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

Code

.model small

.stack 100h

.data

dividend db 0C6h

divisor db 10h

quotient db ?

remainder db ?

msg1 db 'The output for the Quotient: $'

msg2 db 0Dh, 0Ah, 'The output for the Remainder: $'

.code

main proc

mov ax, @data

mov ds, ax

mov al, dividend

mov bl, divisor

xor ah, ah

div bl

mov quotient, al

mov remainder, ah

mov ah, 09h

lea dx, msg1

int 21h

mov al, quotient

call display\_number

mov ah, 09h

lea dx, msg2

int 21h

mov al, remainder

call display\_number

mov ah, 4ch

int 21h

main endp

display\_number proc

cmp al, 10

jb single\_digit

mov ah, 0

mov bl, 0Ah

div bl

add al, 30h

mov dl, al

mov ah, 02h

int 21h

mov al, ah

add al, 30h

mov dl, al

mov ah, 02h

int 21h

ret

single\_digit:

add al, 30h

mov dl, al

mov ah, 02h

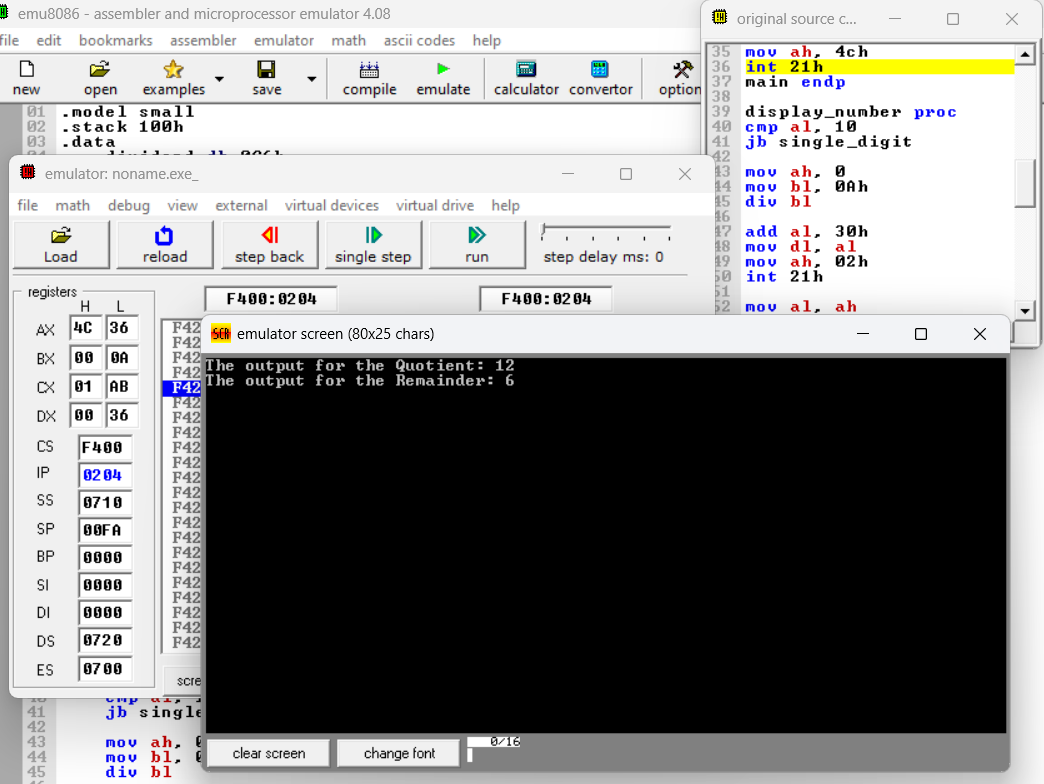
int 21h

ret

display\_number endp

end main

Output:



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

Code

.model small

.stack 100h

.data

dividend dw 1234h

divisor dw 0032h

quotient dw ?

remainder dw ?

msg1 db 'Quotient: $'

msg2 db 0Dh, 0Ah, 'Remainder: $'

.code

main proc

mov ax, @data

mov ds, ax

mov ax, dividend

xor dx, dx

mov bx, divisor

div bx

mov quotient, ax

mov remainder, dx

mov ah, 09h

lea dx, msg1

int 21h

mov ax, quotient

call display\_number

mov ah, 09h

lea dx, msg2

int 21h

mov ax, remainder

call display\_number

mov ah, 4Ch

int 21h

main endp

display\_number proc

push ax

mov cx, 0

mov bx, 10

convert\_digit:

xor dx, dx

div bx

push dx

inc cx

test ax, ax

jnz convert\_digit

print\_digits:

pop dx

add dl, 30h

mov ah, 02h

int 21h

loop print\_digits

pop ax

ret

display\_number endp

end main

Output:

