Coal

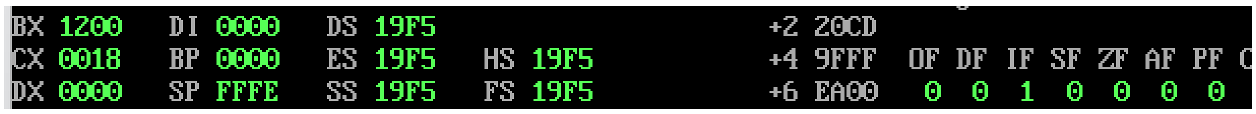
LAB TASK 4

NAME SAJID ISLAM ROLLNO 24p-0745 INSTRUCTOR Muhammad Hassan

Question 1

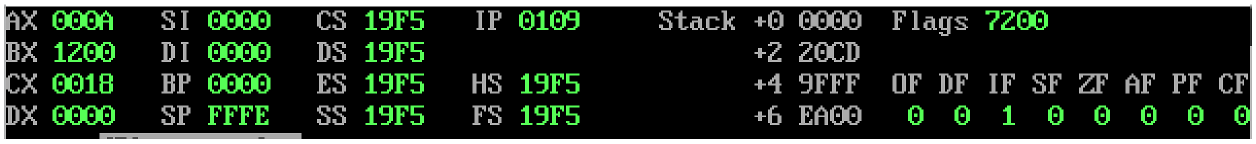
```
[org 0x100]

mov bx, 1200h ; bx gets address 1200h
mov word [bx], 10 ; 10 is then stored as value in it
mov ax, [bx]

add bx, 2 ; we now move to address 1202h
mov word [bx], 20 ; store a new value
mov cx, [bx]

mov ax, 0x4c00
int 0x21
```

First we point **BX** to address 1200h and store the value 10 there.



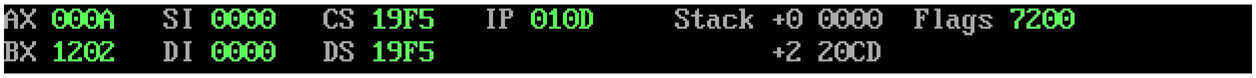
```
BX 1200    DI 0000    DS 19F5                        +2 20CD
CX 0018    BP 0000    ES 19F5    HS 19F5             +4 9FFF   OF DF IF SF ZF AF PF C
DX 0000    SP FFFE    SS 19F5    FS 19F5             +6 EA00    0  0  1  0  0  0  0
```

Then we copy that value from memory into **AX**



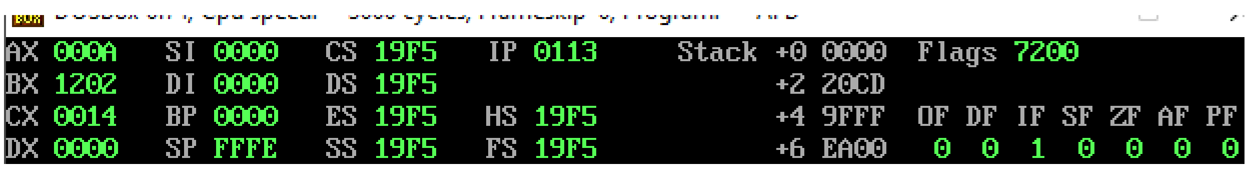
```
AX 000A    SI 0000    CS 19F5    IP 0109     Stack +0 0000   Flags 7200
BX 1200    DI 0000    DS 19F5                      +2 20CD
CX 0018    BP 0000    ES 19F5    HS 19F5           +4 9FFF   OF DF IF SF ZF AF PF CF
DX 0000    SP FFFE    SS 19F5    FS 19F5           +6 EA00    0  0  1  0  0  0  0  0
```

Next we move BX to the next word location (1202h) and store the value 20.



