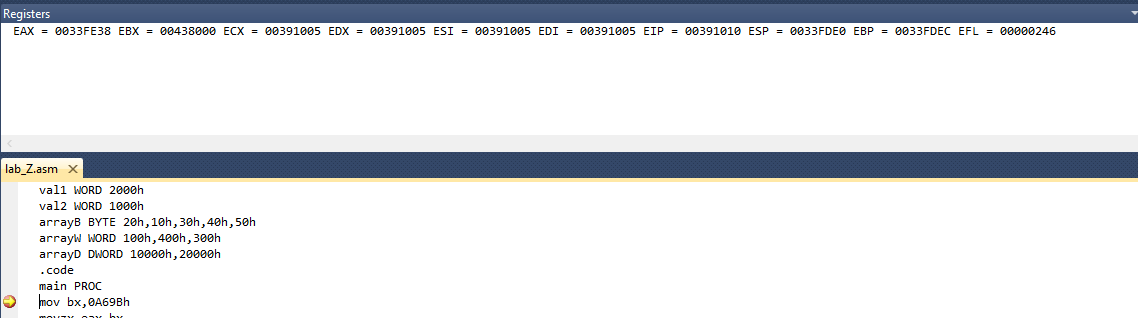
**COAL LAB 4**

Q1)What are the values of the registers and the variables after each group of instructions in the following program .

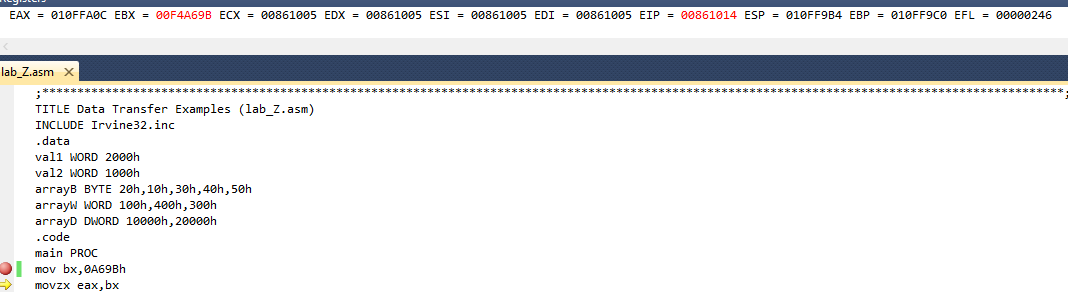
a)Put the break point and notice the value of register in register window and Write down the value of output (i.e register value) or attached the snips of each step.

b)Also write interpretation of the each line of the following program and Attach the output of the register window **not console window** !

