VIS = 'y'.CARDS-ROW = 5.CARDS-COL = 5.

        WHEN 27.CARDS-VIS = 'y'.CARDS-ROW = 6.CARDS-COL = 6.

        WHEN 28.CARDS-ROW = 20.CARDS-COL = 0.

      ENDCASE.

      CARDS-COL = CARDS-COL + 1.

      CARDS-ENTRY = NUMBER.

      APPEND CARDS.

    ENDIF.

*enddo.

  ENDWHILE.

  CURRENT_CARD = MIN_CARD.

ENDFORM.



*---------------------------------------------------------------------*
```

```
*      FORM SHOW_SCREEN                                    *
*-----------------------------------------------------------------------*
*      ........                                            *
*-----------------------------------------------------------------------*
FORM SHOW_SCREEN.
  ROWS_OUT = 1.
  COLS_OUT = 4.
  DO 7 TIMES.
    SKIP TO LINE ROWS_OUT.
    POSITION COLS_OUT.
    WRITE BLANK HOTSPOT.
    COLS_OUT = COLS_OUT + 4.
  ENDDO.

LOOP
 AT CARDS.
    PERFORM MAKE_CARD.
    IF CARDS-ROW < 20.
      ROWS_OUT = CARDS-ROW + 1.
      SKIP TO LINE ROWS_OUT.
      COLS_OUT = CARDS-COL * 4.
      POSITION COLS_OUT.
      IF CARDS-VIS = 'n'.
```

```
    WRITE 'XXX' COLOR OFF INTENSIFIED OFF INVERSE OFF.
  ELSE.
    IF SUIT = 'D' OR SUIT = 'H'.
      WRITE: SUIT NO-GAP COLOR 6 INTENSIFIED INVERSE HOTSPOT,
             CARD COLOR 6 INTENSIFIED INVERSE HOTSPOT.
    ELSE.
   WRITE: SUIT NO-GAP COLOR OFF INTENSIFIED OFF INVERSE OFF HOTSPOT,
          CARD COLOR OFF INTENSIFIED OFF INVERSE OFF HOTSPOT.
    ENDIF.
  ENDIF.
ELSE.
  SKIP TO LINE 20.      COLS_OUT =  CARDS-COL  * 4.
  IF COLS_OUT > 48.
    SKIP TO LINE 21.
    COLS_OUT = COLS_OUT - 48.
  ENDIF.
  POSITION COLS_OUT.
  IF CARDS-COL = CURRENT_CARD.
    PILE_SUIT = SUIT.
    PILE_CARD = CARD.
    PILE_NUM = CARD_NUM.
```

```
      IF SUIT = 'D' OR SUIT = 'H'.
        WRITE: PILE_SUIT NO-GAP COLOR 6 INTENSIFIED INVERSE HOTSPOT,
              PILE_CARD COLOR 6 INTENSIFIED INVERSE HOTSPOT.
      ELSE.
        WRITE: PILE_SUIT NO-GAP COLOR OFF INTENSIFIED OFF HOTSPOT,
              PILE_CARD COLOR OFF INTENSIFIED OFF HOTSPOT.
      ENDIF.
    ELSE.
      IF CARDS-COL < CURRENT_CARD.
        WRITE 'XXX' COLOR OFF INTENSIFIED OFF INVERSE OFF.
      ENDIF.
    ENDIF.
  ENDIF.
ENDLOOP.
SKIP TO LINE 4. POSITION 40.WRITE: DISC_C
     COLOR OFF INTENSIFIED OFF INVERSE OFF HOTSPOT.
SKIP TO LINE 4. POSITION 44. WRITE: DISC_D
     COLOR 6 INTENSIFIED INVERSE HOTSPOT.
SKIP TO LINE 4. POSITION 48. WRITE: DISC_S
     COLOR OFF INTENSIFIED OFF INVERSE OFF HOTSPOT.
SKIP TO LINE 4. POSITION 52. WRITE: DISC_H
     COLOR 6 INTENSIFIED INVERSE HOTSPOT.
```

```
  IF MIN_CARD > 0.
    SKIP TO LINE 23. POSITION 1. WRITE NEXT-CARD HOTSPOT.
  ENDIF.
  SKIP TO LINE 23. POSITION 20. WRITE RESTART HOTSPOT.
*  skip to line 23. position 40. write reveal hotspot.
  SKIP TO LINE 25. POSITION 1. WRITE CURRENTFIELD.
  SY-LSIND = 0.
ENDFORM.


