

 무료 전자 책

배우기

ABAP

Free unaffiliated eBook created from
Stack Overflow contributors.

#abap

.....	1
1: ABAP	2
.....	2
.....	2
Examples.....	2
.....	2
ABAP Hello World.....	2
2: ABAP GRID (ALV)	4
Examples.....	4
ALV	4
ALV	4
ALV	4
ALV	4
ALV	5
ALV	5
ALV	5
3: ABAP	6
Examples.....	6
.....	6
ABAP Globally Locally . ABAP	6
,	6
(,).....	7
.....	7
-	7
.....	7
.....	8
-	8
.....	8
.....	8
.....	9
4: SQL	10

Examples.....	10
SELECT	10
5:	11
Examples.....	11
.....	11
ABAP	11
.....	11
.....	11
.....	11
.....	11
.....	11
,	12
6:	14
Examples.....	14
.....	14
.....	14
7:	16
Examples.....	16
.....	16
.....	16
.....	16
SELECT	16
.....	16
8:	17
Examples.....	17
.....	17
.....	18
.....	18
9:	20
.....	20
Examples.....	20
.....	20

While	20
Do Loop	20
.....	21
10: / MESSAGE	22
.....	22
.....	22
Examples	22
.....	22
.....	22
.....	22
.....	22
.....	22
11:	24
Examples	24
.....	24
.....	24
.....	24
12:	25
.....	25
Examples	25
.....	25
.....	25
13:	26
Examples	26
.....	26
.....	26
.....	26
.....	26
OO	26
14:	28
Examples	28
IF / ELSEIF / ELSE	28

.....	28
.....	28
.....	28
/	28
.....	28
.....	29
.....	29
.....	29
15:	30
Examples	30
.....	30
.....	30
16:	31
.....	31
Examples	31
OO	31
.....	32

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: [abap](#)

It is an unofficial and free ABAP ebook created for educational purposes. All the content is extracted from [Stack Overflow Documentation](#), which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official ABAP.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

1: ABAP

ABAP SAP . SAP .

ABAP ABAP .

ABAP 7.50	2015-10-20
ABAP 7.40	2012-11-29
ABAP 7.0	2006-04-01
ABAP 6.40	2004-04-01
ABAP 6.20	2002-04-01
ABAP 6.10	2001-07-01
ABAP 4.6C	2001-04-01
ABAP 4.6A	1999-12-01
ABAP 4.5	1999-03-01
ABAP 4.0	1998-06-01
ABAP 3.0	1997-02-20

Examples

```
PROGRAM zhello_world.  
START-OF-SELECTION.  
    WRITE 'Hello, World!'.  
.
```

ABAP .

ABAP Hello World

```
PROGRAM zhello_world.  
  
CLASS main DEFINITION FINAL CREATE PRIVATE.  
    PUBLIC SECTION.  
        CLASS-METHODS: start.  
ENDCLASS.  
  
CLASS main IMPLEMENTATION.  
    METHOD start.  
        cl_demo_output=>display( 'Hello World!' ).  
    ENDMETHOD.  
ENDCLASS.  
.
```

```
ENDMETHOD.  
ENDCLASS.  
  
START-OF-SELECTION.  
  main=>start( ).
```

ABAP : <https://riptutorial.com/ko/abap/topic/1196/abap->

2: ABAP GRID (ALV)

Examples

ALV

```
cl_salv_table ALV cl_salv_table . TRY ENDTRY alv->display( ) alv->display( ) .
```

ALV ABAP

```
DATA: t_spfli TYPE STANDARD TABLE OF spfli,  
      alv TYPE REF TO cl_salv_table,  
      error_message TYPE REF TO cx_salv_msg.  
  
" Fill the internal table with example data  
SELECT * FROM spfli INTO TABLE t_spfli.  
  
" Fill ALV object with data from the internal table  
TRY.  
  cl_salv_table=>factory(  
    IMPORTING  
      r_salv_table = alv  
    CHANGING  
      t_table = t_spfli ).  
  CATCH cx_salv_msg INTO error_message.  
    " error handling  
  ENDTRY.  
  
" Use the ALV object's display method to show the ALV on the screen  
alv->display( ) .
```