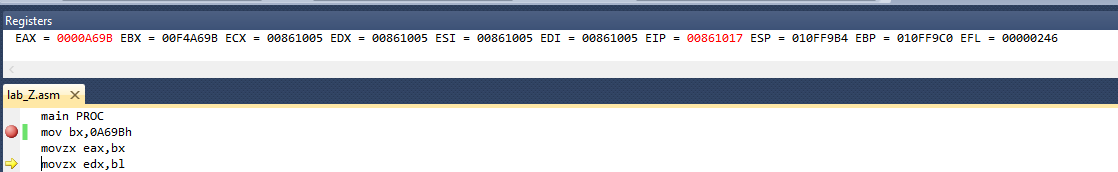
**STEP#01:**

****

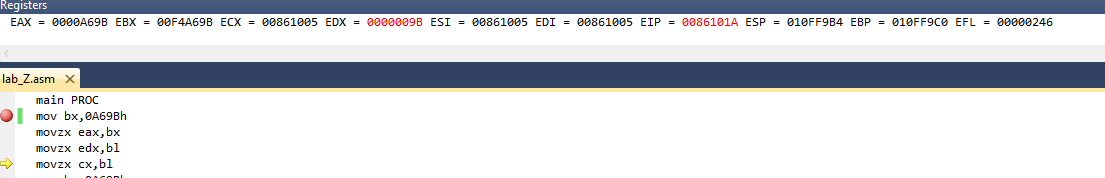
**STEP#02:**

****

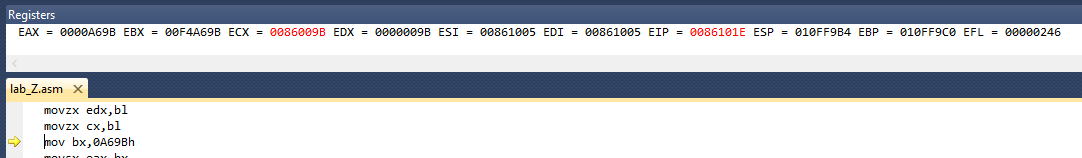
**STEP#03:**

****

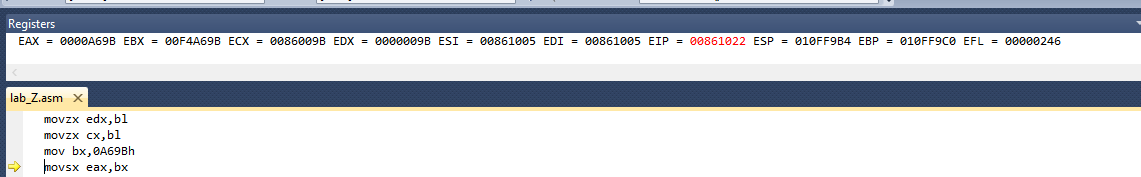
**STEP#04:**

****

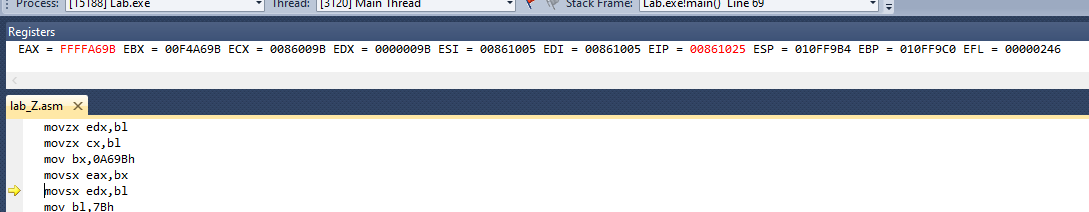
**STEP#05:**

****

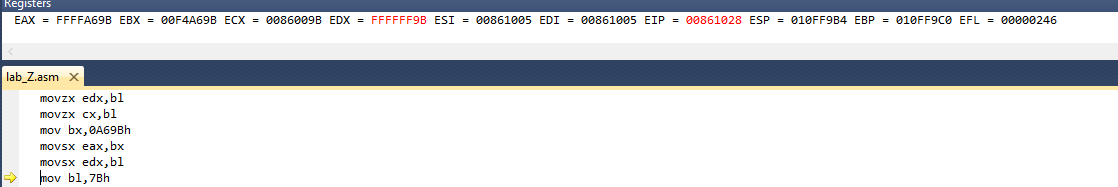
**STEP#06:**

****

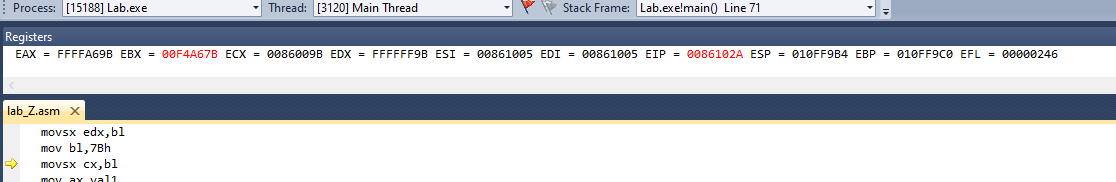
**STEP#07:**

****

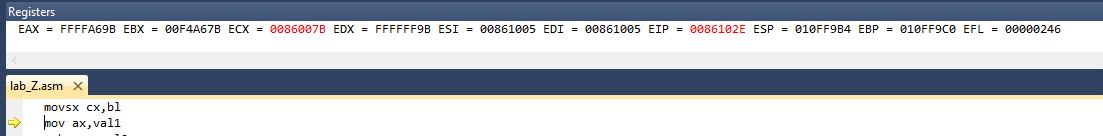
**STEP#08:**

****

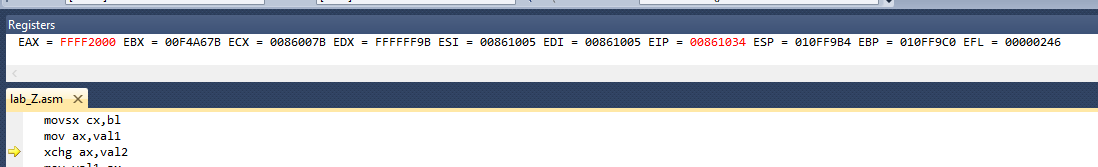
**STEP#09:**

****

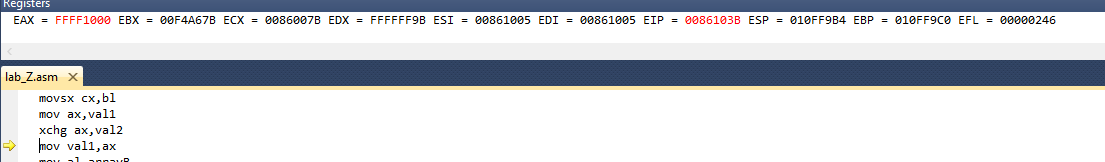
**STEP#10:**

****

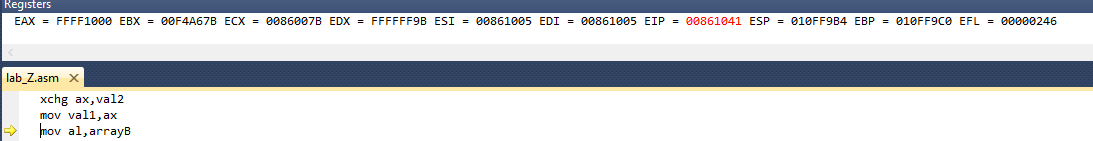
**STEP#11:**

****

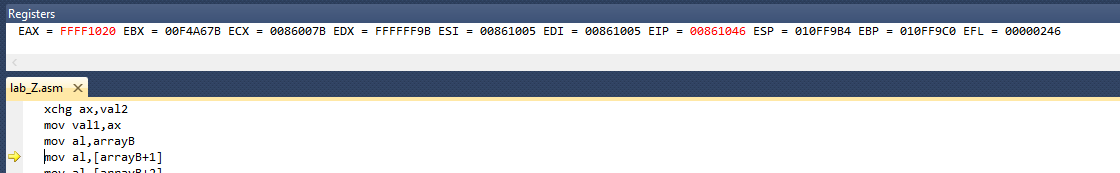
**STEP#12:**

****

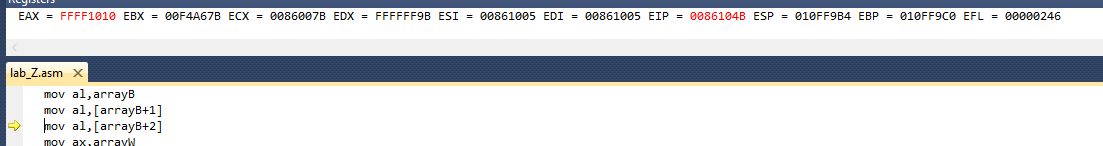
**STEP#13:**

****

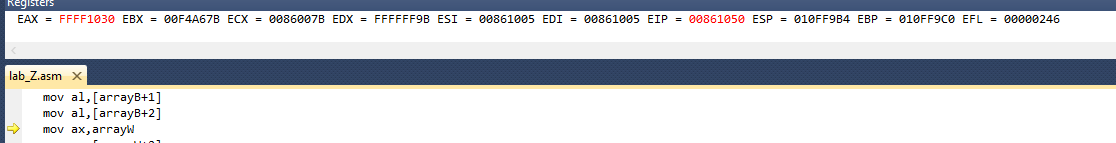
**STEP#14:**

****

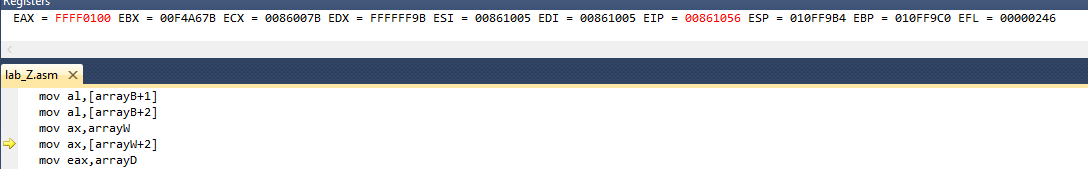
**STEP#15:**

****

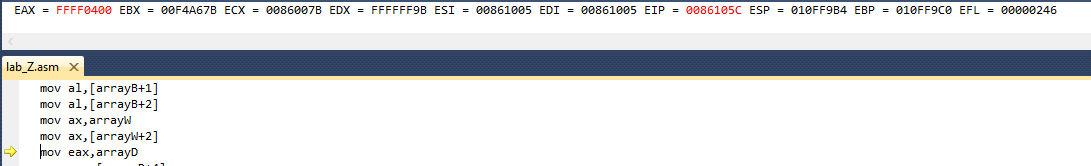
**STEP#16:**

****

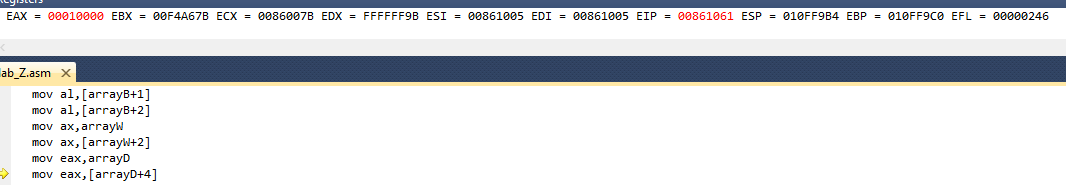
**STEP#17:**

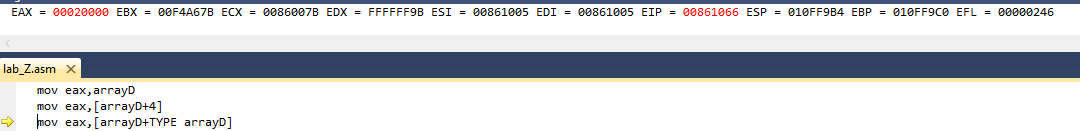
****

**STEP#18:**

****

**STEP#19:**

****

**STEP#20:**

**(b)**

TITLE Data Transfer Examples (Test.asm)

INCLUDE Irvine32.inc

.data

val1 WORD 2000h

val2 WORD 1000h

arrayB BYTE 20h,10h,30h,40h,50h

arrayW WORD 100h,400h,300h

arrayD DWORD 10000h,20000h

.code

main PROC

mov bx,0A69Bh // WE MOVED THE VALUE 0A69Bh IN BX REGISTER.

movzx eax,bx //WE MOVED THE BX VALUE IN EAX REGISTER WITH EXTENDED ZERO’S.

movzx edx,bl //WE MOVED THE BL VALUE IN EDX REGISTER WITH EXTENDED ZERO’S.

movzx cx,bl //WE MOVED THE BL VALUE IN CX REGISTER WITH EXTENDED ZERO’S.

mov bx,0A69Bh //WE MOVED THE VALUE 0A69Bh IN BX REGISTER.