*---------------------------------------------------------------------*
*       FORM PROCESS-INPUT                                            *
*---------------------------------------------------------------------*
*       ........                                                      *
*---------------------------------------------------------------------*
FORM PROCESS-INPUT.
  IF CURRENTFIELD = 'NEXT-CARD'.
    IF CURRENT_CARD = MAX_CARD.
      CURRENT_CARD = MIN_CARD.
    ELSE.
      CURRENT_CARD = CURRENT_CARD + 3.
      IF CURRENT_CARD > MAX_CARD.
        CURRENT_CARD = MAX_CARD.
```

```
      ENDIF.

    ENDIF.

    MOVING_CARD = SPACE.

    CURRENTFIELD = SPACE.

    PERFORM SHOW_SCREEN.

    EXIT.

  ENDIF.

  IF CURRENTFIELD = 'REVEAL'.


LOOP
 AT CARDS.

      CARDS-VIS =  'y'.

      MODIFY CARDS.

    ENDLOOP.

    PERFORM SHOW_SCREEN.

    EXIT.

  ENDIF.

  IF CURRENTFIELD = 'RESTART'.

    PERFORM SHUFFLE.

    CURRENTFIELD = 'Restarting'.

    PERFORM SHOW_SCREEN.

    EXIT.

  ENDIF.
```

```
    IF MOVING_CARD = SPACE.

      IF CURRENTFIELD = 'CARD' OR CURRENTFIELD = 'SUIT'.

        CURRENTFIELD = SPACE.

        ROWS_OUT = SY-CUROW - 1.

        COLS_OUT = SY-CUCOL DIV 4.


LOOP
 AT CARDS.

          IF CARDS-ROW = ROWS_OUT AND CARDS-COL = COLS_OUT.

            CURRENTFIELD = 'Moving '.

            PERFORM MAKE_CARD.

            MOVING_ROW = CARDS-ROW.

            MOVING_COL = CARDS-COL.

            MOVING_CARD = CARD.

            MOVING_SUIT = SUIT.

            MOVING_TYPE = 'm'.

            MOVING_NUM = CARD_NUM.

            CURRENTFIELD+8(1) = SUIT.

            CURRENTFIELD+9(2) = CARD.

          ENDIF.

        ENDLOOP.

      ELSE.

        IF CURRENTFIELD = 'PILE_CARD' OR CURRENTFIELD = 'PILE_SUIT'.
```

```
      MOVING_CARD = PILE_CARD.
      MOVING_SUIT = PILE_SUIT.
      MOVING_TYPE = 'p'.
      MOVING_NUM = PILE_NUM.
      CURRENTFIELD = 'Moving pile card'.
      CURRENTFIELD+17(1) = PILE_SUIT.
      CURRENTFIELD+18(2) = PILE_CARD.
    ELSE.
      CURRENTFIELD = SPACE.
    ENDIF.
  ENDIF.
ELSE.
  IF CURRENTFIELD = 'CARD' OR CURRENTFIELD = 'SUIT' OR
        CURRENTFIELD = 'BLANK'.
    ROWS_OUT = SY-CUROW - 1.
    COLS_OUT = SY-CUCOL DIV 4.
    TEST_ROW = -1.
    IF CURRENTFIELD = 'BLANK'.
      TO_COL = COLS_OUT.
      TO_ROW = ROWS_OUT - 1.
      IF MOVING_CARD = 'K'.
```

```
      PERFORM MOVE_CARD.

    ELSE.

      CURRENTFIELD = 'Can only move K to blank'.

      MOVING_CARD = SPACE.

    ENDIF.

  ELSE.


LOOP
 AT CARDS.

      IF ( CARDS-ROW >= TEST_ROW AND CARDS-ROW < 20 )

           AND CARDS-COL = COLS_OUT.

        TEST_ROW = CARDS-ROW.

        PERFORM MAKE_CARD.

        TO_COL = CARDS-COL.

        TO_ROW = CARDS-ROW.

