FAST-NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES PESHAWAR CAMPUS

NAME: Junaid Saeed

ROLL NO: 22P-9200

SECTION: BS(CS)-3D

COURSE: Coal

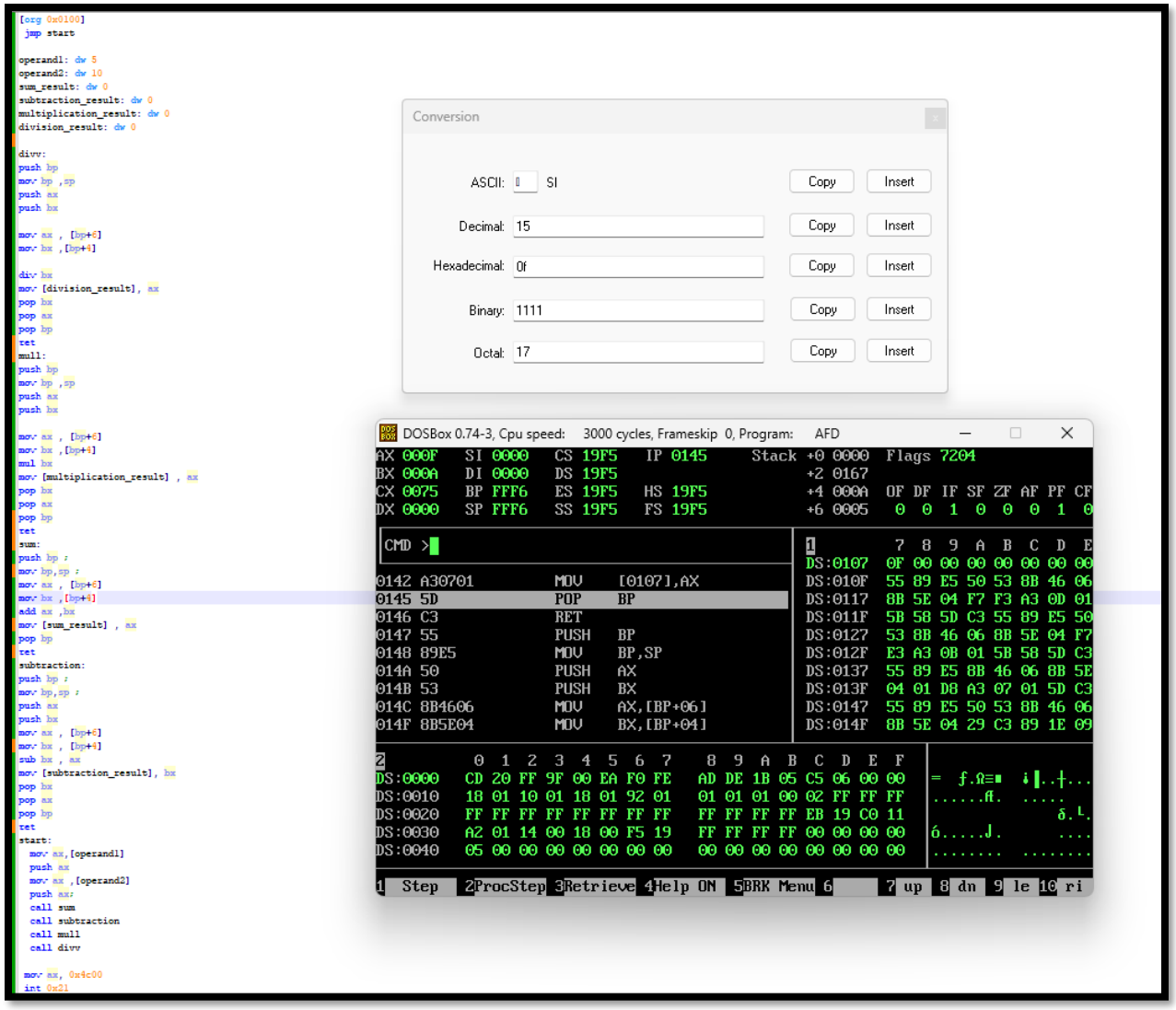
ASSIGNMENT # 3

SUBMITTED TO: Dr . Usman Abbasi

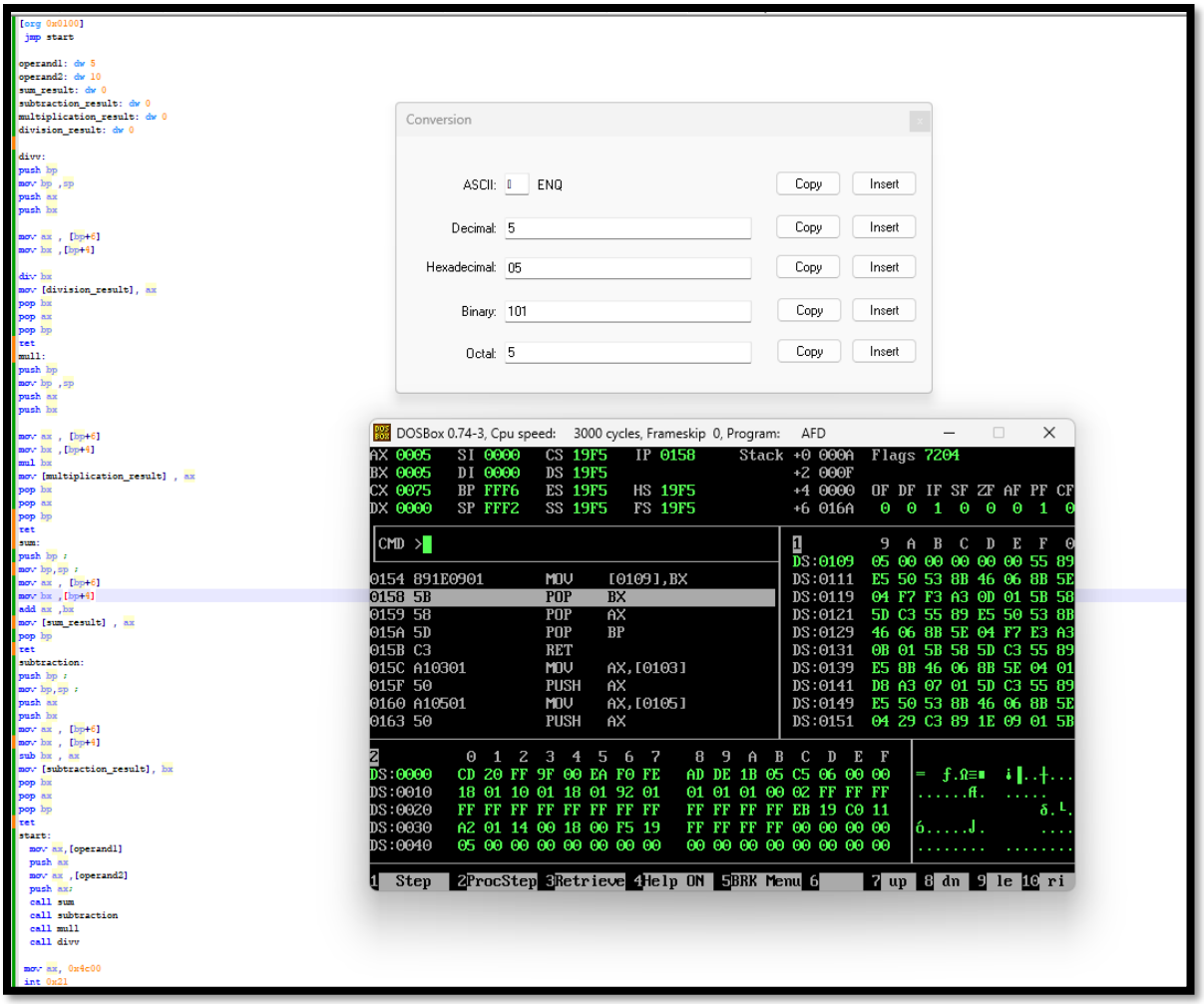
Question # 1:

→ In (question 1) i am using subroutines for add, subtract, multiply and divide and then passing numbers as a parameters in the stack and also storing the results for each of the operation in the variables . after all work then popping back them . and rest of the screenshots and code visualization is given below.

