ABAP Reference

KEYWORDS

Keywords	Description
, [,], {, }	Syntax conventions, Syntax notation
***, "**	Comments
******	Arithm. Operator: Exponentiation (COMPUTE)
+, -, *, /	Arithmetical operators (COMPUTE)
->, =>, ->>, ~, ?=	Operators in ABAP Objects
ABS	Mathematical function: Absolute amount COMPUTE)
ACOS	Mathematical function: Cosine arc (COMPUTE)
ADD	Add
ADD-CORRESPONDING	Field string addition
ADJACENT DUPLICATES	Delete duplicates from internal table (DELETE)
AND	Comparison operator: and
ANY TABLE	Generic table type for internal tables
APPEND	Append line to internaltable
ASIN	Mathematical function: Sine arc (COMPUTE)
ASSIGN	Assign field symbol
AT	Event, control break, field group determination
ATAN	Mathematical function: Tangent arc
AUTHORITY-CHECK	Check authorization
AVG	Aggregate expression: Average (SELECT)
BACK	Positioning in list
BETWEEN	Relational operator: Between
BINARY SEARCH	Binary read of internaltable (READ TABLE)
BIT-NOT	Bit calculation operator: NOT (COMPUTE)
BIT-AND	Bit calculation operator: AND (COMPUTE)
BIT-OR	Bit calculation operator: OR (COMPUTE)
BIT-XOR	Bit calculation operator: AND/OR (COMPUTE)
SET BIT	Set bit of an X field
GET BIT	Read bit of an X field
BLANK LINES	Switch on blank lines in list
BREAK-POINT	Stop processing in debug mode
C	Data type for fixed-length character string
CA	Contains any characters -Relational operator forstring comparison
CALL	Call external component
CASE	Begin case distinction
CATCH	Exception handling (catch runtime errors)
CEIL	Mathematical function: Smallest whole value

CENTERED	Output format: Centered(WRITE)
CHECK	Check condition
CHECKBOX	Display as checkbox
PARAMETERS ... AS CHECKBOX	on the selection screen
WRITE ... AS CHECKBOX	in a list
CLASS	Definition of a class
CLASS-DATA	Static attributes in classes
CLASS-METHODS	Static methods in classes
CLASS-EVENTS	Static events in classes
CLASS-POOL	Introduction for type Kprograms
CLEAR	Initialize data object
CLIENT	Client handling when
DELETE ... CLIENT SPECIFIED	deleting from a database
EXPORT ... TO DATABASE ... CLIENT	Storing a data cluster
IMPORT ... FROM DATABASE ... CLIENT	Reading a data cluster
EXPORT ... TO SHARED BUFFER ... CLIENT	Storing a data cluster
IMPORT ... FROM SHARED BUFFER ... CLIENT	Reading a data cluster
INSERT ... CLIENT SPECIFIED	inserting into a database
MODIFY ... CLIENT SPECIFIED	Insert/Modify in database(s)
SELECT ... CLIENT SPECIFIED	reading from a database
UPDATE ... CLIENT SPECIFIED	updating a database
CLOSE	Close file/cursor
CN	Contains Not Only - Relational operator for character comparison:
CNT	Field groups: Number ofdifferent values
CO	Contains Only - Relational operator for character comparison:
CODE PAGE	Character set
TRANSLATE ... FROM/TOCODE PAGE	Translate character codes
COLLECT	Internal table: Add entries
COLOR	Output format: Color (FORMAT)
COMMENT	Comment on selection screen
SELECTION-SCREEN COMMENT	Generate comment
COMMIT	Close processing unit
COMMUNICATION	Data exchange
COMPUTE	Perform calculations
CONCATENATE	Concatenate character fields
CONDENSE	Condense character fields
CONSTANTS	Defing constants
CONTEXTS	Communicate contexts
CONTINUE	Exit current loop pass
CONTROLS	Define controls for visualization
CONVERT	Convert fields
COS	Mathematical function: Cosine (COMPUTE)
COSH	Mathematical function: Hyperbola cosine (COMPUTE)