        TO_CARD = CARD.

        TO_SUIT = SUIT.

        TO_NUM = CARD_NUM.

      ENDIF.

    ENDLOOP.

    IF TO_COL <> MOVING_COL OR MOVING_TYPE = 'p'.

      IF

        ( ( MOVING_SUIT = 'D' OR MOVING_SUIT = 'H' ) AND
```

```
        ( TO_SUIT = 'D' OR TO_SUIT = 'H' ) )
      OR
       ( ( MOVING_SUIT = 'C' OR MOVING_SUIT = 'S' ) AND
        ( TO_SUIT = 'S' OR TO_SUIT = 'S' ) ).
      CURRENTFIELD = 'Can only put black on red or red on black'.
      MOVING_CARD = SPACE.
     ELSE.
      TEST_NUM = TO_NUM - 1.
      IF TEST_NUM = MOVING_NUM.
       PERFORM MOVE_CARD.
      ELSE.
       CURRENTFIELD = 'Can only put on next higher card'.
       MOVING_CARD = SPACE.
      ENDIF.
     ENDIF.
    ELSE.
     CURRENTFIELD = 'Can only move to another column'.
     MOVING_CARD = SPACE.
    ENDIF.
   ENDIF.
  ELSE.
   IF CURRENTFIELD(4) = 'DISC'.
```

```
      TO_SUIT = CURRENTFIELD+5(1).

      IF MOVING_TYPE = 'p'.

      ELSE.

        TEST_ROW = -1.


LOOP
 AT CARDS.

          IF CARDS-COL = MOVING_COL AND

                CARDS-ROW > TEST_ROW AND CARDS-ROW < 20.

            TEST_ROW = CARDS-ROW.

            PERFORM MAKE_CARD.

            MOVING_SUIT = SUIT.

            MOVING_CARD = CARD.

            MOVING_ROW = CARDS-ROW.

            MOVING_COL = CARDS-COL.

            MOVING_NUM = CARD_NUM.

          ENDIF.

        ENDLOOP.

      ENDIF.

      IF MOVING_SUIT = TO_SUIT.

        CASE MOVING_SUIT.

          WHEN 'C'. TEST_NUM = CURR_C + 1.

          WHEN 'H'. TEST_NUM = CURR_H + 1.
```

```
      WHEN 'D'. TEST_NUM = CURR_D + 1.
      WHEN 'S'. TEST_NUM = CURR_S + 1.
    ENDCASE.
    IF TEST_NUM = MOVING_NUM.
      PERFORM DISCARD_CARD.
      CASE MOVING_SUIT.
        WHEN 'C'. CURR_C = CURR_C + 1.
        WHEN 'D'. CURR_D = CURR_D + 1.
        WHEN 'S'. CURR_S = CURR_S + 1.
        WHEN 'H'. CURR_H = CURR_H + 1.
      ENDCASE.
    ELSE.
      CURRENTFIELD = 'Can only discard on next lower card'.
      MOVING_CARD = SPACE.
    ENDIF.
  ELSE.
    MOVING_CARD = SPACE.
    CURRENTFIELD = 'Can only discard on same suit'.
  ENDIF.
ELSE.
  MOVING_CARD = SPACE.
  CURRENTFIELD = SPACE.
```

```
      ENDIF.
    ENDIF.
  ENDIF.
  PERFORM SHOW_SCREEN.
ENDFORM.
*---------------------------------------------------------------------*
*       FORM MAKE_CARD                                                *
*---------------------------------------------------------------------*
*       ........                                                      *
*---------------------------------------------------------------------*
FORM MAKE_CARD.
  SUIT_NUM = CARDS-ENTRY DIV 13 .
  CASE SUIT_NUM.
    WHEN 0.SUIT = 'D'.
    WHEN 1.SUIT = 'H'.
    WHEN 2.SUIT = 'C'.
    WHEN 3.SUIT = 'S'.
  ENDCASE.
  CARD_NUM = CARDS-ENTRY MOD 13.
  CASE CARD_NUM.
    WHEN 0.CARD = 'A'.
```

```
    WHEN 1.CARD = '2'.

    WHEN 2.CARD = '3'.

    WHEN 3.CARD = '4'.

    WHEN 4.CARD = '5'.