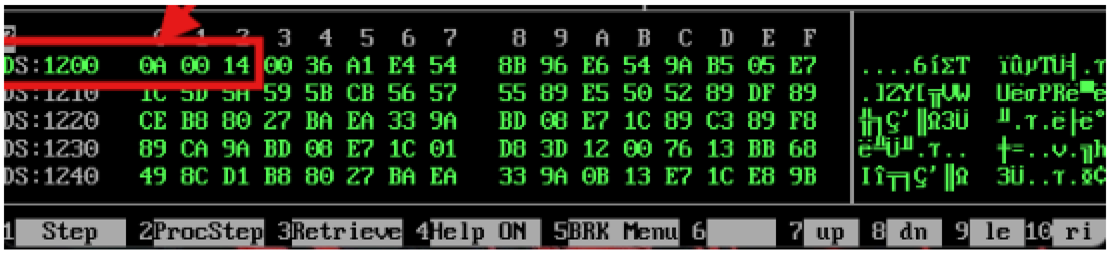
```
AX 000A    SI 0000    CS 19F5    IP 010D      Stack +0 0000  Flags 7200
BX 1202    DI 0000    DS 19F5                       +2 20CD
```

Finally we copy that value into **CX** and and the value of bx also changes to 1202 from 1200.



```
AX 000A    SI 0000    CS 19F5    IP 0113      Stack +0 0000  Flags 7200
BX 1202    DI 0000    DS 19F5                       +2 20CD
CX 0014    BP 0000    ES 19F5    HS 19F5            +4 9FFF  OF DF IF SF ZF AF PF
DX 0000    SP FFFE    SS 19F5    FS 19F5            +6 EA00   0  0  1  0  0  0  0
```

Final SS after program completion



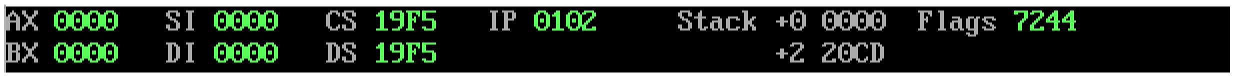
```
            0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F
DS:1200    0A 00 14 00 36 A1 E4 54  8B 96 E6 54 9A B5 05 E7   ....6íΣT  ïûµTU╡.τ
DS:1210    1C 5D 5A 59 5B CB 56 57  55 89 E5 50 52 89 DF 89   .]ZY[╥VW  UëσPRë▀ë
DS:1220    CE B8 80 27 BA EA 33 9A  BD 08 E7 1C 89 C3 89 F8   ╬╕Ç'║Ω3Ü  ╜.τ.ë├ë°
DS:1230    89 CA 9A BD 08 E7 1C 01  D8 3D 12 00 76 13 BB 68   ë╩Ü╜.τ..  ╪=..v.╗h
DS:1240    49 8C D1 B8 80 27 BA EA  33 9A 0B 13 E7 1C E8 9B   Iî╤╕Ç'║Ω  3Ü..τ.Φ¢
1 Step  2ProcStep 3Retrieve 4Help ON 5BRK Menu 6  7 up  8 dn  9 le 10 ri
```

Question 2

Code snippet

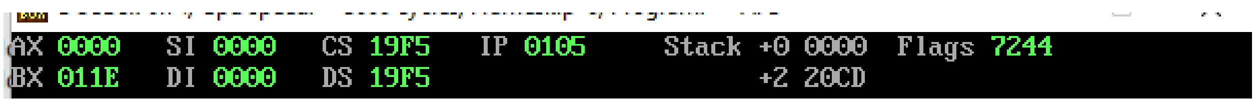
```
[org 0x100]

xor ax, ax

mov bx, val

add ax, [bx]
add bx, 2

add ax, [bx]
add bx, 2

add ax, [bx]
add bx, 2

mov [bx], ax

mov ax, 0x4c00
int 0x21

val: dw 10, 20, 30, 0
```

1. First we clear **AX** so it starts from zero.



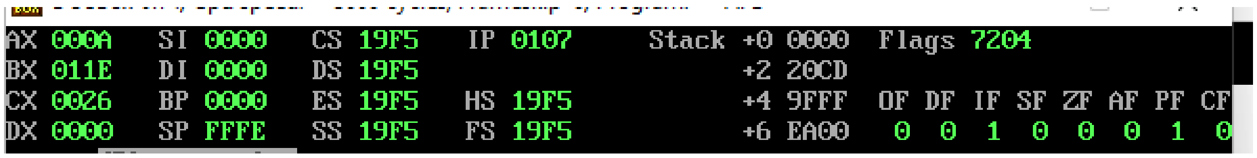
```
AX 0000   SI 0000   CS 19F5   IP 0102      Stack +0 0000  Flags 7244
BX 0000   DI 0000   DS 19F5                      +2 20CD
```

