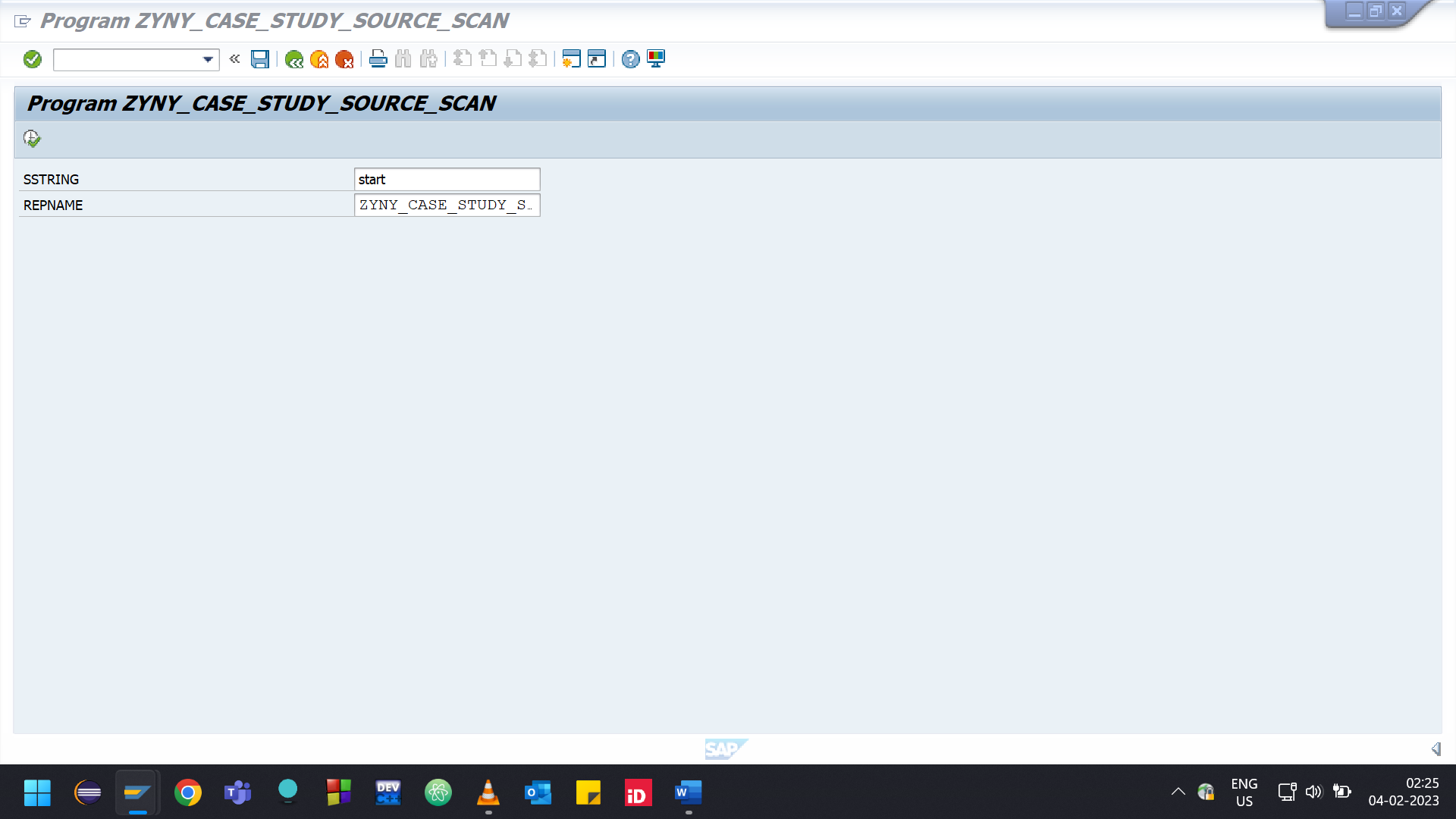
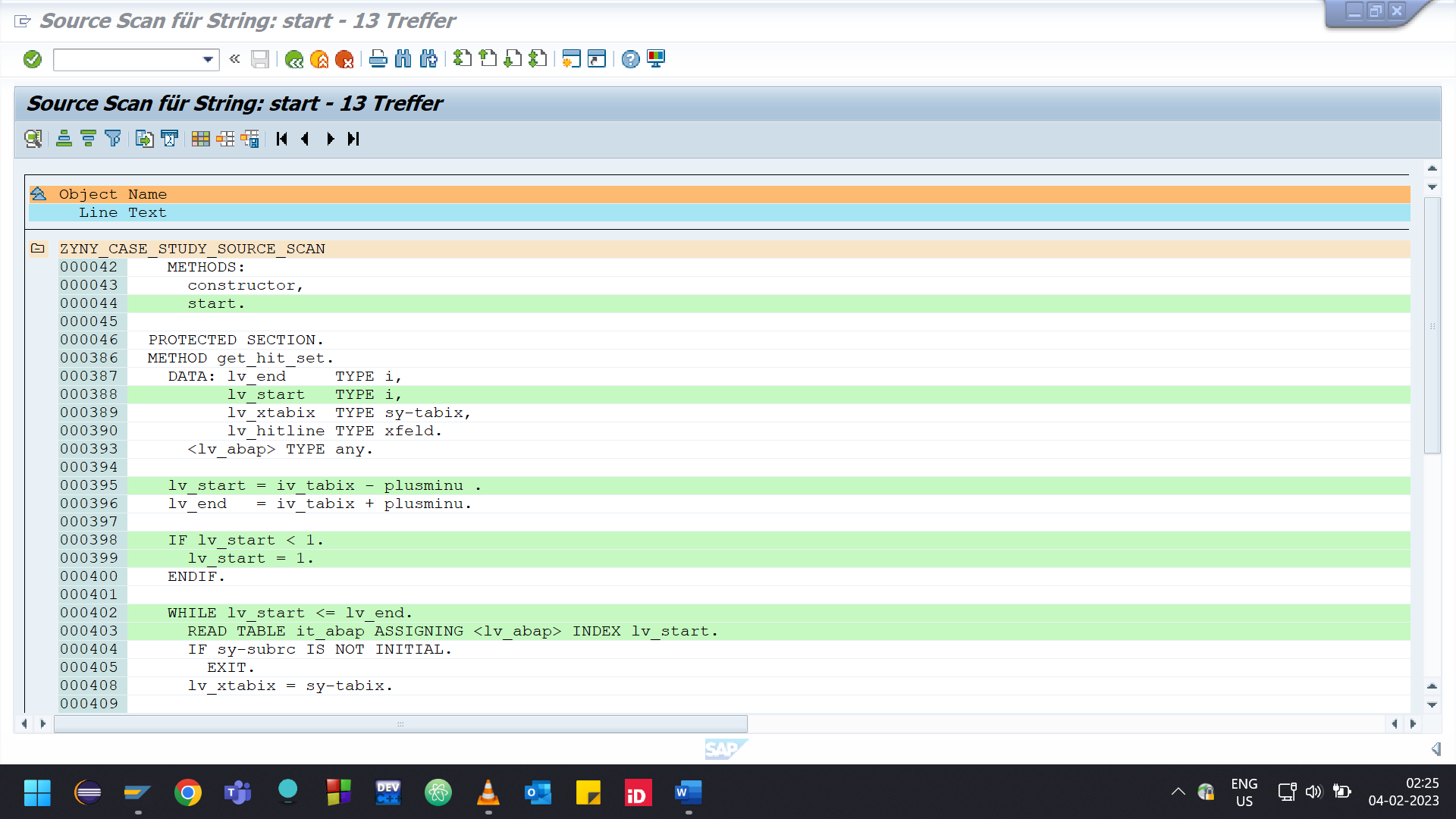
PROGRAM ZYNY\_CASE\_STUDY\_SOURCE\_SCAN.  
  
