```
*&---------------------------------------------------------------------*
*& Report zdpm_practice1
*&---------------------------------------------------------------------*
*&
*&---------------------------------------------------------------------*
REPORT zdpm_practice1.


" subroutines ( from...endform ---> perform ).

*They are used to modularize the code and used for the code reusability
*as it has both global and local scope.
*we can have nested subroutines

*we can pass the values in the subroutine by 3 ways
*1. pass by value.
*2. pass by reference
*3. pass by value and return.



* global declarations.
DATA: n1  TYPE int4 VALUE 1,
      n2  TYPE int4 VALUE 2,
      tot TYPE int4.

data: lt_mara type table of mara,
      ls_mara type mara.

START-OF-SELECTION.
  PERFORM sum USING n1 n2.  " calling the sum subroutine. --> by value

  PERFORM sum1 USING n1 n2.  " calling the sum1 subroutine. --> by reference.

  PERFORM sum2 USING n1 n2 CHANGING tot.  " calling the sum1 subroutine. --> by reference and return.
  WRITE /: tot.

  PERFORM diff USING n1 n2 CHANGING tot.  " calling the sum1 subroutine. --> by value and return.
  WRITE /: tot.

  PERFORM exit_handler USING n1. " using exit keyword.

  PERFORM continue_handler USING n1 n2 CHANGING tot. " using continue keyword.
  IF tot > 0. WRITE /: tot. ENDIF.

  PERFORM check_stmt USING n1 n2 CHANGING tot. " using check statement.
  IF tot > 0. WRITE /: tot. ENDIF.

skip 4.
uline 2.
  PERFORM data_static.  "difference between data and static
skip 3.
uline 2.

perform get_Data. "local used inside but must be declared before.
write:/'after local calling', n1.

perform get_table_data tables lt_mara. " fetch the mara table data.

perform get_wa_data using ls_mara. " fetch the mara table workarea.
end-of-SELECTION.



* 1. pass by value.
FORM sum USING VALUE(p1)
               VALUE(p2).

  DATA : total TYPE int4. " local variable with local scope.
  total =  p1 + p2 .
  WRITE /: total.
ENDFORM.
```

```


* 2. pass by reference
FORM sum1 USING p1 TYPE int4
                p2 TYPE int4.

  DATA: total TYPE int4.
  total = p1 + p2.
  WRITE /: total.
ENDFORM.


* 3. pass by reference and return --> changing parameter
FORM sum2 USING p1 TYPE int4
                p2 TYPE int4
          CHANGING
                total TYPE int4.

  total = p1 + p2.
ENDFORM.


* 3.1 pass by value and return --> changing parameter
FORM diff USING  VALUE(p1)
                 VALUE(p2)
          CHANGING
                total TYPE int4.

  total = p1 - p2.
ENDFORM.


* additional thing need to know in subroutines.

* 1. Exit
FORM exit_handler USING p1 TYPE int4.

  IF p1 > 10.
    p1 = 12.
  ELSE.
    WRITE /'Exit'.
    EXIT.
  ENDIF.

ENDFORM.

* 2. CONTINUE. --> allowed only within loops (DO, WHILE, LOOP, SELECT).
FORM continue_handler USING p1 TYPE int4
                            p2 TYPE int4
                         CHANGING
                           total TYPE int4.
  DO 1 TIMES.
    IF p1 > 10 AND p2 < 10.
      total = p1 + p2.
    ELSE.
      WRITE: /'continue'.
      CONTINUE.
    ENDIF.
  ENDDO.
ENDFORM.


* 3. check statement --> verify data based on condition
FORM check_stmt USING VALUE(p1)
                      VALUE(p2)
                CHANGING VALUE(total).
  WRITE /:'Check'.
  CHECK p1 GE 10.
  CHECK p2 LE 10.

  total = p1 + p2.

ENDFORM.

```

```

* 4. data and static data in subroutines.
" it can be only used in forms, functions and methods.

FORM data_static.

data p1 type c length 5  value 'ABCDE'.
STATICs val TYPE c LENGTH 5 VALUE '12345'. " static declaration.

  WRITE : / 'original data', p1.
  p1 = '432'.
  WRITE : / 'Changed data', p1.
  WRITE : / 'Changed data', p1.


  WRITE: / 'original value', val.
  val = 'GHI'.
  WRITE: / 'changed value', val.
  WRITE: / 'changed value', val.

ENDFORM.

* 5.  use of global and local work-area in subroutine.
form get_data.
local n1.
n1 = '100'.
 write: / 'get_Data local :' , n1.
ENDFORM.


* 6 . passing a table in subroutine
form get_table_data tables lt_mara.
  select * from mara into table lt_mara.
endform.

* 7 . passing a workarea assiging field symbol in subroutine
form get_wa_data using wa type any.
  FIELD-SYMBOLS: <fs_wa>.
  select single * from mara into wa.
  ASSIGN COMPONENT 'MATNR' of STRUCTURE wa to <fs_wa>.
  write / <fs_wa>.
endform.
```