movsx eax,bx //WE MOVED THE BX VALUE IN EAX REGISTER WITH EXTENDED ONE’S.(FFFFh).

movsx edx,bl //WE MOVED THE BL VALUE IN EDX REGISTER WITH EXTENDED ONE’S.(FFFFh).

mov bl,7Bh //WE MOVED THE VALUE 78h IN BL REGISTER.

movsx cx,bl //WE MOVED THE BL VALUE IN CX REGISTER WITH EXTENDED ONE’S.(FFFFh).

mov ax,val1 //WE MOVED THE VAL1(2000h) VALUE IN AX REGISTER.

xchg ax,val2 //WE EXCHANGE THE VALUE OF AX REGISTER WITH VAL2 VARIABLE.

mov val1,ax //WE MOVED THE AX REGISTER IN VAL1 VARIABLE.

mov al,arrayB //WE MOVED THE ARRAYB IN AL REGISTER.

mov al,[arrayB+1] //WE MOVED THe ARRAYB+1 IN AL REGISTER.

mov al,[arrayB+2] //WE MOVED THe ARRAYB+2 IN AL REGISTER.

mov ax,arrayW //WE MOVED THE ARRAYW IN AX REGISTER.

mov ax,[arrayW+2] //WE FIRST ADD THE ARRAYW 1ST INDEX TO 2 AND THEN MOVED IN AX REGISTER.

mov eax,arrayD //WE MOVED THE ARRAYD IN EAX REGISTER.

mov eax,[arrayD+4] //WE FIRST THe ARRAYD 1ST INDEX TO 4 AND THEN MOVED IN EAX REGISTER.

mov eax,[arrayD+TYPE arrayD] //WE FIRST ADD THE ARRAYD 1ST INDEX TO THE SIZE OF ARRAY(TYPE ARRAYD) AND THEN MOVING IN EAX REGISTER.

exit

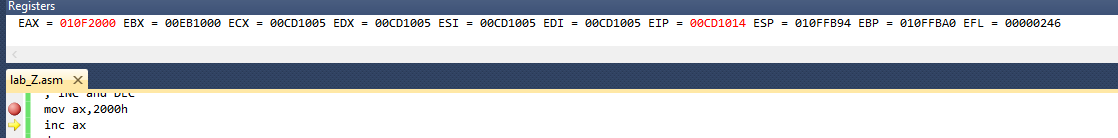
main ENDP

END main

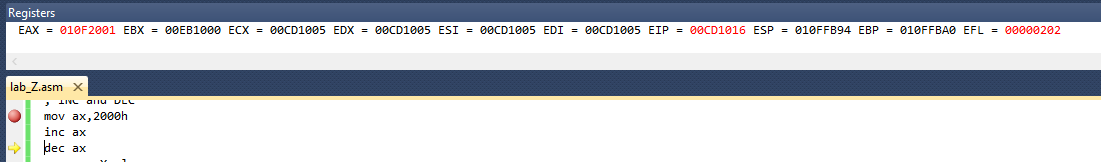
Q2) What are the values of the registers and the variables after each group of instructions in the following program .

a)Put the break point and notice the value of register in the register window and Write down the value of output (i.e register value) or attached the snips of each step.

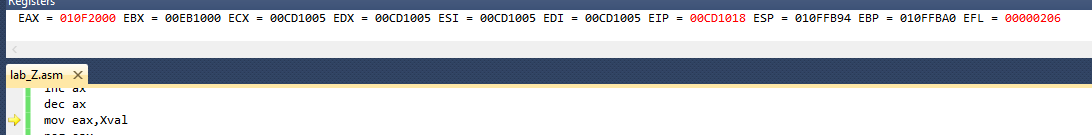
b)Also write interpretation of each line of the following program and Attach the output of the register window **not console window** !