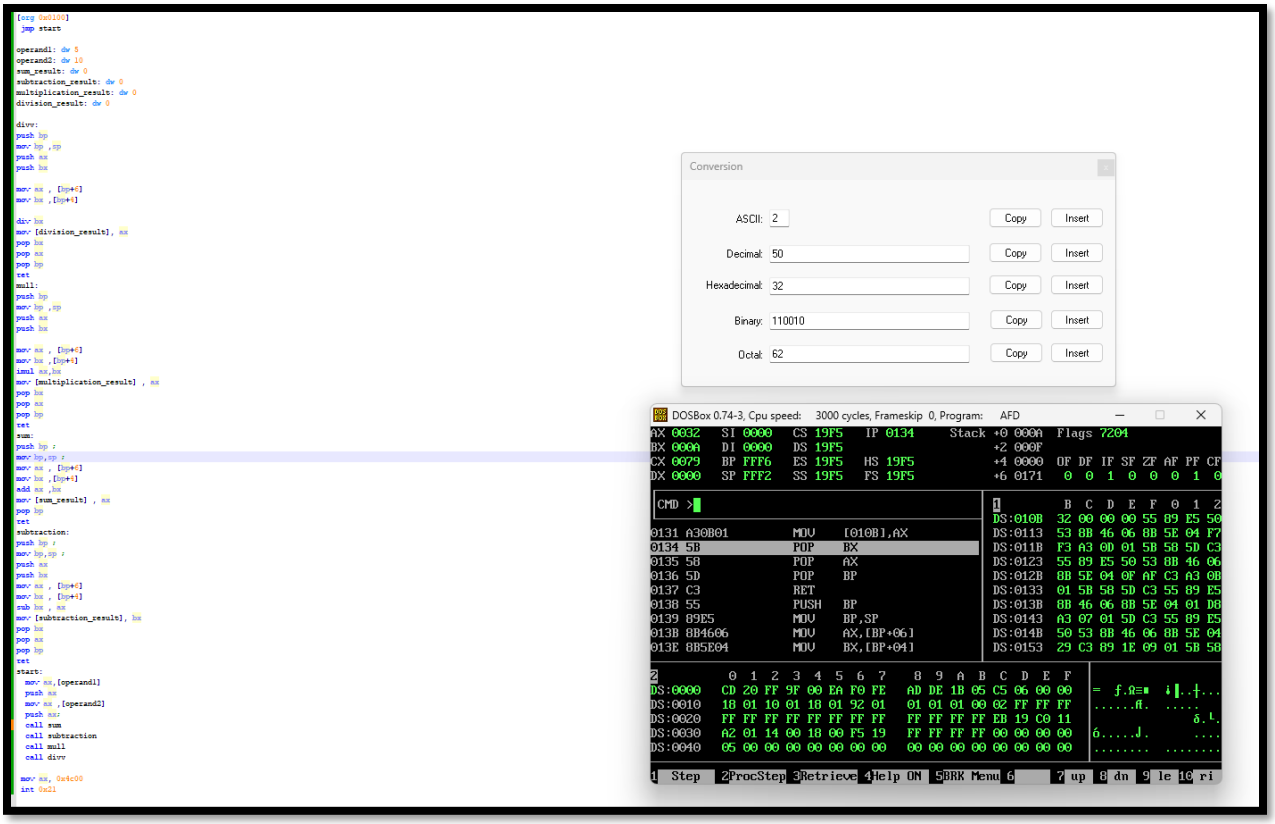
```
[org 0x0100]
 jmp start

operand1: dw 5
operand2: dw 10
sum_result: dw 0
subtraction_result: dw 0
multiplication_result: dw 0
division_result: dw 0

divv:
push bp
mov bp ,sp
push ax
push bx

mov ax , [bp+6]
mov bx ,[bp+4]

div bx
mov [division_result], ax
pop bx
pop ax
pop bp
ret
mull:
push bp
mov bp ,sp
push ax
push bx

mov ax , [bp+6]
mov bx ,[bp+4]
imul ax,bx
mov [multiplication_result] , ax
pop bx
pop ax
pop bp
ret
sum:
push bp ;
mov bp,sp ;
mov ax , [bp+6]
mov bx ,[bp+4]
add ax ,bx
mov [sum_result] , ax
pop bp
ret
subtraction:
push bp ;
mov bp,sp ;
push ax
push bx
mov ax , [bp+6]
mov bx , [bp+4]
sub bx , ax
mov [subtraction_result], bx
pop bx
pop ax
pop bp
ret
start:
  mov ax,[operand1]
  push ax
  mov ax ,[operand2]
  push ax;
  call sum
  call subtraction
  call mull
  call divv

 mov ax, 0x4c00
 int 0x21
```



