National University of Computer and Emerging Sciences

Lab Manual 01
Computer Organization and Assembly Language Lab

Course Instructor	Miss Aleena
Lab Instructor	Maham Saleem
Section	3F1
Semester	Fall 2021

Department of Computer Science

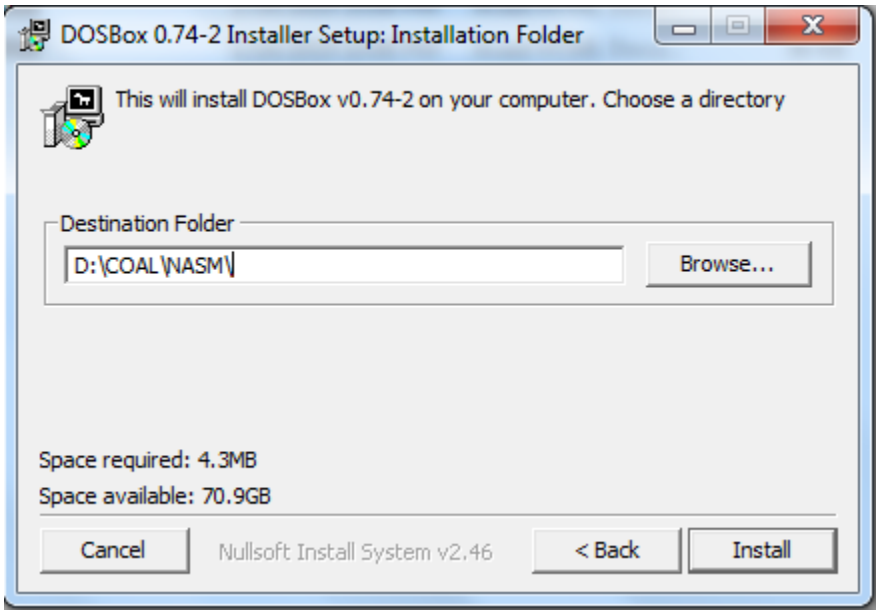
FAST-NU, Lahore, Pakistan

Activity 1: Setup

Download and install NASM, AFD and DOSBOX, according to the instructions, in your NASM folder.

AFD: https://drive.google.com/file/d/1eXnD3JEwBelFiJT6iIk7gluudV2Fu_iX/view?usp=sharing
NASM: https://drive.google.com/file/d/1ZoeE2MxjNaK7DdJKCacYfAJyn006MI_F/view?usp=sharing
Dosbox: https://drive.google.com/file/d/1DnaDIk4RoGBFDP1y4Dr3q89xwM3gx1d1/view?usp=sharing

Tutorial part 1: https://drive.google.com/file/d/1N3lWL8hsN0ZbhF3tlNwCWWwjJ_eHQqk6/view?usp=sharing
Tutorial part 2: https://drive.google.com/file/d/10p8qyaOVOwF5lDighrMKE-uNYQX-c3bL/view?usp=sharing



After installations double click "DOSBox 0.74-2 Options.bat" file and at the end of the file paste following lines:

```
MOUNT C D://COAL//NASM
C:
```

(We are mounting C drive to our folder where we have saved AFD and we will save our .asm file in this directory)

Activity 2: Running your First Program

Follow these step in order to run your first program:

1- Copy/paste following code in notepad

```
; this is a comment. Comment starts with semicolon
```

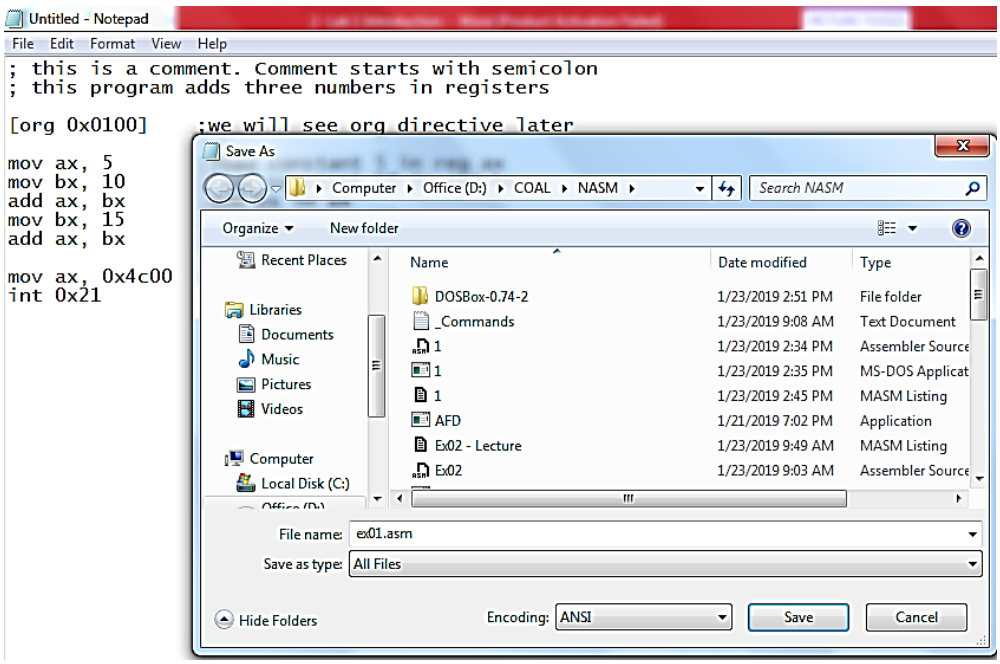
```
; this program adds three numbers in registers

[org 0x0100]     ;we will see org directive later

mov ax, 5        ; AX = 5
mov bx, 10       ; BX = 10
add ax, bx       ; AX = AX + BX
mov bx, 15       ; BX = 15
add ax, bx       ; AX = AX + BX

mov ax, 0x4c00 ;terminate the program
int 0x21
```

2- Save this file as “ex01.asm” in your NASM folder e.g. “D:\COAL\NASM”:



3- Go to NASM installation directory (e.g. “D:\COAL\NASM”). Double click **nasmpath.bat** (batch file) and type following command there. (Your .asm file and nasm should be in one folder)

nasm ex01.asm -o ex01.com -l ex01.lst

4- Above command will assemble your code and create ex01.com and ex01.lst files. Open ex01.lst file in notepad.

a. What is opcode of instruction “mov ax, someConstant”
   B80

b. Verify the above opcode everywhere the instruction has been used.

```
 9 0000000D B8004C                mov ax, 0x4c00 ;terminate the program
```

c. What does “B80500” mean?

