8A] Write an assembly language program to enter elements ina an array and display its contents.

**Program:**

%macro write 2

mov eax, 4

mov ebx, 1

mov ecx, %1

mov edx, %2

int 80h

%endmacro

%macro read 2

mov eax, 3

mov ebx, 2

mov ecx, %1

mov edx, %2

int 80h

%endmacro

section .data

asksize db 'Enter size: '

asksizelen equ $-asksize

ask db 'Enter'

asklen equ $-ask

show db 'Array'

showlen equ $-show

array times 10 dw 0

len equ 10

nl db '', 10

nllen equ $-nl

section .bss

num resb 9

i resb 9

element resb 10

section .text

global \_start:

\_start:

write asksize, asksizelen

read num, 9

write ask, asklen

write nl, nllen

mov byte[i], 0

mov esi, array

input:

read element, 2

mov ebx, [element]

mov [esi], ebx

inc esi

inc byte[i]

mov al, [i]

mov bl, [num]

sub bl, '0'

cmp al, bl

JE exit

JMP input

exit:

write show, showlen

write nl, nllen

mov byte[i], 0

mov esi, array

output:

mov ebx, [esi]

mov [element], ebx

write element, 1

write nl, nllen

inc esi

inc byte[i]

mov al, [i]

mov bl, [num]

sub bl, '0'

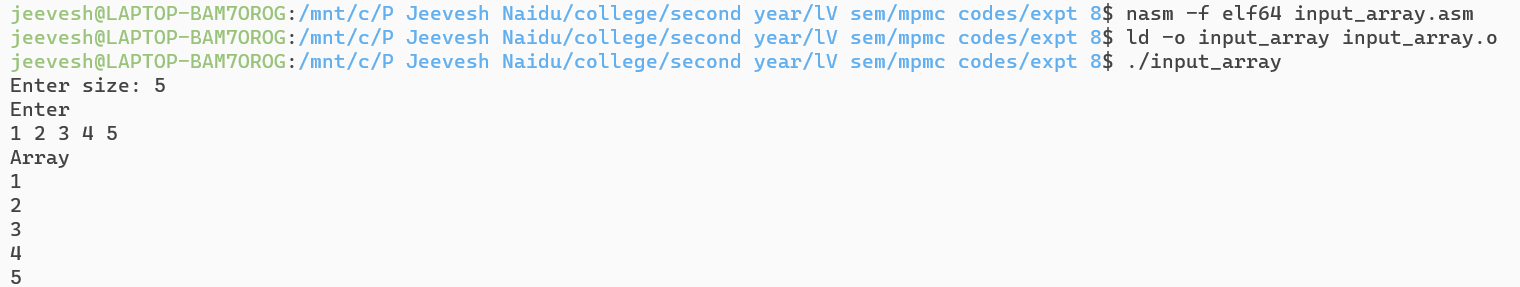
cmp al, bl

JL output

mov eax, 1

int 80h

**Output:**



8B] Write an assembly language program to count number of positive and negative numbers in

an array.

**Program:**

%macro write\_string\_number 2

mov eax, 4

mov ebx, 1

mov ecx, %1

mov edx, %2

int 80h

%endmacro

%macro read\_number\_string 2

mov eax, 3

mov ebx, 2

mov ecx, %1

mov edx, %2

int 80h

%endmacro

; ---->

section .data

display db "Contents of Array : 1, 2, -9, 11, -12, 4, 9, 2, -3",10

dis\_len equ $-display

pos\_msg db "Positive count:"

pos\_len equ $-pos\_msg

neg\_msg db "Negative count:"

neg\_len equ $-neg\_msg

nl db '', 10

nllen equ $-nl

; array declaration and initialization

array dw 1, 2, -9, 11, -12, 4, 9, 2, -3

arrCnt equ 9 ; static array size

posCnt dw 0 ; number of positive numbers

negCnt dw 0 ; no of negative numbers

section .bss

dis\_buffer resb 2

section .text

global \_start

\_start:

; initializing array start address

write\_string\_number display, dis\_len

mov esi, array

mov ecx, arrCnt

UP: BT word[esi], 15

JC NCXT

inc byte[posCnt]

JMP PSKIP

NCXT: inc byte[negCnt]

PSKIP:

inc esi

inc esi

loop UP

; positive count message

write\_string\_number pos\_msg, pos\_len

mov bl, [posCnt]

CALL Hex\_ASCII

; print newline

write\_string\_number nl, nllen

; negative count message

write\_string\_number neg\_msg, neg\_len

mov bl, [negCnt]