COUNT	Aggregate expression: Count (SELECT)
COUNTRY	Set country ID (SET)
CP	Relational operator forcharacter comparison:
DATABASE	Contains Pattern
CREATE	Generate an object or data object
CS	Contains character - Relational operator forcharacter comparison
CURRENCY	Output format: Correct format for currency (WRITE)
CURSOR	Cursor
CLOSE	Close database cursor
FETCH NEXT CURSOR	Read lines with a database cursor
GET CURSOR FIELD	Get field name
OPEN CURSOR	Open database cursor
SET CURSOR	Position cursor
CUSTOMER-FUNCTION	Call customer enhancement
DATA	Define data
DATABASE	Data cluster
DELETE FROM DATABASE	Delete from a database table
EXPORT ... TO DATABASE	Store in a databasetable
IMPORT ... FROM DATABASE	Read from a database table
DATASET	Sequential file
CLOSE DATASET	Close file
DELETE DATASET	Delete file
EXPORT ... TO DATASET	Store data cluster in file
IMPORT ... FROM DATASET	Read data cluster from file
OPEN DATASET Open file	Open file
READ DATASET	Read from a file
TRANSFER	Output to a file
DECIMALS	Output format: Places after the decimal point - (WRITE)
DEFINE	Define macro
DELETE	Delete from tables or from objects
DEMAND	Request information from a context
DESCRIBE	Determine attributes ofdata objects
DIALOG	Call a dialog module (CALL)
DISTINCT	Duplicates
SELECT DISTINCT	Selection set without duplicates
AVG(DISTINCT ...)	Average without duplicates (SELECT)
COUNT(DISTINCT ...)	Sequential file
MAX(DISTINCT ...)	Maximum without duplicates (SELECT)
MIN(DISTINCT ...)	Minimum without duplicates (SELECT)
SUM(DISTINCT ...)	Sum without duplicates (SELECT)
DIV	Arithmetic operator: Whole number division
DIVIDE	Divide
DIVIDE-CORRESPONDINGField string division	Field string division

DO	Loop
DYNPRO Screen	Screen
DELETE DYNPRO Delete	Delete
EXPORT DYNPRO Export	Export
GENERATE DYNPRO Generate	Generate
IMPORT DYNPRO Import	Import
SYNTAX-CHECK FOR DYNPRO Check	Check
EDITOR-CALL	Call editor
ELSE	Query
ELSEIF	Query
END-OF-DEFINITION	End of a macro definition
END-OF-PAGE	Event: End of page handling in lists
END-OF-SELECTION	Event: After processingof all records in a LDB
ENDAT	End of an event introduced by AT
ENDCASE	End of case distinction
ENDCATCH	End of exception handling
ENDDO	End of a DO loop
ENDEXEC	End of a Native SQL statement
ENDFORM	End of a subroutine
ENDFUNCTION	End of a function module
ENDIF	End of a query
ENDINTERFACE	End of an interface definition
ENDLOOP	End of a LOOP
ENDMODULE	End of a module definition
ENDON	End of a conditional statement
ENDPROVIDE	End of a PROVIDE loop
ENDSELECT	End of a SELECT loop
ENDWHILE	End of a WHILE loop
EQ	Relational operator: Equals
EXEC SQL	Native SQL statement
EXIT	Exit loop or terminate processing
EXP	Mathematical function: Exponential function
EXPONENT	Output format: Exponentdisplay (WRITE)
EXPORT	Export data
EXTENDED CHECK	Switch extended syntax check on/off (SET)
EXTRACT	Generate extract dataset
FETCH	Read line from a database table
FIELD-GROUPS	Define field groups
FIELD-SYMBOLS	Define field symbols
FLOOR	Mathematical function:Largest whole value
FORM	Define subroutine
FORMAT	Output format for lists
FOR UPDATE	Read database table with lock (SELECT)