B8 means mov to register ax and 0500 is the value stored which was 5 in this case

d. Verify the opcode of instruction “mov bx, someConstant” throughout the machine code.

```
 4 00000000 B80500                  mov ax, 5 ; AX = 5
 5 00000003 BB0A00                  mov bx, 10 ; BX = 10
 6 00000006 01D8                    add ax, bx ; AX = AX + BX
 7 00000008 BB0F00                  mov bx, 15 ; BX = 15
 8 0000000B 01D8                    add ax, bx ; AX = AX + BX
 9 0000000D B8004C                  mov ax, 0x4c00 ;terminate the program
10 00000010 CD21                    int 0x21
```

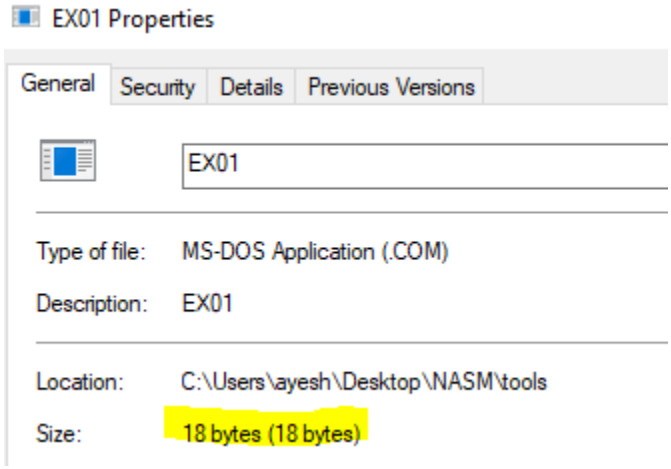
e. What is the offset of first instruction?

```
 4 00000000 B80500                  mov ax, 5 ; AX = 5
```

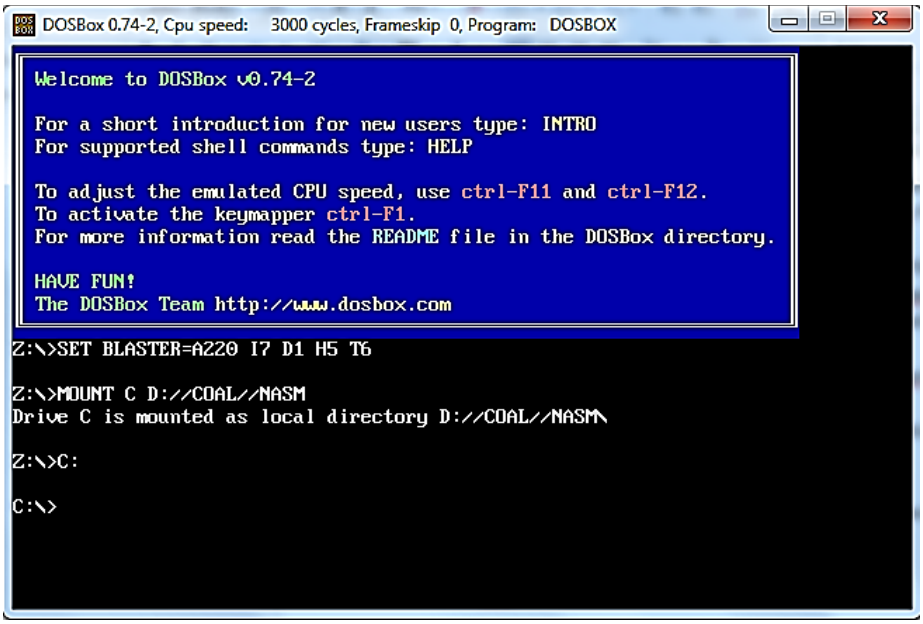
f. Why are offsets of second and third instructions 3 and 6?
Because the previous instructions before 3 and 6 had sizes 3 bytes each
So it was added in the offsets of these instructions.

g. What should be the size of ex01.com file?
18 bytes.

h. Right click ex01.com and verify its size.



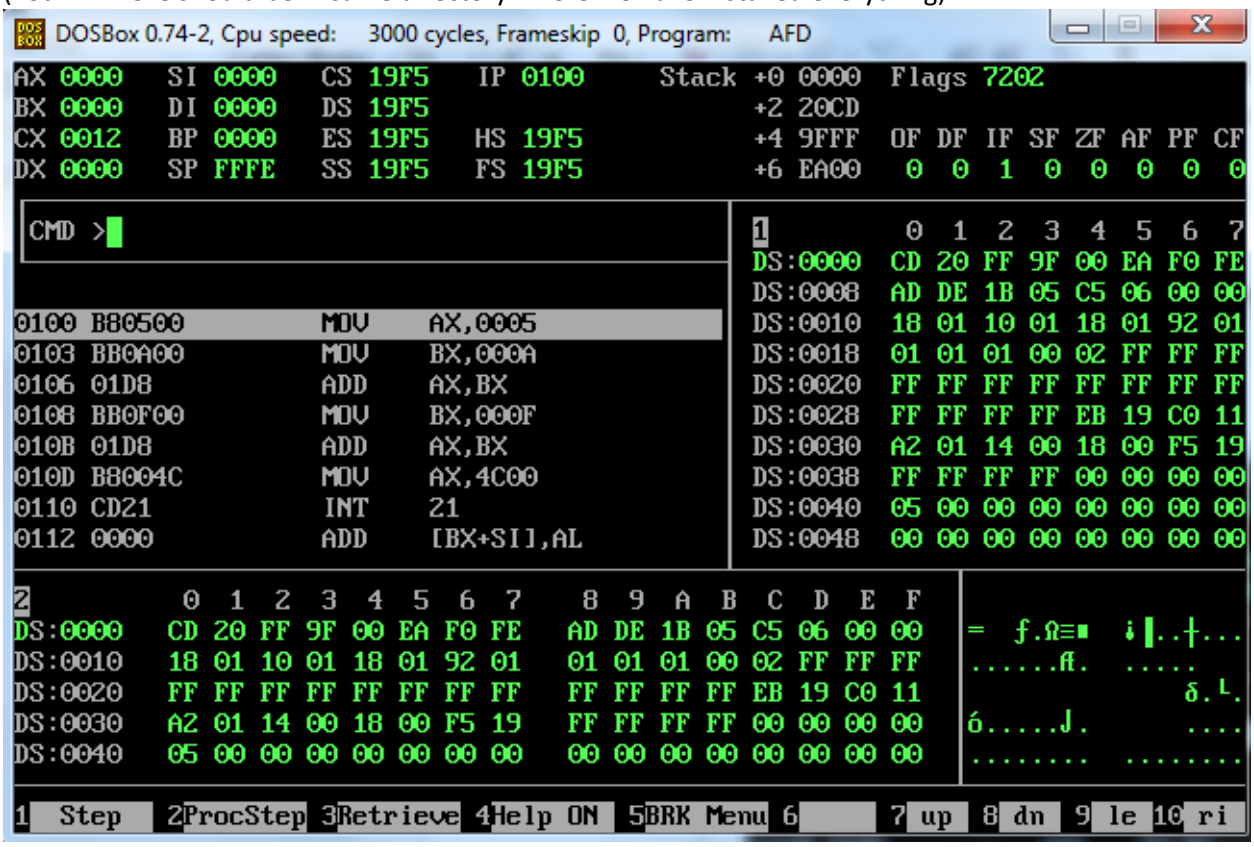
5- Open DOSBox (by double clicking dosbox.exe), following window will appear:



6- Write following command and press enter.

Afd ex01.com

(Your AFD.exe should be in same directory where we have installed everything)

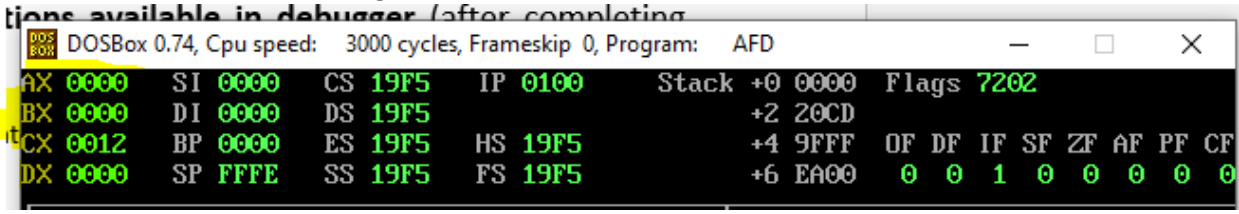


7- Above command will open the debugger and load your ex01.com file in it.

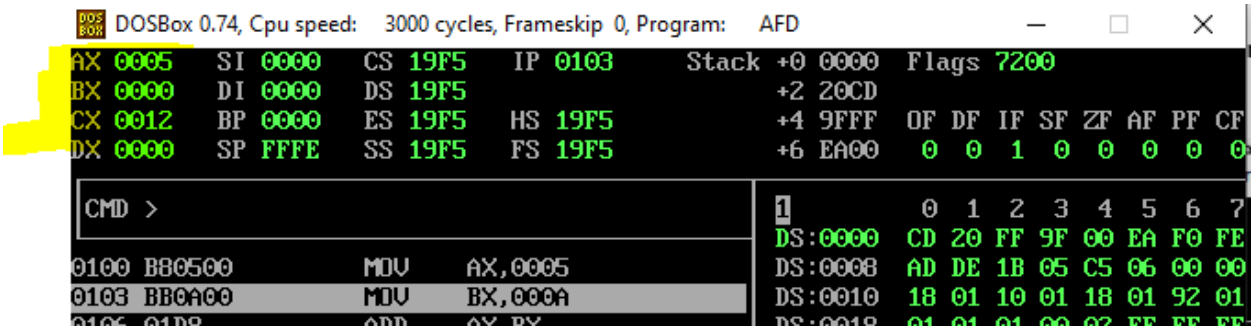
a. What is the value of IP register? And what will be its effect?

Value of IP is 0100 and its effect will be that the first instruction that will be executed will be the one at address 0100.