**STEP#01:**

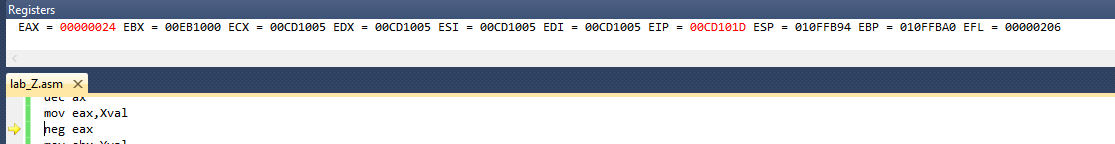
**STEP#02:**

****

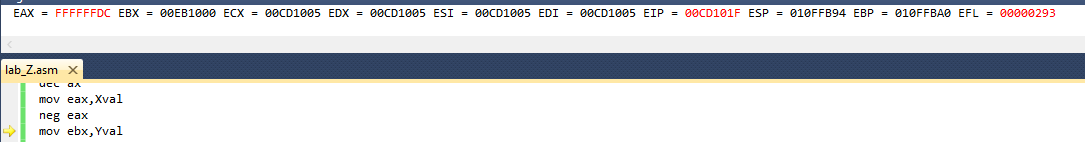
**STEP#03:**

****

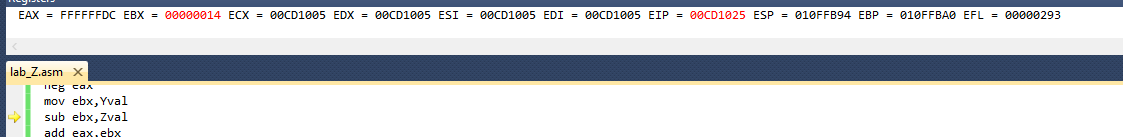
**STEP#04:**

****

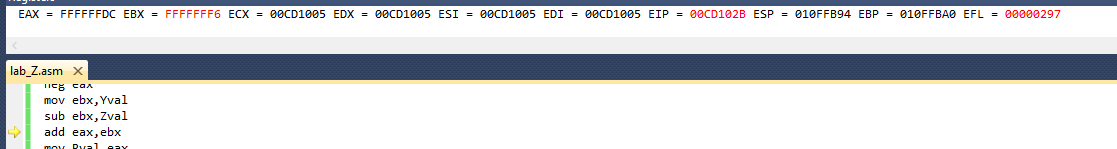
**STEP#05:**

****

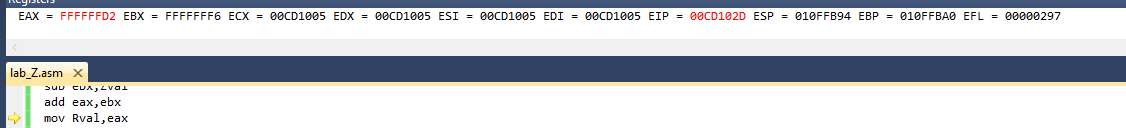
**STEP#06:**

****

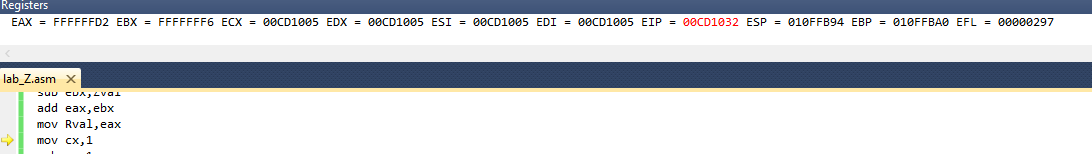
**STEP#07:**

****

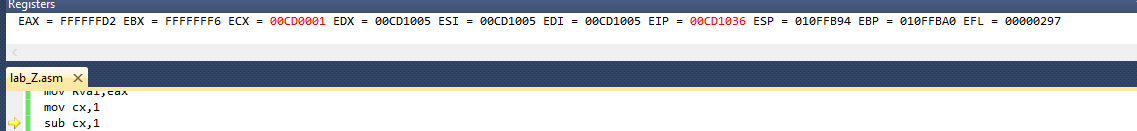
**STEP#08:**

****

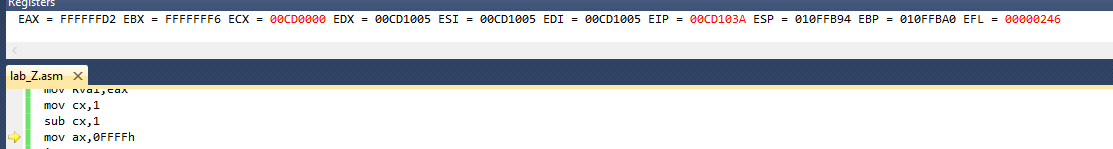
**STEP#09:**

****

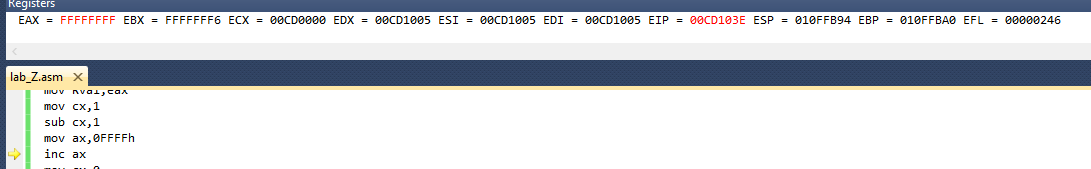
**STEP#10:**

****

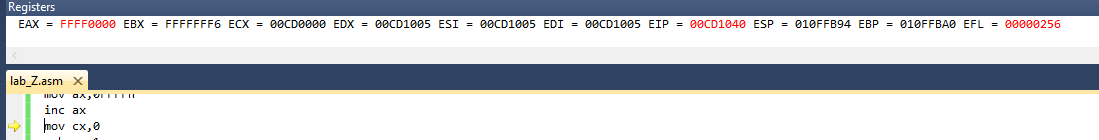
**STEP#11:**

****

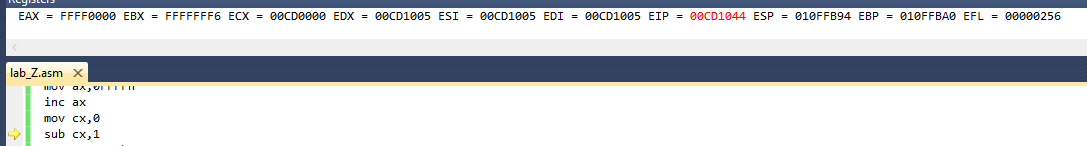
**STEP#12:**

****

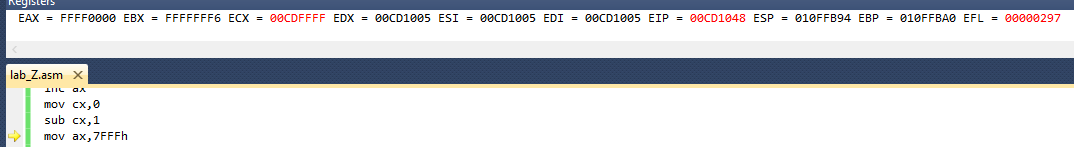
**STEP#13:**

****

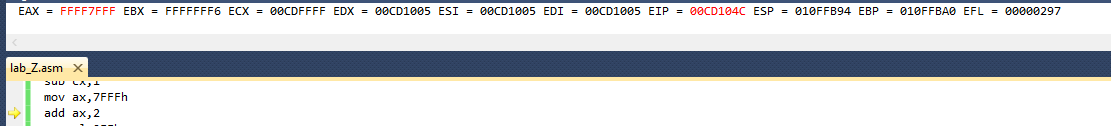
**STEP#14:**

****

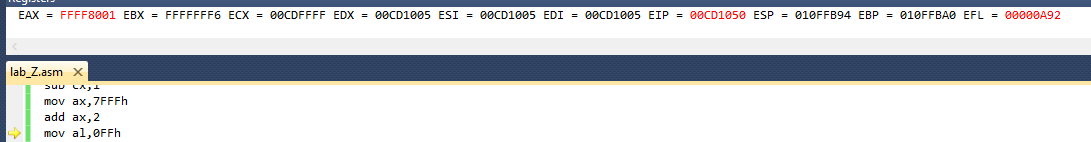
**STEP#15:**

****

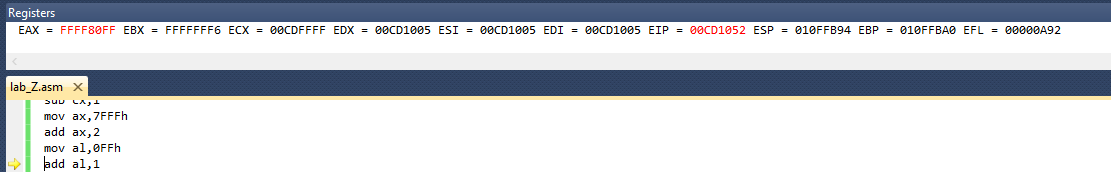
**STEP#16:**

****

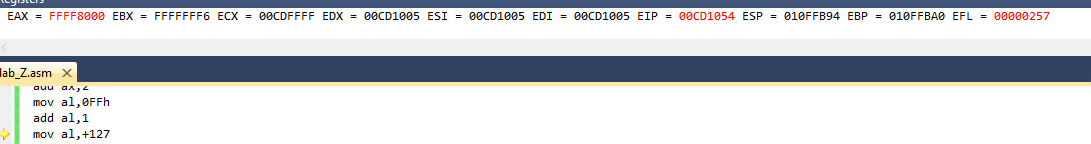
**STEP#17:**

****

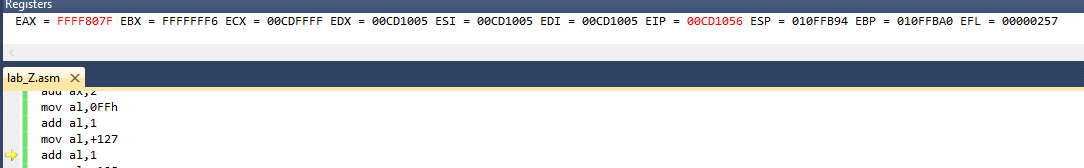
**STEP#18:**

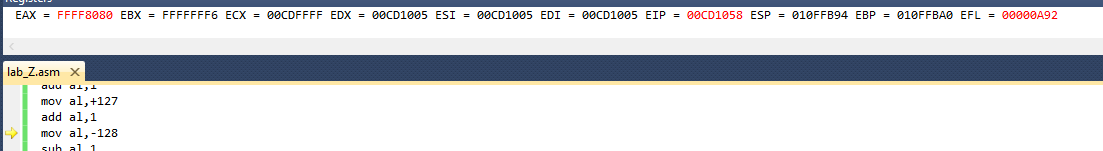
****

**STEP#19:**

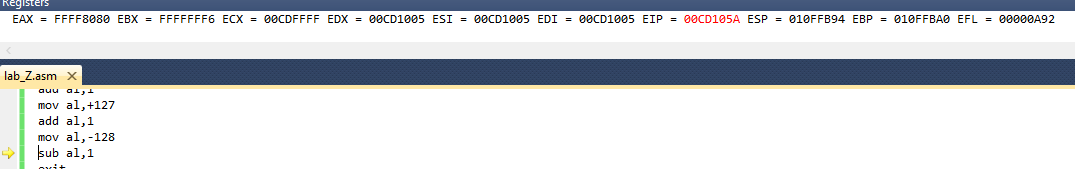
****

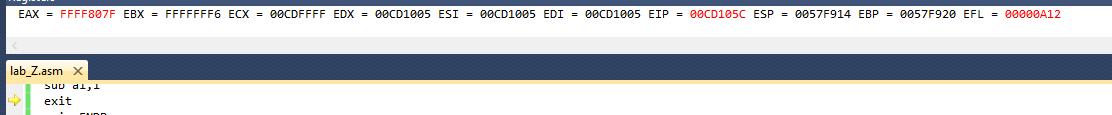