TYPE-POOLS:  
  slis,  
  sscr.  
  
TABLES:  
  seoclasstx,                                                           " Short description class/interface  
  tadir,                                                                " Directory of Repository Objects  
  tlibt,                                                                " Function Group Short Texts  
  d020s,                                                                " System Table TRDIR  
  trdir.  
  
CLASS:  
  lcl\_source\_scan DEFINITION DEFERRED.  
  
DATA:  
  lo\_sscan   TYPE REF TO lcl\_source\_scan,  
  lv\_sstring TYPE text255,  
  lv\_appl    TYPE taplt-appl,                                           " Applications programs, function modules, logical databases  
  plusminu(2) TYPE n value 2,                                           " TO GET 2 LINES BEFORE AND AFTER OF FOUND STRING  
  comment type xfeld value 'X'.  
  
SELECT-OPTIONS sstring FOR lv\_sstring NO INTERVALS no-EXTENSION MODIF ID dsp. " STRING SEARCH  
SELECT-OPTIONS repname FOR trdir-name NO INTERVALS no-EXTENSION MEMORY ID rs\_scan\_repid. " PROGRAM NAME  
  
\*----------------------------------------------------------------------\*  
\*       CLASS lcx\_scan\_exceptions DEFINITION  
\*----------------------------------------------------------------------\*  
\*       Exceptions for source scanning  
\*----------------------------------------------------------------------\*  
CLASS lcx\_scan\_exceptions DEFINITION INHERITING FROM cx\_static\_check FINAL.  
ENDCLASS.                    "lcx\_scan\_exceptions DEFINITION  
  
\*----------------------------------------------------------------------\*  
\*       CLASS lcl\_source\_scan DEFINITION  
\*----------------------------------------------------------------------\*  
\*       ABAP source scanner  
\*----------------------------------------------------------------------\*  
CLASS lcl\_source\_scan DEFINITION FINAL.  
  PUBLIC SECTION.  
    METHODS:  
      constructor,  
      start.  
  
  PROTECTED SECTION.  
    TYPES:  
      BEGIN OF ty\_dynpro,  
        repname LIKE d020s-prog,  
        dynnr  LIKE d020s-dnum,  
      END OF ty\_dynpro.  
  
    TYPES:  
      BEGIN OF ty\_ls\_objname,  
        report TYPE sy-repid,  
        dynnr  TYPE sy-dynnr,  
      END OF ty\_ls\_objname.  
  
    DATA:  
      go\_alv        TYPE REF TO cl\_salv\_hierseq\_table,  
      gv\_hit\_count  TYPE i,  
      gv\_sstring    TYPE string,  
      gv\_dynp\_found TYPE xfeld,  
      gv\_vers\_found TYPE xfeld,  
      gt\_dynpro     TYPE STANDARD TABLE OF ty\_dynpro,  
      gt\_object     TYPE STANDARD TABLE OF tadir-obj\_name,  
      gt\_vrsd       TYPE HASHED TABLE OF vrsd  
                      WITH UNIQUE KEY objname versno,  
      gt\_source TYPE abaptxt255\_tab,  
      gv\_report TYPE syrepid,  
      gv\_dynpro TYPE sydynnr,  
  
      BEGIN OF gs\_alv\_header,  
        repname TYPE tadir-obj\_name,  
        dynnr   TYPE sy-dynnr,  
        expand  TYPE xfeld,  
        versno  TYPE vrsd-versno,  
      END OF gs\_alv\_header,  
  
      gt\_alv\_header LIKE STANDARD TABLE OF gs\_alv\_header,  
  
      BEGIN OF gs\_alv\_item,  
        repname    TYPE sy-repid,  
        dynnr      TYPE sy-dynnr,  
        versno     TYPE vrsd-versno,  
        line\_no    TYPE rsrow,  
        text       TYPE text255,  
        hit        TYPE xfeld,  
        cell\_color TYPE lvc\_t\_scol,  
      END OF gs\_alv\_item,  
  
      gt\_alv\_item LIKE STANDARD TABLE OF gs\_alv\_item.  
  
    CONSTANTS:  
      gc\_x TYPE xfeld VALUE 'X'.  
  
    METHODS:  
      add\_to\_hitlist  
        IMPORTING  
          iv\_report      TYPE clike  
          iv\_dynpro      TYPE clike OPTIONAL  
          iv\_source\_line TYPE clike  
          iv\_tabix       TYPE sy-tabix  
          iv\_hit         TYPE xfeld  
          iv\_versno      TYPE vrsd-versno,  
  
      display,  
  
      get\_alv\_instance,  
  
      get\_hit\_set  
        IMPORTING  
          iv\_report      TYPE clike  
          iv\_dynpro      TYPE clike OPTIONAL  
          it\_abap        TYPE abaptxt255\_tab  
          iv\_tabix       TYPE sy-tabix  
          iv\_versno      TYPE vrsd-versno,  
  
      get\_source\_names,  
  
      get\_source\_by\_version  
        IMPORTING  
          iv\_report TYPE clike  
          iv\_dynpro TYPE clike OPTIONAL  
          iv\_versno TYPE vrsd-versno  
        RETURNING VALUE(rt\_abap) TYPE abaptxt255\_tab,  
  
      search\_abap\_source   RAISING lcx\_scan\_exceptions,  
  
      search\_source        RAISING lcx\_scan\_exceptions,  
  
      set\_alv\_attributes.  
  