FRAC	Mathematical function: Fraction (COMPUTE)
FREE	Release resources no longer needed
FUNCTION	Define function module
CALL FUNCTION	Call function module
FUNCTION-POOL	Introduction for type Fprograms
GE	Relational operator: Greater than or equal
GENERATE	Generate a program or screen
GET	Event, read settings
GT	Relational operator: Greater than
HASHED TABLE	Table type for internalhashed tables
HEADER LINE	Define an internal table with header line (DATA)
HELP-ID	Help ID for F1 help
DESCRIBE FIELD ... HELP-ID	Determine help ID
HELP-REQUEST	Self-programmed help (F1)
PARAMETERS ... HELP-REQUEST	for parameters
SELECT-OPTIONS ... HELP-REQUEST	for selection options
HIDE	Store line information
HOTSPOT	Output format: Hotspot,interaction by simple - mouse click (FORMAT)
ICON	Icons in lists
IF	Query
IMPORT	Import data or a screen
IN	Relational operator: Selection criterion
INCLUDE	Include program components
INDEX	Line index in an internal table
INDEX TABLE	
DELETE ... INDEX	Delete line
INSERT ... INDEX	Insert line
MODIFY ... INDEX	Modify line
READ TABLE ... INDEX	Read line
INFOTYPES	Declare HR info type
INITIAL	Relational operator: Initial value
INITIAL SIZE	Define an internal table type (TYPES)
INITIALIZATION	Event: Before display of the selection screen
INPUT	Output format: Ready for input (FORMAT)
INSERT	Insert into tables or objects
INTENSIFIED	Output format: Intensified (FORMAT)
INTERFACE	Definition of an interface
INTERFACES	Class component interface
INTERFACE-POOL	Introduction fortype J programs
INVERSE	Output format: Inverse (FORMAT)
IS	Relational operator
IS ASSIGNED	Relational operator: Is the field symbol assigned?
IS INITIAL	Relational operator: Initial value

IS REQUESTED	Relational operator: Existence of a formal
parameter	
JOIN	Join (SELECT)
LANGUAGE	Set language for text elements (SET)
LE	Relational operator: Less than or equal
LEAVE	Leave processing
LEFT-JUSTIFIED	Output format: Left-justified (WRITE)
LIKE	Use an existing field as areference
TYPES ... LIKE	Create a type
DATA ... LIKE	Create a field
LINE	Line in a list
MODIFY LINE	Modify line
READ LINE	Read line
LINE-COUNT	Number of lines per page (NEW-PAGE)
LINE-SIZE	Line size (NEW-PAGE)
LIST-PROCESSING	List processing (LEAVE)
LOAD	Load program componentsin internal table
LOAD-OF-PROGRAM	Execution at load time
LOCAL	Rescue actual parameters of a subroutine
LOCAL COPY	Assign local copy to a field symbol
LOCALE	Set text environment (SET)
SET LOCALE	Set text environment
GET LOCALE	Determine text environment
LOG	Mathematical function: Natural logarithm (COMPUTE)
Logical condition	
SELECT ... WHERE	when reading database tables
UPDATE ... WHERE	when changing database tables
DELETE ... WHERE	when deleting fromdatabase tables
SELECT ... FROM ... ON	when reading usinga join
LOG10	Mathematical function: Base 10 logarithm (COMPUTE)
LOOP	Loop
LT	Relational operator: Less than
M	Relational operator: Byte contains zeros and ones
MARGIN	List output: Distance from edge (SET)
MATCHCODE	Matchcode handling
PARAMETERS ... MATCHCODE	for parameters
SELECT-OPTIONS ... MATCHCODE	for selection options
MAX	Aggregate expression: Maximum (SELECT)
MEMORY	ABAP/4 memory
EXPORT ... TO MEMORY	Roll out data to memory
IMPORT ... FROM MEMORY	Restore data from memory
MESSAGE	Output message
MESSAGE-ID	Specify message class (REPORT)

METHOD	Definition of a method
METHODS	Class component method
MIN	Aggregate expression: Minimum (SELECT)
MOD	Arithmetic operator: Remainder after division
	(COMPUTE)
MODIFY	Modify tables or objects
MODULE	Flow logic: Module
MOVE	Assignment
MOVE-CORRESPONDING	Component-by-component assignment
MULTIPLY	Multiply
MULTIPLY-CORRESPONDING	Field string multiplication
NA	Relational operator forcharacter comparison:
	Contains not any characters
NE	Relational operator: Not equal
NEW-LINE	List processing: New line
NEW-PAGE	List processing: New page
NODES	Interface work area forlogical databases
NO-GAP	Output format: Leave nogaps (WRITE)
NO-HEADING	Display no column headers (NEW-PAGE)
NO-SCROLLING	Do not scroll line (NEW-LINE)
NO-SIGN	Output format: No preceding signs (WRITE)
NO-TITLE	Do not display standardpage header (NEW-PAGE)
NO-ZERO	Output format: No leading zeros (WRITE)
NON-UNIQUE	Defines an
TYPES	internal table type
DATA	internal table object
NP	Relational operator forcharacter comparison:
	Does not contain pattern
NS	Relational operator forcharacter comparison:
	Does not contain character
O	Relational operator: Byte positions occupied by1
OBJECT	External object
CREATE OBJECT	Generate
FREE OBJECT	Release
OCCURS	Defines an
TYPES	internal table type
DATA	internal table object
ON CHANGE	Control break
OPEN	Open file/cursor
OR	Relational operator: OR
ORDER BY	Sort table rows (SELECT)
OVERLAY	Overlay character fields
PACK	Conversion