b. Note the initial values of data registers

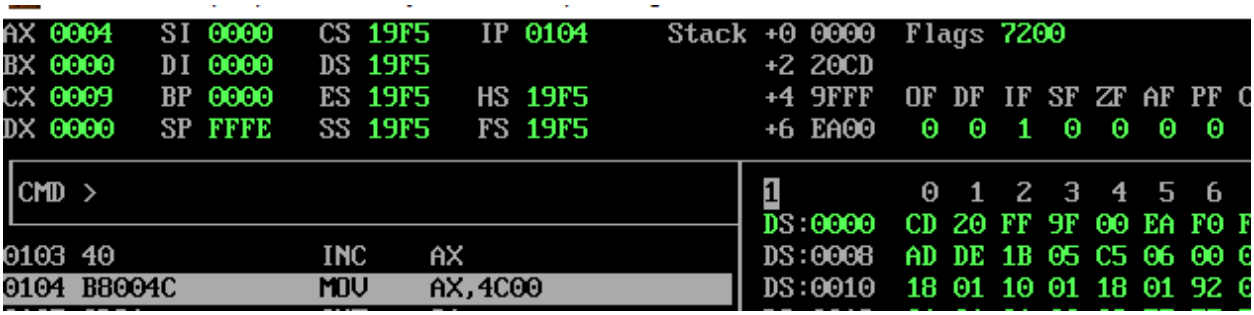


c. Press F1 and watch the values of data registers

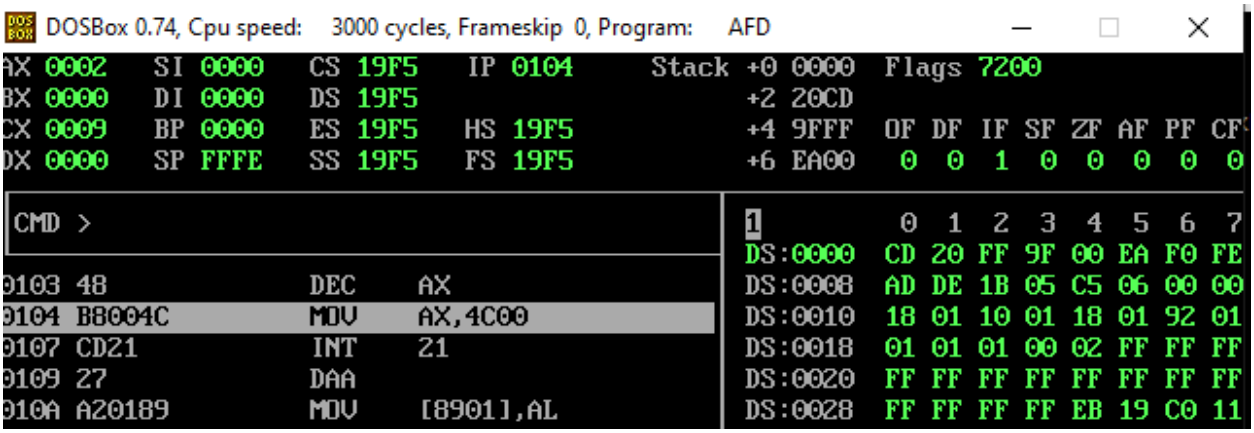


Activity 3: Explore different functions available in debugger (after completing activity 4).