ENDCLASS.                    "lcl\_source\_scan DEFINITION  
  
CLASS lcl\_source\_scan IMPLEMENTATION.  
  
  METHOD constructor.  
    DATA:  
      ls\_restrict    TYPE sscr\_restrict,  
      ls\_opt\_list    TYPE sscr\_opt\_list,  
      ls\_association TYPE sscr\_ass.  
  
    ls\_opt\_list-name       = 'RESTRICT'.  
    ls\_opt\_list-options-cp = gc\_x.  
    ls\_opt\_list-options-eq = gc\_x.  
  
    APPEND ls\_opt\_list TO ls\_restrict-opt\_list\_tab.  
  
    ls\_association-kind    = 'S'.  
    ls\_association-name    = 'SSTRING'.  
    ls\_association-sg\_main = 'I'.  
    ls\_association-op\_main = ls\_association-op\_addy = 'RESTRICT'.  
  
    APPEND ls\_association TO ls\_restrict-ass\_tab.  
  
    CALL FUNCTION 'SELECT\_OPTIONS\_RESTRICT'  
      EXPORTING  
        program     = sy-repid  
        restriction = ls\_restrict  
      EXCEPTIONS  
        OTHERS      = 0.  
  
  ENDMETHOD.                    "constructor  
  
  METHOD set\_alv\_attributes.  
    DATA:  
      lo\_layout    TYPE REF TO cl\_salv\_layout,  
      lo\_functions TYPE REF TO cl\_salv\_functions\_list,  
      lo\_level     TYPE REF TO cl\_salv\_hierseq\_level,  
      lo\_column    TYPE REF TO cl\_salv\_column\_hierseq,  
      lo\_columns   TYPE REF TO cl\_salv\_columns\_hierseq,  
      lt\_columns   TYPE salv\_t\_column\_ref,  
      ls\_columns   LIKE LINE OF lt\_columns,  
      lo\_settings  TYPE REF TO cl\_salv\_display\_settings,  
      lv\_title     TYPE lvc\_title,  
      lv\_hits      TYPE lvc\_title,  
      ls\_color     TYPE lvc\_s\_colo,  
      ls\_layout    TYPE salv\_s\_layout\_key,  
      lt\_functions TYPE salv\_t\_ui\_func.  
  
\*   Layout  
    ls\_layout-report = sy-repid.  
    ls\_layout-handle = 'SCAN'.  
  
    lo\_layout = go\_alv->get\_layout( ).  
    lo\_layout->set\_key( ls\_layout ).  
    lo\_layout->set\_save\_restriction( ).  
  
\*   Function keys/buttons  
    lo\_functions = go\_alv->get\_functions( ).  
    lo\_functions->set\_all( gc\_x ).  
  
\*   exclude the following functions (column paging buttons)  
    lt\_functions = lo\_functions->get\_functions( ).  
  
\*   Display settings  
    lo\_settings = go\_alv->get\_display\_settings( ).  
  
\*   Title  
    lv\_hits = gv\_hit\_count.  
    SHIFT lv\_hits LEFT DELETING LEADING space.  
  
    CONCATENATE lv\_hits  
                'Treffer'(001)  
                INTO lv\_hits SEPARATED BY space.  
  
    lv\_title = 'Source Scan für String:'(002).  
  
    CONCATENATE lv\_title  
                gv\_sstring  
                INTO lv\_title SEPARATED BY space.  
  
    CONCATENATE lv\_title  
                lv\_hits  
                INTO lv\_title SEPARATED BY ' - '.  
  
    lo\_settings->set\_list\_header( lv\_title ).  
  
\*   Field catalog  
    TRY.  
\*       Field catalog/columns - header table  
        lo\_columns  = go\_alv->get\_columns( '1' ).  
        lt\_columns = lo\_columns->get( ).  
  
        TRY.  
            lo\_columns->set\_expand\_column( 'EXPAND' ).  
  
            lo\_level = go\_alv->get\_level( '1' ).  
            lo\_level->set\_items\_expanded( gc\_x ).  
  
          CATCH cx\_salv\_data\_error.  
        ENDTRY.  
  
        LOOP AT lt\_columns INTO ls\_columns.  
          CASE ls\_columns-columnname.  
            WHEN 'EXPAND'.  
              ls\_columns-r\_column->set\_technical( ).  
  
            WHEN 'DYNNR'.  
              IF gv\_dynp\_found IS INITIAL.  
                ls\_columns-r\_column->set\_technical( ).  
              ELSE.  
                ls\_columns-r\_column->set\_output\_length( '15' ).  
              ENDIF.  
  
            WHEN 'VERSNO'.  
              IF gv\_vers\_found IS INITIAL.  
                ls\_columns-r\_column->set\_technical( ).  
              ELSE.  
                ls\_columns-r\_column->set\_leading\_zero( gc\_x ).  
                ls\_columns-r\_column->set\_output\_length( '15' ).  
                TRY.  
                    lo\_column ?= ls\_columns-r\_column.  
                    lo\_column->set\_cell\_type( if\_salv\_c\_cell\_type=>hotspot ).  
                  CATCH cx\_sy\_move\_cast\_error.  
                ENDTRY.  
              ENDIF.  
          ENDCASE.  
        ENDLOOP.  
  
\*       Field catalog/columns - item table  
        lo\_columns = go\_alv->get\_columns( '2' ).  
  
        TRY.  
            lo\_columns->set\_color\_column( 'CELL\_COLOR' ).  
          CATCH cx\_salv\_data\_error.  
        ENDTRY.  
  
        lt\_columns = lo\_columns->get( ).  
  
        LOOP AT lt\_columns INTO ls\_columns.  
          CASE ls\_columns-columnname.  
            WHEN 'REPNAME'.  
              ls\_columns-r\_column->set\_technical( ).  
  
            WHEN 'DYNNR'.  
              ls\_columns-r\_column->set\_technical( ).  
  
            WHEN 'VERSNO'.  
              ls\_columns-r\_column->set\_technical( ).  
  
            WHEN 'CELL\_COLOR'.  
              ls\_columns-r\_column->set\_technical( ).  
  