2. Then we point **BX** to the memory where our numbers are stored.



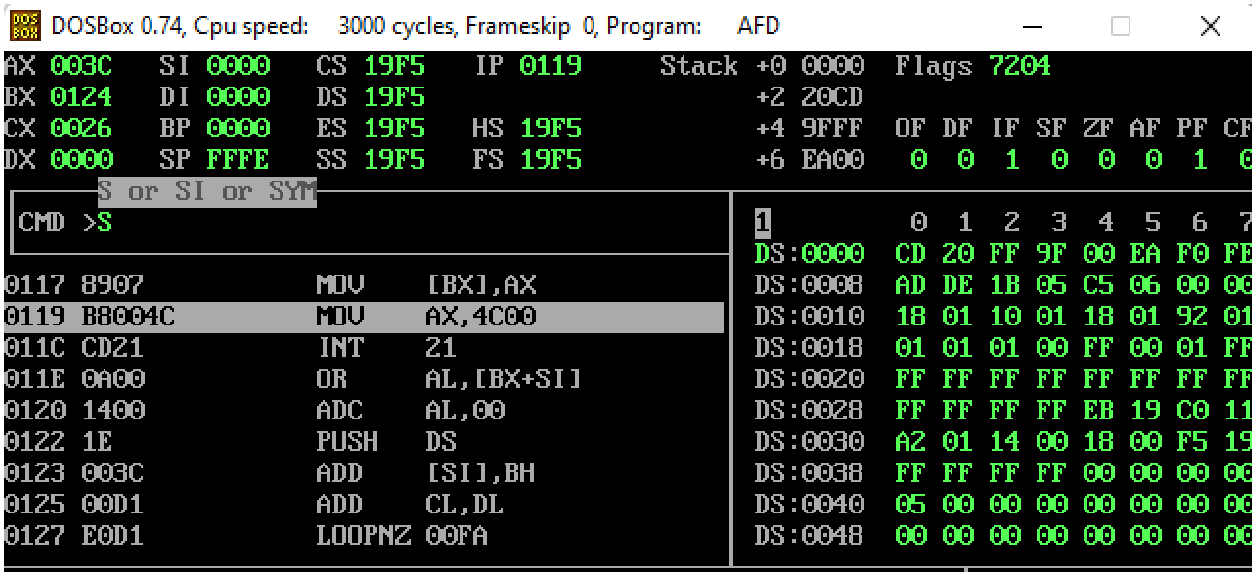
```
AX 0000   SI 0000   CS 19F5   IP 0105      Stack +0 0000  Flags 7244
BX 011E   DI 0000   DS 19F5                      +2 20CD
```

3. We load each number from memory into **AX** one by one (using BX as a pointer) and keep adding them.



```
AX 000A   SI 0000   CS 19F5   IP 0107      Stack +0 0000  Flags 7204
BX 011E   DI 0000   DS 19F5                      +2 20CD
CX 0026   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
```

4. After adding all three numbers, we store the final sum back into the last memory location.



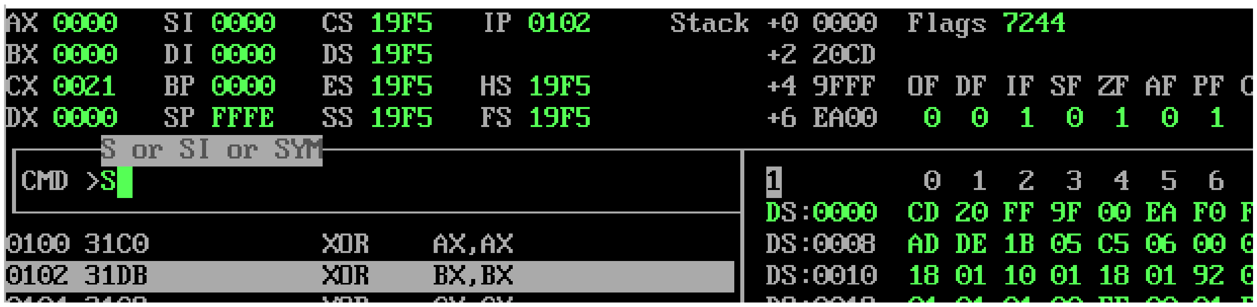
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD

```
AX 003C   SI 0000   CS 19F5   IP 0119      Stack +0 0000  Flags 7204
BX 0124   DI 0000   DS 19F5                      +2 20CD
CX 0026   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
      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
0117 8907          MOV    [BX],AX            DS:0008   AD DE 1B 05 C5 06 00 00
0119 B8004C        MOV    AX,4C00            DS:0010   18 01 10 01 18 01 92 01
011C CD21          INT    21                 DS:0018   01 01 01 00 FF 00 01 FF
011E 0A00          OR     AL,[BX+SI]         DS:0020   FF FF FF FF FF FF FF FF
0120 1400          ADC    AL,00              DS:0028   FF FF FF FF EB 19 C0 11
0122 1E            PUSH   DS                 DS:0030   A2 01 14 00 18 00 F5 19
0123 003C          ADD    [SI],BH            DS:0038   FF FF FF FF 00 00 00 00
0125 00D1          ADD    CL,DL              DS:0040   05 00 00 00 00 00 00 00
0127 E0D1          LOOPNZ 00FA               DS:0048   00 00 00 00 00 00 00 00
```