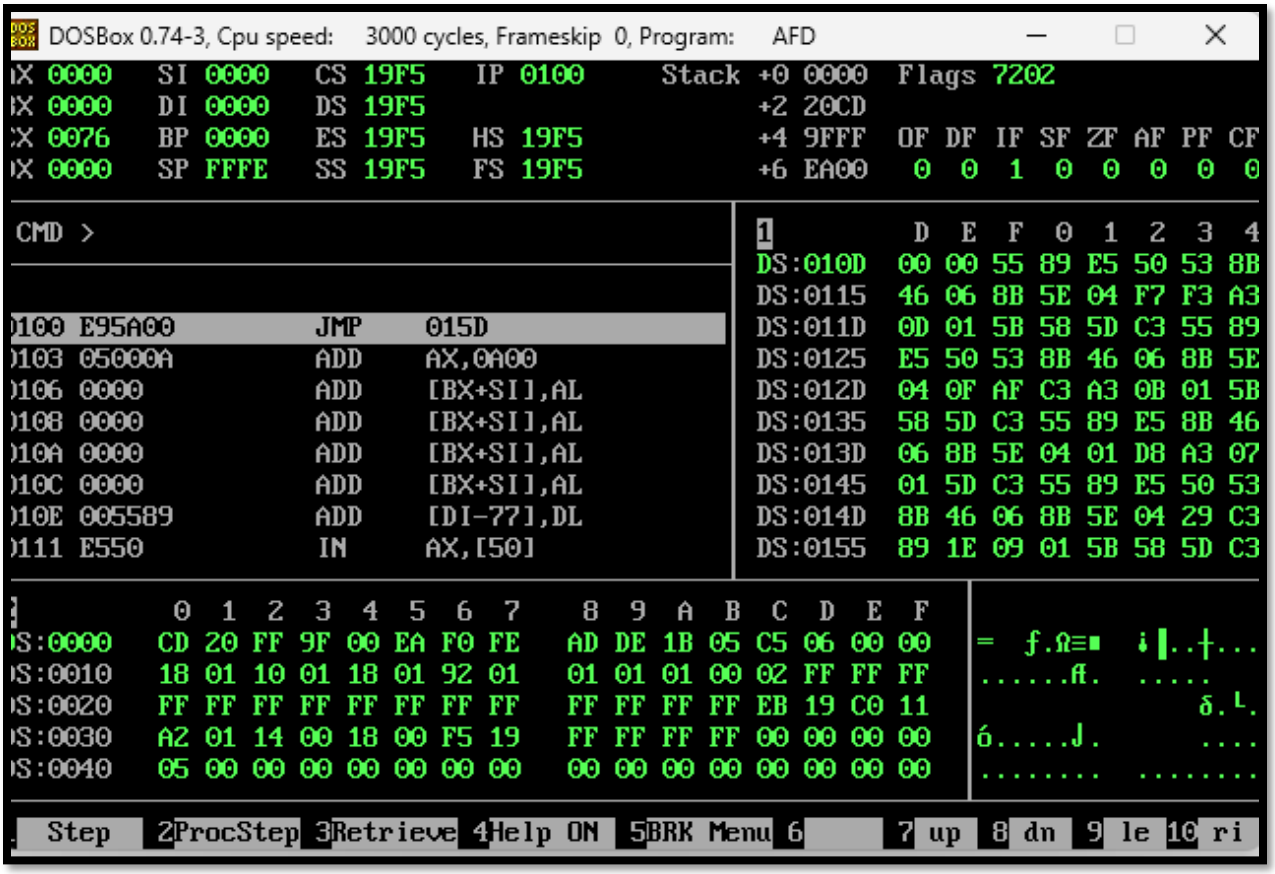
→The upper screen shot is of addition.