            WHEN 'HIT'.  
              ls\_columns-r\_column->set\_technical( ).  
  
            WHEN 'LINE\_NO'.  
              ls\_color-col = '4'.  
              TRY.  
                  lo\_column ?= ls\_columns-r\_column.  
                  lo\_column->set\_color( ls\_color ).  
                  lo\_column->set\_leading\_zero( gc\_x ).  
                CATCH cx\_sy\_move\_cast\_error.  
              ENDTRY.  
  
            WHEN 'TEXT'.  
              TRY.  
                  lo\_column ?= ls\_columns-r\_column.  
                  lo\_column->set\_cell\_type( if\_salv\_c\_cell\_type=>hotspot ).  
                CATCH cx\_sy\_move\_cast\_error.  
              ENDTRY.  
  
          ENDCASE.  
        ENDLOOP.  
      CATCH cx\_salv\_not\_found.  
    ENDTRY.  
  
  ENDMETHOD.                    "set\_alv\_attributes  
  
  METHOD get\_alv\_instance.  
    DATA:  
      lt\_alv\_bind TYPE salv\_t\_hierseq\_binding,  
      ls\_alv\_bind LIKE LINE OF lt\_alv\_bind.  
  
    ls\_alv\_bind-master = ls\_alv\_bind-slave = 'REPNAME'.  
    APPEND ls\_alv\_bind TO lt\_alv\_bind.  
  
    ls\_alv\_bind-master = ls\_alv\_bind-slave = 'DYNNR'.  
    APPEND ls\_alv\_bind TO lt\_alv\_bind.  
  
    ls\_alv\_bind-master = ls\_alv\_bind-slave = 'VERSNO'.  
    APPEND ls\_alv\_bind TO lt\_alv\_bind.  
  
    TRY.  
        CALL METHOD cl\_salv\_hierseq\_table=>factory  
          EXPORTING  
            t\_binding\_level1\_level2 = lt\_alv\_bind  
          IMPORTING  
            r\_hierseq               = go\_alv  
          CHANGING  
            t\_table\_level1          = gt\_alv\_header  
            t\_table\_level2          = gt\_alv\_item.  
  
      CATCH cx\_salv\_data\_error.  
      CATCH cx\_salv\_not\_found.  
    ENDTRY.  
  
  ENDMETHOD.                    "get\_alv\_instance  
  
  METHOD display.  
  
    DATA text TYPE c LENGTH 150.  
  
    IF gv\_hit\_count IS INITIAL.  
      MESSAGE s010(ZNYN\_MESSAGE\_CLASS) with gv\_sstring 'String not found' .  
      RETURN.  
    ENDIF.  
  
    SORT gt\_alv\_item BY repname dynnr versno line\_no hit DESCENDING.  
    DELETE ADJACENT DUPLICATES FROM gt\_alv\_item COMPARING repname dynnr versno line\_no.  
  
    get\_alv\_instance( ).  
    CHECK go\_alv IS NOT INITIAL.  
  
    set\_alv\_attributes( ).  
  
    go\_alv->display( ).  
  
  ENDMETHOD.                    "display  
  
  METHOD add\_to\_hitlist.  
    DATA:  
      ls\_col LIKE LINE OF gs\_alv\_item-cell\_color.  
  
    gs\_alv\_item-repname = iv\_report.  
    gs\_alv\_item-dynnr   = iv\_dynpro.  
    gs\_alv\_item-line\_no = iv\_tabix.  
    gs\_alv\_item-versno  = iv\_versno.  
    gs\_alv\_item-text    = iv\_source\_line.  
  
    IF iv\_hit IS NOT INITIAL.  
      gs\_alv\_item-hit = gc\_x.  
      ADD 1 TO gv\_hit\_count.  
      ls\_col-fname     = 'TEXT'.  
      ls\_col-color-col = '5'.  
      APPEND ls\_col TO gs\_alv\_item-cell\_color.  
    ENDIF.  
  
    APPEND gs\_alv\_item TO gt\_alv\_item.  
  
    CLEAR gs\_alv\_item.  
  ENDMETHOD.                    "add\_to\_hitlist  
  
  METHOD get\_hit\_set.  
    DATA: lv\_end     TYPE i,  
          lv\_start   TYPE i,  
          lv\_xtabix  TYPE sy-tabix,  
          lv\_hitline TYPE xfeld.  
  
    FIELD-SYMBOLS:  
      <lv\_abap> TYPE any.  
  
    lv\_start = iv\_tabix - plusminu .  
    lv\_end   = iv\_tabix + plusminu.  
  
    IF lv\_start < 1.  
      lv\_start = 1.  
    ENDIF.  
  
    WHILE lv\_start <= lv\_end.  
      READ TABLE it\_abap ASSIGNING <lv\_abap> INDEX lv\_start.  
      IF sy-subrc IS NOT INITIAL.  
        EXIT.  
      ENDIF.  
  
      lv\_xtabix = sy-tabix.  
  
      IF lv\_start = iv\_tabix.  
        lv\_hitline = gc\_x.  
      ELSE.  
        CLEAR lv\_hitline.  
      ENDIF.  
  
      ADD 1 TO lv\_start.  
  
      IF comment IS NOT INITIAL.  
          IF <lv\_abap>(1) = '\*'.  
            IF  <lv\_abap>(2) = '\*{' OR <lv\_abap>(2) = '\*}'.  
            ELSE.  
              CONTINUE.  
            ENDIF.  
          ENDIF.  
      ENDIF.  
  
      CALL METHOD add\_to\_hitlist  
        EXPORTING  
          iv\_report      = iv\_report  
          iv\_dynpro      = iv\_dynpro  
          iv\_source\_line = <lv\_abap>  
          iv\_tabix       = lv\_xtabix  
          iv\_hit         = lv\_hitline  
          iv\_versno      = iv\_versno.  
  
    ENDWHILE.  
  
  ENDMETHOD.                    "get\_hit\_set  
  
  METHOD get\_source\_by\_version. "  
    DATA:  
      lv\_object\_name TYPE versobjnam,  
      ls\_object\_name TYPE ty\_ls\_objname,  
      lt\_trdir       TYPE STANDARD TABLE OF trdir,  
      lt\_d022s       TYPE STANDARD TABLE OF d022s.  
  
    IF iv\_dynpro IS INITIAL.  
      lv\_object\_name = iv\_report.  
  
