* This is the debug to check for flags that sir told:

mov ax, 0FFFFh

; Flag is 0000 0010 0000 0010

; Carry is 0

; Parity is 0

; Sign is 0

; Zero is 0

; Auxillary is 0

; Overflow is 0

inc ax

; Flag is 0000 0010 0101 0110

; Carry is 0

; Parity is 1

; Sign is 0

; Zero is 1

; Auxillary is 1

; Overflow is 0

mov bl, 4

; Flag is 0000 0010 0101 0110

; Carry is 0

; Parity is 1

; Sign is 0

; Zero is 1

; Auxillary is 1

; Overflow is 0

sub bl, 5

; Flag is 0000 0010 1001 0111

; Carry is 1

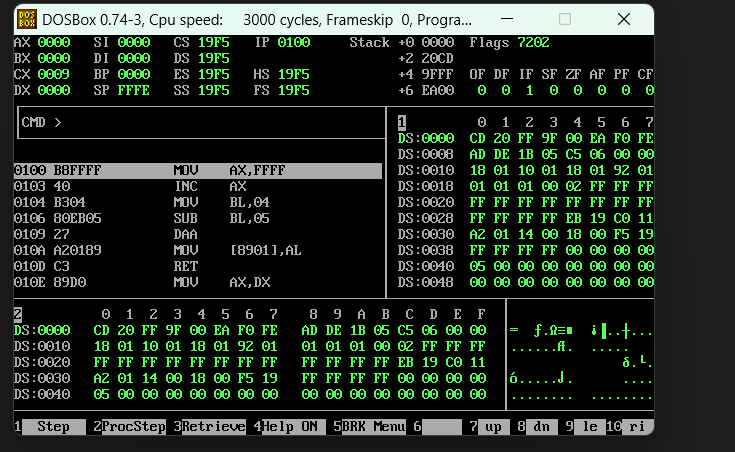
; Parity is 1

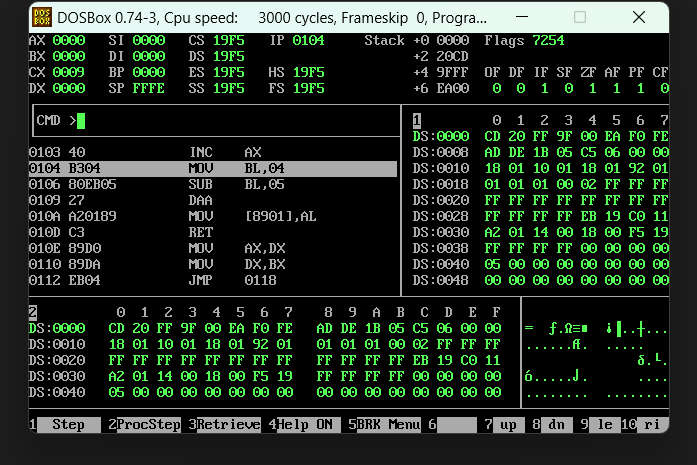
; Sign is 1

; Zero is 0

; Auxillary is 1

; Overflow is 0





A screenshot of a computer

Description automatically generated

**Q.1.**

.model small

.stack 100h

.data

msg1 db 10,13, "Enter First number: $"

msg2 db 10,13, "Enter Second number: $"

msg3 db 10,13, "Numbers are equal $"

msg4 db 10,13, "Numbers are not equal $"

.code

main proc

mov ax, @data

mov ds, ax

; Display message to enter the first number

mov dx, offset msg1

mov ah, 09h

int 21h

; Read the first number

mov ah, 01h

;Function to read a character from STDIN

int 21h

; Call DOS interrupt

sub al, 30h

; Convert ASCII to numeric value

mov cl, al

; Store the first number

; Display message to enter the second number

mov dx, offset msg2

mov ah, 09h

int 21h

; Read the second number

mov ah, 01h

; Function to read a character from STDIN

int 21h

; Call DOS interrupt

sub al, 30h

; Convert ASCII to numeric value

mov dl, al

; Store the second number

; Compare the two numbers

cmp dl, cl

je equal

; If equal, jump to label1

; If not equal, print the message

mov dx, offset msg4

mov ah, 09h

int 21h

jmp end\_prog

equal:

; If equal, print the message

mov dx, offset msg3

mov ah, 09h

int 21h

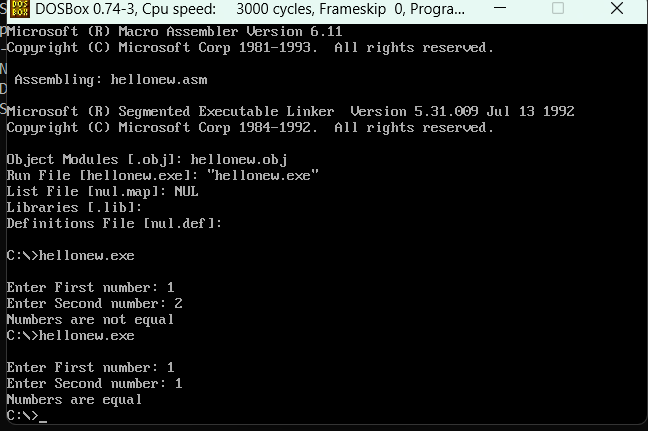
end\_prog:

mov ah,76

int 33

main endp

end main

****

**Q.2.**

dosseg

.model small

.stack 100h

.data

array db 10 Dup(?)

.code

main proc

mov ax, @data

mov ds, ax

mov si, offset array

mov cx, 10

; loop

mov al,48

l1:

mov [si],al

mov dx, [si]

mov ah,2

int 21h

;mov dx, [si+1]

inc si

inc al

loop l1

mov ah, 4ch

int 21h

main endp

end main

**A black screen with white text

Description automatically generated**

**Q.3.**

dosseg

.model small

.stack 100h

.data

array db 10 Dup(?)

.code

main proc

mov ax, @data

mov ds, ax

mov si, offset array

mov cx, 26

; loop

mov al,97

l1:

mov [si],al

mov dx, [si]

mov ah,2

int 21h

;mov dx, [si+1]

inc si

inc al

loop l1

mov ah, 4ch

int 21h

main endp

end main

****

**Q.4.**

dosseg

.model small

.stack 100h

.data

array db 10 Dup(?)

.code

main proc

mov ax, @data

mov ds, ax

mov si, offset array

mov cx, 26

; loop

mov al,65

l1:

mov [si],al

mov dx, [si]

mov ah,2

int 21h

;mov dx, [si+1]

inc si

inc al

loop l1

mov ah, 4ch

int 21h

main endp

end main

**A black screen with white text

Description automatically generated**

**Q.5.**

dosseg

.model small

.stack 100h

.data

var1 db 10,13,'even number..$'

var2 db 10,13,'odd number..$'

.code

main proc

mov ax,@data

mov ds,ax

mov ah,1

int 21h

mov bl,2

div bl

cmp ah,0 ;this si the remainder cehck wether it os 0 or not

je l1 ;then jump to l1

mov dx,offset var2 ;else print odd number

mov ah,9

int 21h

jmp exit ;exit

l1:

mov dx,offset var1

mov ah,9

int 21h

exit:

mov ah,4ch

int 21h

main endp

end main

**A screenshot of a computer program

Description automatically generated**

**Q.6.**

dosseg

.model small

.stack 100h

.data

array db 0,1,2,3,4,5,6,7,8,9

result db 10 Dup(?)

.code

main proc

mov ax, @data

mov ds, ax

mov si, offset array

mov di, offset result

mov cx, 10

l1:

mov al, [si]

mov bl,2

div bl

cmp ah,0

;test ah,ah

je evenNumber

mov byte ptr [di],79

jmp nextElement

evenNumber:

mov byte ptr [di], 101

nextElement:

inc si

inc di

loop l1

mov si, offset result

mov cx, 10

l2:

mov dl, [si]

mov ah, 02h

int 21h

inc si

loop l2 ;

mov ah, 4ch

int 21h

main endp

end main

**A screen shot of a computer

Description automatically generated**

**Q.7.**

dosseg

.model small

.stack 100h

.data

array db 2,1,2,1,2

result db 0

.code

main proc

mov ax, @data

mov ds, ax

mov si, offset array

mov cx, 5

mov ax,0

mov bl,0

l1:

mov al, [si]

add bl,al

inc si

loop l1

mov dl,bl

add dl,48

mov result,dl

mov ah,02h

int 21h

mov ah, 4ch

int 21h

main endp

end main

**A screen shot of a computer

Description automatically generated**