**STEP#20:**

****

**STEP#21:**

**STEP#22:**

****

**STEP#23:**

(B)

**TITLE Addition and Subtraction (lab4.asm)**

INCLUDE Irvine32.inc

.data

Rval SDWORD ?

Xval SDWORD 36

Yval SDWORD 20

Zval SDWORD 30

.code

main PROC

; INC and DEC

mov ax,2000h //we moving the 2000h in ax register.

inc ax //increment in ax register.

dec ax //decrement in ax register.

mov eax,Xval //moving the value of Xval in eax register.

neg eax //it takes the 2’s complement of a number store in eax

mov ebx,Yval //moving the value of Yval in ebx register.

sub ebx,Zval //subtracting the value of Zval from ebx.

add eax,ebx //adding eax and ebx.

mov Rval,eax //moving the eax value in Rval variable.

mov cx,1 //move 1 in cx register.

sub cx,1 //sub 1 from cx register.

mov ax,0FFFFh //moving the 0FFFFh in ax register.

inc ax //incrementing the ax register.

mov cx,0 //moving 0 in cx register.

sub cx,1 //moving 1 in cx register.

mov ax,7FFFh //moving 7FFFh in ax register.

add ax,2 //adding 2 in ax register.

mov al,0FFh //moving 0FFh in al register.

add al,1 //adding 1 in al register.

mov al,+127 //moving positive 127 in al .

add al,1 //adding 1 in al.

mov al,-128 //moving -128 in al.

sub al,1 //subtracting 1 from al register.