Question 3

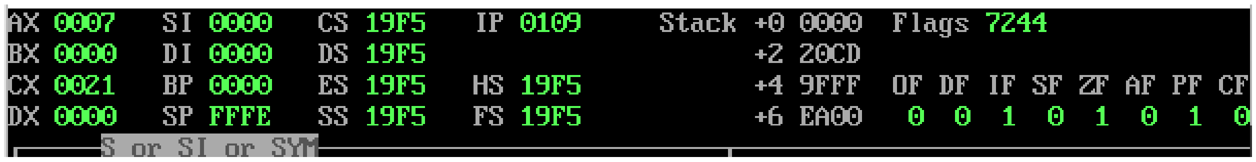
```
[org 0x100]

xor ax,ax
xor bx,bx
xor cx,cx

mov al,[data]
mov bl,[data + 1]
mov cl,[data + 2]

add al,bl
add al,cl

mov [data + 3],al

mov ax,0x4c00
int 0x21

data: db 7, 12, 20, 0
```

We clear the registers **AX, BX, and CX** to start fresh.

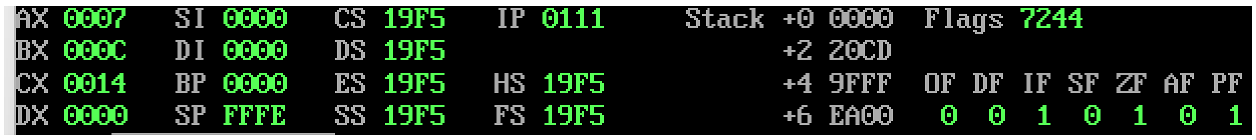


```
AX 0000   SI 0000   CS 19F5   IP 0102      Stack +0 0000  Flags 7244
BX 0000   DI 0000   DS 19F5                      +2 20CD
CX 0021   BP 0000   ES 19F5   HS 19F5            +4 9FFF  OF DF IF SF ZF AF PF C
DX 0000   SP FFFE   SS 19F5   FS 19F5            +6 EA00   0  0  1  0  1  0  1
           S or SI or SYM
 CMD >S                                           1        0  1  2  3  4  5  6
                                                  DS:0000  CD 20 FF 9F 00 EA F0 F
0100 31C0           XOR     AX,AX                 DS:0008  AD DE 1B 05 C5 06 00 0
0102 31DB           XOR     BX,BX                 DS:0010  18 01 10 01 18 01 92 0
```

Then we load the three numbers from memory (7, 12, 20) into **AL, BL, and CL**.

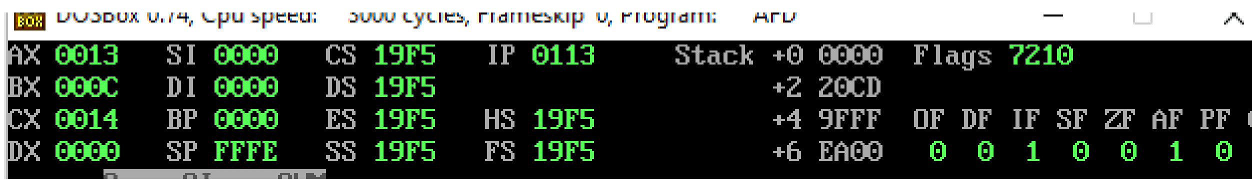


```
AX 0007   SI 0000   CS 19F5   IP 0109      Stack +0 0000  Flags 7244
BX 0000   DI 0000   DS 19F5                      +2 20CD
CX 0021   BP 0000   ES 19F5   HS 19F5            +4 9FFF  OF DF IF SF ZF AF PF CF
DX 0000   SP FFFE   SS 19F5   FS 19F5            +6 EA00   0  0  1  0  1  0  1  0
           S or SI or SYM
```