      CALL FUNCTION 'SVRS\_GET\_REPS\_FROM\_OBJECT'  
        EXPORTING  
          object\_name                  = lv\_object\_name  
          object\_type                  = 'REPS'  
          versno                       = iv\_versno  
          iv\_no\_release\_transformation = 'X'  
        TABLES  
          repos\_tab                    = rt\_abap  
          trdir\_tab                    = lt\_trdir  
        EXCEPTIONS  
          no\_version                   = 1  
          OTHERS                       = 2.  
    ELSE.  
      ls\_object\_name-report = iv\_report.  
      ls\_object\_name-dynnr  = iv\_dynpro.  
  
      lv\_object\_name = ls\_object\_name.  
  
      CALL FUNCTION 'SVRS\_GET\_VERSION\_DYNP\_40'  
        EXPORTING  
          object\_name = lv\_object\_name  
          versno      = iv\_versno  
        TABLES  
          d022s\_tab   = lt\_d022s  
        EXCEPTIONS  
          OTHERS      = 1.  
  
      CHECK sy-subrc IS INITIAL AND lt\_d022s IS NOT INITIAL.  
  
      APPEND LINES OF lt\_d022s TO rt\_abap.  
  
    ENDIF.  
  ENDMETHOD.                    "get\_source\_by\_version  
  
  METHOD search\_abap\_source.  
    DATA:  
      percentage     TYPE i,  
      old\_percentage TYPE i VALUE -1,  
      text           TYPE c LENGTH 150.  
  
    LOOP AT gt\_object INTO gv\_report.  
  
      READ REPORT gv\_report INTO gt\_source. " this will get the source code  
      IF sy-subrc IS NOT INITIAL.  
        CONTINUE.  
      ENDIF.  
  
      search\_source( ).  
  
    ENDLOOP.  
  
    FREE gt\_object.  
  
  ENDMETHOD.  
  
  METHOD search\_source.  
    DATA:  
      lt\_source\_vers  TYPE abaptxt255\_tab,  
      lv\_string\_found TYPE xfeld,  
      lt\_vrsd         TYPE STANDARD TABLE OF vrsd,  
      ls\_vrsd         LIKE LINE OF lt\_vrsd,  
      lv\_number       TYPE i,  
      lv\_index        TYPE i,  
      lt\_results      TYPE match\_result\_tab,  
      ls\_result       LIKE LINE OF lt\_results,  
      ls\_sstring      like line of sstring.  
  
      lv\_number = 1.  
  
    DO lv\_number TIMES.  
      CLEAR lv\_string\_found.  
  
      IF sy-index = 1.  
        CLEAR ls\_vrsd.  
      ENDIF.  
  
      LOOP AT sstring INTO ls\_sstring. " no need for loop  
        REFRESH lt\_results.  
  
          FIND ALL OCCURRENCES OF ls\_sstring-low IN TABLE gt\_source  
            IN CHARACTER MODE  
            IGNORING CASE  
            RESULTS lt\_results.  
  
        CHECK lt\_results IS NOT INITIAL.  
  
        lv\_string\_found = gc\_x.  
  
        SORT lt\_results BY line.  
        DELETE ADJACENT DUPLICATES FROM lt\_results COMPARING line.  
  
        LOOP AT lt\_results INTO ls\_result.  
          CALL METHOD get\_hit\_set  
            EXPORTING  
              iv\_report = gv\_report  
              iv\_dynpro = gv\_dynpro  
              it\_abap   = gt\_source  
              iv\_tabix  = ls\_result-line  
              iv\_versno = ls\_vrsd-versno.  
        ENDLOOP.  
  
      ENDLOOP.  
      IF lv\_string\_found IS NOT INITIAL.  
\*       Add ALV header entry  
        CLEAR gs\_alv\_header.  
  
        gs\_alv\_header-repname = gv\_report.  
        gs\_alv\_header-dynnr   = gv\_dynpro.  
        gs\_alv\_header-versno  = ls\_vrsd-versno.  
        APPEND gs\_alv\_header TO gt\_alv\_header.  
  
        IF gv\_dynpro IS NOT INITIAL.  
          gv\_dynp\_found = gc\_x.  
        ENDIF.  
  
        IF ls\_vrsd-versno IS NOT INITIAL.  
          gv\_vers\_found = gc\_x.  
        ENDIF.  
      ENDIF.  
    ENDDO.  
  
  ENDMETHOD.  
  
  METHOD get\_source\_names.  
    IF repname[] IS NOT INITIAL.  
  
      SELECT name APPENDING TABLE gt\_object  
        FROM trdir  
        WHERE name IN repname.  
    ENDIF.  
  
  ENDMETHOD.                    "get\_source\_names  
  
  METHOD start.  
    DATA:  
     ls\_sstring LIKE LINE OF sstring[].  
  
    IF sstring[] IS INITIAL.  
\*     Please specifiy a search term  
      MESSAGE s010(ZNYN\_MESSAGE\_CLASS) with 'Enter a search term' DISPLAY LIKE 'E'.  
      RETURN.  
    ENDIF.  
  
    READ TABLE sstring[] INTO ls\_sstring INDEX 1.  
    IF lines( sstring[] ) = 1.  
      gv\_sstring = ls\_sstring-low.  
    ENDIF.  
  
    get\_source\_names( ).  
  
      TRY.  
          search\_abap\_source( ).  
        CATCH lcx\_scan\_exceptions.  
          RETURN.  
      ENDTRY.  
  
    display( ).  
  ENDMETHOD.                    "start  
  
ENDCLASS.                    "lcl\_source\_scan IMPLEMENTATION  
  
INITIALIZATION.  
  CREATE OBJECT lo\_sscan.  
  
START-OF-SELECTION.  
  lo\_sscan->start( ).





PROGRAM ZYNY\_CASE\_STUDY\_SOURCE\_SCAN.  
  
TYPE-POOLS:  
  SLIS,  
  SSCR.  
  
TABLES:  
  D020S,                                                                *" System Table TRDIR*  
  TRDIR.  
  
CLASS:  
  LCL\_SOURCE\_SCAN DEFINITION DEFERRED.  
  
DATA:  
  LO\_SSCAN    TYPE REF TO LCL\_SOURCE\_SCAN,  
  PLUSMINU(2) TYPE N VALUE 2,                                           *" TO GET 2 LINES BEFORE AND AFTER OF FOUND STRING*  
  COMMENT     TYPE XFELD VALUE 'X'.  
  
