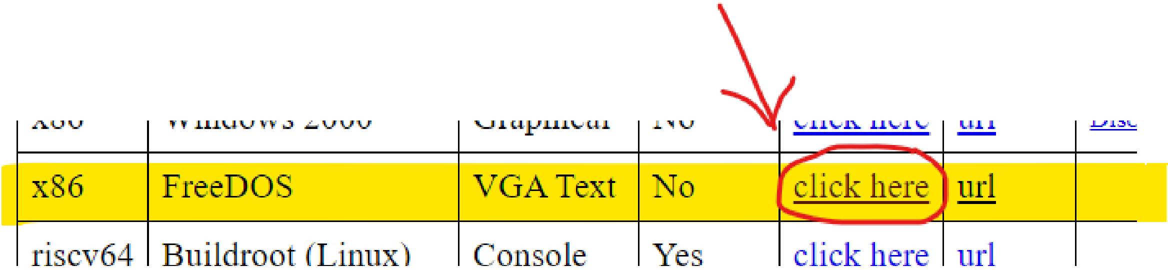
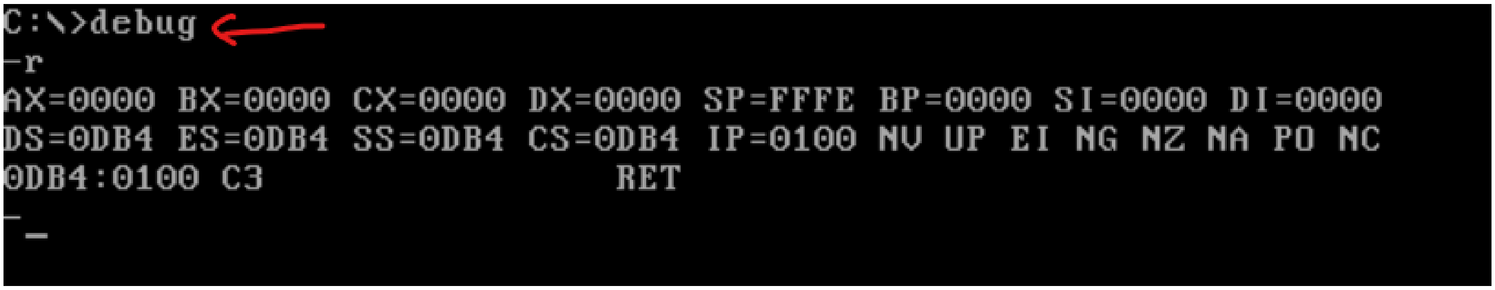
No E-Mail submissions will be accepted.
Submission formats and file naming:
File name : Pts_firstName_lastName_Assembly
File format: pdf or MS Word format
e.g. Pts_Donald_Trump_Assembly.pdf

https://bellard.org/jslinux/



x86	Windows 2000	Graphical	No	click here	url	Disc
x86	FreeDOS	VGA Text	No	click here	url	
riscv64	Buildroot (Linux)	Console	Yes	click here	url	



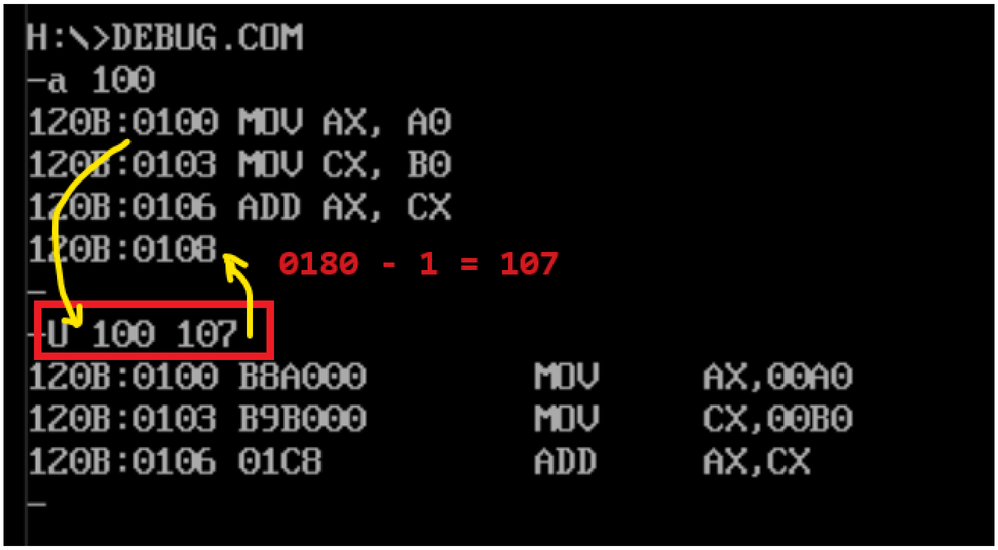
```
C:\>debug
-r
AX=0000 BX=0000 CX=0000 DX=0000 SP=FFFE BP=0000 SI=0000 DI=0000
DS=0DB4 ES=0DB4 SS=0DB4 CS=0DB4 IP=0100 NV UP EI NG NZ NA PO NC
0DB4:0100 C3            RET
-_
```