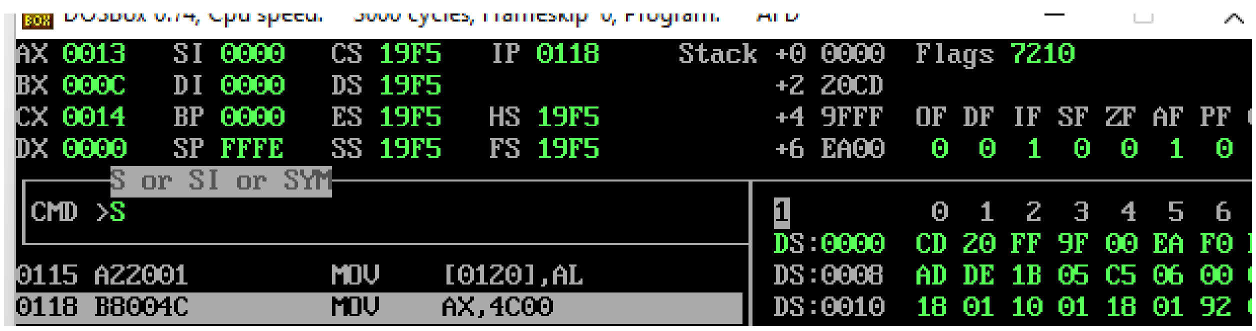
```
AX 0007   SI 0000   CS 19F5   IP 0111     Stack +0 0000  Flags 7244
BX 000C   DI 0000   DS 19F5                     +2 20CD
CX 0014   BP 0000   ES 19F5   HS 19F5           +4 9FFF  OF DF IF SF ZF AF PF
DX 0000   SP FFFE   SS 19F5   FS 19F5           +6 EA00   0  0  1  0  1  0  1
```

We add them together step by step using **AL** as the accumulator.



```
DOSBox 0.74, Cpu speed:   3000 cycles, Frameskip  0, Program:   AFD
AX 0013   SI 0000   CS 19F5   IP 0113     Stack +0 0000  Flags 7210
BX 000C   DI 0000   DS 19F5                     +2 20CD
CX 0014   BP 0000   ES 19F5   HS 19F5           +4 9FFF  OF DF IF SF ZF AF PF
DX 0000   SP FFFE   SS 19F5   FS 19F5           +6 EA00   0  0  1  0  0  1  0
```

Finally, the total sum is stored back in memory at the last location.

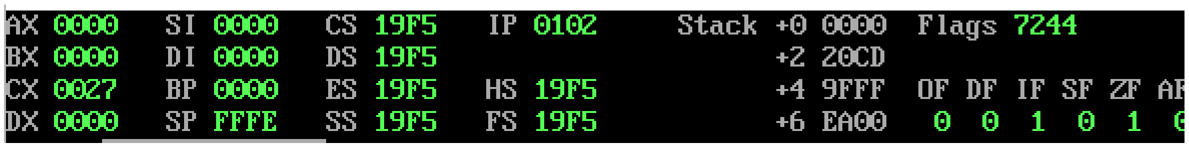


```
DOSBox 0.74, Cpu speed:   3000 cycles, Frameskip  0, Program:   AFD
AX 0013   SI 0000   CS 19F5   IP 0118     Stack +0 0000  Flags 7210
BX 000C   DI 0000   DS 19F5                     +2 20CD
CX 0014   BP 0000   ES 19F5   HS 19F5           +4 9FFF  OF DF IF SF ZF AF PF
DX 0000   SP FFFE   SS 19F5   FS 19F5           +6 EA00   0  0  1  0  0  1  0
          S or SI or SYM
CMD >S                                    1          0  1  2  3  4  5  6
                                          DS:0000   CD 20 FF 9F 00 EA F0
0115 A22001          MOV    [0120],AL     DS:0008   AD DE 1B 05 C5 06 00
0118 B8004C          MOV    AX,4C00       DS:0010   18 01 10 01 18 01 92
```

