Write an assembly program using 8086 instructions to count the number of characters entered by the user until the "Enter" key (carriage return) is pressed. The program should then display the total count of characters.

.model small

.stack 100h

.data

count db 0 ; Variable to store character count

.code

main proc

mov ax, @data

mov ds, ax ; Initialize data segment

mov cl, 0 ; Initialize count

mov ah, 1 ; Read character function

while\_:

int 21h ; Read a character

cmp al, 13 ; Check for Enter (ASCII 13)

je word\_count ; If Enter, go to word\_count

inc cl ; Increment count for each character

jmp while\_ ; loop while

word\_count:

mov ah, 2

mov dl, 10

int 21h

mov dl, 13

int 21h

mov ah, 2 ; Display character function

add cl, '0' ; Convert number to ASCII

mov dl, cl ; Load DL with count

int 21h ; Print the count

mov ah, 4Ch ; Terminate program

int 21h

main endp

end main

Write a assembly code to display the char ‘\*’ 20 times using loop instructions or using only conditional/unconditional jumps

.model small

.stack 100h

.data

count db 0 ; Variable to store character count

.code

main proc

mov ax, @data

mov ds, ax ; Initialize data segment

mov cx, 20

mov ah, 2

mov dl, '\*'

print\_:

int 21h

;loop print\_

dec cx

jz exit\_

jmp print\_

exit\_:

mov ah, 4Ch ; Terminate program

int 21h

main endp

end main

**//printing all ascii 256 chars**

.model small

.stack 100h

.data

.code

main proc

mov ax, @data

mov ds, ax

mov cx, 256

mov dl, 0

mov ah, 2

print\_loop:

int 21h

inc dl

dec cx

jnz print\_loop

;loop print\_loop we can also use this line by commenint out upper two lines

mov ah, 4ch

main endp

end main

**//Put the sum of the first 50 terms of the arithmetic sequence 1, 5, 9, 13, ... in DX. Hints: Employ LOOP instructions to do the following**

.model small

.stack 100h

.data

.code

main proc

mov ax, @data

mov ds, ax

mov cx, 50

mov ax, 1

mov bx, 0

sum\_loop:

add bx, ax

add ax, 4

loop sum\_loop

mov dx, bx

mov ah, 4ch

main endp

end main

**//Put the sum 100 + 95 + 90 + ... + 5 in AX. Hints: Employ LOOP instructions to do the following.**

.model small

**.**stack 100h

.data

.code

main proc

mov ax, @data

mov ds, ax

mov cx, 19

mov ax, 100

mov bx, 0

sum\_loop:

add bx, ax

sub ax, 5

loop sum\_loop

mov ax, bx

mov ah, 4ch

main endp

end main

**//Read a character and display it 50 times on the next line. Hints: use LOOP instructions and put cx = 50**

.model small

.stack 100h

.data

.code

main proc

mov ax, @data

mov ds, ax

mov ah, 1

int 21h

mov bl, al

mov ah, 2

mov dl, 10

int 21h

mov dl, 13

int 21h

mov cx, 50

mov dl, bl

mov ah, 2

display\_char\_50\_times:

int 21h

loop display\_char\_50\_times

mov ah, 4ch

main endp

end main

**//Write a program to check whether a given input character is a vowel or not.**

.model small

.stack 100h

.data

vowel db "it is a vowel$"

n db "it is not a vowel$"

newline db 0ah, 0dh, "$"

;newline db 10, 13, "$"

.code

main proc

mov ax, @data

mov ds, ax

mov ah, 1

int 21h

mov bl, al

mov ah, 9

lea dx, newline

int 21h

cmp bl, 'a'

je is\_vowel

cmp bl, 'e'

je is\_vowel

cmp bl, 'i'

je is\_vowel

cmp bl, 'o'

je is\_vowel

cmp bl, 'u'

je is\_vowel

jmp no\_vowel

is\_vowel:

mov ah, 9

lea dx, vowel

int 21h

jmp exit\_

no\_vowel:

mov ah, 9

lea dx, n

int 21h

exit\_:

mov ah, 4ch

main endp

end main

**//Take an input character from user. Check it for letter and convert upper to lower or lower to upper using logical instructions**

.model small

.stack 100h

.data

vowel db "it is a vowel$"

n db "it is not a vowel$"

newline db 0ah, 0dh, "$"

;newline db 10, 13, "$"