ALV

.

```
alv->get_columns( )->set_optimize( ) .
```

ALV

```
ALV MANDT ( ) . get_column( ) .
```

```
alv->get_columns( )->get_column( 'MANDT' )->set_visible( if_salv_c_bool_sap=>false ) .
```

ALV

. .

```
set_short_text 10
```

set_medium_text	20
set_long_text	40

```
. column alv->get_columns( )->get_column( 'DISTID' ) . . .
```

```
DATA column TYPE REF TO cl_salv_column.
column = alv->get_columns( )->get_column( 'DISTID' ).

column->set_short_text( 'Dist. Unit' ).
column->set_medium_text( 'Unit of Distance' ).
column->set_long_text( 'Mass Unit of Distance (kms, miles)' ).
```

ALV

```
, .
```

```
alv->get_functions( )->set_all( ).
```

ALV

```
.
```

```
alv->get_display_settings( )->set_stripped_pattern( if_salv_c_bool_sap=>true ).
```

ALV

```
ALV . 70 . .
```

```
alv->get_display_settings( )->set_list_header( |Flight Schedule - { lines( t_spfli ) }
records| ).
```

ABAP GRID (ALV) : <https://riptutorial.com/ko/abap/topic/4660/abap-grid---alv->

3: ABAP

Examples

ABAP Globally Locally . ABAP . .

```
CLASS lcl_abap_class DEFINITION.  
  PUBLIC SECTION.  
  PROTECTED SECTION.  
  PRIVATE SECTION.  
ENDCLASS.  
  
CLASS lcl_abap_class IMPLEMENTATION.  
ENDCLASS.
```

```
CLASS lcl_abap_class DEFINITION.  
  PUBLIC SECTION.  
    METHODS: constructor,  
             method1.  
  PROTECTED SECTION.  
  PRIVATE SECTION.  
    METHODS: method2,  
             method3.  
ENDCLASS.  
  
CLASS lcl_abap_class IMPLEMENTATION.  
  METHOD constructor.  
    "Logic  
  ENDMETHOD.  
  
  METHOD method1.  
    "Logic  
  ENDMETHOD.  
  
  METHOD method2.  
    "Logic  
    method3( ).  
  ENDMETHOD.  
  
  METHOD method3.  
    "Logic  
  ENDMETHOD.  
ENDCLASS.
```

```
DATA lo_abap_class TYPE REF TO lcl_abap_class.  
CREATE OBJECT lo_abap_class. "Constructor call  
lo_abap_class->method1( ).
```

(,,)

:

```
CLASS lcl_abap_class DEFINITION.  
  PRIVATE SECTION.  
    METHODS method1 IMPORTING iv_string TYPE string  
                    CHANGING cv_string TYPE string  
                    EXPORTING ev_string TYPE string.  
ENDCLASS.  
  
CLASS lcl_abap_class IMPLEMENTATION.  
  METHOD method1.  
    cv_string = iv_string.  
    ev_string = 'example'.  
  ENDMETHOD.  
ENDCLASS.
```

:

```
method1 (  
  EXPORTING iv_string = lv_string  
  IMPORTING ev_string = lv_string2  
  CHANGING cv_string = lv_string3  
).
```

:

```
CLASS lcl_abap_class DEFINITION.  
  PRIVATE SECTION.  
    METHODS method1 RETURNING VALUE(rv_string) TYPE string.  
ENDCLASS.  
  
CLASS lcl_abap_class IMPLEMENTATION.  
  METHOD method1.  
    rv_string = 'returned value'.  
  ENDMETHOD.  
ENDCLASS.
```

:

```
lv_string = method1( ).
```

RETURNING .

-

. INHERITING FROM .

DEFINITION.

.

```

CLASS lcl_vehicle DEFINITION.
ENDCLASS.

CLASS lcl_vehicle IMPLEMENTATION.
ENDCLASS.

CLASS lcl_car DEFINITION INHERITING FROM lcl_vehicle.
ENDCLASS.

CLASS lcl_car IMPLEMENTATION.
ENDCLASS.