Question 4

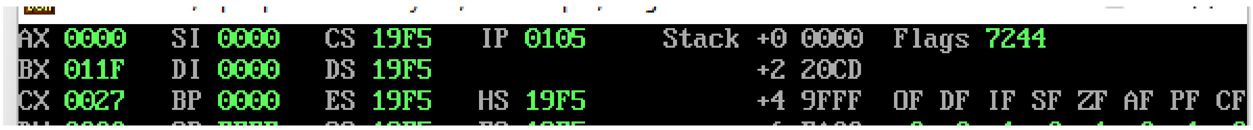
```
[org 0x100]

xor ax,ax

mov bx, array

add ax,[bx]
add bx,2

add ax,[bx]
add bx,2

add ax,[bx]
add bx,2

mov [result],ax

mov ax,0x4c00
int 0x21

array: dw 7, 12, 20
result: dw 0
```

We clear **AX** so it starts from zero.



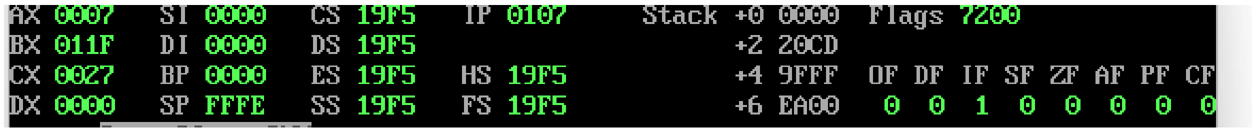
```
AX 0000   SI 0000   CS 19F5   IP 0102     Stack +0 0000  Flags 7244
BX 0000   DI 0000   DS 19F5                     +2 20CD
CX 0027   BP 0000   ES 19F5   HS 19F5           +4 9FFF  OF DF IF SF ZF AF
DX 0000   SP FFFE   SS 19F5   FS 19F5           +6 EA00   0  0  1  0  1  0
```

Then we point **BX** to the start of the array where the three word values are stored.

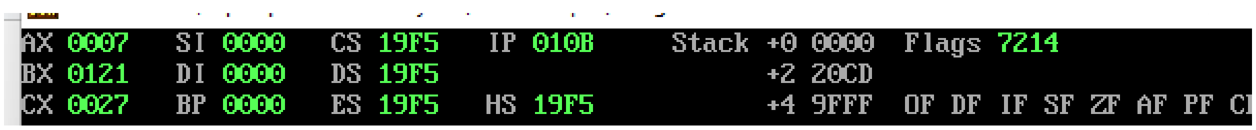


```
AX 0000   SI 0000   CS 19F5   IP 0105     Stack +0 0000  Flags 7244
BX 011F   DI 0000   DS 19F5                     +2 20CD
CX 0027   BP 0000   ES 19F5   HS 19F5           +4 9FFF  OF DF IF SF ZF AF PF CF
```

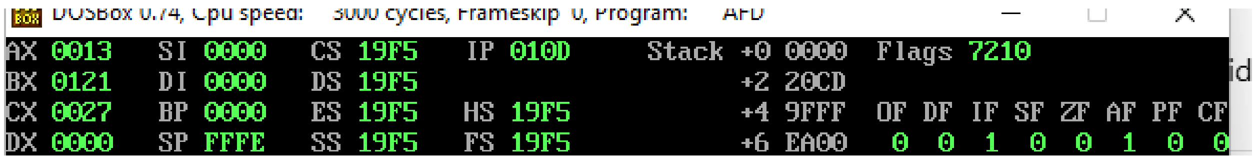
Using **BX as a pointer**, we fetch each number one by one, add it into **AX**, and move BX forward each time.



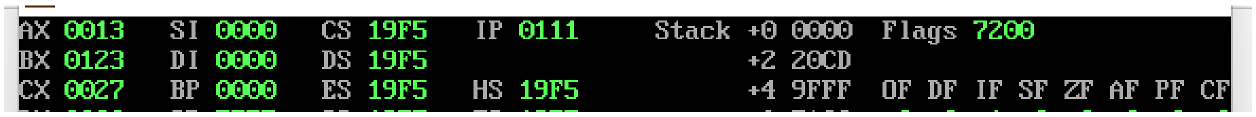
```
AX 0007   SI 0000   CS 19F5   IP 0107      Stack +0 0000  Flags 7200
BX 011F   DI 0000   DS 19F5                      +2 20CD
CX 0027   BP 0000   ES 19F5   HS 19F5            +4 9FFF   OF DF IF SF ZF AF PF CF
DX 0000   SP FFFE   SS 19F5   FS 19F5            +6 EA00    0  0  1  0  0  0  0  0
```



```
AX 0007   SI 0000   CS 19F5   IP 010B      Stack +0 0000  Flags 7214
BX 0121   DI 0000   DS 19F5                      +2 20CD
CX 0027   BP 0000   ES 19F5   HS 19F5            +4 9FFF   OF DF IF SF ZF AF PF CI
```

