Lecture 18

External Subprograms: Passing Parameters

Text:
Chapter 22 (5^{th} edition, continued)
Chapter 23 (4^{th} edition, continued)

Parameter Passing

- agree in number, type and order
- formal parameters (names given in subprogram)
- actual parameters (information passed to subprogram)
- pass by value (copy of actual given to formal)
- pass by reference (address of actual given to formal)

Example:

```
program Add;
procedure Add2( FirstOP, SecondOP: integer;
                var Answer: integer);
begin
   Answer := FirstOP+SecondOP
end; {Add2}
var X,Y,Z: integer;
begin {main}
   X:= 3;
   Y:= 4;
   Add2(X,Y,Z);
   Writeln(X,Y,Z)
end.  {main}
```

Main program variables:

X	Y	Z

Subprogram variables:

FirstOP	SecondOP	Answer

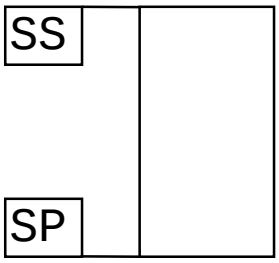
Parameters are passed on the **stack.**

The stack is common to both programs.

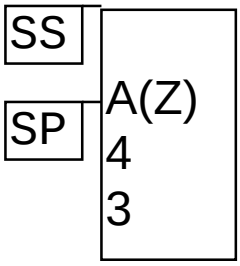
MAIN:

```
    PUSH    X
    PUSH    Y
    PUSH    Z
```

Before



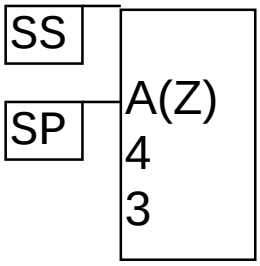
After



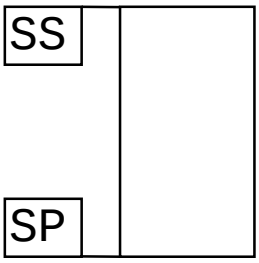
SUB:

```
    POP     Answer
    POP     SecondOP
    POP     FirstOP
```

Before



After



Before

X	Y	Z
3	4	?

FirstOP	SecondOP	Answer
?	?	?

After

X	Y	Z
3	4	?

FirstOP	SecondOP	Answer
3	4	A(Z)

Note:

This is simplified, as the CS and IP are also pushed on the stack when the subprogram is called.

```
                page 60,132
TITLE   P23MAIN6 (EXE)  Passing parameters
        EXTRN   P23SUB6:FAR
; -----------------------------------------------
STACKSG SEGMENT PARA STACK 'Stack'
        DW      64 DUP(?)
STACKSG ENDS
; -----------------------------------------------
DATASG  SEGMENT PARA 'Data'
QTY     DW      0140H
PRICE   DW      2500H
DATASG  ENDS
; -----------------------------------------------
CODESG  SEGMENT PARA PUBLIC 'Code'
BEGIN   PROC    FAR
        ASSUME CS:CODESG,DS:DATASG,SS:STACKSG
        MOV     AX,DATASG
        MOV     DS,AX
        PUSH    PRICE
        PUSH    QTY
        CALL    P23SUB6    ;Call subprogram
        MOV     AX,4C00H   ;Exit to DOS
        INT     21H
BEGIN   ENDP
CODESG  ENDS
        END     BEGIN
```

The parameters are passed through the stack

```
         page    60,132
TITLE    P23SUB6 Called subprogram
CODESG   SEGMENT PARA PUBLIC 'Code'
P23SUB6  PROC    FAR
         ASSUME CS:CODESG
         PUBLIC P23SUB6
         PUSH    BP          ;save BP
         MOV     BP,SP       ;point to stack
         MOV     AX,[BP+8]   ;Get price
         MOV     BX,[BP+6]   ;Get quantity
         MUL     BX          ;DX:AX = product
         POP     BP          ;restore BP
         RET     4
P23SUB6  ENDP
CODESG   ENDS
         END
```

The BP is saved and replaced with the SP. Now we point directly into the stack.

The parameters are moved off the stack and into the registers.

The RET instruction will pop the CS and IP off the stack, but we must also get rid of the parameters. The “4” tells the RET to pop four additional bytes (the two words which were the parameters QTY and PRICE).