    WHEN 5.CARD = '6'.

    WHEN 6.CARD = '7'.

    WHEN 7.CARD = '8'.

    WHEN 8.CARD = '9'.

    WHEN 9.CARD = '10'.

    WHEN 10.CARD = 'J'.

    WHEN 11.CARD = 'Q'.

    WHEN 12.CARD = 'K'.

  ENDCASE.

ENDFORM.

*---------------------------------------------------------------------*

*       FORM MOVE_CARD                                                *

*---------------------------------------------------------------------*

*       ........                                                      *

*---------------------------------------------------------------------*

FORM MOVE_CARD.

  IF CURRENTFIELD <> 'BLANK'.

    CURRENTFIELD = 'Moving xxx to'.
```

```
      MOVE MOVING_SUIT TO CURRENTFIELD+7(1).

      MOVE MOVING_CARD TO CURRENTFIELD+8(2).

      MOVE TO_SUIT TO CURRENTFIELD+14(1).

      MOVE TO_CARD TO CURRENTFIELD+15(2).

    ELSE.

      CURRENTFIELD = 'Starting emtpy column'.

    ENDIF.

    IF MOVING_TYPE = 'p'.

      MOVE 'from pile' TO CURRENTFIELD+18.


LOOP
 AT CARDS.

        IF CARDS-ROW = 20 AND CARDS-COL = CURRENT_CARD.

          CARDS-COL = TO_COL.

          CARDS-ROW = TO_ROW + 1.

          CARDS-VIS = 'y'.

        ENDIF.

        IF CARDS-ROW = 20 AND CARDS-COL > CURRENT_CARD.

          CARDS-COL = CARDS-COL - 1.

        ENDIF.

        MODIFY CARDS.

      ENDLOOP.

      CURRENT_CARD = CURRENT_CARD - 1.
```

```
      MAX_CARD = MAX_CARD - 1.

      IF MAX_CARD < MIN_CARD.

        MIN_CARD = MIN_CARD - 1.

        MAX_CARD = MIN_CARD.

      ENDIF.

      IF CURRENT_CARD < 1.

        CURRENT_CARD = MIN_CARD.

      ENDIF.

    ELSE.


LOOP
 AT CARDS.

        IF CARDS-COL = MOVING_COL AND

         ( CARDS-ROW >= MOVING_ROW AND CARDS-ROW < 20 ).

          CARDS-COL = TO_COL.

          CARDS-ROW = TO_ROW + 1 + CARDS-ROW - MOVING_ROW.

          MODIFY CARDS.

        ENDIF.

      ENDLOOP.

      TEST_ROW = MOVING_ROW - 1.


LOOP
 AT CARDS.

        IF CARDS-COL = MOVING_COL AND CARDS-ROW = TEST_ROW.
```

```
        CARDS-VIS = 'y'.
        MODIFY CARDS.
      ENDIF.
    ENDLOOP.
  ENDIF.
  MOVING_CARD = SPACE.
ENDFORM.
*---------------------------------------------------------------------*
*       FORM DISCARD_CARD                                             *
*---------------------------------------------------------------------*
*       ........                                                      *
*---------------------------------------------------------------------*
FORM DISCARD_CARD.
  MOVE 'Discarding' TO CURRENTFIELD.
  MOVE MOVING_SUIT TO CURRENTFIELD+13(1).
  MOVE MOVING_CARD TO CURRENTFIELD+14(2).
  CASE TO_SUIT.
    WHEN 'C'.
      DISC_C+1(2) = MOVING_CARD.
    WHEN 'D'.
      DISC_D+1(2) = MOVING_CARD.
    WHEN 'H'.
```

```
      DISC_H+1(2) = MOVING_CARD.

    WHEN 'S'.

      DISC_S+1(2) = MOVING_CARD.

  ENDCASE.

  IF MOVING_TYPE = 'p'.

    MOVE 'from pile' TO CURRENTFIELD+18.


LOOP
 AT CARDS.

      IF CARDS-ROW = 20 AND CARDS-COL = CURRENT_CARD.

        CARDS-COL = 0.

      ENDIF.

      IF CARDS-ROW = 20 AND CARDS-COL > CURRENT_CARD.

