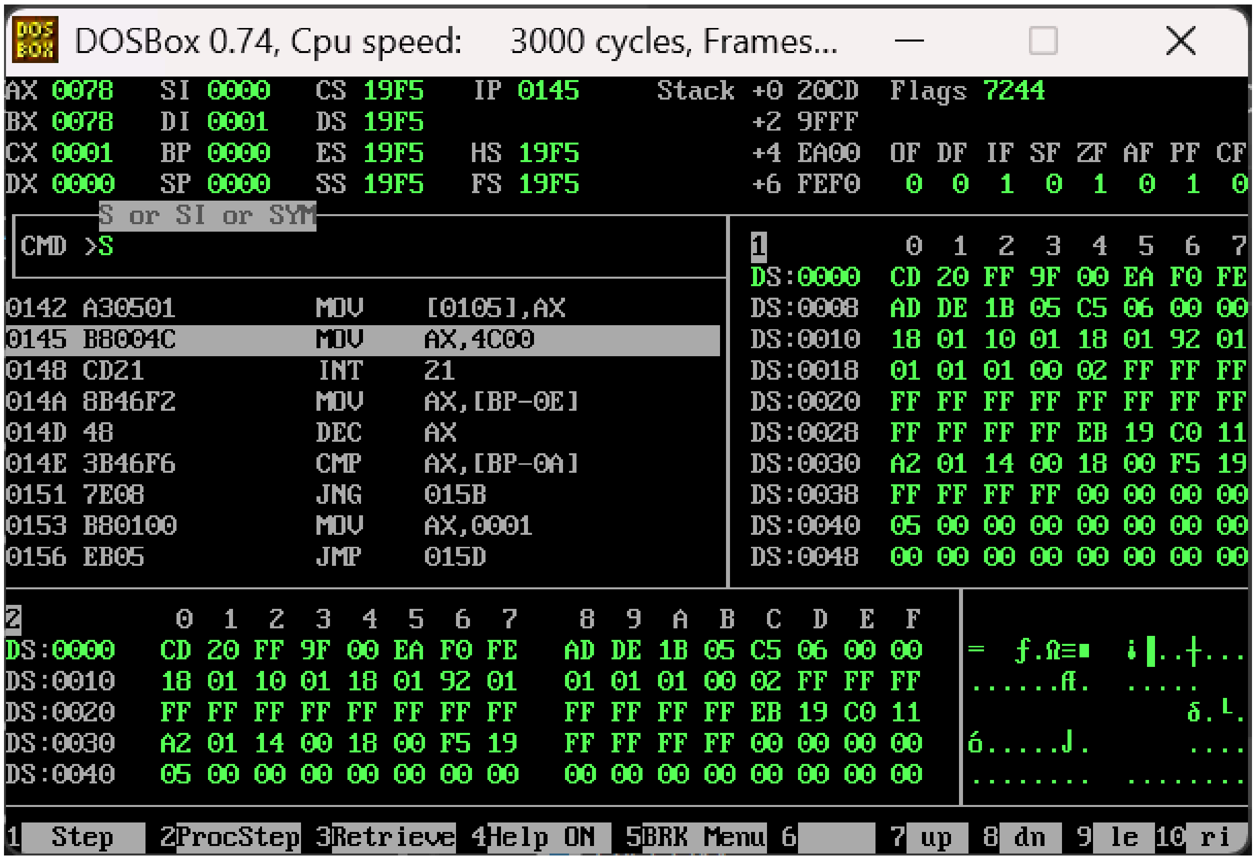
Q1



```
DOSBox 0.74, Cpu speed:     3000 cycles, Frames...
AX 0078   SI 0000   CS 19F5   IP 0145      Stack +0 20CD  Flags 7244
BX 0078   DI 0001   DS 19F5                      +2 9FFF
CX 0001   BP 0000   ES 19F5   HS 19F5            +4 EA00  OF DF IF SF ZF AF PF CF
DX 0000   SP 0000   SS 19F5   FS 19F5            +6 FEF0   0  0  1  0  1  0  1  0
      S or SI or SYM
 CMD >S                                           1        0  1  2  3  4  5  6  7
                                                  DS:0000  CD 20 FF 9F 00 EA F0 FE
0142 A30501          MOV     [0105],AX            DS:0008  AD DE 1B 05 C5 06 00 00
0145 B8004C          MOV     AX,4C00              DS:0010  18 01 10 01 18 01 92 01
0148 CD21            INT     21                   DS:0018  01 01 01 00 02 FF FF FF
014A 8B46F2          MOV     AX,[BP-0E]           DS:0020  FF FF FF FF FF FF FF FF
014D 48              DEC     AX                   DS:0028  FF FF FF FF EB 19 C0 11
014E 3B46F6          CMP     AX,[BP-0A]           DS:0030  A2 01 14 00 18 00 F5 19
0151 7E08            JNG     015B                 DS:0038  FF FF FF FF 00 00 00 00
0153 B80100          MOV     AX,0001              DS:0040  05 00 00 00 00 00 00 00
0156 EB05            JMP     015D                 DS:0048  00 00 00 00 00 00 00 00

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 02 FF FF FF  ......Æ.  .....
DS:0020    FF FF FF 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
```