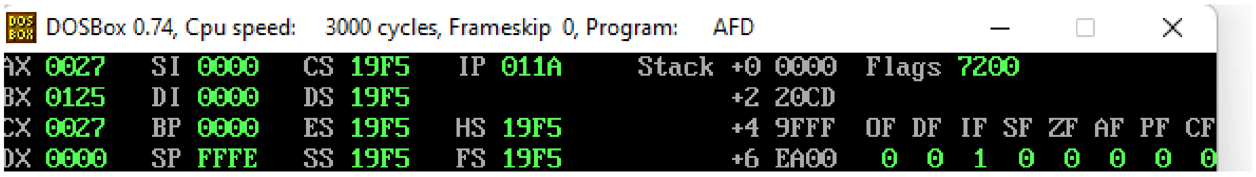


```
DOSBox 0.74, Cpu speed:    3000 cycles, Frameskip  0, Program:    AFD
AX 0013   SI 0000   CS 19F5   IP 010D      Stack +0 0000  Flags 7210
BX 0121   DI 0000   DS 19F5                      +2 20CD
CX 0027   BP 0000   ES 19F5   HS 19F5            +4 9FFF   OF DF IF SF ZF AF PF CF
DX 0000   SP FFFE   SS 19F5   FS 19F5            +6 EA00    0  0  1  0  0  1  0  0
```

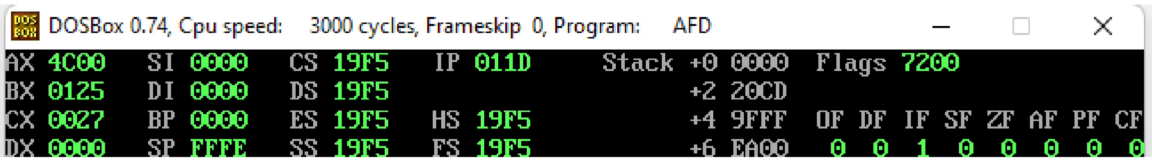


```
AX 0013   SI 0000   CS 19F5   IP 0111      Stack +0 0000  Flags 7200
BX 0123   DI 0000   DS 19F5                      +2 20CD
CX 0027   BP 0000   ES 19F5   HS 19F5            +4 9FFF   OF DF IF SF ZF AF PF CF
```

After all additions, we save the final sum into the separate memory variable called **result**.



```
DOSBox 0.74, Cpu speed:    3000 cycles, Frameskip  0, Program:    AFD
AX 0027   SI 0000   CS 19F5   IP 011A      Stack +0 0000  Flags 7200
BX 0125   DI 0000   DS 19F5                      +2 20CD
CX 0027   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
```



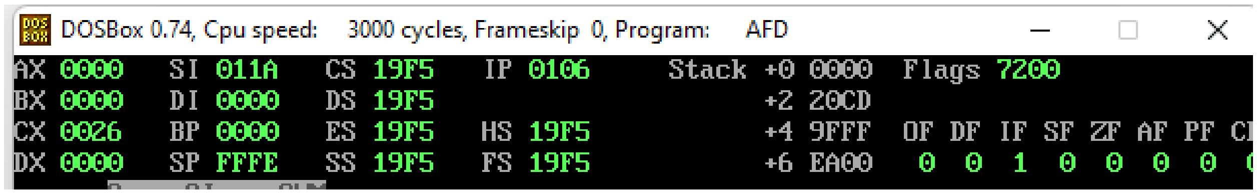
```
DOSBox 0.74, Cpu speed:    3000 cycles, Frameskip  0, Program:    AFD
AX 4C00   SI 0000   CS 19F5   IP 011D      Stack +0 0000  Flags 7200
BX 0125   DI 0000   DS 19F5                      +2 20CD
CX 0027   BP 0000   ES 19F5   HS 19F5            +4 9FFF   OF DF IF SF ZF AF PF CF
DX 0000   SP FFFE   SS 19F5   FS 19F5            +6 EA00    0  0  1  0  0  0  0  0
```