        CARDS-COL = CARDS-COL - 1.

      ENDIF.

      MODIFY CARDS.

    ENDLOOP.

    CURRENT_CARD = CURRENT_CARD - 1.

    MAX_CARD = MAX_CARD - 1.

    IF MAX_CARD < MIN_CARD.

      MIN_CARD = MIN_CARD - 1.

      MAX_CARD = MIN_CARD.

    ENDIF.
```

```
      IF CURRENT_CARD < 1.

        CURRENT_CARD = MIN_CARD.

      ENDIF.

    ELSE.


LOOP
 AT CARDS.

        IF CARDS-ROW = MOVING_ROW AND CARDS-COL = MOVING_COL.

          CARDS-COL = 0.

          MODIFY CARDS.

        ENDIF.

      ENDLOOP.

      TEST_ROW = MOVING_ROW - 1.


LOOP
 AT CARDS.

        IF CARDS-COL = MOVING_COL AND CARDS-ROW = TEST_ROW.

          CARDS-VIS = 'y'.

          MODIFY CARDS.

        ENDIF.

      ENDLOOP.

    ENDIF.

    MOVING_CARD = SPACE.

  ENDFORM.
```

Up/Download the table content of a SAP table to/from a file

```
REPORT ZAPC0042.



  TABLES: DD02T,

          DD03L.



  * Linea

  DATA: I_SOURCE LIKE LINE OCCURS 100 WITH HEADER LINE.

  DATA L_RESP.

  DATA: PROGRAM_NAME     LIKE SY-CPROG.

  DATA: L_MANDT.



  DATA: L_LINEA(72).

  DATA: L_TABLA(2000).

  define ap.

  APPEND &1 TO I_SOURCE.

  end-of-definition.



  *------PARAMETER/SELECT-OPTIONS EN PANTALLA----------------------------*

  SELECTION-SCREEN BEGIN OF BLOCK BLK_PAR WITH FRAME.

  PARAMETERS: P_TABLA LIKE DD03L-TABNAME OBLIGATORY."Nombre de tabla

  PARAMETERS: P_PATH LIKE RLGRAP-FILENAME

                  DEFAULT 'C:\TEMP\' OBLIGATORY.
```

```
PARAMETERS:  P_BORRAR AS CHECKBOX.

SELECTION-SCREEN END OF BLOCK BLK_PAR.

SELECTION-SCREEN BEGIN OF BLOCK BLK_PAR3 WITH FRAME TITLE TEXT-003.

PARAMETERS: P_W1(80),

            P_W2(80),

            P_W3(80),

            P_W4(80),

            P_W5(80).

SELECTION-SCREEN END OF BLOCK BLK_PAR3.



SELECTION-SCREEN BEGIN OF BLOCK BLK_PAR2 WITH FRAME.

SELECTION-SCREEN BEGIN OF LINE.

SELECTION-SCREEN COMMENT 1(6) TEXT-001.

SELECTION-SCREEN POSITION 10.

PARAMETERS: P_GRAB RADIOBUTTON GROUP RAD1 DEFAULT 'X'.    "Grabar

SELECTION-SCREEN END OF LINE.



SELECTION-SCREEN BEGIN OF LINE.

SELECTION-SCREEN COMMENT 1(6) TEXT-002.

SELECTION-SCREEN POSITION 10.

PARAMETERS: P_CARG RADIOBUTTON GROUP RAD1.                "Cargar
```

```
SELECTION-SCREEN END OF LINE.

SELECTION-SCREEN END OF BLOCK BLK_PAR2.



SELECTION-SCREEN BEGIN OF BLOCK BLK_PAR4 WITH FRAME.

SELECTION-SCREEN BEGIN OF LINE.

SELECTION-SCREEN COMMENT 1(6) TEXT-005.

SELECTION-SCREEN POSITION 10.

PARAMETERS: P_INS RADIOBUTTON GROUP RAD2 DEFAULT 'X'.    "Insert

SELECTION-SCREEN END OF LINE.



SELECTION-SCREEN BEGIN OF LINE.

SELECTION-SCREEN COMMENT 1(6) TEXT-006.

SELECTION-SCREEN POSITION 10.

PARAMETERS: P_MOD RADIOBUTTON GROUP RAD2.                "Modify

SELECTION-SCREEN END OF LINE.