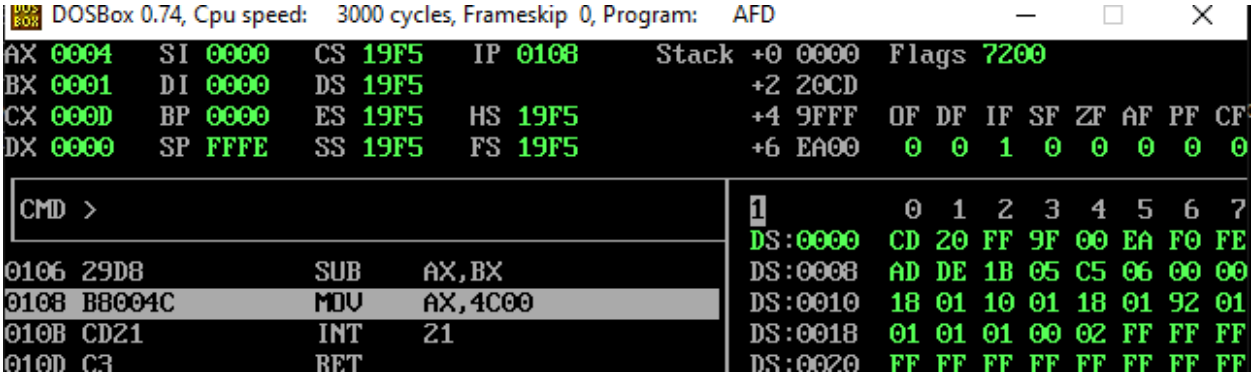
1) **INC function**



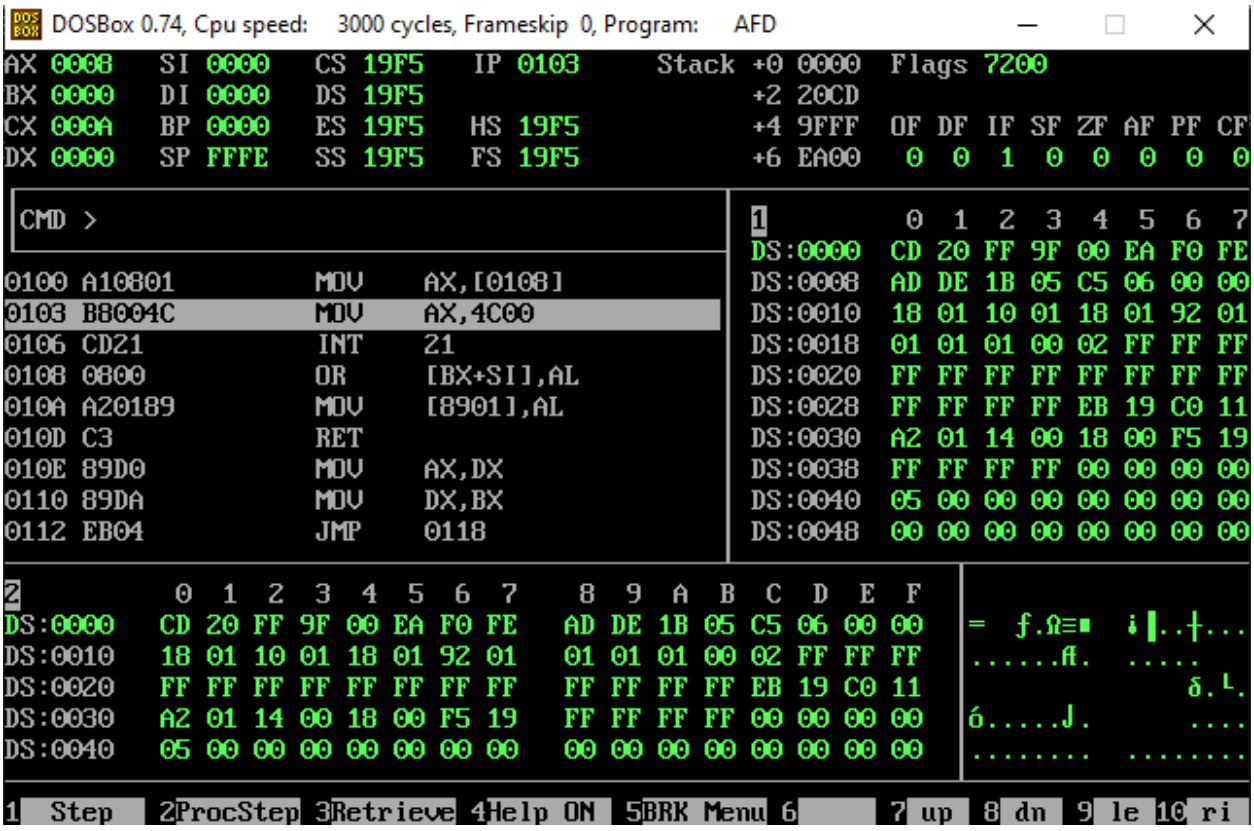
2) **DEC function**



3) **SUB function**

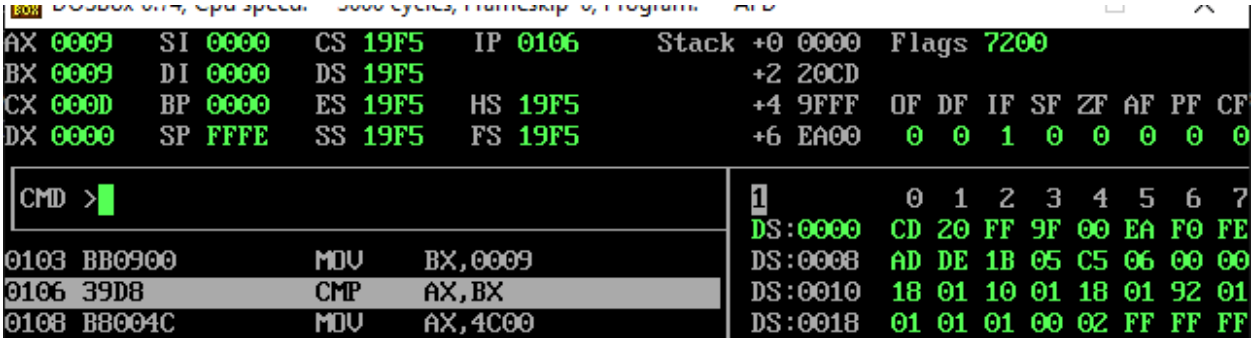


4) **Data Declaration**

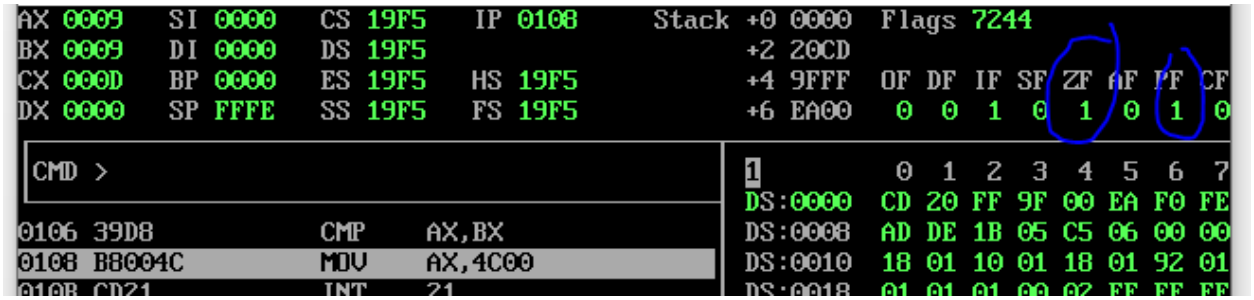


5) **CMP Function**

Before