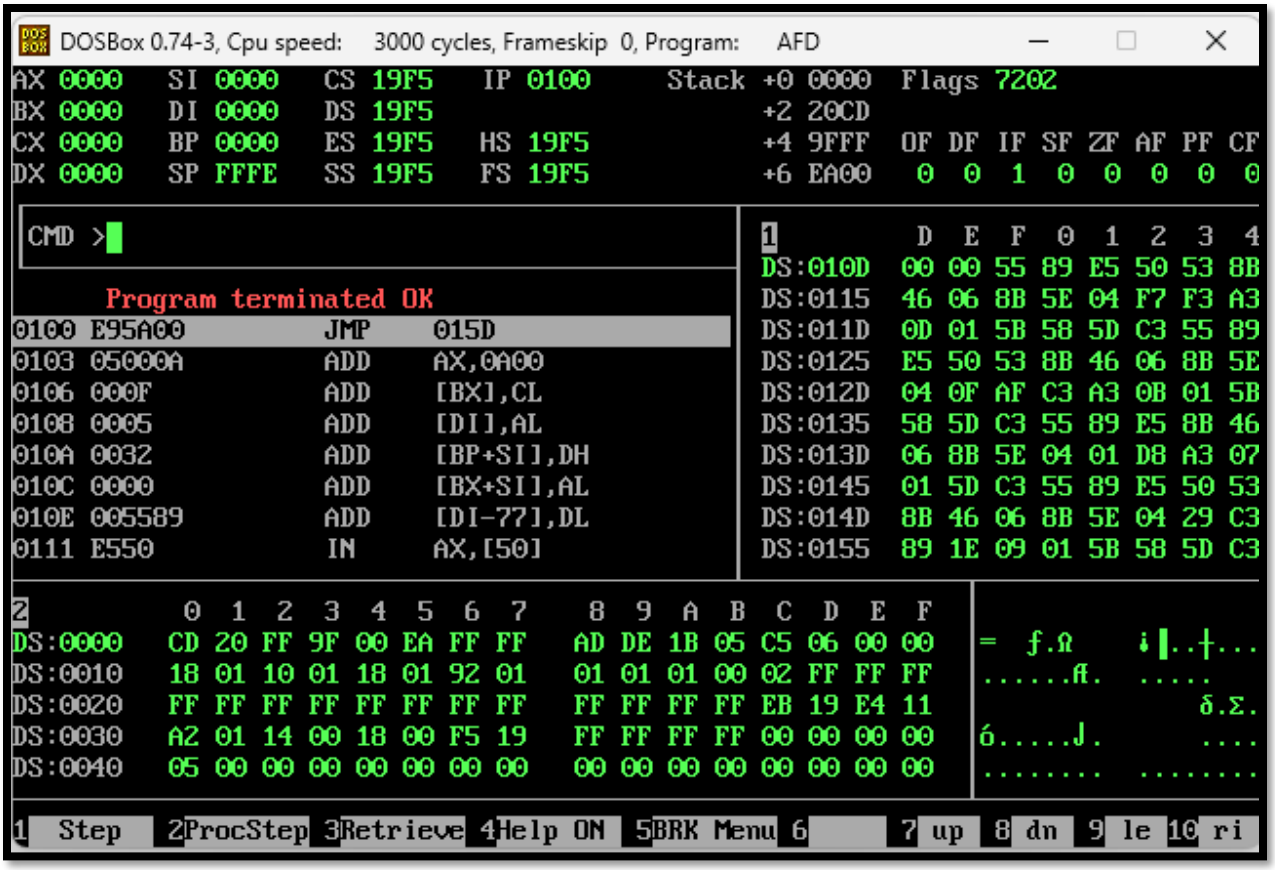
→ The upper screen shot is of substraction.



→ The upper screen shot is of multiplication.



→ The upper screen shot is of division. In this dividing 5 by 10 it gives us 0.5 but in this it will be shown in the output as 0.(not in points).



→Program terminated sucessfully.

<u>Question # 2: (now using recursion)</u>

```
[org 0x0100]

jmp start

data:   dw 5
count:  dw 11
result: dw 0

recursion:
        push bp
        mov bp, sp
        mov ax, [bp + 4]      ; Load the argument from the stack
        add [result], ax
        dec word[count]
        cmp word[count], 0
        jle end_recursion     ; Jump if less than or equal to zero

        ; Recursive call
        push word[result]     ; Save the current result on the stack
        push word[count]      ; Save the current count on the stack
        call recursion        ; Recursive call
        add sp, 4             ; Clean up the stack

end_recursion:
        pop bp
        ret

start:  mov ax, [data]
        push ax
        mov cx, [count]
        push cx
        call recursion

        mov ax, 0x4c00
        int 0x21
```

→steps:

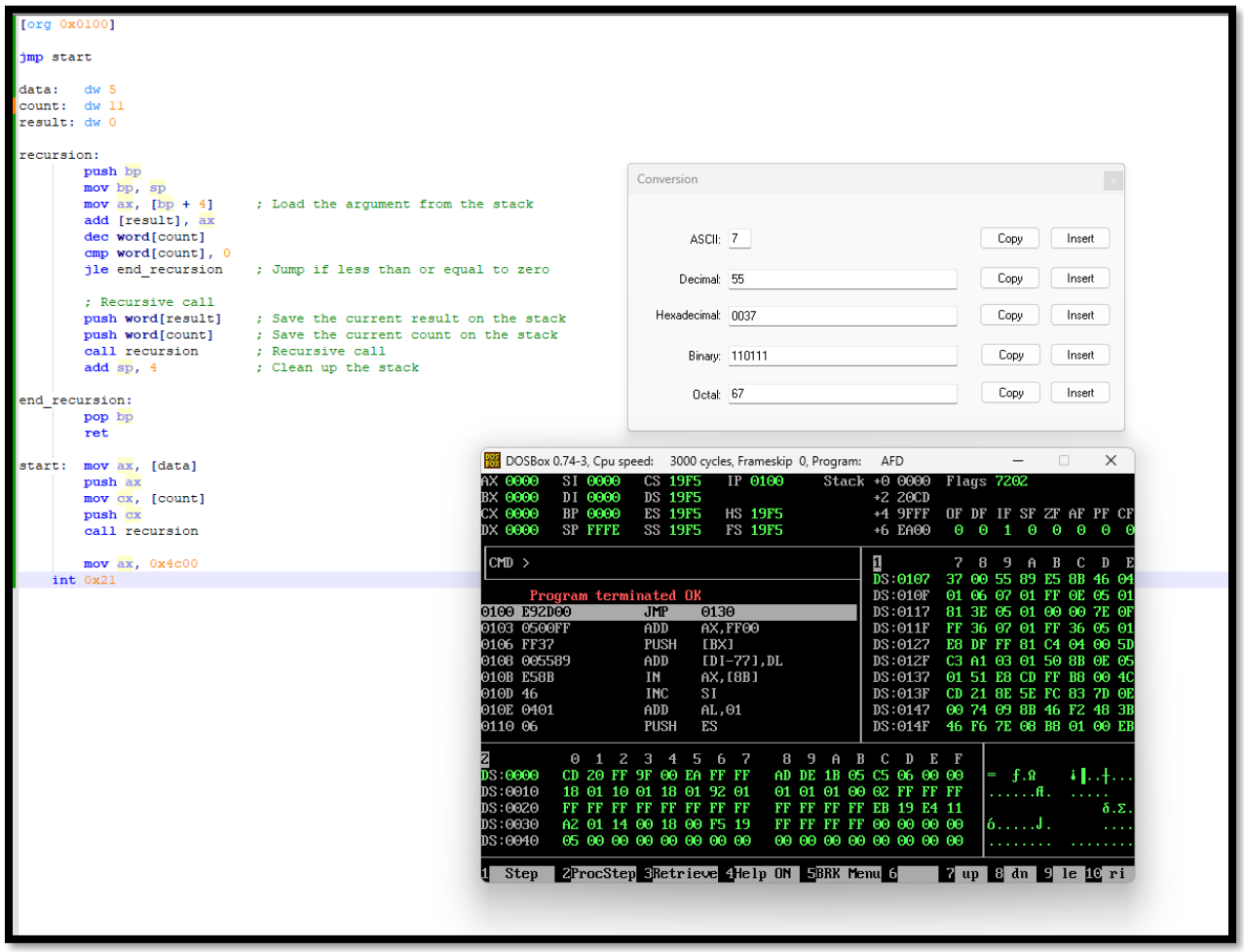
```
DOSBox 0.74-3, Cpu speed:   3000 cycles, Frameskip  0, Program:   AFD
AX 0007   SI 0000   CS 19F5   IP 0117     Stack +0 FFE6  Flags 7204
BX 0000   DI 0000   DS 19F5                    +2 012A
CX 000A   BP FFDE   ES 19F5   HS 19F5           +4 0007  OF DF IF SF ZF AF PF CF
DX 0000   SP FFDE   SS 19F5   FS 19F5           +6 001B   0  0  1  0  0  0  1  0

CMD >                                                   1        7  8  9  A  B  C  D  E
                                                 0006   DS:0107  22 00 55 89 E5 8B 46 04
0113 FF0E0501        DEC    W/[0105]                    DS:010F  01 06 07 01 FF 0E 05 01
0117 813E05010000    CMP    [0105],0000                 DS:0117  81 3E 05 01 00 00 7E 0F
011D 7E0F            JNG    012E                        DS:011F  FF 36 07 01 FF 36 05 01
011F FF360701        PUSH   [0107]                      DS:0127  E8 DF FF 81 C4 04 00 5D
0123 FF360501        PUSH   [0105]                      DS:012F  C3 A1 03 01 50 8B 0E 05
0127 E8DFFF          CALL   0109                        DS:0137  01 51 E8 CD FF B8 00 4C
012A 81C40400        ADD    SP,0004                     DS:013F  CD 21 8E 5E FC 83 7D 0E
012E 5D              POP    BP                          DS:0147  00 74 09 8B 46 F2 48 3B
012F C3              RET                                DS:014F  46 F6 7E 08 B8 01 00 EB

DS:0000  CD 20 FF 9F 00 EA F0 FE  AD DE 1B 05 C5 06 00 00  = ƒ.Ω≡■  ¡▌..┼...
DS:0010  18 01 10 01 18 01 92 01  01 01 01 00 FF 00 01 00  ......Æ.  .... ...
DS:0020  01 00 01 FF FF FF FF FF  FF FF FF FF EB 19 C0 11  ...          δ.└.
DS:0030  A2 01 14 00 18 00 F5 19  FF FF FF FF 00 00 00 00  ó.....J.      ....
DS:0040  05 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........  ........

 Step  2ProcStep 3Retrieve 4Help ON  5BRK Menu 6      7 up  8 dn  9 le 10 ri
```

```
DOSBox 0.74-3, Cpu speed:   3000 cycles, Frameskip  0, Program:   AFD
AX 0006   SI 0000   CS 19F5   IP 0123     Stack +0 0028  Flags 7204
BX 0000   DI 0000   DS 19F5                    +2 FFDE
CX 000A   BP FFD6   ES 19F5   HS 19F5          +4 012A  OF DF IF SF ZF AF PF CF
DX 0000   SP FFD4   SS 19F5   FS 19F5          +6 0006   0  0  1  0  0  0  1  0

CMD >                                                   1        7  8  9  A  B  C  D  E
                                                 -0005-
011F FF360701       PUSH   [0107]                       DS:0107  28 00 55 89 E5 8B 46 04
0123 FF360501       PUSH   [0105]                       DS:010F  01 06 07 01 FF 0E 05 01
