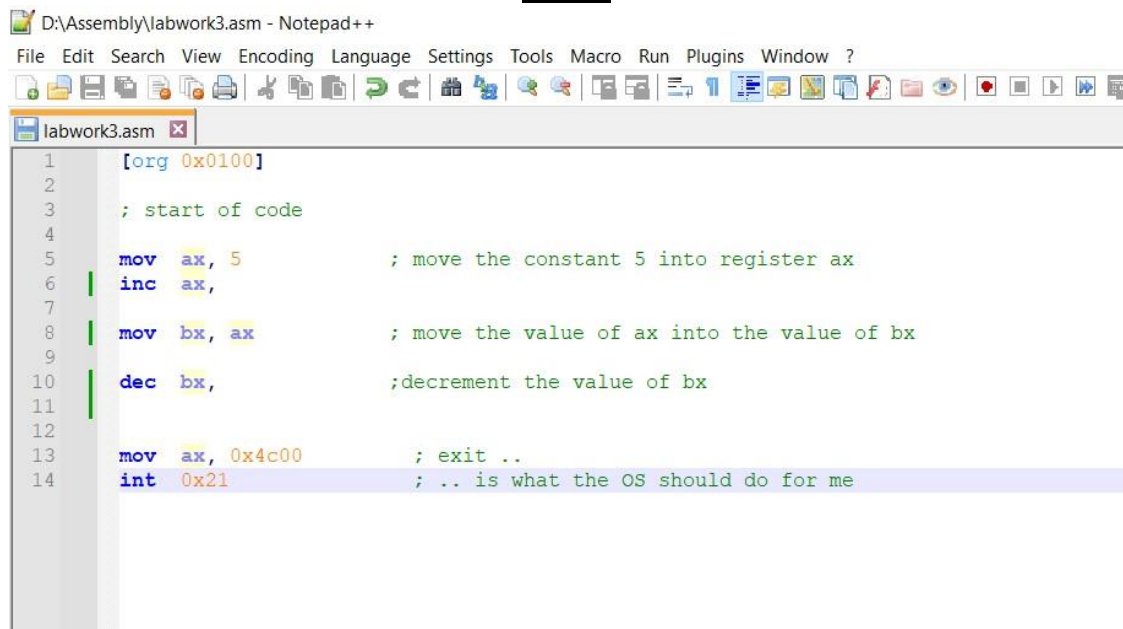


Lab 3

Execution Of Tasks Lab 3

Task 1

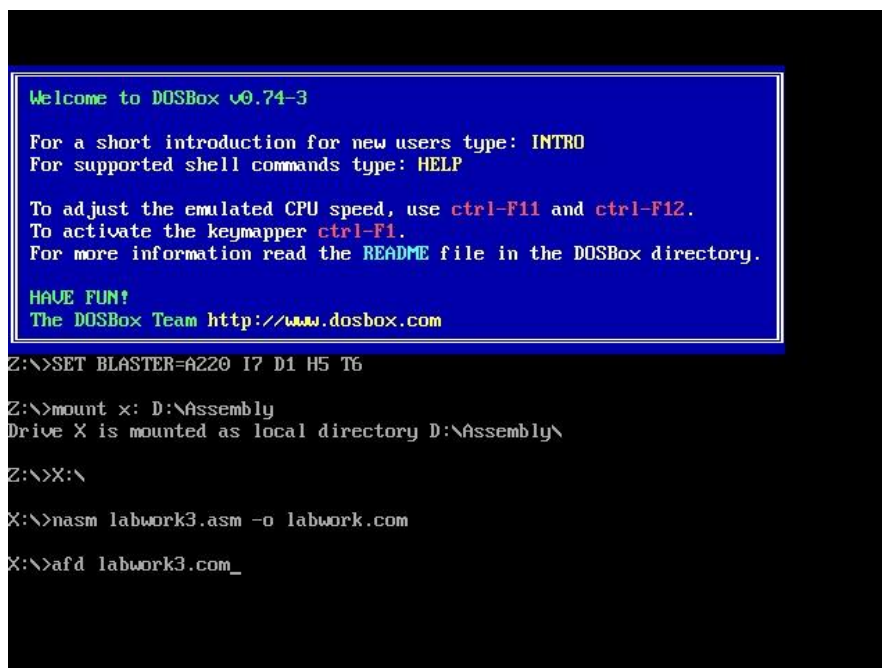


```
D:\Assembly\labwork3.asm - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
labwork3.asm
1  [org 0x0100]
2
3  ; start of code
4
5  mov ax, 5           ; move the constant 5 into register ax
6  inc ax,             ; increment the value of ax by 1
7
8  mov bx, ax          ; move the value of ax into the value of bx
9
10 dec bx,             ; decrement the value of bx by 1
11
12
13 mov ax, 0x4c00       ; exit ..
14 int 0x21             ; .. is what the OS should do for me
```