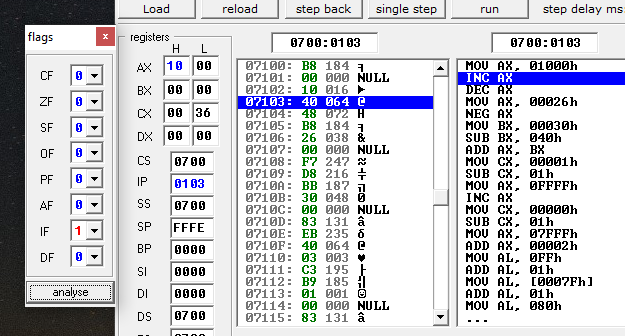
exit

main ENDP

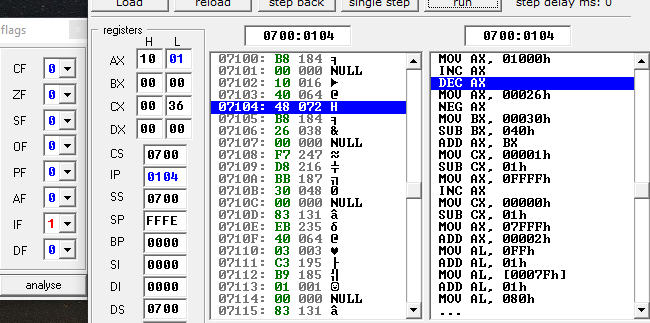
END main

Q3 RUN the following program at emu8086 and notice the value of lower byte and higher byte register and status of CPU flag bit. Attached the output of running program. Also write or attached each step of output .