PARAMETERS : SSTRING TYPE TEXT255 DEFAULT 'start',                      *" STRING SEARCH*  
             REPNAME TYPE TRDIR-NAME DEFAULT 'ZYNY\_CASE\_STUDY\_SOURCE\_SCAN'. *" PROGRAM NAME*  
  
*\*----------------------------------------------------------------------\**  
*\*       CLASS lcx\_scan\_exceptions DEFINITION*  
*\*----------------------------------------------------------------------\**  
*\*       Exceptions for source scanning*  
*\*----------------------------------------------------------------------\**  
CLASS LCX\_SCAN\_EXCEPTIONS DEFINITION INHERITING FROM CX\_STATIC\_CHECK FINAL.  
ENDCLASS.                    *"lcx\_scan\_exceptions DEFINITION*  
  
*\*----------------------------------------------------------------------\**  
*\*       CLASS lcl\_source\_scan DEFINITION*  
*\*----------------------------------------------------------------------\**  
*\*       ABAP source scanner*  
*\*----------------------------------------------------------------------\**  
CLASS LCL\_SOURCE\_SCAN DEFINITION FINAL.                                 *" source scan class*  
  PUBLIC SECTION.  
    METHODS:  
      START.  
  
    TYPES:  
      BEGIN OF TY\_LS\_OBJNAME,  
        REPORT TYPE SY-REPID,  
        DYNNR  TYPE SY-DYNNR,  
      END OF TY\_LS\_OBJNAME.  
  
    DATA:  
      GO\_ALV        TYPE REF TO CL\_SALV\_HIERSEQ\_TABLE,  
      GV\_HIT\_COUNT  TYPE I,  
      GV\_SSTRING    TYPE STRING,  
      GV\_DYNP\_FOUND TYPE XFELD,  
      GV\_VERS\_FOUND TYPE XFELD,  
      GS\_OBJECT     TYPE TADIR-OBJ\_NAME,  
      GT\_SOURCE     TYPE ABAPTXT255\_TAB,  
      GV\_REPORT     TYPE SYREPID,  
  
      BEGIN OF GS\_ALV\_HEADER,  
        REPNAME TYPE TADIR-OBJ\_NAME,  
        EXPAND  TYPE XFELD,  
      END OF GS\_ALV\_HEADER,  
  
      GT\_ALV\_HEADER LIKE STANDARD TABLE OF GS\_ALV\_HEADER,  
  
      BEGIN OF GS\_ALV\_ITEM,  
        REPNAME    TYPE SY-REPID,  
        LINE\_NO    TYPE RSROW,  
        TEXT       TYPE TEXT255,  
        HIT        TYPE XFELD,  
        CELL\_COLOR TYPE LVC\_T\_SCOL,  
      END OF GS\_ALV\_ITEM,  
  
      GT\_ALV\_ITEM LIKE STANDARD TABLE OF GS\_ALV\_ITEM.  
  
    CONSTANTS:  
      GC\_X TYPE XFELD VALUE 'X'.  
  
    METHODS:  
      ADD\_TO\_HITLIST  
        IMPORTING  
          IV\_REPORT      TYPE CLIKE  
          IV\_SOURCE\_LINE TYPE CLIKE  
          IV\_TABIX       TYPE SY-TABIX  
          IV\_HIT         TYPE XFELD,  
  
      DISPLAY,  
  
      GET\_ALV\_INSTANCE,  
  
      GET\_HIT\_SET  
        IMPORTING  
          IV\_REPORT TYPE CLIKE  
          IT\_ABAP   TYPE ABAPTXT255\_TAB  
          IV\_TABIX  TYPE SY-TABIX,  
  
      GET\_SOURCE\_NAMES,  
  
      SEARCH\_ABAP\_SOURCE   RAISING LCX\_SCAN\_EXCEPTIONS,  
  
      SEARCH\_SOURCE        RAISING LCX\_SCAN\_EXCEPTIONS,  
  
      SET\_ALV\_ATTRIBUTES.  
  
ENDCLASS.                    *"lcl\_source\_scan DEFINITION*  
  
CLASS LCL\_SOURCE\_SCAN IMPLEMENTATION.  
  
  METHOD SET\_ALV\_ATTRIBUTES.  
    DATA:  
      LO\_LAYOUT    TYPE REF TO CL\_SALV\_LAYOUT,  
      LO\_FUNCTIONS TYPE REF TO CL\_SALV\_FUNCTIONS\_LIST,  
      LO\_LEVEL     TYPE REF TO CL\_SALV\_HIERSEQ\_LEVEL,  
      LO\_COLUMN    TYPE REF TO CL\_SALV\_COLUMN\_HIERSEQ,  
      LO\_COLUMNS   TYPE REF TO CL\_SALV\_COLUMNS\_HIERSEQ,  
      LT\_COLUMNS   TYPE SALV\_T\_COLUMN\_REF,  
      LS\_COLUMNS   LIKE LINE OF LT\_COLUMNS,  
      LO\_SETTINGS  TYPE REF TO CL\_SALV\_DISPLAY\_SETTINGS,  
      LV\_TITLE     TYPE LVC\_TITLE,  
      LV\_HITS      TYPE LVC\_TITLE,  
      LS\_COLOR     TYPE LVC\_S\_COLO,  
      LS\_LAYOUT    TYPE SALV\_S\_LAYOUT\_KEY,  
##NEEDED LT\_FUNCTIONS TYPE SALV\_T\_UI\_FUNC.  
  
*\*   Layout*  
    LS\_LAYOUT-REPORT = SY-REPID.  
    LS\_LAYOUT-HANDLE = 'SCAN'.  
  
    LO\_LAYOUT = GO\_ALV->GET\_LAYOUT( ).  
    LO\_LAYOUT->SET\_KEY( LS\_LAYOUT ).  
    LO\_LAYOUT->SET\_SAVE\_RESTRICTION( ).  
  
*\*   Function keys/buttons*  
    LO\_FUNCTIONS = GO\_ALV->GET\_FUNCTIONS( ).  
    LO\_FUNCTIONS->SET\_ALL( GC\_X ).  
  
*\*   exclude the following functions (column paging buttons)*  
    LT\_FUNCTIONS = LO\_FUNCTIONS->GET\_FUNCTIONS( ).  
  
*\*   Display settings*  
    LO\_SETTINGS = GO\_ALV->GET\_DISPLAY\_SETTINGS( ).  
  
*\*   Title*  
    LV\_HITS = GV\_HIT\_COUNT.  
    SHIFT LV\_HITS LEFT DELETING LEADING SPACE.  
  