Explanation:

Here in this code, we first add 5 to register ax, and then using **inc** command we increment the value stored in ax by 1, after that we are moving the value from ax to bx register and then using **dec**, we decrement the value in bx by 1 as per question requirement.

Following are the screenshots of execution of code:



```
Welcome to DOSBox v0.74-3
For a short introduction for new users type: INTRO
For supported shell commands type: HELP

To adjust the emulated CPU speed, use ctrl-F11 and ctrl-F12.
To activate the keymapper ctrl-F1.
For more information read the README file in the DOSBox directory.

HAVE FUN!
The DOSBox Team http://www.dosbox.com

Z:\>SET BLASTER=A220 I7 D1 H5 T6

Z:\>mount x: D:\Assembly
Drive X is mounted as local directory D:\Assembly\

Z:\>X:\

X:\>nasm labwork3.asm -o labwork.com

X:\>afd labwork3.com_
```

```

AX 0006 SI 0000 CS 19F5 IP 0104 Stack +0 0000 Flags 7204
BX 0000 DI 0000 DS 19F5 +2 20CD
CX 000C BP 0000 ES 19F5 HS 19F5 +4 9FFF 0F 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 >
0103 40 INC AX
0104 89C3 MOV BX,AX
0106 4B DEC BX
0107 B8004C MOV AX,4C00
010A CD21 INT 21
010C 89C3 MOV BX,AX
010E 89D0 MOV AX,DX
0110 89DA MOV DX,BX
0112 EB04 JMP 0118

DS:0000 CD 20 FF 9F 00 EA F0 FE AD DE 1B 05 C5 06 00 00
DS:0008 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:0018 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:0028 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:0038 FF FF FF FF 00 00 00 00
DS:0040 05 00 00 00 00 00 00 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 = f.Ω≡ i |..†...
DS:0010 18 01 10 01 18 01 92 01 01 01 01 00 02 FF FF FF .....ff. ....
DS:0020 FF FF FF FF FF FF FF FF FF FF FF FF EB 19 C0 11 δ.L.
DS:0030 A2 01 14 00 18 00 F5 19 FF FF FF FF 00 00 00 00 6.....J. ....
DS:0040 05 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

```

The value is added and incremented in ax

```

AX 0006 SI 0000 CS 19F5 IP 0107 Stack +0 0000 Flags 7204
BX 0005 DI 0000 DS 19F5 +2 20CD
CX 000C BP 0000 ES 19F5 HS 19F5 +4 9FFF 0F 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 4B DEC BX
0107 B8004C MOV AX,4C00
010A CD21 INT 21
010C 89C3 MOV BX,AX
010E 89D0 MOV AX,DX
0110 89DA MOV DX,BX
0112 EB04 JMP 0118
0114 31D2 XOR DX,DX
0116 31C0 XOR AX,AX

DS:0000 CD 20 FF 9F 00 EA F0 FE AD DE 1B 05 C5 06 00 00 = f.Ω≡ i |..†...
DS:0008 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 .....ff. ....
DS:0018 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 δ.L.
DS:0028 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 6.....J. ....
DS:0038 FF FF FF FF 00 00 00 00
DS:0040 05 00 00 00 00 00 00 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 = f.Ω≡ i |..†...
DS:0010 18 01 10 01 18 01 92 01 01 01 01 00 02 FF FF FF .....ff. ....
DS:0020 FF FF FF FF FF FF FF FF FF FF FF FF EB 19 C0 11 δ.L.
DS:0030 A2 01 14 00 18 00 F5 19 FF FF FF FF 00 00 00 00 6.....J. ....
DS:0040 05 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

```

The value is moved to bx and decremented.