CALL Hex\_ASCII

; print newline

write\_string\_number nl, nllen

; exit

mov eax, 1

mov ebx, 0

int 80h

; HEX to ASCII procedure ----->

Hex\_ASCII:

mov ecx, 2

mov edi, dis\_buffer

DUP:

rol bl, 04

mov al, bl

and al, 0fh

cmp al, 09h

jbe NEXT

add al, 07h

NEXT:

add al, 30h

mov [edi], al

inc edi

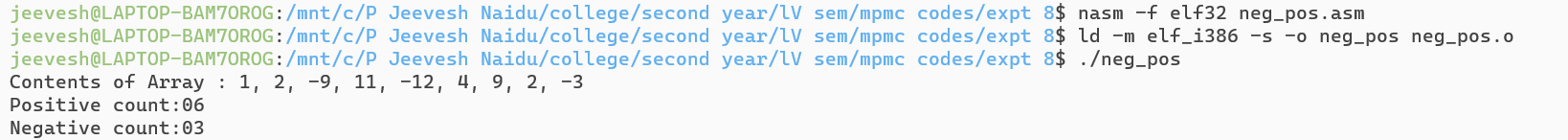
loop DUP

; display count

write\_string\_number dis\_buffer, 2

ret

**Output:**



8C] Write an assembly language program to count number of odd and even numbers in an array.

Program:

%macro write 2

mov eax, 4

mov ebx, 1

mov ecx, %1

mov edx, %2

int 80h

%endmacro

%macro read 2

mov eax, 3

mov ebx, 2

mov ecx, %1

mov edx, %2

int 80h

%endmacro

section .data

asknum db 'Enter the number of elements: '

asknumlen equ $-asknum

ask db 'Enter elements of the array: '

asklen equ $-ask

showe db 'Even: '

showelen equ $-showe

showo db 'Odd: '

showolen equ $-showo

nl db '', 10

nllen equ $-nl

array times 10 dw 0

len equ 10

section .bss

num resb 10

i resb 10

el resb 10

rem resb 5

neven resb 5

nodd resb 5

section .text

global \_start

\_start:

write asknum, asknumlen

read num, 10

write ask, asklen

mov byte[neven], 0

mov byte[nodd], 0

mov byte[i], 0

mov esi, array

input:

read el, 2

mov ebx, [el]

mov [esi], ebx

inc esi

inc byte[i]

mov al, [i]

mov bl, [num]

sub bl, '0'

cmp al, bl

jl input

mov byte[i], 0

mov esi, array

check:

mov al, [esi]

mov bl, '2'

sub bl, '0'

div bl

cmp ah, 0

JE even

JMP odd

even:

inc byte[neven]

jmp loop

odd:

inc byte[nodd]

jmp loop

loop:

inc esi

inc byte[i]

mov al, [i]

mov bl, [num]

sub bl, '0'

cmp al, bl

JL check

JE output

output:

add [neven], byte '0'

add [nodd], byte '0'

write showe, showelen

write neven, 5

write nl, nllen

write showo, showolen

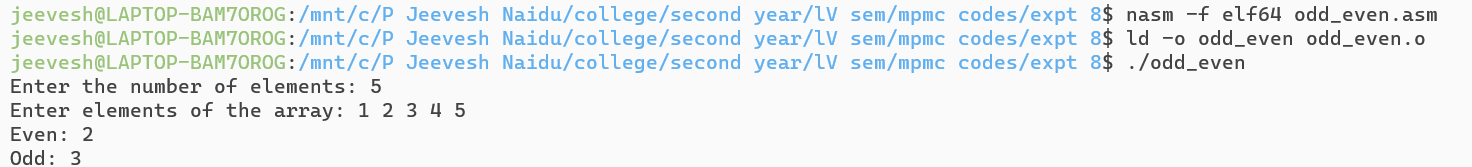
write nodd, 5

write nl, nllen

mov eax, 1

int 80h

Output:



8D] Write an assembly language program to numbers in array above and below 50.

Program:

; Count the num of numbers above 50 and below 50

section .data

display db "Contents of Array : 2, 4, 34, 66, 57",10

dis\_len equ $-display

msg1 db 'Numbers above 50: '

msg1len equ $-msg1

msg2 db 'Numbers below 50: '

msg2len equ $-msg2

newline db '',10

n1 equ $-newline

%macro write 2

mov eax,4

mov ebx,1

mov ecx, %1

mov edx, %2

int 80h

%endmacro

%macro read 2

mov eax,3

mov ebx,2

mov ecx,%1

mov edx,%2

int 80h

%endmacro

global arr

arr: db 2,4,34,66,57

section .bss

above resb 1

below resb 1

temp resb 5

section .text

global \_start

\_start:

write display,dis\_len

call count

write msg1,msg1len

write above, 1

write newline,n1

write msg2,msg2len

write below, 1

write newline,n1

; exit

mov eax ,1

mov ebx ,0

int 80h

; procedure

count:

mov eax, 0

mov [above],eax

mov [below], eax

mov ecx, 5

mov esi, 0

label:

movzx edi, byte[arr+esi]

mov [temp],edi

mov al,[temp]

mov bl,50

cmp al, bl

jg l1

; if number below 50

inc byte[below]

jmp end

l1:

; if number above 50

inc byte[above]

end:

inc esi

loop label

mov eax, [below]

add eax, '0'

mov [below], eax

mov eax, [above]

add eax, '0'

mov [above], eax

ret

**Output:**