    CONCATENATE LV\_HITS  
                'Treffer'(001)  
                INTO LV\_HITS SEPARATED BY SPACE.  
  
    LV\_TITLE = 'Source Scan für String:'(002).  
  
    CONCATENATE LV\_TITLE  
                GV\_SSTRING  
                INTO LV\_TITLE SEPARATED BY SPACE.  
  
    CONCATENATE LV\_TITLE  
                LV\_HITS  
                INTO LV\_TITLE SEPARATED BY ' - '.  
  
    LO\_SETTINGS->SET\_LIST\_HEADER( LV\_TITLE ).  
  
*\*   Field catalog*  
    TRY.  
*\*       Field catalog/columns - header table*  
        LO\_COLUMNS  = GO\_ALV->GET\_COLUMNS( '1' ).  
        LT\_COLUMNS = LO\_COLUMNS->GET( ).  
  
        TRY.  
            LO\_COLUMNS->SET\_EXPAND\_COLUMN( 'EXPAND' ).  
  
            LO\_LEVEL = GO\_ALV->GET\_LEVEL( '1' ).  
            LO\_LEVEL->SET\_ITEMS\_EXPANDED( GC\_X ).  
  
          CATCH CX\_SALV\_DATA\_ERROR.  
        ENDTRY.  
  
        LOOP AT LT\_COLUMNS INTO LS\_COLUMNS.  
          CASE LS\_COLUMNS-COLUMNNAME.  
            WHEN 'EXPAND'.  
              LS\_COLUMNS-R\_COLUMN->SET\_TECHNICAL( ).  
          ENDCASE.  
        ENDLOOP.  
  
*\*       Field catalog/columns - item table*  
        LO\_COLUMNS = GO\_ALV->GET\_COLUMNS( '2' ).  
  
        TRY.  
            LO\_COLUMNS->SET\_COLOR\_COLUMN( 'CELL\_COLOR' ).  
          CATCH CX\_SALV\_DATA\_ERROR.  
        ENDTRY.  
  
        LT\_COLUMNS = LO\_COLUMNS->GET( ).  
  
        LOOP AT LT\_COLUMNS INTO LS\_COLUMNS.  
          CASE LS\_COLUMNS-COLUMNNAME.  
            WHEN 'REPNAME'.  
              LS\_COLUMNS-R\_COLUMN->SET\_TECHNICAL( ).  
  
            WHEN 'CELL\_COLOR'.  
              LS\_COLUMNS-R\_COLUMN->SET\_TECHNICAL( ).  
  
            WHEN 'HIT'.  
              LS\_COLUMNS-R\_COLUMN->SET\_TECHNICAL( ).  
  
            WHEN 'LINE\_NO'.  
              LS\_COLOR-COL = '4'.  
              TRY.  
                  LO\_COLUMN ?= LS\_COLUMNS-R\_COLUMN.  
                  LO\_COLUMN->SET\_COLOR( LS\_COLOR ).  
                  LO\_COLUMN->SET\_LEADING\_ZERO( GC\_X ).  
                CATCH CX\_SY\_MOVE\_CAST\_ERROR.  
              ENDTRY.  
  
            WHEN 'TEXT'.  
              TRY.  
                  LO\_COLUMN ?= LS\_COLUMNS-R\_COLUMN.  
                  LO\_COLUMN->SET\_CELL\_TYPE( IF\_SALV\_C\_CELL\_TYPE=>HOTSPOT ).  
                CATCH CX\_SY\_MOVE\_CAST\_ERROR.  
              ENDTRY.  
  
          ENDCASE.  
        ENDLOOP.  
      CATCH CX\_SALV\_NOT\_FOUND.  
    ENDTRY.  
  
  ENDMETHOD.                    *"set\_alv\_attributes*  
  
  METHOD GET\_ALV\_INSTANCE.  
    DATA:  
      LT\_ALV\_BIND TYPE SALV\_T\_HIERSEQ\_BINDING,    *" Binding Between Two Tables*  
      LS\_ALV\_BIND LIKE LINE OF LT\_ALV\_BIND.  
  
    LS\_ALV\_BIND-MASTER = LS\_ALV\_BIND-SLAVE = 'REPNAME'.  
    APPEND LS\_ALV\_BIND TO LT\_ALV\_BIND.  
  
    TRY.  
        CALL METHOD CL\_SALV\_HIERSEQ\_TABLE=>FACTORY  
          EXPORTING  
            T\_BINDING\_LEVEL1\_LEVEL2 = LT\_ALV\_BIND  
          IMPORTING  
            R\_HIERSEQ               = GO\_ALV  
          CHANGING  
            T\_TABLE\_LEVEL1          = GT\_ALV\_HEADER  
            T\_TABLE\_LEVEL2          = GT\_ALV\_ITEM.  
  
      CATCH CX\_SALV\_DATA\_ERROR.  
      CATCH CX\_SALV\_NOT\_FOUND.  
    ENDTRY.  
  
  ENDMETHOD.                    *"get\_alv\_instance*  
  
  METHOD DISPLAY.  
  
    IF GV\_HIT\_COUNT IS INITIAL.  
      MESSAGE S010(ZNYN\_MESSAGE\_CLASS) WITH GV\_SSTRING 'String not found' .  
      RETURN.  
    ENDIF.  
  
    SORT GT\_ALV\_ITEM BY REPNAME LINE\_NO HIT DESCENDING.  
    DELETE ADJACENT DUPLICATES FROM GT\_ALV\_ITEM COMPARING REPNAME LINE\_NO.  
  
    GET\_ALV\_INSTANCE( ).  
    CHECK GO\_ALV IS NOT INITIAL.  
  
    SET\_ALV\_ATTRIBUTES( ).  
  
    GO\_ALV->DISPLAY( ).  
  
  ENDMETHOD.                    *"display*  
  
  METHOD ADD\_TO\_HITLIST.  
    DATA:  
      LS\_COL LIKE LINE OF GS\_ALV\_ITEM-CELL\_COLOR.  
  
    GS\_ALV\_ITEM-REPNAME = IV\_REPORT.  
    GS\_ALV\_ITEM-LINE\_NO = IV\_TABIX.  
    GS\_ALV\_ITEM-TEXT    = IV\_SOURCE\_LINE.  
  
    IF IV\_HIT IS NOT INITIAL. *" iv\_hit represent the desired line containing desired string*  
      GS\_ALV\_ITEM-HIT = GC\_X.  
      ADD 1 TO GV\_HIT\_COUNT.  
      LS\_COL-FNAME     = 'TEXT'.  
      LS\_COL-COLOR-COL = '5'.  
      APPEND LS\_COL TO GS\_ALV\_ITEM-CELL\_COLOR.  
    ENDIF.  
  