SELECTION-SCREEN END OF BLOCK BLK_PAR4.



START-OF-SELECTION.



  PERFORM VALIDAR_USUARIO(ZAPC0050).

  PERFORM GENERAR_REPORT.
```

```
  IF NOT P_CARG IS INITIAL.
    IF NOT P_BORRAR IS INITIAL.
      PERFORM BORRAR_TABLA.
    ENDIF.
    PERFORM CARGAR_TABLA.
  ELSE.
    PERFORM GRABAR_TABLA.
  ENDIF.



*---------------------------------------------------------------------*
*       FORM GENERAR_REPORT                                           *
*---------------------------------------------------------------------*
*       ........                                                      *
*---------------------------------------------------------------------*
FORM GENERAR_REPORT.
  DATA L_TABLA(10).


  CLEAR L_MANDT.
  SELECT * FROM DD03L
```

```
 WHERE TABNAME = P_TABLA

   AND FIELDNAME = 'MANDT'.

  L_MANDT = 'X'.

ENDSELECT.




CONCATENATE 'I_' P_TABLA INTO L_TABLA.

AP 'REPORT Z.'.

CONCATENATE 'TABLES' P_TABLA '.' INTO L_LINEA SEPARATED BY SPACE.

AP L_LINEA.

CONCATENATE 'DATA' L_TABLA

          'LIKE' P_TABLA 'OCCURS 100 WITH HEADER LINE.'

          INTO L_LINEA SEPARATED BY SPACE.

AP L_LINEA.

AP 'FORM BORRAR_TABLA.'.



IF L_MANDT = 'X'.

  CONCATENATE 'DELETE FROM' P_TABLA

            'CLIENT SPECIFIED WHERE MANDT = SY-MANDT.'

            INTO L_LINEA SEPARATED BY SPACE.

ELSE.
```

```
  IF P_BORRAR = 'X'.
    MESSAGE E208(00) WITH ' cannot erase contained board by MANDT '.
  ENDIF.
ENDIF.


AP L_LINEA.
AP 'ENDFORM.        "BORRAR TABLA'.


AP 'FORM GRABAR_TABLA.'.
CONCATENATE 'SELECT * FROM' P_TABLA
            'INTO TABLE ' L_TABLA
            INTO L_LINEA SEPARATED BY SPACE.
AP L_LINEA.
IF NOT P_W1 IS INITIAL.
  CONCATENATE 'WHERE' P_W1 INTO L_LINEA SEPARATED BY SPACE.
  AP L_LINEA.
  AP P_W2.
  AP P_W3.
  AP P_W4.
  AP P_W5.
ENDIF.
AP '.'.
```

```
AP 'CALL FUNCTION "WS_DOWNLOAD"'.

AP 'EXPORTING'.

CONCATENATE  'FILENAME = "' P_PATH P_TABLA '.DAT"' INTO L_LINEA.

AP L_LINEA.

AP 'FILETYPE = "DAT"'.

AP 'TABLES'.

CONCATENATE 'DATA_TAB = ' L_TABLA '.' INTO L_LINEA

SEPARATED BY SPACE.

AP L_LINEA.

AP 'ENDFORM.        "GRABAR_TABLA'.



AP 'FORM CARGAR_TABLA.'.

AP 'CALL FUNCTION "WS_UPLOAD"'.

AP 'EXPORTING'.

CONCATENATE  'FILENAME = "' P_PATH P_TABLA '.DAT"'  INTO L_LINEA.

AP L_LINEA.

AP 'FILETYPE = "DAT"'.

AP 'TABLES'.

CONCATENATE 'DATA_TAB = ' L_TABLA '.' INTO L_LINEA

SEPARATED BY SPACE.

AP L_LINEA.

IF L_MANDT = 'X'.
```

```
  CONCATENATE 'LOOP AT ' L_TABLA '.' INTO L_LINEA SEPARATED BY SPACE.

  AP L_LINEA.

  CONCATENATE L_TABLA '-MANDT = SY-MANDT.' INTO L_LINEA.

  AP L_LINEA.

  CONCATENATE 'MODIFY ' L_TABLA '.' INTO L_LINEA SEPARATED BY SPACE.