Question 5

```
[org 0x100]

mov ax,0 ; adding values
mov si,array ; for moving through values
mov cx,5 ; to run the loop

loop_start:
    add ax, [si]
    add si, 2

    dec cx
    jnz loop_start

mov [result],ax

mov ax,0x4c00
int 0x21

array: dw 7, 12, 20, 35, 9
result: dw 0
```

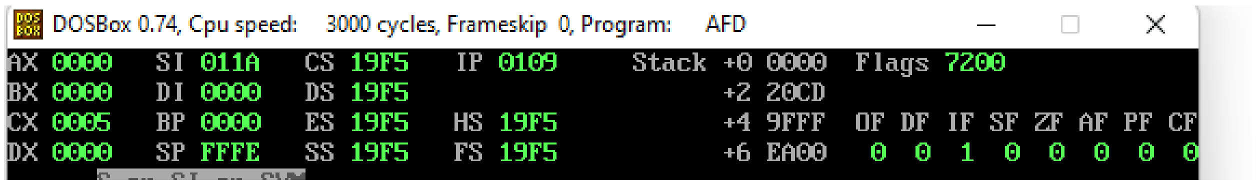
We clear **AX** to start adding from zero.

Then we point **SI** to the start of the array and set **CX = 5** for the loop counter.



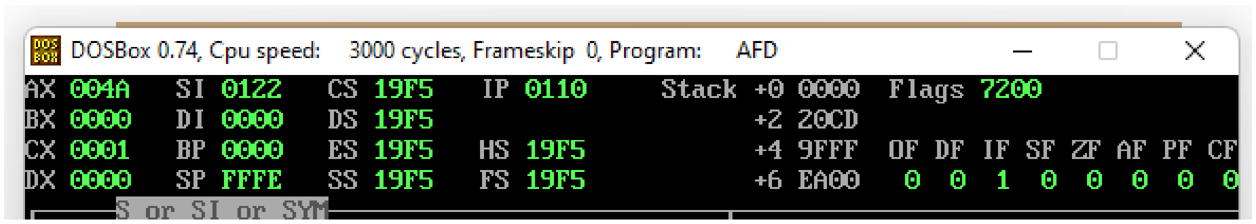
```
DOSBox 0.74, Cpu speed:    3000 cycles, Frameskip  0, Program:    AFD
AX 0000    SI 011A    CS 19F5    IP 0106      Stack +0 0000   Flags 7200
BX 0000    DI 0000    DS 19F5                       +2 20CD
CX 0026    BP 0000    ES 19F5    HS 19F5            +4 9FFF   OF DF IF SF ZF AF PF CF
DX 0000    SP FFFE    SS 19F5    FS 19F5            +6 EA00    0  0  1  0  0  0  0  0
```

Inside the loop, we add each array value into **AX**, move **SI** to the next word, and decrease **CX**.


DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD

```
AX 0000   SI 011A   CS 19F5   IP 0109      Stack +0 0000  Flags 7200
BX 0000   DI 0000   DS 19F5                      +2 20CD
CX 0005   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
```

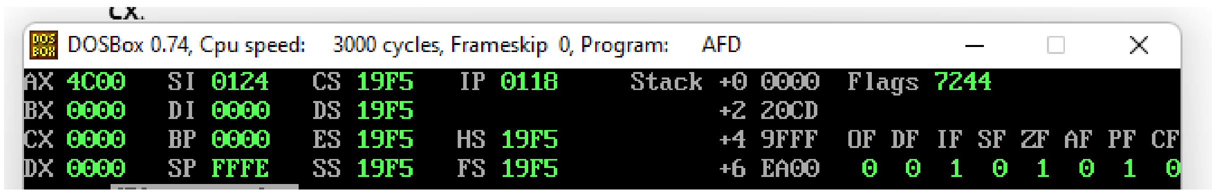
The loop keeps running until all 5 numbers are added.


DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD

```
AX 004A   SI 0122   CS 19F5   IP 0110      Stack +0 0000  Flags 7200
BX 0000   DI 0000   DS 19F5                      +2 20CD
CX 0001   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
```

Counting register value is decreasing after each loop

Finally, the total sum is stored into the memory variable **result**


CX.

DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD

```
AX 4C00   SI 0124   CS 19F5   IP 0118      Stack +0 0000  Flags 7244
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  1  0  1  0
```