1) Write an assembly code to calculate AX XOR BX. You are only allowed to use:

1. MOV (to load a value into a register e.g. AX)
2. NOT, OR, AND
3. AX, BX, DX, CX registers

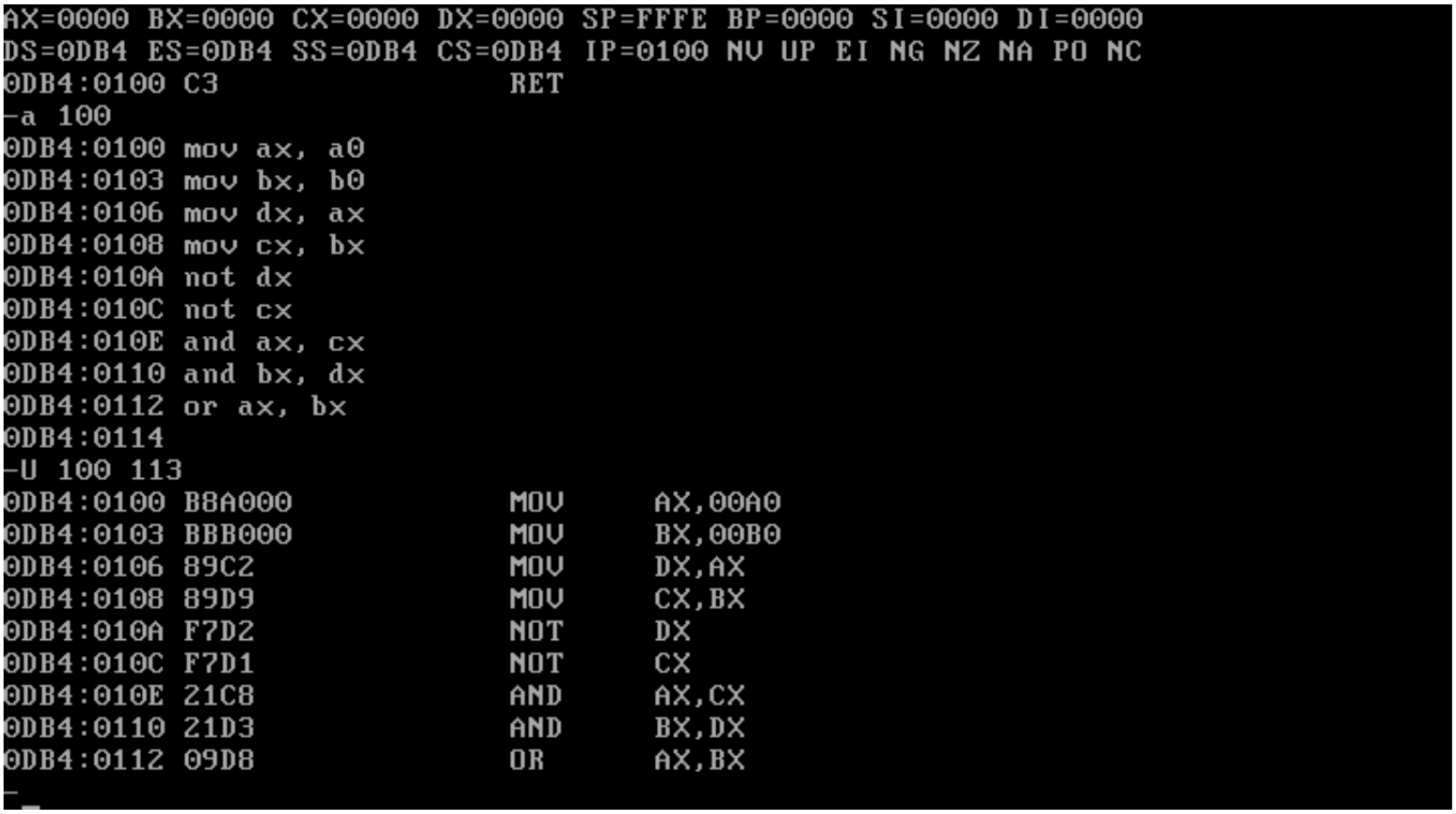
Include a screenshot of the debug environment displaying your assembly code.