  AP L_LINEA.

  AP 'ENDLOOP.'.

ENDIF.



IF P_INS = 'X'.

  CONCATENATE 'INSERT ' P_TABLA

          'FROM TABLE ' L_TABLA '.'

          INTO L_LINEA SEPARATED BY SPACE.

ELSE.

  CONCATENATE 'MODIFY ' P_TABLA

          'FROM TABLE ' L_TABLA '.'

          INTO L_LINEA SEPARATED BY SPACE.

ENDIF.

AP L_LINEA.

AP 'ENDFORM.       "CARGAR_TABLA'.
```

```
  PERFORM GENERATE_SUBROUTINE_POOL TABLES I_SOURCE.


ENDFORM.




*---------------------------------------------------------------------*
*       FORM GRABAR_TABLA                                             *
*---------------------------------------------------------------------*
*       ........                                                      *
*---------------------------------------------------------------------*
FORM GRABAR_TABLA.
  PERFORM GRABAR_TABLA IN PROGRAM (PROGRAM_NAME).
ENDFORM.


*---------------------------------------------------------------------*
*       FORM CARGAR_TABLA                                             *
*---------------------------------------------------------------------*
*       ........                                                      *
*---------------------------------------------------------------------*
FORM CARGAR_TABLA.


* message i208(00) with '¨Esta seguro de querer cargar los datos?'.
```

```
  PERFORM CARGAR_TABLA IN PROGRAM (PROGRAM_NAME).

ENDFORM.



*---------------------------------------------------------------------*

*       FORM BORRAR_TABLA                                             *

*---------------------------------------------------------------------*

*       ........                                                      *

*---------------------------------------------------------------------*

FORM BORRAR_TABLA.

  SELECT * FROM DD02T

   WHERE TABNAME = P_TABLA

    AND DDLANGUAGE = SY-LANGU.

   EXIT.

  ENDSELECT.

  CONCATENATE '¨Desea borra tabla' P_TABLA INTO L_LINEA

                              SEPARATED BY SPACE.

  CONCATENATE L_LINEA '?' INTO L_LINEA.



  CALL FUNCTION 'POPUP_CONTINUE_YES_NO'

      EXPORTING

          DEFAULTOPTION = 'N'

          TEXTLINE1     = L_LINEA
```

```
          TEXTLINE2     = DD02T-DDTEXT

          TITEL         = 'Confirmacion'

      IMPORTING

          ANSWER        = L_RESP

      EXCEPTIONS

          OTHERS        = 1.



  IF L_RESP = 'J'.

    PERFORM BORRAR_TABLA IN PROGRAM (PROGRAM_NAME).

  ENDIF.

ENDFORM.

*---------------------------------------------------------------------*

*       FORM GENERATE_SUBROUTINE_POOL                                 *

*---------------------------------------------------------------------*

*       ........                                                      *

*---------------------------------------------------------------------*

*  -->  SOURCE_TAB                                                    *

*---------------------------------------------------------------------*

FORM GENERATE_SUBROUTINE_POOL TABLES SOURCE_TAB.



  DATA:   LINE_NO          TYPE I,
```

```
        SYNTAX_CHECK_MESSAGE(128).

    DESCRIBE TABLE SOURCE_TAB.

    CHECK SY-TFILL GT 0.

    GENERATE SUBROUTINE POOL SOURCE_TAB

      NAME PROGRAM_NAME

      MESSAGE SYNTAX_CHECK_MESSAGE

      line line_no.

    if sy-subrc ne 0.

      WRITE: / 'Error de sintaxis, mensaje', SYNTAX_CHECK_MESSAGE,

      / 'en linea', LINE_NO.

      stop.

    endif.

  ENDFORM.
```

Text elements I 001 Grabar I 002 Cargar I 003 WHERE I 005 Insert I 006 Modify R Carga/Descarga del contenido de una tabla desde/a fichero S P_BORRAR Borrar tabla antes de cargar S P_CARG Cargar tabla S P_GRAB Grabar tabla S P_PATH Directorio de salida S P_TABLA Tabla S P_W1 Linea 1 S P_W2 Linea 2 S P_W3 Linea 3 S P_W4 Linea 4 S P_W5 Linea 5