.code

main proc

mov ax, @data

mov ds, ax

mov ah, 1

int 21h

mov bl, al

mov ah, 9

lea dx, newline

int 21h

cmp bl, 'a'

jl check\_upper

cmp bl, 'z'

jg check\_upper

lower\_to\_upper:

and bl, 0dfh

mov dl, bl

mov ah, 2

int 21h

jmp exit\_

check\_upper:

cmp bl, 'A'

jl exit\_

cmp bl, 'Z'

jg exit\_

upper\_to\_lower:

or bl, 20h

mov dl, bl

mov ah, 2

int 21h

exit\_:

mov ah, 4ch

main endp

end main

**//Take an input character from user. Check it for number and find whether it is odd or even using TEST instruction.**

.model small

.stack 100h

.data

error db "it is not a number$"

even db "even number $"

odd db "odd number $"

newline db 0ah, 0dh, "$"

;newline db 10, 13, "$"

.code

main proc

mov ax, @data

mov ds, ax

mov ah, 1

int 21h

mov bl, al

mov ah, 9

lea dx, newline

int 21h

cmp bl, '0'

jl error\_

cmp bl, '9'

jg error\_

ascii\_to\_number:

and bl, 0fh ; same as sub al, 48 ; sub al, 30h, sub al, '0'

check\_even\_odd:

test bl, 1

jz print\_even

print\_odd:

mov ah, 9

lea dx, odd

int 21h

jmp exit\_

print\_even:

mov ah, 9

lea dx, even

int 21h

jmp exit\_

error\_:

mov ah, 9

lea dx, error

int 21h

exit\_:

mov ah, 4ch

main endp

end main

/**/Write an assembly language program for Binary Input and Output**

.model small

.stack 100h

.data

error db "it is not a number$"

even db "even number $"

odd db "odd number $"

newline db 0ah, 0dh, "$"

;newline db 10, 13, "$"

.code

main proc

mov ax, @data

mov ds, ax

xor bx, bx

mov cx, 8

input\_binary:

mov ah, 1

int 21h

and al, 0fh

shl bl, 1

or bl, al

loop input\_binary

mov ah, 9

lea dx, newline

int 21h

mov cx, 8

output\_binary:

mov ah, 2

shl bl, 1

mov dl, '0'

jnc print\_binary

mov dl, '1'

print\_binary:

int 21h

loop output\_binary

exit\_:

mov ah, 4ch

main endp

end main

//**binary input and reverse binary output**

.model small

.stack 100h

.data

error db "it is not a number$"

even db "even number $"

odd db "odd number $"

newline db 0ah, 0dh, "$"

;newline db 10, 13, "$"

.code

main proc

mov ax, @data

mov ds, ax

xor bl, bl

mov cx, 8

input\_binary:

mov ah, 1

int 21h

and al, 0fh ; same as sub al, 48

shl bl, 1

or bl, al

loop input\_binary

mov ah, 9

lea dx, newline

int 21h

mov cx, 8

reverse\_binary:

mov ah, 2

shr bl, 1

mov dl, '0'

jnc print\_reverse

mov dl, '1'

print\_reverse:

int 21h

loop reverse\_binary

exit\_:

mov ah, 4ch

main endp

end main

**//count the number of characters in the input until user press enter**

.model small

.stack 100h

.data

newline db 0ah, 0dh, "$"

;newline db 10, 13, "$"

.code

main proc

mov ax, @data

mov ds, ax

mov bl, 0

mov ah, 1

int 21h

count\_char:

cmp al, 0dh

je display\_char\_count

inc bl

int 21h

jmp count\_char

display\_char\_count:

mov ah, 9

lea dx, newline

int 21h

mov ah, 2

;add bl, '0'

or bl, 30h

mov dl, bl

int 21h

exit\_:

mov ah, 4ch

main endp

end main

**//enter a char and display corresponding binary with the count of bit 1 of output binary**

;binary input and reverse binary output(using shit and rotate)

.model small

.stack 100h

.data

input db "Pleae enter a binary number(8 bit):$"

output db "the number of 1 bits is : $"

newline db 0ah, 0dh, "$"

;newline db 10, 13, "$"

.code

main proc

mov ax, @data

mov ds, ax

mov ah, 9

lea dx, input

int 21h

mov ah, 1

int 21h

mov bl, al

mov ah, 9

lea dx, newline

int 21h

mov cx, 8

mov bh, 0

print\_binary:

mov ah, 2

shl bl, 1

mov dl, '0'

jnc print

mov dl, '1'

inc bh

print:

int 21h

loop print\_binary

count\_bit\_number:

mov ah, 9

lea dx, newline

int 21h

mov ah, 9

lea dx, output

int 21h

mov ah,2

or bh, 30h

mov dl, bh

int 21h

exit\_:

mov ah, 4ch

main endp

end main