```

ABSTRACT FINAL METHODS CLASS

```

. . .
final . . .

```

```

CLASS lcl_abstract DEFINITION ABSTRACT.
  PUBLIC SECTION.
    METHODS: abstract_method ABSTRACT,
             final_method FINAL
             normal_method.
ENDCLASS.

CLASS lcl_abstract IMPLEMENTATION.
  METHOD final_method.
    "This method can't be redefined in child class!
  ENDMETHOD.

  METHOD normal_method.
    "Some logic
  ENDMETHOD.

    "We can't implement abstract_method here!
ENDCLASS.

CLASS lcl_abap_class DEFINITION INHERITING FROM lcl_abstract.
  PUBLIC SECTION.
    METHODS: abstract_method REDEFINITION,
             abap_class_method.
ENDCLASS.

CLASS lcl_abap_class IMPLEMENTATION.
  METHOD abstract_method.
    "Abstract method implementation
  ENDMETHOD.

  METHOD abap_class_method.

```

```
"Logic  
ENDMETHOD.  
ENDCLASS.
```

```
▪  
▪
```

```
DATA lo_class TYPE REF TO lcl_abap_class.  
CREATE OBJECT lo_class.  
  
lo_class->abstract_method( ).  
lo_class->normal_method( ).  
lo_class->abap_class_method( ).  
lo_class->final_method( ).
```

ABAP : <https://riptutorial.com/ko/abap/topic/2244/abap->

4: SQL

Examples

SELECT

SELECT Open-SQL .

1. * This returns all records into internal table lt_mara.

```
SELECT * FROM mara
      INTO lt_mara.
```
2. * This returns single record if table consists multiple records with same key.

```
SELECT SINGLE * INTO TABLE lt_mara
      FROM mara
      WHERE matnr EQ '400-500'.
```
3. * This returns records with distinct values.

```
SELECT DISTINCT * FROM mara
      INTO TABLE lt_mara
      ORDER BY matnr.
```
4. * This puts the number of records present in table MARA into the variable lv_var

```
SELECT COUNT( * ) FROM mara
      INTO lv_var.
```

SQL : <https://riptutorial.com/ko/abap/topic/6885/sql->

5:

Examples

```
DATA: <TABLE NAME> TYPE <SORTED|STANDARD|HASHED> TABLE OF <TYPE NAME>
      WITH <UNIQUE|NON-UNIQUE> KEY <FIELDS FOR KEY>.
```

```
WITH UNIQUE | NON-UNIQUE KEY . . . . . INSERT INSERT .
```

```
WITH UNIQUE | NON-UNIQUE KEY . - . STANDARD SORTED .
```

ABAP

```
" Declaration of type
TYPES: BEGIN OF ty_flightb,
        id          TYPE fl_id,
        dat         TYPE fl_date,
        seatno     TYPE fl_seatno,
        firstname  TYPE fl_fname,
        lastname   TYPE fl_lname,
        fl_smoke   TYPE fl_smoker,
        classf     TYPE fl_class,
        classb     TYPE fl_class,
        classe     TYPE fl_class,
        meal       TYPE fl_meal,
        service    TYPE fl_service,
        discout    TYPE fl_discnt,
      END OF ty_flightb.

" Declaration of internal table
DATA t_flightb TYPE STANDARD TABLE OF ty_flightb.
```

```
DATA t_flightb TYPE STANDARD TABLE OF flightb.
```

ABAP > 7.4 .

```
TYPES t_itab TYPE STANDARD TABLE OF i WITH EMPTY KEY.

DATA(t_inline) = VALUE t_itab( ( 1 ) ( 2 ) ( 3 ) ).
```

ABAP

:

i_compc_all i_compc_all compc_str.

```
DATA: i_compc_all TYPE STANDARD TABLE OF compc_str WITH HEADER LINE.
DATA: i_compc_all TYPE STANDARD TABLE OF compc_str.
```