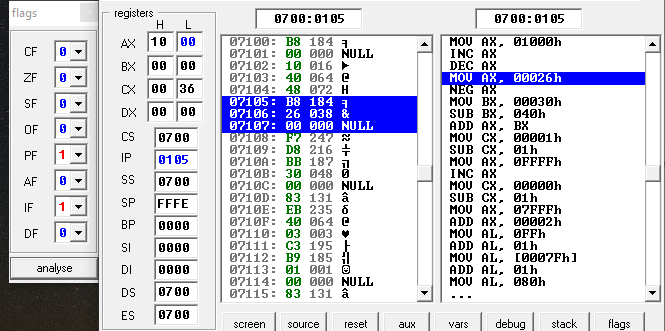
**STEP#01:**

****

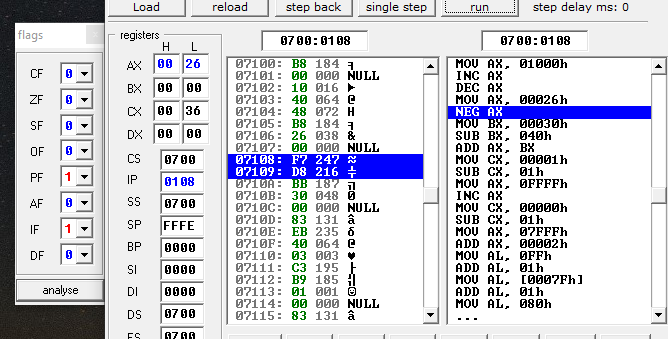
**STEP#02:**

****

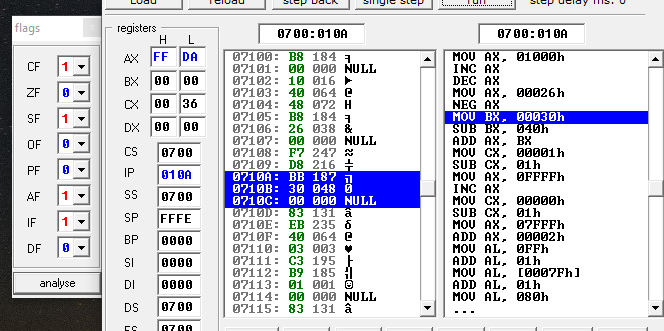
**STEP#03:**

****

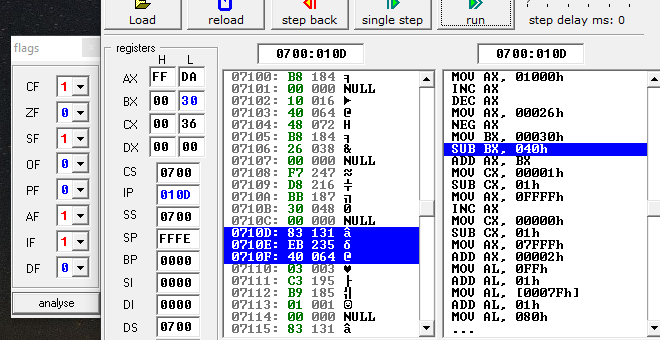
**STEP#04:**

****

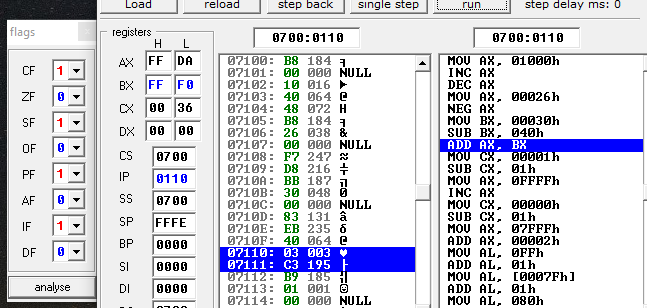
**STEP#05:**

****

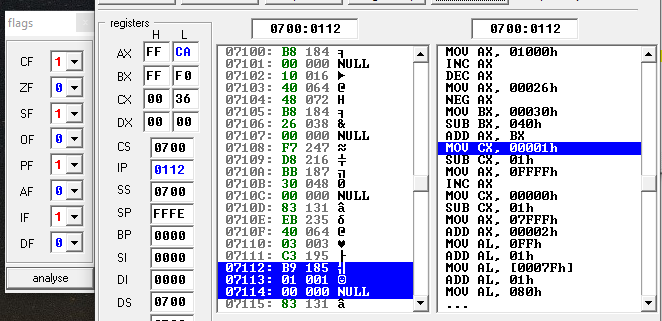
**STEP#06:**

****

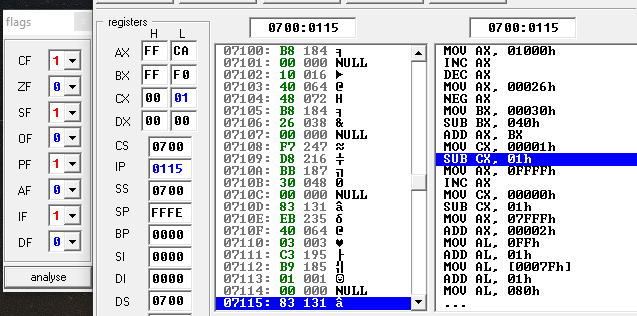
**STEP#07:**

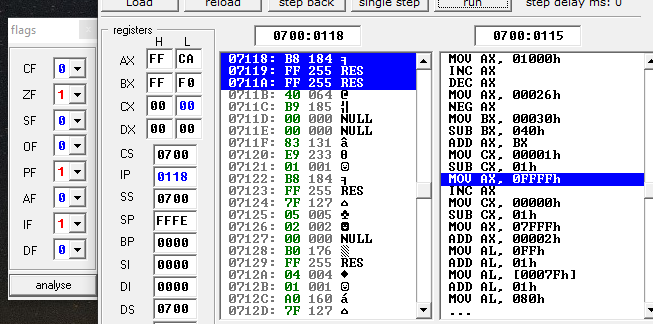
****

**STEP#08:**

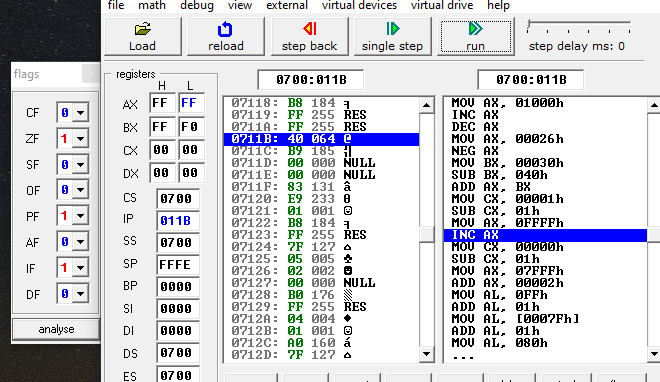
****

**STEP#09:**

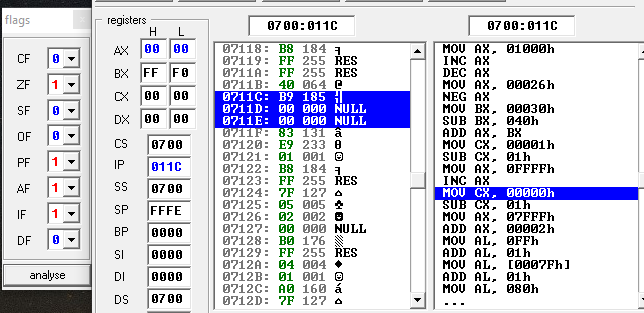
****

**STEP#10:**

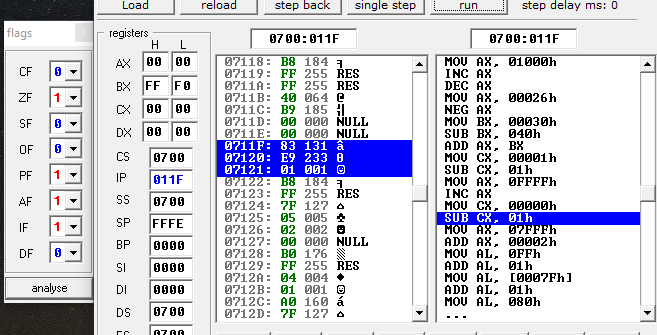
**STEP#11:**

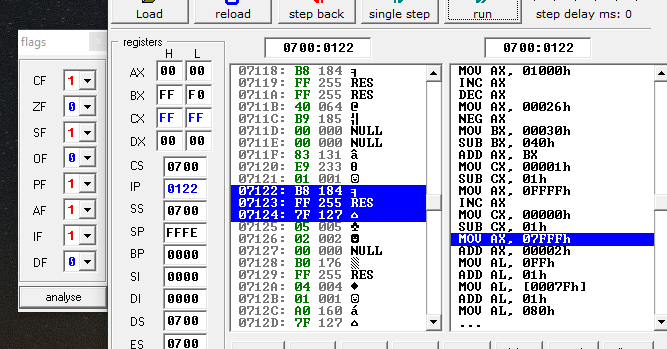
****

**STEP#12:**

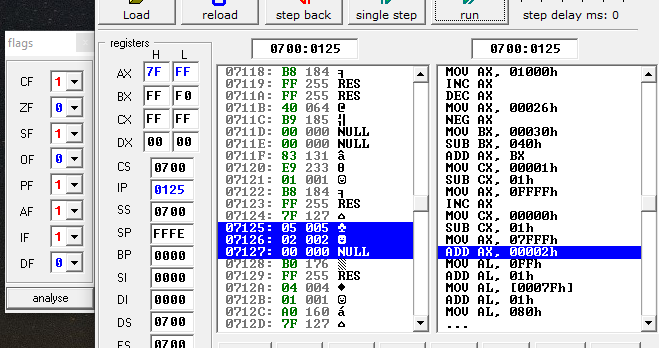
****

**STEP#13:**

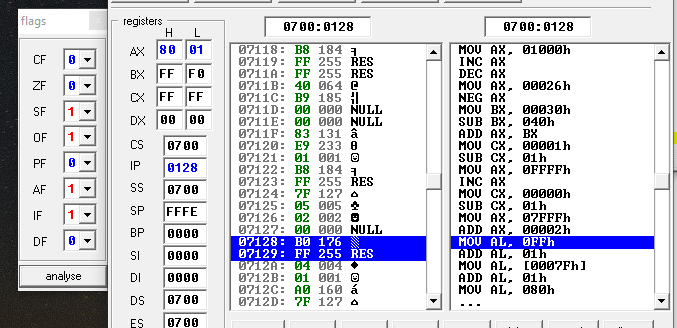
****

**STEP#14:**

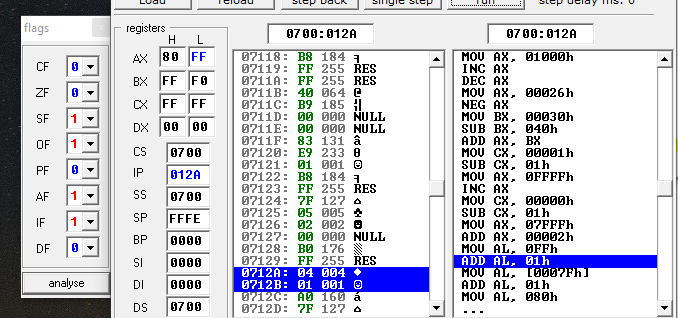
**STEP#15:**

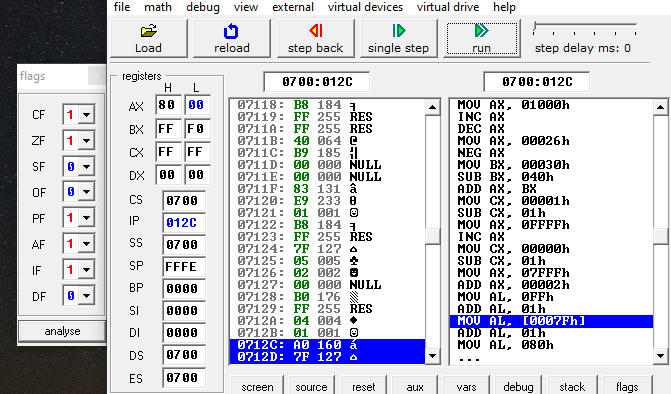
****

**STEP#16:**

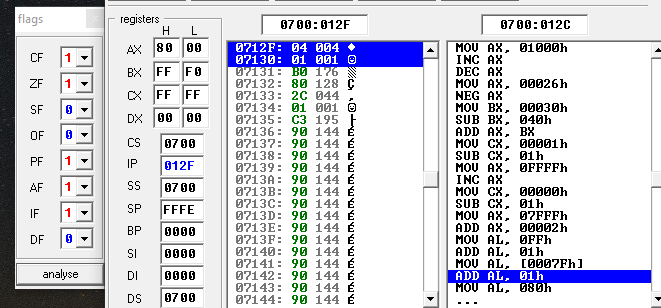
****

**STEP#17:**

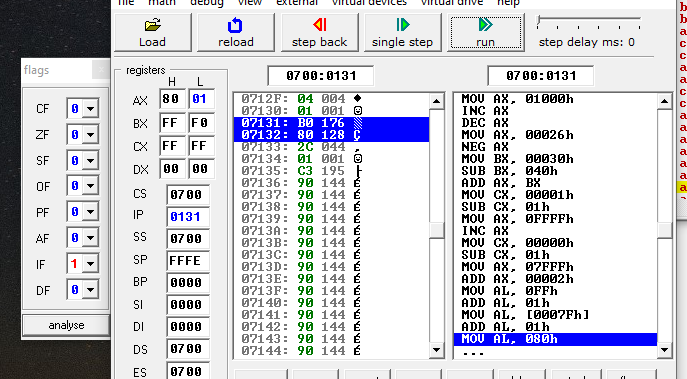
****

**STEP#18:**

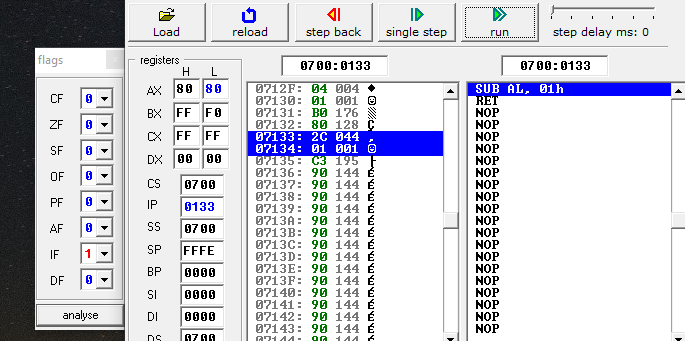
**STEP#19:**

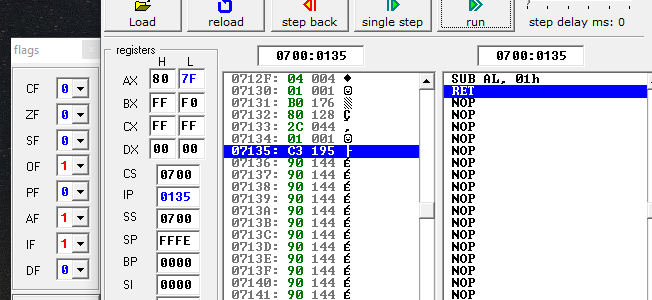
****

**STEP#20:**

****

**STEP#21:**

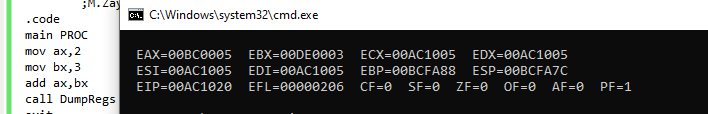