0127 E8DFFF         CALL   0109                         DS:0117  81 3E 05 01 00 00 7E 0F
012A 81C40400       ADD    SP,0004                      DS:011F  FF 36 07 01 FF 36 05 01
012E 5D             POP    BP                           DS:0127  E8 DF FF 81 C4 04 00 5D
012F C3             RET                                 DS:012F  C3 A1 03 01 50 8B 0E 05
0130 A10301         MOV    AX,[0103]                    DS:0137  01 51 E8 CD FF B8 00 4C
0133 50             PUSH   AX                           DS:013F  CD 21 8E 5E FC 83 7D 0E
0134 8B0E0501       MOV    CX,[0105]                    DS:0147  00 74 09 8B 46 F2 48 3B
                                                        DS:014F  46 F6 7E 08 B8 01 00 EB

2          0  1  2  3  4  5  6  7    8  9  A  B  C  D  E  F
DS:0000   CD 20 FF 9F 00 EA F0 FE   AD DE 1B 05 C5 06 00 00   = ƒ.Ω≡■  ¡▌..┼...
DS:0010   18 01 10 01 18 01 92 01   01 01 01 00 FF 00 01 00   ......ª.  .... ...
DS:0020   01 00 01 FF FF FF FF FF   FF FF FF FF EB 19 C0 11   ...          δ.└.
DS:0030   A2 01 14 00 18 00 F5 19   FF FF FF FF 00 00 00 00   ó.....J.     ....
DS:0040   05 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00   ........  ........

1  Step   2ProcStep 3Retrieve 4Help ON  5BRK Menu 6      7 up  8 dn  9 le 10 ri
```

```
DOSBox 0.74-3, Cpu speed:   3000 cycles, Frameskip  0, Program:   AFD
AX 0003   SI 0000   CS 19F5   IP 011F     Stack +0 FFC6  Flags 7200
BX 0000   DI 0000   DS 19F5                    +2 012A
CX 000A   BP FFBE   ES 19F5   HS 19F5          +4 0003  OF DF IF SF ZF AF PF CF
DX 0000   SP FFBE   SS 19F5   FS 19F5          +6 0031   0  0  1  0  0  0  0  0

CMD >                                                   1        7  8  9  A  B  C  D  E
                                                 -0034-
011D 7E0F           JNG    012E                         DS:0107  34 00 55 89 E5 8B 46 04
011F FF360701       PUSH   [0107]                       DS:010F  01 06 07 01 FF 0E 05 01
0123 FF360501       PUSH   [0105]                       DS:0117  81 3E 05 01 00 00 7E 0F
0127 E8DFFF         CALL   0109                         DS:011F  FF 36 07 01 FF 36 05 01
012A 81C40400       ADD    SP,0004                      DS:0127  E8 DF FF 81 C4 04 00 5D
012E 5D             POP    BP                           DS:012F  C3 A1 03 01 50 8B 0E 05
012F C3             RET                                 DS:0137  01 51 E8 CD FF B8 00 4C
0130 A10301         MOV    AX,[0103]                    DS:013F  CD 21 8E 5E FC 83 7D 0E
0133 50             PUSH   AX                           DS:0147  00 74 09 8B 46 F2 48 3B