:

```
DATA: i_map_rules_c TYPE HASHED TABLE OF /bic/ansdomm0100 WITH HEADER LINE
DATA: i_map_rules_c TYPE HASHED TABLE OF /bic/ansdomm0100
```

(WA) , (a WA).

```
DATA: i_compc_all_line LIKE LINE OF i_compc_all.
```

,

, :

```
" Read from table with header (using a loop):
LOOP AT i_compc_all.           " Loop over table i_compc_all and assign header line
  CASE i_compc_all-fctype.     " Read cell ftype from header line from table i_compc_all
    WHEN 'B'.                 " Bill-to customer number transformation
      i_compc_bil = i_compc_all. " Assign header line of table i_compc_bil with content of
header line i_compc_all
      APPEND i_compc_bil.      " Insert header line of table i_compc_bil into table
i_compc_bil
      " ... more WHENs
    ENDCASE.
  ENDLLOOP.
```

: . .

, :

```
" Loop over table i_compc_all and assign current line to structure i_compc_all_line
LOOP AT i_compc_all INTO i_compc_all_line.
  CASE i_compc_all_line-fctype. " Read column ftype from current line (which as
assigned into i_compc_all_line)
    WHEN 'B'.                 " Bill-to customer number transformation
      i_compc_bil_line = i_compc_all_line. " Copy structure
      APPEND i_compc_bil_line TO i_compc_bil. " Append structure to table
      " more WHENs ...
    ENDCASE.
  ENDLLOOP.
```

```
" Insert into table with Header:
INSERT TABLE i_sap_knb1.           " insert into TABLE WITH HEADER: insert table
header into it's content
insert i_sap_knb1_line into table i_sap_knb1. " insert into HASHED TABLE: insert structure
i_sap_knb1_line into hashed table i_sap_knb1
APPEND p_t_errorlog_line to p_t_errorlog. " insert into STANDARD TABLE: insert structure /
```

```
wa p_t_errorlog_line into table p_t_errorlog_line
```

: [https://riptutorial.com/ko/abap/topic/1647/-](https://riptutorial.com/ko/abap/topic/1647/)

6:

Examples

include .

.

```
CLASS lcl_test DEFINITION
    FOR TESTING
    DURATION SHORT
    RISK LEVEL HARMLESS.

PRIVATE SECTION.
    DATA:
        mo_cut TYPE REF TO zcl_dummy.

    METHODS:
        setup,

        "***** 30 chars *****|
        dummy_test          for testing.
ENDCLASS.

CLASS lcl_test IMPLEMENTATION.
    METHOD setup.
        CREATE OBJECT mo_cut.
    ENDMETHOD.

    METHOD dummy_test.
        cl_aunit_assert=>fail( ).
    ENDMETHOD.
ENDCLASS.
```

FOR TESTING .setup .

▪

. . .

. , mudule , . . .

.

SCARR

ZIF_DB_SCARR :

```
INTERFACE zif_db_scarr
    PUBLIC.
    METHODS get_all
    RETURNING
        VALUE(rt_scarr) TYPE scarr_tab .
```

```
ENDINTERFACE.
```

```
:
```

```
CLASS lcl_db_scarr DEFINITION.  
  PUBLIC SECTION.  
    INTERFACES: zif_db_scarr.  
ENDCLASS.  
  
CLASS lcl_db_scarr IMPLEMENTATION.  
  METHOD zif_db_scarr~get_all.  
    " generate static data here  
  ENDMETHOD.  
ENDCLASS.  
  
CLASS lcl_test DEFINITION  
  FOR TESTING  
  DURATION SHORT  
  RISK LEVEL HARMLESS.  
  
  PRIVATE SECTION.  
    DATA:  
      mo_cut TYPE REF TO zcl_main_class.  
  
    METHODS:  
      setup.  
ENDCLASS.  
  
CLASS lcl_test IMPLEMENTATION.  
  METHOD setup.  
    DATA: lo_db_scarr TYPE REF TO lcl_db_scarr.  
  
    CREATE OBJECT lo_db_scarr.  
  
    CREATE OBJECT mo_cut  
      EXPORTING  
        io_db_scarr = lo_db_scarr.  
  ENDMETHOD.  
ENDCLASS.
```

```
ZCL_MAIN_CLASS SELECT ZIF_DB_SCARR ZIF_DB_SCARR include .
```