PARAMETER	Parameter in global SAP memory
GET	Read parameter
SET	Set parameter
PARAMETERS	Define report parameters
PERFORM	Execute subroutine
PF-STATUS	Set GUI status
POSITION	List processing: Defineoutput position
PRINT	Print formatting (NEW-PAGE)
PRINT-CONTROL	Define print format
PRIVATE	Class area not visible from outside
PROGRAM	Introduction for type Mand S programs
LEAVE PROGRAM	Leave program
PROPERTY	Object property
GET PROPERTY	Get property
SET PROPERTY	Set property
PROVIDE	Internal tables: Interval-related processing
PUT	Trigger event
RADIOBUTTON	Radio button (PARAMETERS)
RAISE	Raise exceptions and events
RAISING	Raise error message in function module
RANGES	Define internal table for selection criterion
READ	Read tables or objects
RECEIVE	Receive results (RFC)
REFRESH	Delete internal table
REFRESH CONTROL	Initialize control
REJECT	Do not process current database line further
REPLACE	Replace characters
REPORT	Introduction for type 1programs
DELETE REPORT	Delete program
EDITOR-CALL FOR REPORT	Call ABAP program editor
INSERT REPORT	Insert program in library
READ REPORT	Read program
RESERVE	List processing: Conditional new page
RESET	Output format: Reset all formats (FORMAT)
RIGHT-JUSTIFIED	Output format: Right justified (WRITE)
ROLLBACK	Roll back database changes
ROUND	Output format: Scaled (WRITE)
RTTI	Runtime type identification
RUN TIME ANALYZER	Activate/Deactivate runtime analysis (SET)
SCAN	Analyze ABAP/4 source code
SCREEN	Screen
CALL SCREEN	Call screen
SET SCREEN	Set next screen

LEAVE SCREEN	Leave screen
LEAVE TO SCREEN	Branch to a screen
LOOP AT SCREEN	Loop through screen fields
MODIFY SCREEN	Modify screen fields
SCROLL	List processing: Scroll
SCROLL-BOUNDARY	List processing: Fix lead columns (SET)
SEARCH	Find character
SELECT	Read database table
SELECT-OPTIONS	Define selection criterion
SELECTION-SCREEN	Design selection screen
AT SELECTION-SCREENEvent:	After editing ofselection screen
SHARED BUFFER	Cross-transaction application buffer
DELETE FROM SHARED BUFFER	delete from application buffer
EXPORT ... TO SHARED BUFFER	Store data in application buffer
IMPORT ... FROM SHARED BUFFER	Read data from application buffer
SELECTION-TABLE	Selection table (SUBMIT)
SET	Set different processing parameters
SHIFT	Move character
SIGN	Mathematical function: Sign (COMPUTE)
SIN	Mathematical function: Sine (COMPUTE)
SINGLE	Select single record (SELECT)
SINH	Mathematical function: Hyperbola sine (COMPUTE)
SKIP	List processing: Outputblank line
SORT	Sort internal table or extract dataset
SORTED TABLE	Table type for internaltables that are always kept
SPLIT	Split character fields
SQRT	Mathematical function: Square root (COMPUTE)
STANDARD TABLE	Table type for standardinternal tables
START-OF-SELECTION	Event: Before first access to LDB
STATICS	Define static data
STOP	Stop data selection (LDB)
STRING	Data type for variable-length character sequence
STRLEN	Character function: Current length (COMPUTE)
STRUCTURE	Data structure
INCLUDE STRUCTURE	Use structure
SUBMIT	Program call
SUBTRACT	Subtract
SUBTRACT-CORRESPONDING	Field string subtraction
SUM	Calculate control total
SELECT ... SUM	Aggregate expression: Total
SUPPLY	Supply context key fields
SUPPRESS DIALOG	Suppress dialog
SYMBOL	Output as symbol (WRITE)