****

**STEP#22:**

Q4)Indicate whether or not each of the following instructions is valid OR not.Run the each instruction in .code segment and attached the snip.

Register assume to any value. For example you may take ax=2 or any other integer.

**a. add ax,bx**

****

**This statement is valid!!!!.**

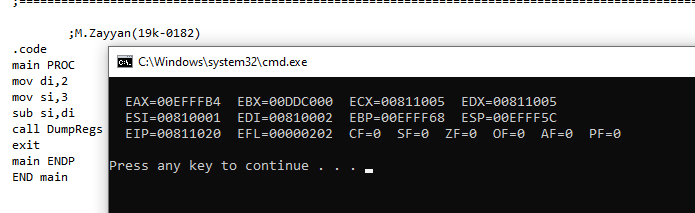
**b. add dx,bl**

**This statement is invalid because the size of destination and source register is different.**

**c. add ecx,dx**

**This statement is invalid because the size of destination and source register is different.**

**d. sub si,di**

****

**This statement is valid!!!!.**

**e. add bx,90000**

**This statement is invalid!!.**

**f. sub ds,1**

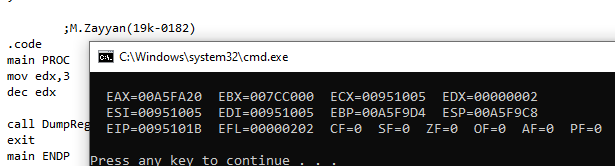
**This statement is invalid!!**

**g. dec ip**

**This statement is invalid because ip is undefined.**

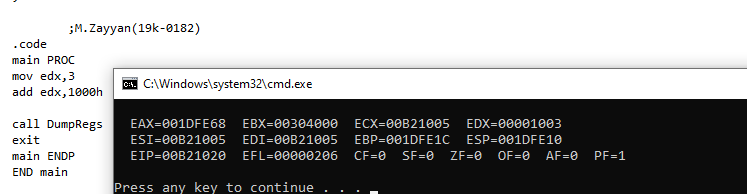
**h. dec edx**

**This statement is valid!!.**

****

**i. add edx,1000h**

**This statement is valid.**

****

**j. sub ah,126h**

**This statement is invalid because ah can’t store 126h due to less size.**

**k. sub al,256h**

**This statement is invalid due to invalid instruction operands.**

**l. inc ax,1**

**This statement is invalid because of syntax error.**

**(correct=inc ax)**