: [https://riptutorial.com/ko/abap/topic/3999/-](https://riptutorial.com/ko/abap/topic/3999/)

7:

Examples

```
LOOP AT lt_sflight INTO DATA(ls_sflight).  
    WRITE ls_sflight-carrid.  
ENDLOOP.
```

```
DATA begda TYPE sy-datum.
```

```
DATA: begda TYPE sy-datum,  
      endda TYPE sy-datum.
```

SELECT

```
SELECT...ENDSELECT SELECT SINGLE @ DATA(lv_cityto) . (lv_carrid lv_carrid).
```

```
DATA lv_carrid TYPE s_carr_id VALUE 'LH'.  
SELECT SINGLE cityto FROM spfli  
    INTO @DATA(lv_cityto)  
    WHERE carrid = @lv_carrid  
    AND connid = 2402.  
WRITE: / lv_cityto.
```

```
BERLIN .
```

```
DATA: lv_string TYPE string, " standard declaration  
      lv_char TYPE c, " declares a character variable of length 1  
      lv_char5(5) TYPE c, " declares a character variable of length 5  
      l_packed TYPE p LENGTH 10 DECIMALS 5 VALUE '1234567890.123456789'. " evaluates to  
1,234,567,890.12346
```

: <https://riptutorial.com/ko/abap/topic/1646/>-

8:

Examples

ABAP ().

```
FIELD-SYMBOLS . ( ANY [...] TABLE ) .
```

```
FIELD-SYMBOLS: <fs_line>      TYPE any,      "generic
                <fs_struct>   TYPE knal.    "non-generic
```

```
unassigned . catch . IS ASSIGNED
```

```
IF <fs> IS ASSIGNED.
*... symbol is assigned
ENDIF.
```

```
. DATA .
```

```
DATA: w_name TYPE string VALUE `Max`,
      w_index TYPE i      VALUE 1.
```

```
FIELD-SYMBOLS <fs_name> TYPE any.
```

```
ASSIGN w_name TO <fs_name>. "<fs_name> now gets w_name
<fs_name> = 'Manni'.      "Changes to <fs_name> now also affect w_name
```

```
* As <fs_name> is generic, it can also be used for numbers
```

```
ASSIGN w_index TO <fs_name>. "<fs_name> now refers to w_index.
ADD 1 TO <fs_name>.        "w_index gets incremented by one
```

```
. UNASSIGN .
```

```
UNASSIGN <fs>.
* Access on <fs> now leads to an exception again
```

```
.
```

```
LOOP AT itab INTO DATA(wa).
```

```
* Only modifies wa_line
```

```
wa-name1 = 'Max'.
```

```
ENDLOOP.
```

```
LOOP AT itab ASSIGNING FIELD-SYMBOL(<fs>).
```

```
* Directly refers to a line of itab and modifies its values
```

```
<fs>-name1 = 'Max'.
```

```
ENDLOOP.
```

! . .

TYPE REF TO .

data .

```
DATA wa TYPE REF TO data.
```

wa CREATE DATA . TYPE .

:

```
CREATE DATA wa TYPE knal
```

- .

:

```
CREATE DATA wa TYPE (lw_name_as_string)
```

- `lw_name_as_string` .
- `CX_SY_CREATE_DATA_ERROR` *throw*.

HANDLE . HANDLE CL_ABAP_DATADESCR .

```
CREATE DATA dref TYPE HANDLE obj
```

- obj YPE erivces **S R T T**
- dref (a DataReference , ->*) (-) datacontainer

(:RTTS) .