Task 2 Following

is the code of the task 2:

```

1  [org 0x0100]
2
3  ; start of code
4
5  mov bl, 4          ; move the constant 5 into register bl
6  mov dx, 6          ; move the constant 6 to dx
7
8  mov cl,bl          ; move dl to cl
9  mov ax,dx          ; move dx to ax
10
11 mov al,0x12        ; move 0x12 to al
12 mov ax,0x1234      ; move 0x1234 to ax
13 mov ax,0xffff      ; move 0xffff to ax
14
15 mov ax,0x4c00
16 int 0x21

```

Following are the screenshots of code execution and their explanation:

AX 0006	SI 0000	CS 19F5	IP 0109	Stack +0 0000	Flags 7200
BX 0004	DI 0000	DS 19F5		+2 20CD	
CX 0004	BP 0000	ES 19F5	HS 19F5	+4 9FFF	OF DF IF SF ZF AF PF CF
DX 0006	SP FFFE	SS 19F5	FS 19F5	+6 EA00	0 0 1 0 0 0 0 0

CMD >				1	0	1	2	3	4	5	6	7
0107 89D0				MOV	AX,DX	DS:0000 CD 20 FF 9F 00 EA FF FF						
0109 B012				MOV	AL,12	DS:0008 AD DE 1B 05 C5 06 00 00						
010B B83412				MOV	AX,1234	DS:0010 18 01 10 01 18 01 92 01						
010E B8FFFF				MOV	AX,FFFF	DS:0018 01 01 01 00 02 FF FF FF						
0111 B8004C				MOV	AX,4C00	DS:0020 FF FF FF FF FF FF FF FF						
0114 CD21				INT	21	DS:0028 FF FF FF FF EB 19 E6 11						
0116 31C0				XOR	AX,AX	DS:0030 A2 01 14 00 18 00 F5 19						
0118 8956E4				MOV	[BP-1C1],DX	DS:0038 FF FF FF FF 00 00 00 00						
011B 8946E6				MOV	[BP-1A1],AX	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	FF	FF	AD	DE	1B	05	C5	06	00	00	= f.n i ..+...
DS:0010	18	01	10	01	18	01	92	01	01	01	01	00	02	FF	FF	FFf.
DS:0020	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	EB	19	E6	11δ.μ.
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	2 ProcStep	3 Retrieve	4 Help ON	5 BRK Menu	6	7 up	8 dn	9 le	10 ri
--------	------------	------------	-----------	------------	---	------	------	------	-------

- The value is added to bl ,4 (bl has half of memory size as of bx)
- The value is added to bx, 6
- The value is moved from bl to cl (same as bl, cl also is a register containing half of memory as cx)
- The value is moved from dx to ax


```

AX 0012  SI 0000  CS 19F5  IP 010B  Stack +0 0000  Flags 7200
BX 0004  DI 0000  DS 19F5          +2 20CD
CX 0004  BP 0000  ES 19F5  HS 19F5  +4 9FFF  OF DF IF SF ZF AF PF CF
DX 0006  SP FFFE  SS 19F5  FS 19F5  +6 EA00  0 0 1 0 0 0 0 0

CMD >

0109 B012      MOV     AL,12
010B B83412     MOV     AX,1234
010E B8FFFF     MOV     AX,FFFF
0111 B8004C     MOV     AX,4C00
0114 CD21       INT     21
0116 31C0       XOR     AX,AX
0118 8956E4     MOV     [BP-1C],DX
011B 8946E6     MOV     [BP-1A],AX
011E C746F60000 MOV     [BP-0A],0000

DS:0000  CD 20 FF 9F 00 EA FF FF  AD DE 1B 05 C5 06 00 00  = f.ñ  i|.+....
DS:0010  18 01 10 01 18 01 92 01  01 01 01 00 02 FF FF FF  .....ñ. ....
DS:0020  FF FF FF FF FF FF FF FF  FF FF FF FF EB 19 E6 11  .....δ.μ.
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