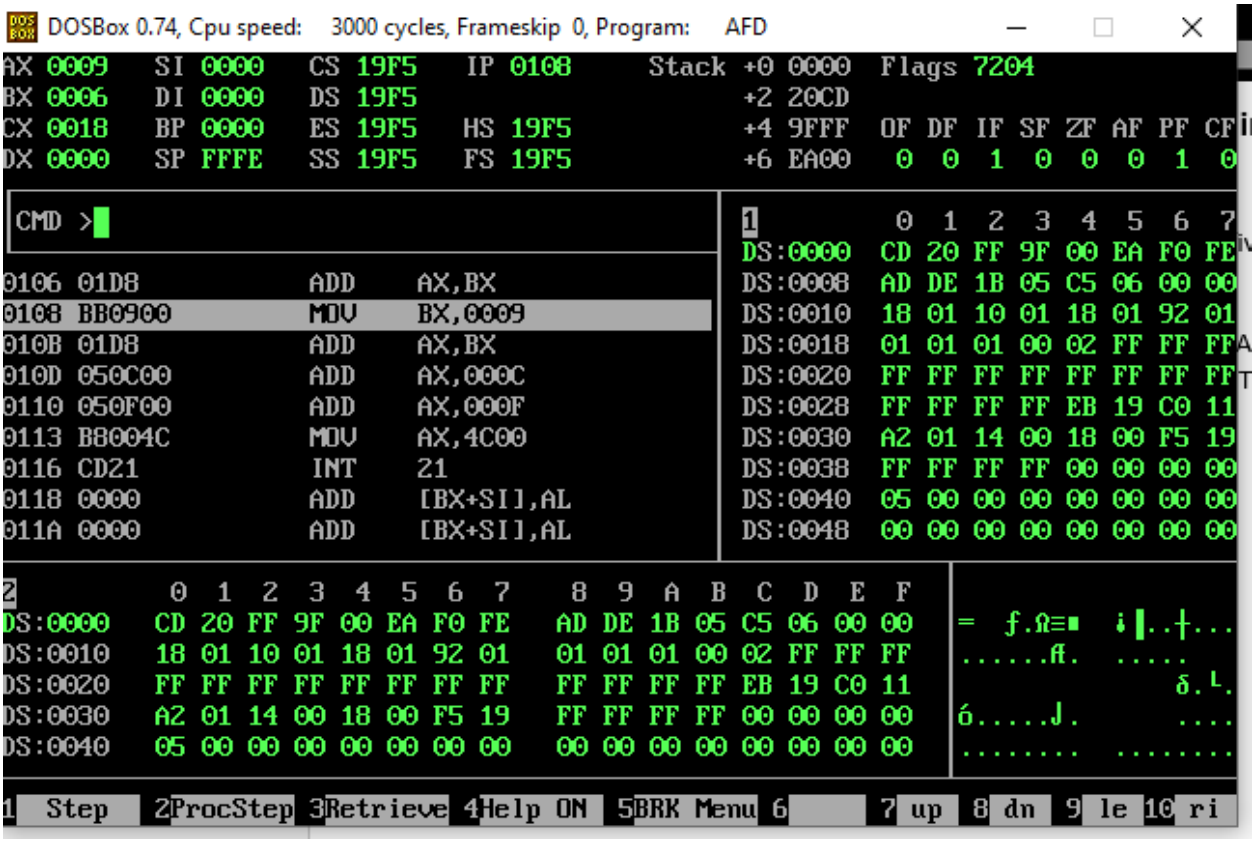


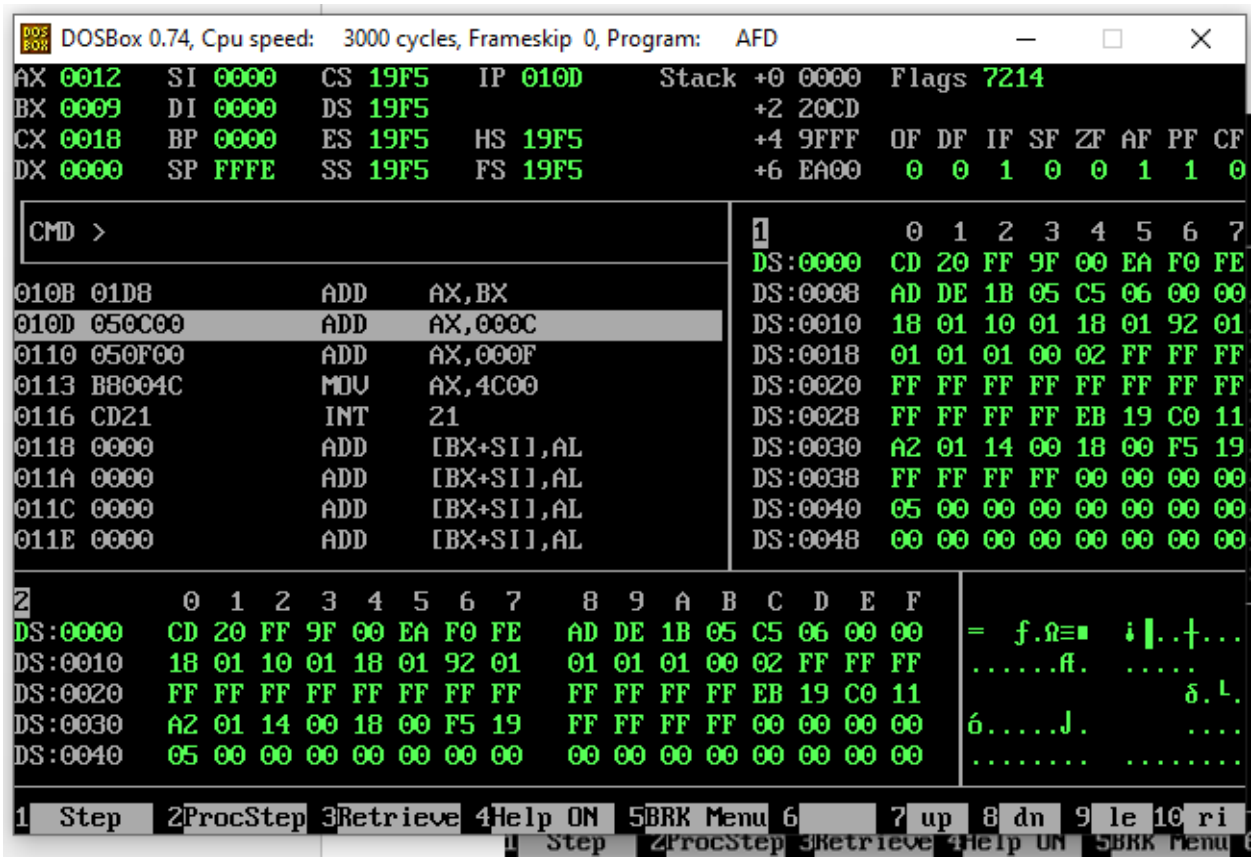
After

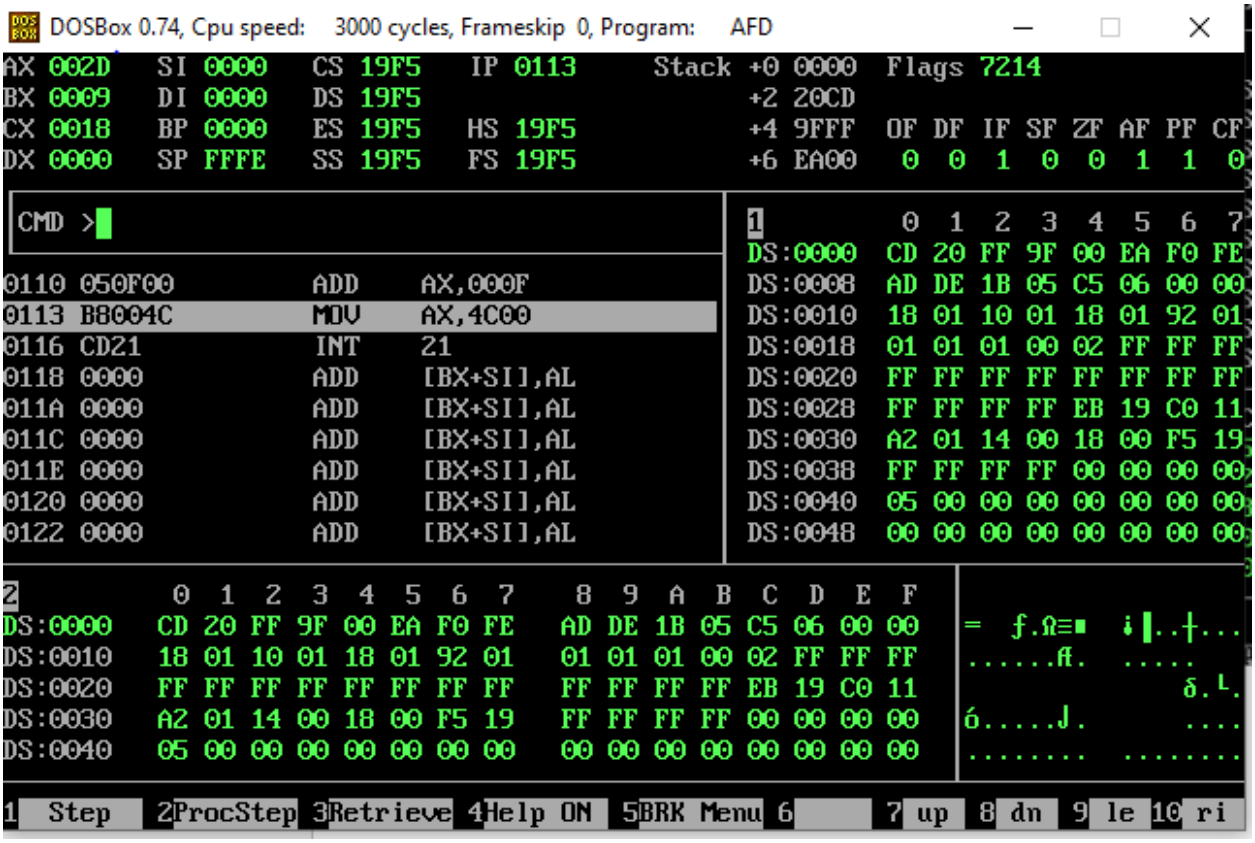


Activity 4: Modify this program to generate the sum of first five entries of table of 3, using registers, and watch its execution in the debugger.