e.g.



```
H:\>DEBUG.COM
-a 100
120B:0100 MOV AX, A0
120B:0103 MOV CX, B0
120B:0106 ADD AX, CX
120B:0108
-
-U 100 107
120B:0100 B8A000          MOV     AX,00A0
120B:0103 B9B000          MOV     CX,00B0
120B:0106 01C8            ADD     AX,CX
-
```

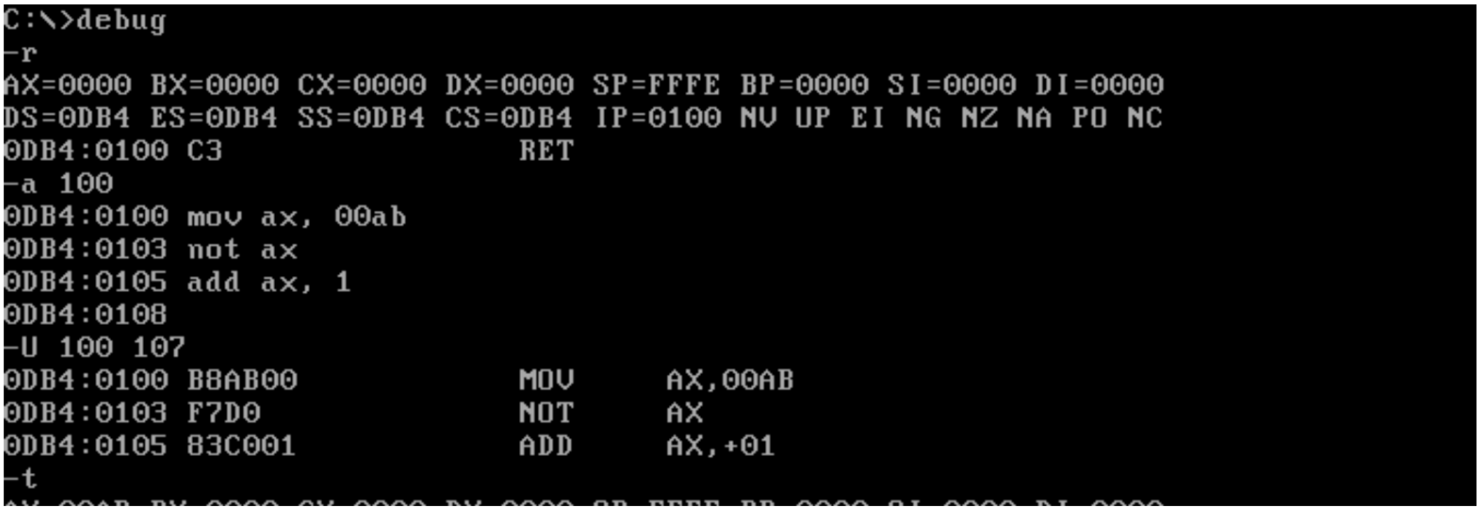
v



```
AX=0000 BX=0000 CX=0000 DX=0000 SP=FFFE BP=0000 SI=0000 DI=0000
DS=0DB4 ES=0DB4 SS=0DB4 CS=0DB4 IP=0100 NV UP EI NG NZ NA PO NC
0DB4:0100 C3                 RET
-a 100
0DB4:0100 mov ax, a0
0DB4:0103 mov bx, b0
0DB4:0106 mov dx, ax
0DB4:0108 mov cx, bx
0DB4:010A not dx
0DB4:010C not cx
0DB4:010E and ax, cx
0DB4:0110 and bx, dx
0DB4:0112 or ax, bx
0DB4:0114
-U 100 113
0DB4:0100 B8A000             MOV     AX,00A0
0DB4:0103 BBB000             MOV     BX,00B0
0DB4:0106 89C2               MOV     DX,AX
0DB4:0108 89D9               MOV     CX,BX
0DB4:010A F7D2               NOT     DX
0DB4:010C F7D1               NOT     CX
0DB4:010E 21C8               AND     AX,CX
0DB4:0110 21D3               AND     BX,DX
0DB4:0112 09D8               OR      AX,BX
-
```

2) Write an assembly code obtain two complements of AX = 00AB (use MOV to load this value). Include a screenshot of the debug environment displaying your assembly code. You are only allowed to use:

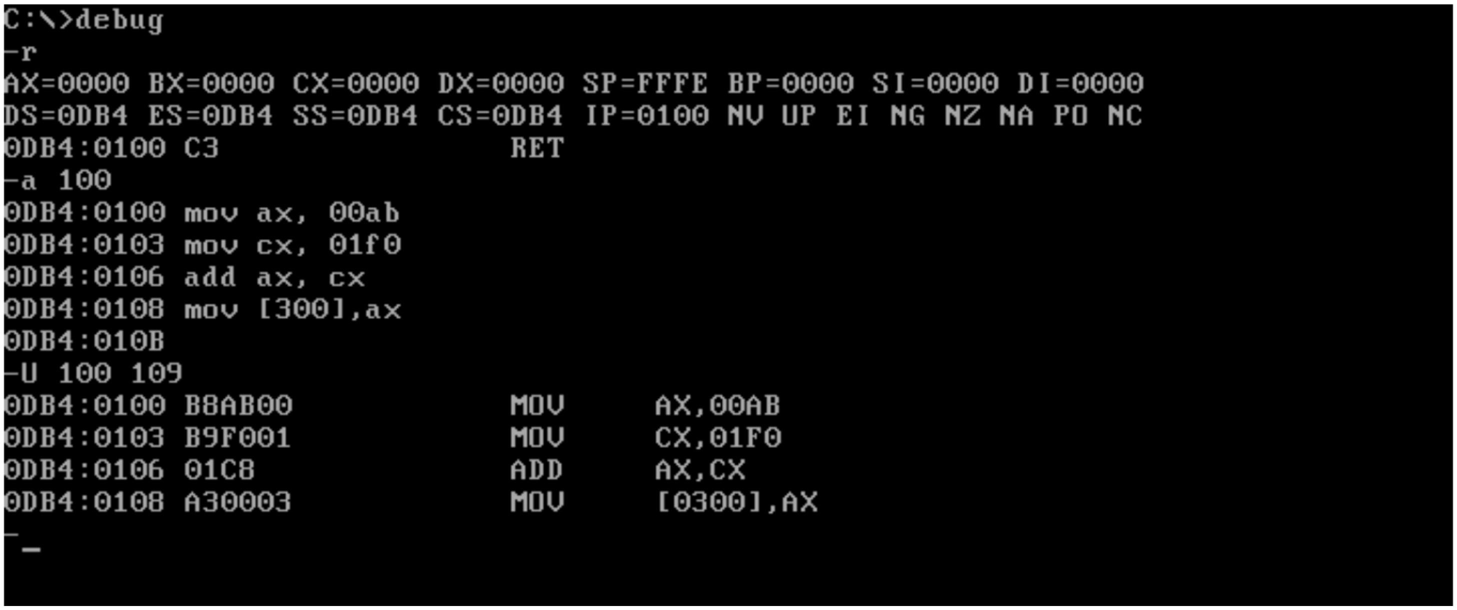
1. MOV (to load a value into a register e.g. AX)
2. NOT, ADD
3. AX register



```
C:\>debug
-r
AX=0000 BX=0000 CX=0000 DX=0000 SP=FFFE BP=0000 SI=0000 DI=0000
DS=0DB4 ES=0DB4 SS=0DB4 CS=0DB4 IP=0100 NV UP EI NG NZ NA PO NC
0DB4:0100 C3                 RET
-a 100
0DB4:0100 mov ax, 00ab
0DB4:0103 not ax
0DB4:0105 add ax, 1
0DB4:0108
-U 100 107
0DB4:0100 B8AB00             MOV     AX,00AB
0DB4:0103 F7D0               NOT     AX
0DB4:0105 83C001             ADD     AX,+01
-t
AX=00AB BX=0000 CX=0000 DX=0000 SP=FFFE BP=0000 SI=0000 DI=0000
```

3) Write an assembly code to calculate the sum of AX and CX. Save the result in AX and store the AX value in memory location 300. Assume AX=00AB and CX=01F0 (use MOV to load these values). Include a screenshot of the debug environment displaying your assembly code. You are only allowed to use:

1. MOV (to load a value into a register e.g. AX)
2. ADD
3. AX and CX registers



```
C:\>debug
-r
AX=0000 BX=0000 CX=0000 DX=0000 SP=FFFE BP=0000 SI=0000 DI=0000
DS=0DB4 ES=0DB4 SS=0DB4 CS=0DB4 IP=0100 NV UP EI NG NZ NA PO NC
0DB4:0100 C3                RET
-a 100
0DB4:0100 mov ax, 00ab
0DB4:0103 mov cx, 01f0
0DB4:0106 add ax, cx
0DB4:0108 mov [300],ax
0DB4:010B
-U 100 109
0DB4:0100 B8AB00            MOV     AX,00AB
0DB4:0103 B9F001            MOV     CX,01F0
0DB4:0106 01C8              ADD     AX,CX
0DB4:0108 A30003            MOV     [0300],AX
-
```