```

The value 0x12 is moved to al.

```

AX 1234  SI 0000  CS 19F5  IP 010E  Stack +0 0000  Flags 7200
BX 0004  DI 0000  DS 19F5          +2 20CD
CX 0004  BP 0000  ES 19F5  HS 19F5  +4 9FFF  OF DF IF SF ZF AF PF CF
DX 0006  SP FFFE  SS 19F5  FS 19F5  +6 EA00  0 0 1 0 0 0 0 0

CMD >

010B B83412     MOV     AX,1234
010E B8FFFF     MOV     AX,FFFF
0111 B8004C     MOV     AX,4C00
0114 CD21       INT     21
0116 31C0       XOR     AX,AX
0118 8956E4     MOV     [BP-1C],DX
011B 8946E6     MOV     [BP-1A],AX
011E C746F60000 MOV     [BP-0A],0000
0123 8B46F6     MOV     AX,[BP-0A]

DS:0000  CD 20 FF 9F 00 EA FF FF  AD DE 1B 05 C5 06 00 00  = f.ñ  i|.+....
DS:0010  18 01 10 01 18 01 92 01  01 01 01 00 02 FF FF FF  .....ñ. ....
DS:0020  FF FF FF FF FF FF FF FF  FF FF FF FF EB 19 E6 11  .....δ.μ.
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

```

The value 0x1234 is moved to al.

```

AX FFFF  SI 0000  CS 19F5  IP 0111  Stack +0 0000  Flags 7200
BX 0004  DI 0000  DS 19F5  +2 20CD
CX 0004  BP 0000  ES 19F5  HS 19F5  +4 9FFF  OF DF IF SF ZF AF PF CF
DX 0006  SP FFFE  SS 19F5  FS 19F5  +6 EA00  0 0 1 0 0 0 0 0

CMD >

010E B8FFFF  MOV  AX,FFFF
0111 B8004C  MOV  AX,4C00
0114 CD21    INT  21
0116 31C0    XOR  AX,AX
0118 8956E4  MOV  [BP-1C],DX
011B 8946E6  MOV  [BP-1A],AX
011E C746F60000 MOV  [BP-0A],0000
0123 8B46F6  MOV  AX,[BP-0A]
0126 D1E0    SHL  AX,1

1  0 1 2 3 4 5 6 7
DS:0000 CD 20 FF 9F 00 EA FF FF
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 E6 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 FF FF  AD DE 1B 05 C5 06 00 00  = f.n  i |..+...
DS:0010 18 01 10 01 18 01 92 01  01 01 01 00 02 FF FF FF  .....f. ....
DS:0020 FF FF FF FF FF FF FF FF  FF FF FF FF EB 19 E6 11  δ.μ.
DS:0030 A2 01 14 00 18 00 F5 19  FF FF FF FF 00 00 00 00  6.....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

```

The value 0xffff is moved to al register.

Task 3

```

D:\Assembly\labwork3.asm - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window

labwork3.asm
1  [org 0x0100]
2
3  ; start of code
4
5  mov ax, 6 ;move the constant 6 into register ax
6  mov bx, 0
7  add ax, 6 ;add 6 to ax
8  inc bx
9  add ax, 6 ;add 6 to ax
10 inc bx
11 add ax, 6 ;add 6 to ax
12 inc bx
13 add ax, 6 ;add 6 to ax
14 inc bx
15 add ax, 6 ;add 6 to ax
16 inc bx
17
18
19 mov ax, 0x4c00
20 int 0x21

```

Following are the screenshots of execution of code followed by explanation.

```

AX 0006 SI 0000 CS 19F5 IP 0106 Stack +0 0000 Flags 7200
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 >

0103 BB0000      MOV     BX,0000
0106 050600      ADD     AX,0006
0109 43          INC     BX
010A 050600      ADD     AX,0006
010D 43          INC     BX
010E 050600      ADD     AX,0006
0111 43          INC     BX
0112 050600      ADD     AX,0006
0115 43          INC     BX

DS:0000  CD 20 FF 9F 00 EA FF FF
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 E6 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 FF FF  AD DE 1B 05 C5 06 00 00  = f.0 i|..+...
DS:0010  18 01 10 01 18 01 92 01  01 01 01 00 02 FF FF FF  .....f. ....
DS:0020  FF FF FF FF FF FF FF FF  FF FF FF FF EB 19 E6 11  .....δ.μ.
DS:0030  A2 01 14 00 18 00 F5 19  FF FF FF FF 00 00 00 00  6.....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