- (RunTime Type Creation, :RTTC)
- (, :RTTI)

```
CL_ABAP_TYPEDESCR
|
|--CL_ABAP_DATADESCR
|  |
|  |--CL_ABAP_ELEMDESCR
|  |--CL_ABAP_REFDESCR
|  |--CL_ABAP_COMPLEXDESCR
|      |
|      |--CL_ABAP_STRUCTDESCR
|      |--CL_ABAP_TABLEDESCR
|
|--CL_ABAP_OBJECTDESCR
|
|--CL_ABAP_CLASSDESCR
|--CL_ABAP_INTFDESCR
```

CL_ABAP_TYPEDESCR . .

- DESCRIBE_BY_DATA
- DESCRIBE_BY_NAME
- DESCRIBE_BY_OBJECT_REF

- DESCRIBE_BY_DATA_REF

: [https://riptutorial.com/ko/abap/topic/4442/-](https://riptutorial.com/ko/abap/topic/4442/)

9:

, ASSIGN INTO . , INTO long / wide .

Examples

```
LOOP AT itab INTO wa.  
ENDLOOP.  
  
FIELD-SYMBOLS <fs> LIKE LINE OF itab.  
LOOP AT itab ASSIGNING <fs>.  
ENDLOOP.  
  
LOOP AT itab ASSIGNING FIELD-SYMBOL(<fs>).  
ENDLOOP.  
  
LOOP AT itab REFERENCE INTO dref.  
ENDLOOP.  
  
LOOP AT itab TRANSPORTING NO FIELDS.  
ENDLOOP.
```

WHERE .

```
LOOP AT itab INTO wa WHERE f1 = 'Max'.  
ENDLOOP.
```

While

ABAP WHILE -Loop . sy-index .

```
WHILE condition.  
* do something  
ENDWHILE
```

Do Loop

DO -Loop . sy-index .

```
DO.  
* do something... get it?  
* call EXIT somewhere  
ENDDO.
```

TIMES (amount i).

```
DO amount TIMES.  
* do several times  
ENDDO.
```

EXIT .

```
DO.  
  READ TABLE itab INDEX sy-index INTO DATA(wa).  
  IF sy-subrc <> 0.  
    EXIT. "Stop this loop if no element was found  
  ENDIF.  
  " some code  
ENDDO.
```

CONTINUE .

```
DO.  
  IF sy-index MOD 1 = 0.  
    CONTINUE. " continue to next even index  
  ENDIF.  
  " some code  
ENDDO.
```

CHECK CONTINUE . CONTINUE . : .

CHECK ...

```
DO.  
  " some code  
  CHECK sy-index < 10.  
  " some code  
ENDDO.
```

... ..

```
DO.  
  " some code  
  IF sy-index >= 10.  
    CONTINUE.  
  ENDIF.  
  " some code  
ENDDO.
```

: <https://riptutorial.com/ko/abap/topic/2270/>

10: / MESSAGE

```
MESSAGE . , SE91 SE91 .
```

```
& 72 .
```

Examples

```
PROGRAM zprogram MESSAGE-ID sabapdemos.
```

```
.MESSAGE-ID sabapdemos . MESSAGE .
```

```
PROGRAM zprogram MESSAGE-ID za.  
...  
MESSAGE i000 WITH TEXT-i00.
```

```
i00 . i(i000 ) MESSAGE .
```

```
za MESSAGE-ID .
```

```
PROGRAM zprogram.  
...  
MESSAGE i050(sabapdemos).
```

```
MESSAGE . sabapdemos 050 .
```

```
DATA: msgid TYPE sy-msgid VALUE 'SABAPDEMOS',  
      msgty TYPE sy-msgty VALUE 'I',  
      msgno TYPE sy-msgno VALUE '050'.
```

```
MESSAGE ID mid TYPE mtype NUMBER num.
```

```
MESSAGE MESSAGE i050(sapdemos). MESSAGE i050(sapdemos)..
```

```
& .
```

777 **of** sabapdemos :

```
Message with type &1 &2 in event &3
```

```
MESSAGE i050(sabapdemos) WITH 'E' '010' 'START-OF-SELECTION` .
```

```
Message with type E 010 in event START-OF-SELECTION . & .
```

888 sabapdemos :

```
& & & &
```

.

```
MESSAGE i050(sabapdemos) WITH 'param1' 'param2' 'param3' 'param4'.
```

param1 param2 param3 param4 .

