**LAB TASK – 10**

1. (a) Write a program in assembly language to find L.C.M of two single-digit numbers.

org 100h

jmp start

input1 db "Enter the First Digit: $"

input2 db 0Dh,0Ah,"Enter the Second Digit: $"

output db 0Dh,0Ah,"LCM of the Given two numbers is: $"

start:

; Display "Enter the First Digit:"

mov dx, offset input1

mov ah, 09h

int 21h

; Read first digit

mov ah, 01h

int 21h

sub al, '0' ; Convert ASCII to numerical value

mov cl, al ; Store first number in CL

; Display "Enter the Second Digit:"

mov dx, offset input2

mov ah, 09h

int 21h

; Read second digit

mov ah, 01h

int 21h

sub al, '0' ; Convert ASCII to numerical value

mov ch, al ; Store second number in CH

; Copy CL and CH to registers for GCD calculation

mov al, cl ; AL = first number

mov bl, ch ; BL = second number

GCD\_LOOP:

cmp bl, 0

je DONE\_GCD

mov ah, 0

div bl ; Divide AL by BL, remainder in AH, quotient in AL

mov al, bl

mov bl, ah

jmp GCD\_LOOP

DONE\_GCD:

; Now AL contains the GCD

mov bh, al ; Store GCD in BH

; Calculate LCM = (CL \* CH) / GCD

mov al, cl ; AL = first number

mul ch ; AX = AL \* CH (product of the two numbers)

mov bl, bh ; Move GCD to BL

div bl ; AX / GCD, result in AL

mov bl, al

; Display "LCM of the Given two numbers is:"

mov dx, offset output

mov ah, 09h

int 21h

mov ax, 0 ; Clear AX register

mov al, bl ; Move the result from BL to AL

mov cx, 10 ; Set divisor to 10 (for decimal conversion)

dec\_to\_ascii:

xor dx, dx ; Clear DX for division

div cx ; AX / 10 -> Quotient in AL, Remainder in DL

add dl, '0' ; Convert remainder to ASCII

push dx ; Store remainder on stack

test al, al ; Check if quotient is zero

jnz dec\_to\_ascii ; Repeat if quotient is not zero

print\_decimal:

pop dx ; Get character from stack

mov ah, 02h ; DOS function to display character in DL

int 21h ; Print character

cmp sp, 0 ; Check if stack is empty

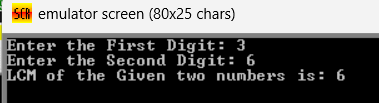
jne print\_decimal ; Repeat if stack not empty

; Exit program

mov ah, 4Ch

int 21h

OUTPUT:



(b) Write an assembly language program to display the nth term of a fibonacci series. “n” must be a single digit number which may be taken from the user.

mov ah, 01h

int 21h

sub al, '0'

mov cl, al

mov al, 0

mov bl, 1

cmp cl, 1

je display\_al

cmp cl, 2

je display\_bl

mov ch, 2

next\_term:

mov dl, al

add al, bl

mov bl, dl

inc ch

cmp ch, cl

jne next\_term

display\_al:

add al, '0'

mov dl, al

mov ah, 02h

int 21h

jmp end\_program

display\_bl:

add bl, '0'

mov dl, bl

mov ah, 02h

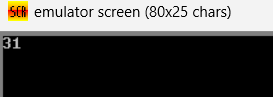
int 21h

end\_program:

mov ah, 4Ch

int 21h

OUTPUT:



2. Write an assembly language program to find the factorial of a given single-digit number.

org 100h ;

start:

; Read the single-digit number

mov ah, 01h ; Function 01h - Read character from standard input

int 21h

sub al, '0' ; Convert ASCII to integer

mov cl, al ; Save number in CL for loop counter

mov ax, 1 ; Initialize AX with 1 (factorial base case)

factorial\_loop:

cmp cl, 1 ; If CL <= 1, we're done

jle done

mul cl ; Multiply AX by CL

dec cl ; Decrement CL

jmp factorial\_loop

done:

; Display the result in AX as a decimal number

mov bx, 10 ; Base 10 for division

mov cx, 0 ; Clear CX for high word of dividend

lea dx, buffer ; Point to the buffer for result storage

add dx, 4 ; Start filling from the end of buffer

convert\_to\_ascii:

div bx ; Divide AX by 10

add dl, '0' ; Convert remainder to ASCII

dec dx ; Move to the previous position

or ax, ax ; Check if AX is zero

jnz convert\_to\_ascii

inc dx ; Move DX to the first character of the number

; Display the result

mov ah, 09h

int 21h

; Exit program

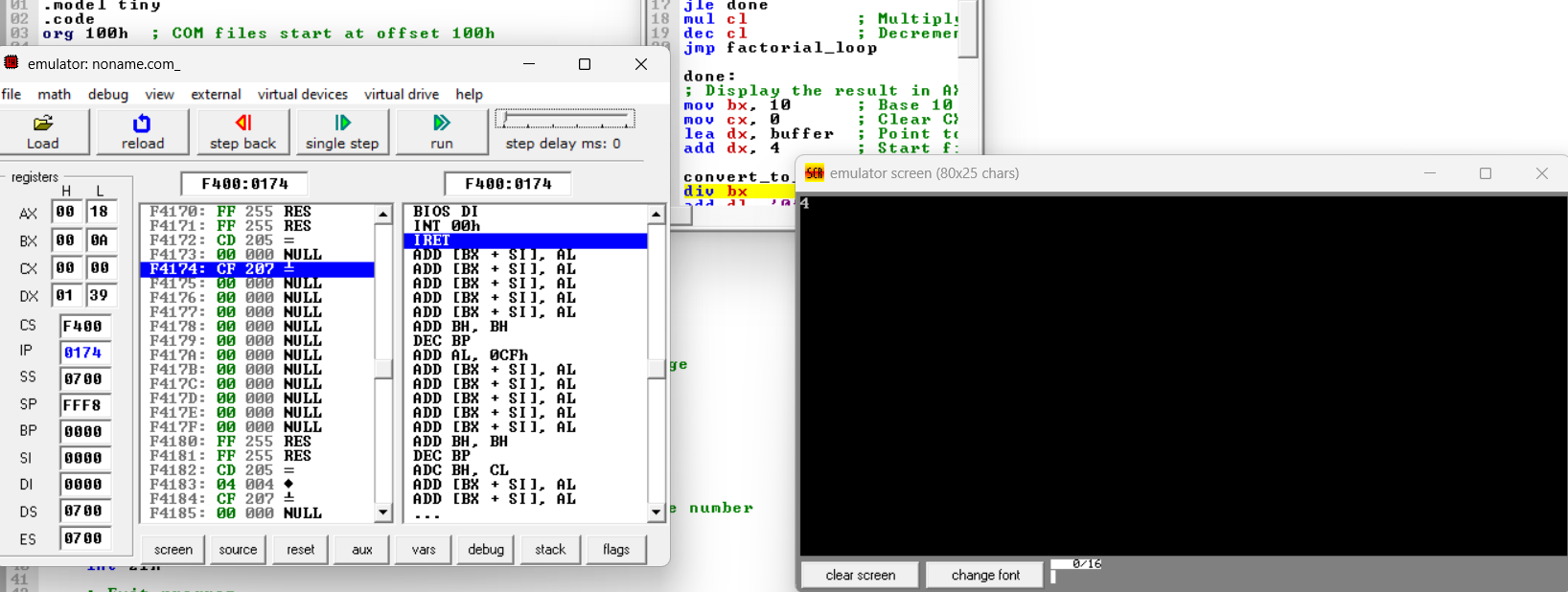
mov ah, 4Ch

int 21h

buffer db '0000$', '$' ; Buffer to hold up to 4 digits plus end marker

end start

OUTPUT:



Github link:

<https://github.com/SRIVALLI2005/-ABM-MODULE->