```

The value of 6 is moved to the register ax.

```

AX 0018 SI 0000 CS 19F5 IP 0111 Stack +0 0000 Flags 7204
BX 0002 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 1 0

CMD >

010E 050600      ADD     AX,0006
0111 43          INC     BX
0112 050600      ADD     AX,0006
0115 43          INC     BX
0116 050600      ADD     AX,0006
0119 43          INC     BX
011A BB004C      MOV     AX,4C00
011D CD21        INT     21
011F 46          INC     SI

DS:0000  CD 20 FF 9F 00 EA FF FF
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 E6 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 FF FF  AD DE 1B 05 C5 06 00 00  = f.0 i|..+...
DS:0010  18 01 10 01 18 01 92 01  01 01 01 00 02 FF FF FF  .....f. ....
DS:0020  FF FF FF FF FF FF FF FF  FF FF FF FF EB 19 E6 11  .....δ.μ.
DS:0030  A2 01 14 00 18 00 F5 19  FF FF FF FF 00 00 00 00  6.....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

```

The value is added again followed the incrementation to bx, bx is 2 here and ax is 18, means the process of addition is being repeated 3 times until now...


```

AX 001E  SI 0000  CS 19F5  IP 0115  Stack +0 0000  Flags 7204
BX 0003  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 1 0

CMD >

0112 050600      ADD     AX,0006
0115 43          INC     BX
0116 050600      ADD     AX,0006
0119 43          INC     BX
011A B8004C      MOV     AX,4C00
011D CD21        INT     21
011F 46          INC     SI
0120 F60000      TEST    [BX+SI],00
0123 8B46F6      MOV     AX,[BP-0A]

DS:0000  CD 20 FF 9F 00 EA FF FF  AD DE 1B 05 C5 06 00 00
DS:0008  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:0018  01 01 01 00 02 FF FF FF
DS:0020  FF FF FF FF FF FF FF FF  FF FF FF FF EB 19 E6 11
DS:0028  FF FF FF FF FF EB 19 E6 11
DS:0030  A2 01 14 00 18 00 F5 19  FF FF FF FF 00 00 00 00
DS:0038  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
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 FF FF  AD DE 1B 05 C5 06 00 00  = f.Ω  i|.+....
DS:0010  18 01 10 01 18 01 92 01  01 01 01 00 02 FF FF FF  .....f. ....
DS:0020  FF FF FF FF FF FF FF FF  FF FF FF FF EB 19 E6 11  .....δ.μ.
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

```

```

AX 0024  SI 0000  CS 19F5  IP 0119  Stack +0 0000  Flags 7214
BX 0004  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 1 1 0

CMD >

0116 050600      ADD     AX,0006
0119 43          INC     BX
011A B8004C      MOV     AX,4C00
011D CD21        INT     21
011F 46          INC     SI
0120 F60000      TEST    [BX+SI],00
0123 8B46F6      MOV     AX,[BP-0A]
0126 D1E0        SHL     AX,1
0128 D1E0        SHL     AX,1

DS:0000  CD 20 FF 9F 00 EA FF FF  AD DE 1B 05 C5 06 00 00
DS:0008  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:0018  01 01 01 00 02 FF FF FF
DS:0020  FF FF FF FF FF FF FF FF  FF FF FF FF EB 19 E6 11
DS:0028  FF FF FF FF FF EB 19 E6 11
DS:0030  A2 01 14 00 18 00 F5 19  FF FF FF FF 00 00 00 00
DS:0038  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
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 FF FF  AD DE 1B 05 C5 06 00 00  = f.Ω  i|.+....
DS:0010  18 01 10 01 18 01 92 01  01 01 01 00 02 FF FF FF  .....f. ....
DS:0020  FF FF FF FF FF FF FF FF  FF FF FF FF EB 19 E6 11  .....δ.μ.
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

```

Until now the process is repeated by 6 times giving us the value in bx 4, as we initiated with 0, ax has value of 24 and process continues till we get 36...

```

AX 4C00  SI 0000  CS 19F5  IP 011D  Stack +0 0000  Flags 7204
BX 0005  DI 0000  DS 19F5
CX 0000  BP 0000  ES 19F5  HS 19F5  +2 20CD
DX 0000  SP FFFE  SS 19F5  FS 19F5  +4 9FFF  OF DF IF SF ZF AF PF CF
                                         +6 EA00  0 0 1 0 0 0 1 0

CMD >

011A B8004C      MOV     AX,4C00
011D CD21        INT     21
011F 46          INC     SI
0120 F60000      TEST    [BX+SI],00
0123 8B46F6      MOV     AX,[BP-0A]
0126 D1E0        SHL     AX,1
0128 D1E0        SHL     AX,1
012A C55ED8      LDS     BX,[BP-28]
012D 01C3        ADD     BX,AX

DS:0000  CD 20 FF 9F 00 EA FF FF  AD DE 1B 05 C5 06 00 00
DS:0008  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:0018  01 01 01 00 02 FF FF FF
DS:0020  FF FF FF FF FF FF FF FF  FF FF FF FF EB 19 E6 11
DS:0028  FF FF FF FF EB 19 E6 11
DS:0030  A2 01 14 00 18 00 F5 19  FF FF FF FF 00 00 00 00
DS:0038  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
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 FF FF  AD DE 1B 05 C5 06 00 00  = f.Ω  i|..†...
DS:0010  18 01 10 01 18 01 92 01  01 01 01 00 02 FF FF FF  .....f. ....
DS:0020  FF FF FF FF FF FF FF FF  FF FF FF FF EB 19 E6 11  δ.μ.
DS:0030  A2 01 14 00 18 00 F5 19  FF FF FF FF 00 00 00 00  6.....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