[org 0x0100]

jmp start
fact: dw 5
answer: dw 0

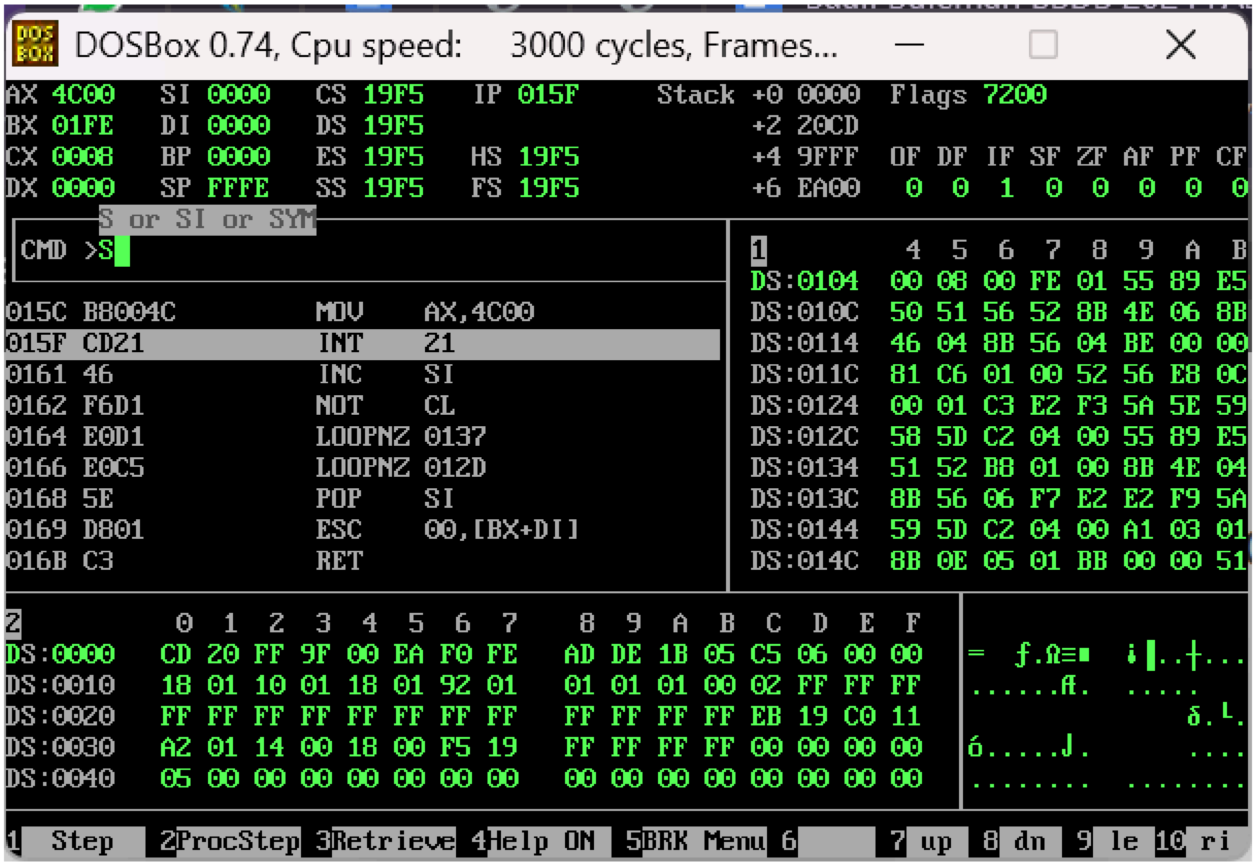
factorial :
push bp
mov bp,sp

mov bx,[bp+4]
mov di,[bp+4]
sub di,1
mov ax,0
mov cx,0

l1:
add ax,bx
add cx,1
cmp cx ,di

```
jne l1

mov bx,ax
mov cx,1
sub di,1
cmp di,1
jne l1

mov sp,bp
pop bp
push ax
pop ax
ret 4

start :
mov ax, [fact]
push ax
call factorial
mov[answer],ax

mov ax,0x4c00
int 0x21
```

Q2



DOSBox 0.74, Cpu speed: 3000 cycles, Frames...

```
AX 4C00   SI 0000   CS 19F5   IP 015F      Stack +0 0000  Flags 7200
BX 01FE   DI 0000   DS 19F5                      +2 20CD
CX 0008   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
      S or SI or SYM
 CMD >S                                          1       4  5  6  7  8  9  A  B
                                                 DS:0104 00 08 00 FE 01 55 89 E5
015C B8004C          MOV    AX,4C00              DS:010C 50 51 56 52 8B 4E 06 8B
015F CD21            INT    21                   DS:0114 46 04 8B 56 04 BE 00 00
0161 46              INC    SI                   DS:011C 81 C6 01 00 52 56 E8 0C
0162 F6D1            NOT    CL                   DS:0124 00 01 C3 E2 F3 5A 5E 59
0164 E0D1            LOOPNZ 0137                 DS:012C 58 5D C2 04 00 55 89 E5
0166 E0C5            LOOPNZ 012D                 DS:0134 51 52 B8 01 00 8B 4E 04
0168 5E              POP    SI                   DS:013C 8B 56 06 F7 E2 E2 F9 5A
0169 D801            ESC    00,[BX+DI]           DS:0144 59 5D C2 04 00 A1 03 01
016B C3              RET                         DS:014C 8B 0E 05 01 BB 00 00 51

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≡▪  i▌..┼...
DS:0010   18 01 10 01 18 01 92 01  01 01 01 00 02 FF FF FF   ......Æ.  .....