    APPEND GS\_ALV\_ITEM TO GT\_ALV\_ITEM.  
  
    CLEAR GS\_ALV\_ITEM.  
  ENDMETHOD.                    *"add\_to\_hitlist*  
  
  METHOD GET\_HIT\_SET.  
    DATA: LV\_END     TYPE I,  
          LV\_START   TYPE I,  
          LV\_XTABIX  TYPE SY-TABIX,  
          LV\_HITLINE TYPE XFELD.  
  
    FIELD-SYMBOLS:  
      <LV\_ABAP> TYPE ANY.  
  
    LV\_START = IV\_TABIX - PLUSMINU .  *" 2 line before*  
    LV\_END   = IV\_TABIX + PLUSMINU.   *" 2 lines after*  
  
    IF LV\_START < 1. *"  edge cases*  
      LV\_START = 1.  
    ENDIF.  
  
    WHILE LV\_START <= LV\_END.  
      READ TABLE IT\_ABAP ASSIGNING <LV\_ABAP> INDEX LV\_START.  
      IF SY-SUBRC IS NOT INITIAL. *" if failed then exit*  
        EXIT.  
      ENDIF.  
  
      LV\_XTABIX = SY-TABIX. *" also increase the line counter*  
  
      IF LV\_START = IV\_TABIX.  
        LV\_HITLINE = GC\_X.    *" marking the line where the desired string is found*  
      ELSE.  
        CLEAR LV\_HITLINE.  
      ENDIF.  
  
      ADD 1 TO LV\_START.  
  
      IF COMMENT IS NOT INITIAL.  
        IF <LV\_ABAP>(1) = '\*'.  
          IF  <LV\_ABAP>(2) = '\*{' OR <LV\_ABAP>(2) = '\*}'.  
          ELSE.  
            CONTINUE.  
          ENDIF.  
        ENDIF.  
      ENDIF.  
  
      CALL METHOD ADD\_TO\_HITLIST  
        EXPORTING  
          IV\_REPORT      = IV\_REPORT  
          IV\_SOURCE\_LINE = <LV\_ABAP>  
          IV\_TABIX       = LV\_XTABIX  
          IV\_HIT         = LV\_HITLINE.  
  
    ENDWHILE.  
  
  ENDMETHOD.                    *"get\_hit\_set*  
  
  METHOD SEARCH\_ABAP\_SOURCE.  
    READ REPORT GS\_OBJECT INTO GT\_SOURCE. *" this will get the source code (REPORT KWY WORD EXCLUSIVELY)*  
  
    IF SY-SUBRC IS INITIAL.  
      SEARCH\_SOURCE( ).  
    ENDIF.  
  
    FREE GS\_OBJECT.  
  ENDMETHOD.  
  
  METHOD SEARCH\_SOURCE.  
    DATA:  
      LV\_STRING\_FOUND TYPE XFELD,  
      LT\_VRSD         TYPE STANDARD TABLE OF VRSD,  
      LS\_VRSD         LIKE LINE OF LT\_VRSD,  
      LT\_RESULTS      TYPE MATCH\_RESULT\_TAB,  
      LS\_RESULT       LIKE LINE OF LT\_RESULTS,  
      LS\_SSTRING      TYPE TEXT255.  
  
    CLEAR LV\_STRING\_FOUND.  
  
    CLEAR LS\_VRSD.  
  
    REFRESH LT\_RESULTS.  
  
    FIND ALL OCCURRENCES OF SSTRING IN TABLE GT\_SOURCE    *" now searching for the desired string in source code*  
        IN CHARACTER MODE  
        IGNORING CASE  
        RESULTS LT\_RESULTS.  
  
    CHECK LT\_RESULTS IS NOT INITIAL. *" if failed this will come out of method*  
  
    LV\_STRING\_FOUND = GC\_X.   *" flag is string is found*  
  
    SORT LT\_RESULTS BY LINE.  
    DELETE ADJACENT DUPLICATES FROM LT\_RESULTS COMPARING LINE. *" removing duplicates*  
  
    LOOP AT LT\_RESULTS INTO LS\_RESULT.  
      CALL METHOD GET\_HIT\_SET  
        EXPORTING  
          IV\_REPORT = GV\_REPORT  
          IT\_ABAP   = GT\_SOURCE  
          IV\_TABIX  = LS\_RESULT-LINE.  
    ENDLOOP.  
  
    IF LV\_STRING\_FOUND IS NOT INITIAL. *" if we got some data then create header for alv*  
*\*       Add ALV header entry*  
      CLEAR GS\_ALV\_HEADER.  
  
      GS\_ALV\_HEADER-REPNAME = GV\_REPORT.  
      APPEND GS\_ALV\_HEADER TO GT\_ALV\_HEADER.  
    ENDIF.  
  
  ENDMETHOD.  
  
  METHOD GET\_SOURCE\_NAMES.  
  
    IF REPNAME IS NOT INITIAL.  
      SELECT NAME INTO GS\_OBJECT  
        FROM TRDIR  
        WHERE NAME EQ REPNAME.  
      ENDSELECT.  
    ENDIF.  
  
  ENDMETHOD.                    *"get\_source\_names*  
  
  METHOD START.  
    DATA:  
     LS\_SSTRING TYPE TEXT255.       *" char size 255*  
  
    IF SSTRING IS INITIAL.  
*\*     Please specifiy a search term*  
      MESSAGE S010(ZNYN\_MESSAGE\_CLASS) WITH 'Enter a search term' DISPLAY LIKE 'E'.  
      RETURN.  
    ENDIF.  
  
*\*   initializing local string and global variable*  
    LS\_SSTRING = SSTRING.  
    GV\_SSTRING = SSTRING.                                   *" class variable*  
  
    GET\_SOURCE\_NAMES( ).  
  
    TRY.  
        SEARCH\_ABAP\_SOURCE( ).  
      CATCH LCX\_SCAN\_EXCEPTIONS.  
        RETURN.  
    ENDTRY.  
  
    DISPLAY( ).  
  ENDMETHOD.                    *"start*  
  
ENDCLASS.                    *"lcl\_source\_scan IMPLEMENTATION*  
  
INITIALIZATION.  
  CREATE OBJECT LO\_SSCAN.  
  
START-OF-SELECTION.  
  LO\_SSCAN->START( ).