```

Ax has value 4c00, for exit as we have coded in code...

Task 4

Following is the screenshot of how the given values are stored in the registers and explained:

```

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
AX 1234  SI 0000  CS 19F5  IP 0103  Stack +0 0000  Flags 7200
BX 0000  DI 0000  DS 19F5
CX 000E  BP 0000  ES 19F5  HS 19F5  +2 20CD
DX 0000  SP FFFE  SS 19F5  FS 19F5  +4 9FFF  OF DF IF SF ZF AF PF CF
                                         +6 EA00  0 0 1 0 0 0 0 0

CMD >

0100 B83412      MOV     AX,1234
0103 B8FCAB      MOV     AX,ABFC
0106 B800B1      MOV     AX,B100
0109 B800B8      MOV     AX,B800
010C CD21        INT     21
010E 89D0        MOV     AX,DX
0110 89DA        MOV     DX,BX
0112 EB04        JMP     0118
0114 31D2        XOR     DX,DX

DS:0100  B8 34 12 B8 FC AB B8 00
DS:0108  B1 B8 00 B8 CD 21 89 D0
DS:0110  89 DA EB 04 31 D2 31 C0
DS:0118  89 56 E4 89 46 E6 C7 46
DS:0120  F6 00 00 8B 46 F6 D1 E0
DS:0128  D1 E0 C5 5E D8 01 C3 8B
DS:0130  07 8B 57 02 85 D2 75 04
DS:0138  85 C0 74 1C C7 46 DC 00
DS:0140  00 8E 5E FC 83 7D 0E 00
DS:0148  74 09 8B 46 F2 48 3B 46

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  = f.Ω= i|..†...
DS:0010  18 01 10 01 18 01 92 01  01 01 01 00 02 FF FF FF  .....f. ....
DS:0020  FF FF FF FF FF FF FF FF  FF FF FF FF EB 19 C0 11  δ.L.
DS:0030  A2 01 14 00 18 00 F5 19  FF FF FF FF 00 00 00 00  6.....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

```

Here the value 34 is stored at address 0101 and value 12 is stored at address 0102

Because intel architecture follows little-Endian method to store value

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

AX	ABFC	SI	0000	CS	19F5	IP	0106	Stack	+0 0000	Flags	7200
BX	0000	DI	0000	DS	19F5				+2 20CD		
CX	000E	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 >

0103	B8FCAB	MOV	AX,ABFC
0106	B800B1	MOV	AX,B100
0109	B800B8	MOV	AX,B800
010C	CD21	INT	21
010E	89D0	MOV	AX,DX
0110	89DA	MOV	DX,BX
0112	EB04	JMP	0118
0114	31D2	XOR	DX,DX
0116	31C0	XOR	AX,AX

1	0	1	2	3	4	5	6	7
DS:0100	B8	34	12	B8	FC	AB	B8	00
DS:0108	B1	B8	00	B8	CD	21	89	D0
DS:0110	89	DA	EB	04	31	D2	31	C0
DS:0118	89	56	E4	89	46	E6	C7	46
DS:0120	F6	00	00	8B	46	F6	D1	E0
DS:0128	D1	E0	C5	5E	D8	01	C3	8B
DS:0130	07	8B	57	02	85	D2	75	04
DS:0138	85	C0	74	1C	C7	46	DC	00
DS:0140	00	8E	5E	FC	83	7D	0E	00
DS:0148	74	09	8B	46	F2	48	3B	46

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	00	02	FF	FF	FF	FF
DS:0020	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

Here value FC is stored at address 0104 and AB is stored at address 0105 following little-endian method.

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

AX	B100	SI	0000	CS	19F5	IP	0109	Stack	+0 0000	Flags	7200
BX	0000	DI	0000	DS	19F5				+2 20CD		
CX	000E	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 >

0106	B800B1	MOV	AX,B100
0109	B800B8	MOV	AX,B800
010C	CD21	INT	21
010E	89D0	MOV	AX,DX
0110	89DA	MOV	DX,BX
0112	EB04	JMP	0118
0114	31D2	XOR	DX,DX
0116	31C0	XOR	AX,AX
0118	8956E4	MOV	[BP-1C],DX

1	0	1	2	3	4	5	6	7
DS:0100	B8	34	12	B8	FC	AB	B8	00
DS:0108	B1	B8	00	B8	CD	21	89	D0
DS:0110	89	DA	EB	04	31	D2	31	C0
DS:0118	89	56	E4	89	46	E6	C7	46
DS:0120	F6	00	00	8B	46	F6	D1	E0
DS:0128	D1	E0	C5	5E	D8	01	C3	8B
DS:0130	07	8B	57	02	85	D2	75	04
DS:0138	85	C0	74	1C	C7	46	DC	00
DS:0140	00	8E	5E	FC	83	7D	0E	00
DS:0148	74	09	8B	46	F2	48	3B	46

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	00	02	FF	FF	FF	FF
DS:0020	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

Here value 00 is stored at address 0107 and B1 is stored at address 0108

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

AX B800 SI 0000 CS 19F5 IP 010C Stack +0 0000 Flags 7200
 BX 0000 DI 0000 DS 19F5 +2 20CD
 CX 000E 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 >

Address	Instruction	Comment
0109 B800B8	MOV	AX, B800
010C CD21	INT	21
010E 89D0	MOV	AX, DX
0110 89DA	MOV	DX, BX
0112 EB04	JMP	0118
0114 31D2	XOR	DX, DX
0116 31C0	XOR	AX, AX
0118 8956E4	MOV	[BP-1C], DX
011B 8946E6	MOV	[BP-1A], AX

1 0 1 2 3 4 5 6 7
 DS:0100 B8 34 12 B8 FC AB B8 00
 DS:0108 B1 B8 00 B8 CD 21 89 D0
 DS:0110 89 DA EB 04 31 D2 31 C0
 DS:0118 89 56 E4 89 46 E6 C7 46
 DS:0120 F6 00 00 8B 46 F6 D1 E0
 DS:0128 D1 E0 C5 5E D8 01 C3 8B
 DS:0130 07 8B 57 02 85 D2 75 04
 DS:0138 85 C0 74 1C C7 46 DC 00
 DS:0140 00 8E 5E FC 83 7D 0E 00
 DS:0148 74 09 8B 46 F2 48 3B 46

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 = f.Ω≡ i |.+. ...
 DS:0010 18 01 10 01 18 01 92 01 01 01 01 00 02 FF FF FFff.
 DS:0020 FF FF FF FF FF FF FF FF FF FF FF FF EB 19 C0 11δ. L.
 DS:0030 A2 01 14 00 18 00 F5 19 FF FF FF FF 00 00 00 00 6.....J.
 DS:0040 05 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

Here 00 is stored at address 0108,2 and B1 is stored at 0108,1 , all these values are stored in little endian method since it's a intel architecture.

Little-Endian: Means that the least significant byte is stored at the lower memory address, while the most significant byte.

Big-Endian is the opposite; the most significant byte is stored at the lower memory address, and the least significant byte is stored at the higher memory address.

a. **1234**

- In Little-Endian: Memory Address: 34 (lower), 12 (higher)
- In Big-Endian: Memory Address: 12 (lower), 34 (higher)

b. **ABFC**

- In Little-Endian: Memory Address: FC (lower), AB (higher)
- In Big-Endian: Memory Address: AB (lower), FC (higher)

c. **B100**

- In Little-Endian: Memory Address: 00 (lower), B1 (higher)
- In Big-Endian: Memory Address: B1 (lower), 00 (higher)

d. **B800**

- In Little-Endian: Memory Address: 00 (lower), B8 (higher)

- In Big-Endian: Memory Address: B8 (lower), 00 (higher)