                                                        DS:014F  46 F6 7E 08 B8 01 00 EB

2          0  1  2  3  4  5  6  7    8  9  A  B  C  D  E  F
DS:0000   CD 20 FF 9F 00 EA F0 FE   AD DE 1B 05 C5 06 00 00   = ƒ.Ω≡■  ¡▌..┼...
DS:0010   18 01 10 01 18 01 92 01   01 01 01 00 FF 00 01 00   ......ª.  .... ...
DS:0020   01 00 01 FF FF FF FF FF   FF FF FF FF EB 19 C0 11   ...          δ.└.
DS:0030   A2 01 14 00 18 00 F5 19   FF FF FF FF 00 00 00 00   ó.....J.     ....
DS:0040   05 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00   ........  ........

1  Step   2ProcStep 3Retrieve 4Help ON  5BRK Menu 6      7 up  8 dn  9 le 10 ri
```

```
DOSBox 0.74-3, Cpu speed:   3000 cycles, Frameskip  0, Program:   AFD
AX 0000   SI 0000   CS 19F5   IP 0100     Stack +0 0000  Flags 7202
BX 0000   DI 0000   DS 19F5                    +2 20CD
CX 0000   BP 0000   ES 19F5   HS 19F5          +4 9FFF  OF DF IF SF ZF AF PF CF
DX 0000   SP FFFE   SS 19F5   FS 19F5          +6 EA00   0  0  1  0  0  0  0  0

CMD >

   > Program terminated OK
0100 E92D00          JMP    0130
0103 050000          ADD    AX,0000
0106 0037            ADD    [BX],DH
0108 005589          ADD    [DI-77],DL
010B E58B            IN     AX,[8B]
010D 46              INC    SI
010E 0401            ADD    AL,01
0110 06              PUSH   ES

1          7  8  9  A  B  C  D  E
DS:0107   37 00 55 89 E5 8B 46 04
DS:010F   01 06 07 01 FF 0E 05 01
DS:0117   81 3E 05 01 00 00 7E 0F
DS:011F   FF 36 07 01 FF 36 05 01
DS:0127   E8 DF FF 81 C4 04 00 5D
DS:012F   C3 A1 03 01 50 8B 0E 05
DS:0137   01 51 E8 CD FF B8 00 4C
DS:013F   CD 21 8E 5E FC 83 7D 0E
DS:0147   00 74 09 8B 46 F2 48 3B
DS:014F   46 F6 7E 08 B8 01 00 EB

2          0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F
DS:0000   CD 20 FF 9F 00 EA FF FF   AD DE 1B 05 C5 06 00 00   =  ƒ.Ω    ¡▌..┼...
DS:0010   18 01 10 01 18 01 92 01   01 01 01 00 FF 00 01 00   ......ª.  .... ...
DS:0020   01 00 01 FF FF FF FF FF   FF FF FF FF EB 19 E4 11   ...           δ.Σ.
DS:0030   A2 01 14 00 18 00 F5 19   FF FF FF FF 00 00 00 00   ó.....J.      ....
DS:0040   05 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00   ........  ........

1 Step  2ProcStep 3Retrieve 4Help ON  5BRK Menu 6      7 up  8 dn  9 le 10 ri
```

→FINAL OUTPUt:



→In this I am using recursion and also using subroutine to add 5 (11 times) in this I am also passing parameters and saving values.and at the end popping back .and the result as shown in the upper screenshot is 55.

--THANKS--