The main program (MAIN.ASM)

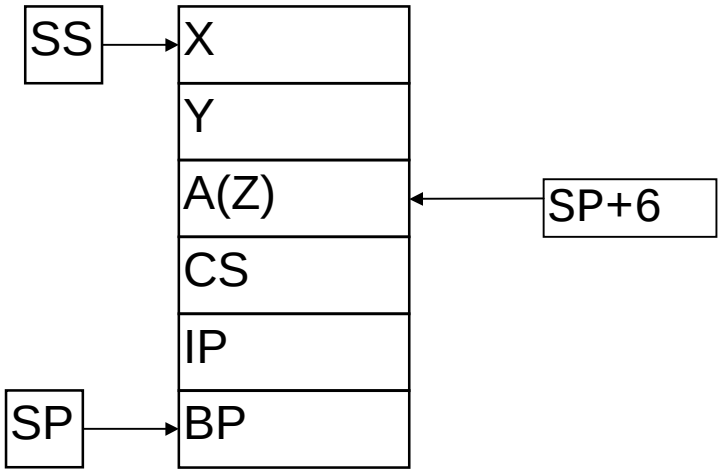
Passes A and B by value by pushing them on the stack.
Passes C by reference by passing its offset on the stack.

```
        title    Pass parameters by ref.
        EXTRN    ADDSUB:FAR
STACKSG SEGMENT  PARA  STACK 'Stack'
        DW       32
STACKSG ENDS
DATASG  SEGMENT  PARA 'Data'
A       DW       3
B       DW       4
C       DW       9
DATASG  ENDS
CODESG  SEGMENT  PARA 'Code'
BEGIN   PROC     FAR
        ASSUME   SS:STACKSG,DS:DATASG,CS:CODESG
        MOV     AX,DATASG
        MOV     DS,AX
;
        PUSH    A         ;pass A by value
        PUSH    B         ;pass B by value
        LEA     AX,C      ;get addr of C and
        PUSH    AX        ; pass it by reference
        CALL    ADDSUB
;
        MOV      AX,4C00H
        INT      21H
BEGIN   ENDP
CODESG  ENDS
        END      BEGIN
```

The subprogram finds the parameters in the stack. A is at SP+10 and B is at SP+8. It adds them, and stores the answer at the address obtained using the offset for the answer (at SP+6) combined with the DS register which still points to the data segment of the main program.

```
         page    60,132
         title   Subprogram w/va&ref param.
         PUBLIC   ADDSUB
CODESG   SEGMENT  PARA 'Code'
ADDSUB   PROC     FAR
         ASSUME   CS:CODESG
         PUSH     BP
         MOV      BP,SP
         MOV      AX,[BP+10]   ;GET FIRSTOP
         ADD      AX,[BP+8]    ;ADD SECONDOP
         MOV      BX,[BP+6]  ;GET ADDR OF ANSWER
         MOV      [BX],AX    ;PUT ANSWER IN DS:BX
         POP      BP
         RET    6
ADDSUB   ENDP
CODESG   ENDS
         END
```

The Stack:



Exercises - Lecture 18

Consider an external subprogram that will determine the minimum, maximum and sum of the numbers in an array. The subprogram should have five parameters:

- The address of the array
- The number of words in the array
- The address of a word in which to put the minimum
- The address of a word in which to put the maximum
- The address of a word in which to put the sum.

Complete the following main program which will set up the parameters and do the call:

```
        EXTRN   _________________
STACKSG SEGMENT  PARA  STACK 'Stack'
        DW       32
STACKSG ENDS
DATASG  SEGMENT  PARA 'Data'
Numbers DW       4,12,7,19,32,15,46,25,8,1
Asize   DW       10
Minimum DW       ?
Maximum DW       ?
Sum     DW       ?
Minimum
DATASG  ENDS
CODESG  SEGMENT  PARA 'Code'
BEGIN   PROC     FAR
        ASSUME   SS:STACKSG,DS:DATASG,CS:CODESG
        MOV     AX,DATASG
        MOV     DS,AX
;
```

```
        MOV      AX,4C00H
        INT      21H
BEGIN   ENDP
CODESG  ENDS
```

Write the external subprogram:

```
        PUBLIC  _________________
CODESG  SEGMENT PARA 'Code'
ADDSUB  PROC    FAR
        ASSUME  CS:CODESG
        PUSH    BP
        MOV     BP,SP




        RET     _____________
ADDSUB  ENDP
CODESG  ENDS
        END     _____________
```