SYNTAX-CHECK	Syntax check for programs and screens
SYNTAX-TRACE	Syntax check log
SYSTEM-CALL	Call to various system services
SYSTEM-EXCEPTIONS	Catch runtime errors (CATCH)
TABLE LINE	Unstructured lines in internal tables
TABLE_LINE	Unstructured lines in internal tables
TABLES	Declare database table
TABLE	Set or array operations for database tables
DELETE ... FROM TABLE	Delete block of lines
INSERT ... FROM TABLE	Insert block of lines
MODIFY ... FROM TABLE	Insert/update block of lines
UPDATE ... FROM TABLE	Update block of lines
SELECT ... INTO TABLE	Copy block of lines to internal table
TAN	Mathematical function: Tangent (COMPUTE)
TANH	Mathematical function: Hyperbola tangent (COMPUTE)
TEXT	Locale-specific
CONVERT TEXT	Set format
SORT itab AS TEXT	Sort an internal table
SORT AS TEXT	Sort an extract dataset
TEXTPOOL	Text elements
DELETE TEXTPOOL	Delete
INSERT TEXTPOOL	Insert
READ TEXTPOOL	Read
TIME	Time measurement
GET RUN TIME	Get runtime
GET TIME	Get time
SET RUN TIME ANALYZER	Switch runtime analysison/off
TIME STAMP	Time stamp
GET TIME STAMP	Get time stamp
CONVERT TIME STAMP	Convert time stamps to date/time
WRITE f TIME ZONE	Output of time stamps to lists
TITLEBAR	Set screen title (SET)
TOP-OF-PAGE	Event: Top of page handling in lists
TRANSACTION	SAP transaction
CALL TRANSACTION	Call
LEAVE TO TRANSACTION	Leave to
TRANSFER	Output to file
TRANSLATE	Character conversion incharacter fields
TRANSPORTING	Selective field transport
MODIFY ... TRANSPORTING	Modify lines of an internal table
READ ... TRANSPORTING	Read lines of an internal table
LOOP ... TRANSPORTING	Loop through an internal table
TRUNC	Mathematical function: Whole number part (COMPUTE)

TYPE	Define a type
TYPES ... TYPE	Define a type
DATA ... TYPE	Define a field
TYPE-POOL	Introduction for type Tprograms
TYPE-POOLS	Include type group
TYPES	Define types
ULINE	List processing: Underscore
UNDER	Output format: One under the other (WRITE)
UNIQUE	Define an
TYPES	internal table type
DATA	internal table object
UNIT	Output format: Unit (WRITE)
UNPACK	Conversion
UPDATE	Update database table
USER-COMMAND	List processing: Execute command immediately (SET)
USING	Use parameter or format
USING	Parameter of a subroutine
USING EDIT MASK	Output format: Use template (WRITE)
VALUE-REQUEST	Self-programmed value help(F4)
PARAMETERS ... VALUE-REQUEST	for parameters
SELECT-OPTIONS ... VALUE-REQUEST	for selection options
WHEN	Case distinction
SELECT ... WHERE	when reading from databasetables
UPDATE ... WHERE	when changing database tables
DELETE ... WHERE	when deleting database tables
LOOP AT ... WHERE	when looping at internal tables
DELETE ... WHERE	when deleting from internal tables
WHILE	Loop
WINDOW	List processing: Outputin window
WITH-TITLE	Output standard page header (NEW-PAGE)
WORK	Processing unit
COMMIT WORK	Close unit
ROLLBACK WORK	Close unit, but undo changes
WRITE	List processing: Output
WRITE TO	Correct type output in a variable
X	Data type for fixed-length byte sequence
XSTRING	Data type for variable-length byte sequence
Z	Relational bit operator: Bit positions occupiedby

STATEMENTS:

Statement is lines of codes which instruct an order. In ABAP, every statement always close by dot(.) at the end.