/ MESSAGE : <https://riptutorial.com/ko/abap/topic/10691/----message->

11:

Examples

ABAP

'...'		1-255	
`...`	CString	0-255	
...	CString	0-255	

```
. CString C .
```

.

```
WRITE |Hello, { lv_name }, nice to meet you!|.
```

.

```
WRITE |The order was completed on { lv_date DATE = USER } and can not be changed|.
```

.

```
WRITE |Your token is { to_upper( lv_token ) }|.
WRITE |Version is: { cond #( when lv_date < sy-datum then 'out of date' else 'up to date' )
}|.
```

```
! ( ).
```

char ABAP CONCATENATE CONCATENATE . .

:

```
CONCATENATE var1 var2 var3 INTO result.
"result now contains the values of var1, var2 & var3 stringed together without spaces
```

ABAP && () .

```
DATA(lw_result) = `Sum: ` && lw_sum.
```

```
! ( ).
```

: <https://riptutorial.com/ko/abap/topic/3531/>

12:

- , _ .
- .
- L .
- G .
- I (import) .
- E () .
- S .
- T .

Examples

```
data: lv_temp type string.  
data: ls_temp type sy.  
data: lt_temp type table of sy.
```

```
data: gv_temp type string.  
data: gs_temp type sy.  
data: gt_temp type table of sy.
```

: <https://riptutorial.com/ko/abap/topic/6770/-->

13:

Examples

```
REPLACE .
```

```
DATA(lv_test) = 'The quick brown fox'.
REPLACE ALL OCCURRENCES OF REGEX '\wo' IN lv_test WITH 'XX'.
```

```
lv_test The quick bXXwn XXx .
```

```
FIND .
```

```
DATA(lv_test) = 'The quick brown fox'.

FIND REGEX '..ck' IN lv_test.
" sy-subrc == 0

FIND REGEX 'a[sdf]g' IN lv_test.
" sy-subrc == 4
```

```
CL_ABAP_REGEX .
```

```
DATA: lv_test TYPE string,
      lo_regex TYPE REF TO cl_abap_regex.

lv_test = 'The quick brown fox'.
CREATE OBJECT lo_regex
  EXPORTING
    pattern = 'q(...)\w'.

DATA(lo_matcher) = lo_regex->create_matcher( text = lv_test ).
WRITE: / lo_matcher->find_next( ).      " X
WRITE: / lo_matcher->get_submatch( 1 ). " uic
WRITE: / lo_matcher->get_offset( ).     " 4
```

```
matches .
```

```
IF matches( val = 'Not a hex string'
            regex = '[0-9a-f]*' ).
  cl_demo_output=>display( 'This will not display' ).
ELSEIF matches( val = '6c6f7665'
                regex = '[0-9a-f]*' ).
  cl_demo_output=>display( 'This will display' ).
ENDIF.
```

00

```
GET_SUBMATCH CL_ABAP_MATCHER / .
```

: " .

```
DATA: lv_pattern TYPE string VALUE 'type\s+(\w+)',  
      lv_test TYPE string VALUE 'data lwa type mara'.
```

```
CREATE OBJECT ref_regex  
  EXPORTING  
    pattern      = lv_pattern  
    ignore_case = c_true.
```

```
ref_regex->create_matcher(  
  EXPORTING  
    text      = lv_test  
  RECEIVING  
    matcher = ref_matcher  
  ).
```

```
ref_matcher->get_submatch(  
  EXPORTING  
    index = 0  
  RECEIVING  
    submatch = lv_smatch.
```

lv_smatch MARA .