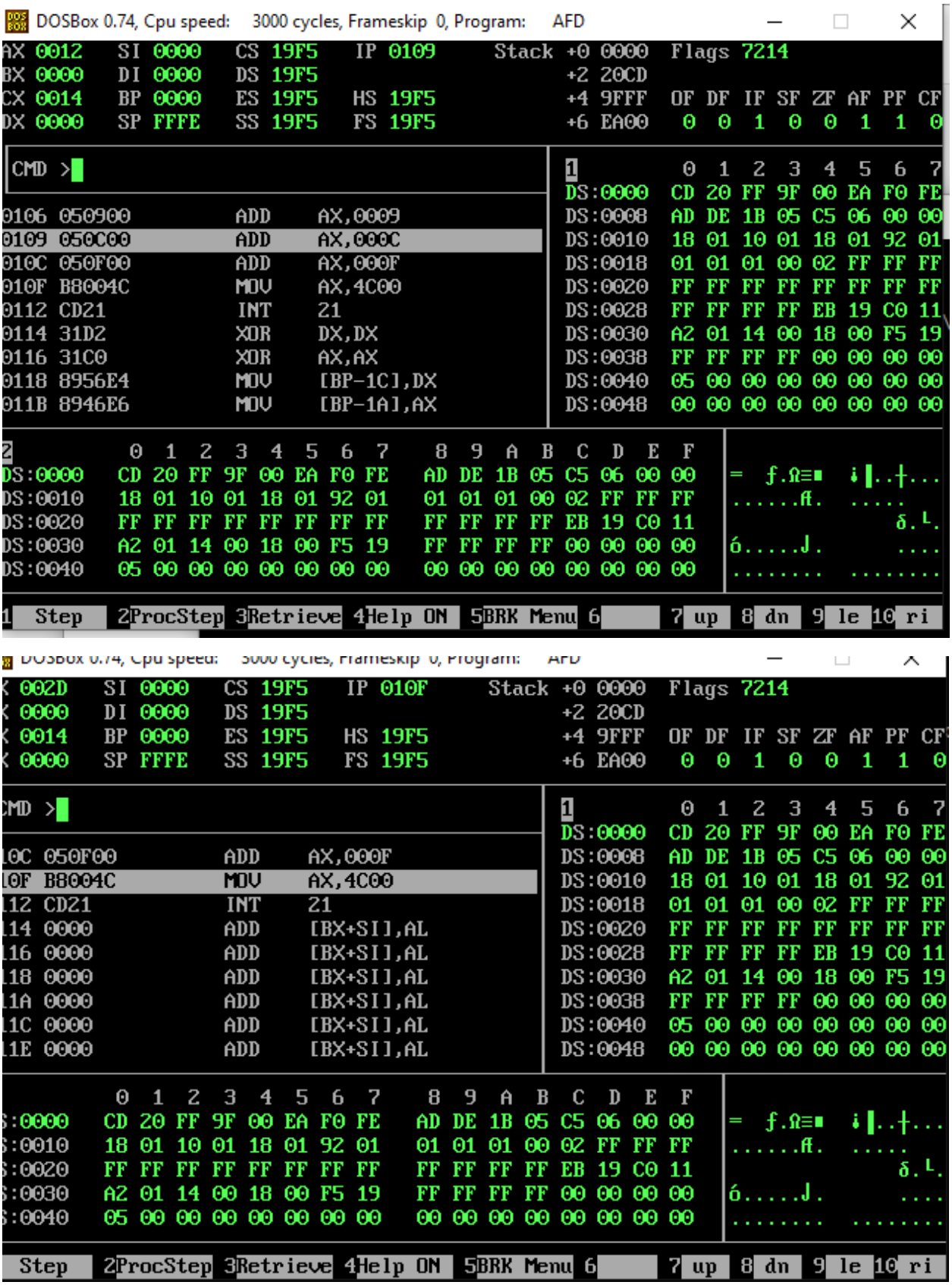
Help: [Approach 1] Can you do this using two registers only? [Approach 2] Can you do this using one register only if we have **add ax, 3** available in our instruction set? Try both of these approaches and watch the first five entries of table of 3 in AX.


DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD
AX 0009 SI 0000 CS 19F5 IP 0108 Stack +0 0000 Flags 7204
BX 0006 DI 0000 DS 19F5 +2 20CD
CX 0018 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 1 0
CMD >
0106 01D8 ADD AX,BX
0108 BB0900 MOV BX,0009
010B 01D8 ADD AX,BX
010D 050C00 ADD AX,000C
0110 050F00 ADD AX,000F
0113 B8004C MOV AX,4C00
0116 CD21 INT 21
0118 0000 ADD [BX+SI],AL
011A 0000 ADD [BX+SI],AL
DS:0000 CD 20 FF 9F 00 EA F0 FE
DS:0008 AD DE 1B 05 C5 06 00 00
DS:0010 18 01 10 01 18 01 92 01
DS:0018 01 01 01 00 02 FF FF FF
DS:0020 FF FF FF FF FF FF FF FF
DS:0028 FF FF FF FF EB 19 C0 11
DS:0030 A2 01 14 00 18 00 F5 19
DS:0038 FF FF FF FF 00 00 00 00
DS:0040 05 00 00 00 00 00 00 00
DS:0048 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, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD
AX 0012 SI 0000 CS 19F5 IP 010D Stack +0 0000 Flags 7214
BX 0009 DI 0000 DS 19F5 +2 20CD
CX 0018 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 1 1 0
CMD >
010B 01D8 ADD AX,BX
010D 050C00 ADD AX,000C
0110 050F00 ADD AX,000F
0113 B8004C MOV AX,4C00
0116 CD21 INT 21
0118 0000 ADD [BX+SI],AL
011A 0000 ADD [BX+SI],AL
011C 0000 ADD [BX+SI],AL
011E 0000 ADD [BX+SI],AL
DS:0000 CD 20 FF 9F 00 EA F0 FE
DS:0008 AD DE 1B 05 C5 06 00 00
DS:0010 18 01 10 01 18 01 92 01
DS:0018 01 01 01 00 02 FF FF FF
DS:0020 FF FF FF FF FF FF FF FF
DS:0028 FF FF FF FF EB 19 C0 11
DS:0030 A2 01 14 00 18 00 F5 19
DS:0038 FF FF FF FF 00 00 00 00
DS:0040 05 00 00 00 00 00 00 00
DS:0048 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, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD
AX 002D SI 0000 CS 19F5 IP 0113 Stack +0 0000 Flags 7214
BX 0009 DI 0000 DS 19F5 +2 20CD
CX 0018 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 1 1 0
CMD >
0110 050F00 ADD AX,000F
0113 B8004C MOV AX,4C00
0116 CD21 INT 21
0118 0000 ADD [BX+SI],AL
011A 0000 ADD [BX+SI],AL
011C 0000 ADD [BX+SI],AL
011E 0000 ADD [BX+SI],AL
0120 0000 ADD [BX+SI],AL
0122 0000 ADD [BX+SI],AL
1 0 1 2 3 4 5 6 7
DS:0000 CD 20 FF 9F 00 EA F0 FE
DS:0008 AD DE 1B 05 C5 06 00 00
DS:0010 18 01 10 01 18 01 92 01
DS:0018 01 01 01 00 02 FF FF FF
DS:0020 FF FF FF FF FF FF FF FF
DS:0028 FF FF FF FF EB 19 C0 11
DS:0030 A2 01 14 00 18 00 F5 19
DS:0038 FF FF FF FF 00 00 00 00
DS:0040 05 00 00 00 00 00 00 00
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
DS:0010 18 01 10 01 18 01 92 01 01 01 01 00 02 FF FF FF
DS:0020 FF FF FF FF FF FF FF FF FF FF FF FF EB 19 C0 11
DS:0030 A2 01 14 00 18 00 F5 19 FF FF FF FF 00 00 00 00
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

Approach 2


DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD
AX 0012 SI 0000 CS 19F5 IP 0109 Stack +0 0000 Flags 7214
BX 0000 DI 0000 DS 19F5 +2 20CD
CX 0014 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 1 1 0
CMD >
0106 050900 ADD AX,0009
0109 050C00 ADD AX,000C
010C 050F00 ADD AX,000F
010F B8004C MOV AX,4C00
0112 CD21 INT 21
0114 31D2 XOR DX,DX
0116 31C0 XOR AX,AX
0118 8956E4 MOV [BP-1C],DX
011B 8946E6 MOV [BP-1A],AX
DS:0000 CD 20 FF 9F 00 EA F0 FE
DS:0008 AD DE 1B 05 C5 06 00 00
DS:0010 18 01 10 01 18 01 92 01
DS:0018 01 01 01 00 02 FF FF FF
DS:0020 FF FF FF FF FF FF FF FF
DS:0028 FF FF FF FF EB 19 C0 11
DS:0030 A2 01 14 00 18 00 F5 19
DS:0038 FF FF FF FF 00 00 00 00
DS:0040 05 00 00 00 00 00 00 00
DS:0048 00 00 00 00 00 00 00 00
DS:0000 CD 20 FF 9F 00 EA F0 FE AD DE 1B 05 C5 06 00 00
DS:0010 18 01 10 01 18 01 92 01 01 01 01 00 02 FF FF FF
DS:0020 FF FF FF FF FF FF FF FF FF FF FF FF EB 19 C0 11
DS:0030 A2 01 14 00 18 00 F5 19 FF FF FF FF 00 00 00 00
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
AX 002D SI 0000 CS 19F5 IP 010F Stack +0 0000 Flags 7214
BX 0000 DI 0000 DS 19F5 +2 20CD
CX 0014 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 1 1 0
CMD >
010C 050F00 ADD AX,000F
010F B8004C MOV AX,4C00
0112 CD21 INT 21
0114 0000 ADD [BX+SI],AL
0116 0000 ADD [BX+SI],AL
0118 0000 ADD [BX+SI],AL
011A 0000 ADD [BX+SI],AL
011C 0000 ADD [BX+SI],AL
011E 0000 ADD [BX+SI],AL
DS:0000 CD 20 FF 9F 00 EA F0 FE
DS:0008 AD DE 1B 05 C5 06 00 00
DS:0010 18 01 10 01 18 01 92 01
DS:0018 01 01 01 00 02 FF FF FF
DS:0020 FF FF FF FF FF FF FF FF
DS:0028 FF FF FF FF EB 19 C0 11
DS:0030 A2 01 14 00 18 00 F5 19
DS:0038 FF FF FF FF 00 00 00 00
DS:0040 05 00 00 00 00 00 00 00
DS:0048 00 00 00 00 00 00 00 00
DS:0000 CD 20 FF 9F 00 EA F0 FE AD DE 1B 05 C5 06 00 00
DS:0010 18 01 10 01 18 01 92 01 01 01 01 00 02 FF FF FF
DS:0020 FF FF FF FF FF FF FF FF FF FF FF FF EB 19 C0 11
DS:0030 A2 01 14 00 18 00 F5 19 FF FF FF FF 00 00 00 00
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