These are few examples of statements:

>> Source Code

```
write 'Hello, World !!'.
```

>> Display

abap_syntax0001

Statements are always begined by ABAP Keywords (Example: TYPES , DATA , write dst). Keyword determine statement Catagory.
These are reference of statement at ABAP Program:

1. Data Types
2. Declaration of Data Type

This statement use to define a data type or object as variable of program.
Example :

```
TYPES <variable_name> TYPE <type>.
DATA <variable_name> TYPE <type>.
```

3. Compound Statement

Compound Statement is group of statements of a block coding. It's also called as "Block Statement".
Example :

```
FORM <form_name>.
Statement 1.
Statement 2.
...
Statement n.
ENDFORM.

FUNCTION <function_name>.
Statement 1.
Statement 2.
...
Statement n.
ENDFUNCTION.

MODULE <module_name>.
Statement 1.
Statement 2.
...
Statement n.
ENDMODULE.
```

4. Control Statement

Control Statement is used to control statement flow according to conditions that siutable to algoritma of program we want.
Example :

```
IF <condition1>.
```

```
<statement block>
ELSEIF <condition2>
<statement block>.
ELSEIF <condition3>.
<statement block>
.....
ELSE.
<statement block>
ENDIF.
```

5. Operator Statement

Operator Statement is keyword for doing some process of type/object that declared in a program.
Example :

```
WRITE <statement>.
ADD <variable 1> to <variable 2>.
```

6. Database Statement

Database Statement is statement that used for accessing database. In ABAP, there are 2 statement for accessing database, these are Open SQL and Native SQL.

7.

In ABAP, structure is usually called as Workarea or Header Line. In other language program, structure is equal as one dimension array.

>> Source Code

```
*First Style
types : begin of ty_wa,
        a type i,
        b type i,
        hasil type p decimals 2,
      end of ty_wa.

Data : l_wa1 type ty_wa.          " Sturcture (Workarea)


*Second Style
data: begin of l_wa2,
        a type i,
        b type i,
        hasil type p decimals 2,
    end of l_wa2.              " Sturcture (Workarea)


parameter p_a like l_wa1-a default 12.
parameter p_b like l_wa1-b default 14.

START-OF-SELECTION.
l_wa1-a = p_a.
l_wa1-b = p_b.

l_wa1-hasil = l_wa1-a / l_wa1-b.

write : / 'WA dari variabel A = ', l_wa1-a,
      / 'WA dari variabel B = ', l_wa1-b,
```

```
    / 'WA dari variabel Hasil = ', l_wa1-hasil.
```

>> Display :

Selection Screen

Result

```
WA dari variabel A =          12
WA dari variabel B =          14
WA dari variabel Hasil =              0,86
```

8.

In ABAP, tables are usually called as internal table (temporary table) or table (real). In other language program, tables are equal as multi dimension array.

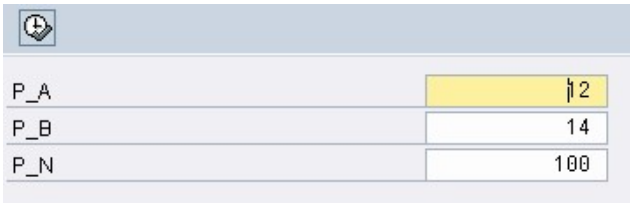
>> Source Code

```
TYPES : BEGIN OF ty_wa,
        a TYPE i,
        b TYPE i,
        hasil TYPE p DECIMALS 2,
       END OF ty_wa.

* Internal Tables Declaration
DATA : gi_itab TYPE STANDARD TABLE OF ty_wa WITH HEADER LINE.

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

DATA : lv_counter TYPE i,
     lv_puluhan TYPE i.

PARAMETER p_a LIKE gi_itab-a DEFAULT 12.
PARAMETER p_b LIKE gi_itab-b DEFAULT 14.
PARAMETER p_n TYPE i DEFAULT 100.


START-OF-SELECTION.
* Insert data to itab
CLEAR lv_counter.
DO p_n TIMES.
lv_counter = lv_counter + 1.

gi_itab-a = p_a + lv_counter.

lv_puluhan = gi_itab-a MOD 10.
IF lv_puluhan <> 0.
  gi_itab-b = p_b + lv_counter.
  gi_itab-hasil = p_a + p_b.
  APPEND gi_itab.
ENDIF.
ENDDO.
```