DS:0020   FF FF FF 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
```

```
[org 0x0100]
jmp start
r: dw 2
l: dw 8
result: dw 0

series:
push bp
mov bp,sp
push ax
push cx
push si
push dx

mov cx,[bp+6]
mov ax,[bp+4]
mov dx,[bp+4]
mov si,0

l1:
add si,1
```

```
push dx
push si
call mulli
add bx,ax
loop l1

pop dx
pop si
pop cx
pop ax
pop bp
ret 4

mulli:
push bp
mov bp,sp
push cx
push dx

mov ax,1

mov cx,[bp+4]

l2:
mov dx,[bp+6]
mul dx
loop l2

pop dx
pop cx
pop bp
ret 4

start:
mov ax,[r]
mov cx,[l]
mov bx,0
push cx
push ax
call series

terminate:
mov [result],bx
mov ax,0x4c00
```

```
int 0x21
```

Q3

(i) How can we access all parameters in function/subroutine?

All parameters are accessed via `[BP+offset]`

Parameters can be accessed using positive offsets from BP:

[BP+6]

[BP+8]

(ii) How can we place return value of function?

Usually the return value is placed in register AX

mov ax, [result]

(iii) How can we access local stack variables of the function/sub routine?

Local variables are created by reserving space on the stack after setting up BP.

Example: sub sp, 4

They are accessed as negative offsets from BP:

`[BP-2]` = first local variable

`[BP-4]` = second local variable

(iv) How to empty stack before & after leaving subroutine?

All registers are required to be popped in the same order they were pushed in.

```
mov sp, bp     ; deallocate locals
pop bp         ; restore old base pointer
ret n
```

(v) How can we pass parameters and retrieve return value in Caller?

Push parameters in reverse order, then call the subroutine:

```
mov ax, p4
push ax
mov ax, p3
push ax
mov ax, p2
push ax
mov ax, p1
push ax
call myFunction
```

Retrieve the return value:

```
mov [result], ax
```