: <https://riptutorial.com/ko/abap/topic/51113/>-

14:

Examples

IF / ELSEIF / ELSE

```
IF lv_foo = 3.  
    WRITE: / 'lv_foo is 3'.  
ELSEIF lv_foo = 5.  
    WRITE: / 'lv_foo is 5'.  
ELSE.  
    WRITE: / 'lv_foo is neither 3 nor 5'.  
ENDIF.
```

```
CASE lv_foo.  
    WHEN 1.  
        WRITE: / 'lv_foo is 1'.  
    WHEN 2.  
        WRITE: / 'lv_foo is 2'.  
    WHEN 3.  
        WRITE: / 'lv_foo is 3'.  
    WHEN OTHERS.  
        WRITE: / 'lv_foo is something else'.  
ENDCASE
```

```
CHECK false .
```

```
METHOD do_something.  
    CHECK iv_input IS NOT INITIAL. "Exits method immediately if iv_input is initial  
  
    "The rest of the method is only executed if iv_input is not initial  
ENDMETHOD.
```

```
ASSERT . ASSERT ( ASSERTION_FAILED ).
```

```
ASSERT 1 = 1. "No Problem - Program continues  
  
ASSERT 1 = 2. "ERROR
```

/

```
SWITCH COND . IF CASE , . .
```

```
COND . .
```

```
COND <type>(  
    WHEN <condition> THEN <value>  
    ...
```

```
[ ELSE <default> | throw <exception> ]
).
```

```
" Set screen element active depending on radio button
screen-active = COND i(
  WHEN p_radio = abap_true THEN 1
  ELSE 0 " optional, because type 'i' defaults to zero
).
```

```
" Check how two operands are related to each other
" COND determines its type from rw_compare
rw_compare = COND #(
  WHEN op1 < op2 THEN 'LT'
  WHEN op1 = op2 THEN 'EQ'
  WHEN op1 > op2 THEN 'GT'
).
```

SWITCH COND . **throw** . . .

```
SWITCH <type>(
  <variable>
  WHEN <value> THEN <new_value>
  ...
  [ ELSE <default> | throw <exception> ]
).
```

```
DATA(lw_language) = SWITCH string(
  sy-langu
  WHEN 'E' THEN 'English'
  WHEN 'D' THEN 'German'
  " ...
  ELSE THROW cx_sy_conversion_unknown_langu( )
).
```

: <https://riptutorial.com/ko/abap/topic/7289/-->

15:

Examples

" " .

```
DATA ls_booking TYPE flightb. " Commented text
```

* .* .

```
* DATA ls_booking TYPE flightb. Nothing on this line will be executed.
```

: <https://riptutorial.com/ko/abap/topic/1644/>

16:

- CLASS DEFINITION ABSTRACT FINAL . . .

Examples

OO

```
REPORT z_template.

CLASS lcl_program DEFINITION ABSTRACT FINAL.

    PUBLIC SECTION.

        CLASS-METHODS start_of_selection.
        CLASS-METHODS initialization.
        CLASS-METHODS end_of_selection.

ENDCLASS.

CLASS lcl_program IMPLEMENTATION.

    METHOD initialization.

    ENDMETHOD.

    METHOD start_of_selection.

    ENDMETHOD.

    METHOD end_of_selection.

    ENDMETHOD.

ENDCLASS.

INITIALIZATION.

    lcl_program=>initialization( ).

START-OF-SELECTION.

    lcl_program=>start_of_selection( ).

END-OF-SELECTION.

    lcl_program=>end_of_selection( ).
```

: [https://riptutorial.com/ko/abap/topic/10552/-](https://riptutorial.com/ko/abap/topic/10552/)

S. No		Contributors
1	ABAP	Christian, Community, gkubed, Jagger, mkysoft
2	ABAP GRID (ALV)	Achuth hadnoor, gkubed
3	ABAP	Community, Michał Majer, Thomas Matecki
4	SQL	AKHIL RAJ, gkubed
5		Community, gkubed, Michał Majer, Rahul Kadukar, Thorsten Niehues
6		maillard
7		Christian, gkubed
8		Community, gkubed
9		Christian, Community, gkubed, Stu G
10	/ MESSAGE	gkubed
11		Achuth hadnoor, Community, maillard, nexxus, Suncatcher
12		mkysoft
13		AKHIL RAJ, gkubed, maillard
14		Community, gkubed, maillard
15		4444, Christian, gkubed
16		nath