1. Write an assembly language program to perform division of 8-bit data.

**Ans:**

**Code:**

org 100h

start:

mov ax, cs

mov ds, ax

mov al, [dividend]

mov bl, [divisor]

xor ah, ah

div bl

mov [quotient], al

mov [remainder], ah

mov dx, offset msg\_quotient

mov ah, 09h

int 21h

mov al, [quotient]

add al, '0'

mov dl, al

mov ah, 02h

int 21h

mov dl, 0Dh

int 21h

mov dl, 0Ah

int 21h

mov dx, offset msg\_remainder

mov ah, 09h

int 21h

mov al, [remainder]

add al, '0'

mov dl, al

mov ah, 02h

int 21h

mov ah, 4Ch

int 21h

dividend db 25

divisor db 7

quotient db 0

remainder db 0

msg\_quotient db 'Quotient: $'

msg\_remainder db 'Remainder: $'

**Output**:



1. Write a program in assembly language to perform division of 16-bit data.

**Ans**:

**Code:**

org 100h

start:

mov ax, cs

mov ds, ax

mov ax, [dividend]

mov cx, [divisor]

xor dx, dx

div cx

mov [quotient], ax

mov [remainder], dx

mov dx, offset msg\_quotient

mov ah, 09h

int 21h

mov ax, [quotient]

call print\_number

mov dl, 0Dh

int 21h

mov dl, 0Ah

int 21h

mov dx, offset msg\_remainder

mov ah, 09h

int 21h

mov dx, [remainder]

call print\_number

mov ah, 4Ch

int 21h

print\_number:

mov bx, 10

xor cx, cx

xor dx, dx

test ax, ax

jz zero\_number

convert\_loop:

xor dx, dx

div bx

push dx

inc cx

test ax, ax

jnz convert\_loop

zero\_number:

test cx, cx

jz print\_zero

print\_digits:

pop dx

add dl, '0'

mov ah, 02h

int 21h

loop print\_digits

ret

print\_zero:

mov dl, '0'

mov ah, 02h

int 21h

ret

dividend dw 1234h

divisor dw 12

quotient dw 0

remainder dw 0

msg\_quotient db 'Quotient: $'

msg\_remainder db 'Remainder: $'

****