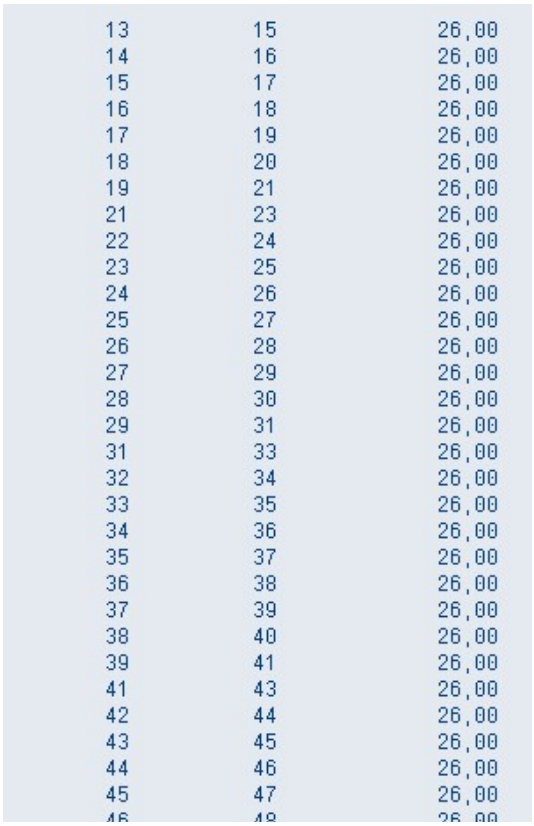
```
*Display itab data
LOOP AT gi_itab.
WRITE : / gi_itab-a, gi_itab-b, gi_itab-hasil.
ENDLOOP.
```

>> Display :

Selection Screen



Display



LITERALS

Literal is the way to declare directly a value in language program.
There are 2 type of literal at ABAP, these are:

1.	Numeric Literal

Example :

```
a = 10.
b = 100.
```

2. Text Literal

Text Literal initiate dan close by sitation (').

Example :

```
write 'Hello, World !!'.
```

Chained Statements

ABAP has classical language program caracteristic, that is, in every statement in main program or in block statement will be executed in sequence from the top to bottom.

Example :

No.	Statement
1	Data : a TYPE i,
	b TYPE i,
	c TYPE p DECIMALS 2.
2	START-OF-SELECTION.
3	a = 10.
4	b = 100.
5	c = a / b.
6	WRITE : / 'Hitung : ', a, '/', b, '=', c.

From the example above, ABAP will execute from number 1 to 6.

COMMENTS

Inside the source code, comment is explanation or infomation for statement or block statement. It's important to give explanation about steps or algoritm of ABAP program. If someone else want edit that program, it's not difficult to study the logic algoritm in code of program.

There are 2 kind of comment in ABAP, these are:

1. Use key star (*), and locate it in the first character at line program..

Example :

```
FORM fm_process_data.
*Get other requirement information
LOOP AT gi_header.
*   Get Material Description
  READ TABLE gi_makt WITH KEY matnr = gi_header-matnr.
  IF sy-subrc = 0.
    gi_header-maktx = gi_makt-maktx.
  ENDIF.
```

```
*   Get Measurement Text
  READ TABLE gi_t006a WITH KEY msehi = gi_header-meins.
  IF sy-subrc = 0.
    gi_header-mseht = gi_t006a-mseht.
  ENDIF.

  MODIFY gi_header.
ENDLOOP.

ENDFORM.
```

2. Use double sitation ("), it's free to locate it.

Example :

```
FORM fm_process_data.
"Get other requirement information
LOOP AT gi_header.
  "Get Material Description
  READ TABLE gi_makt WITH KEY matnr = gi_header-matnr.
  IF sy-subrc = 0.
    gi_header-maktx = gi_makt-maktx.
  ENDIF.

  "Get Measurement Text
  READ TABLE gi_t006a WITH KEY msehi = gi_header-meins.
  IF sy-subrc = 0.
    gi_header-mseht = gi_t006a-mseht.
  ENDIF.

  MODIFY gi_header.
ENDLOOP.

ENDFORM.
```