**System Programming Lab**

**BCSE 3rd year 1st Semester**

**Name: Sayantan Biswas**

**Roll No: 001910501057**

**----------------------------------------------------------------------**

***>> I have done this assignment using emu8086***

* **Assignment 2**

**Q1:**

; Write and test a MASM program to add and subtract two 16 bit numbers.

.model small

.stack 300h

.data

msg1 db 0AH,0DH,'ENTER 1ST NUMBER: $'

msg2 db 0AH,0DH,'ENTER 2ND NUMBER: $'

msg3 db 0AH,0DH,'THE RESULT AFTER ADDITION: $'

msg4 db 0AH,0DH,'THE RESULT AFTER SUBTRACTION: $'

space db ' $'

endl db 0AH,0DH,'$'

val1 dw ?

val2 dw ?

.code

print macro msg

push ax

push dx

mov ah, 09h

lea dx, msg

int 21h

pop dx

pop ax

endm

main proc

mov ax,@data

mov ds,ax

start:

print msg1

call readnum

mov val1, ax

print msg2

call readnum

mov val2, ax

print msg3

mov ax, val1

mov bx, val2

add ax,bx

call writenum

print msg4

mov ax, val1

mov bx, val2

sub ax,bx

call writenum

exit:

mov ah, 4ch

int 21h

main endp

readnum proc near

; this procedure will take a number as input from user and store in AX

; input : none

; output : AX

push bx

push cx

mov cx,0ah

mov bx,00h

loopnum:

mov ah,01h

int 21h

cmp al,'0'

jb skip

cmp al,'9'

ja skip

sub al,'0'

push ax

mov ax,bx

mul cx

mov bx,ax

pop ax

mov ah,00h

add bx,ax

jmp loopnum

skip:

mov ax,bx

pop cx

pop bx

ret

readnum endp

writenum proc near

; this procedure will display a decimal number

; input : AX

; output : none

push ax

push bx

push cx

push dx

xor cx, cx

mov bx, 0ah

@output:

xor dx, dx

div bx ; divide AX by BX

push dx ; push remainder onto the STACK

inc cx

or ax, ax

jne @output

mov ah, 02h ; output

@display:

pop dx ; pop a value(remainder) from STACK to DX

or dl, 30h ; convert decimal to ascii code

int 21h

loop @display

pop dx

pop cx

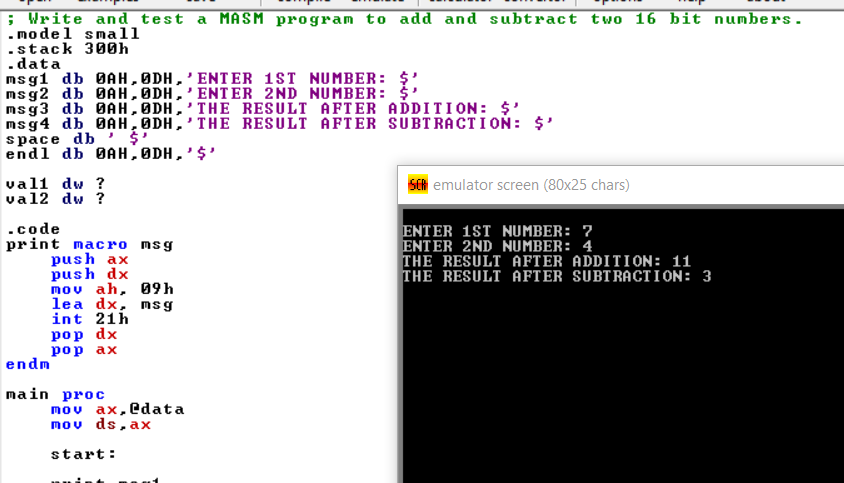
pop bx

pop ax

ret

writenum endp

end main



**Q3:**

; Write and test a program to print pairs of even numbers where the summation of the numbers in each pair is 100.

.model small

.stack 300h

.data

char1 db '($'

char2 db ')$'

space db ' $'

endl db 0AH,0DH,'$'

val1 dw ?

val2 dw ?

.code

print macro msg

push ax

push dx

mov ah, 09h

lea dx, msg

int 21h

pop dx

pop ax

endm

main proc

mov ax,@data

mov ds,ax

start:

mov bx, 100

mov ax, 100

loop1:

print char1

call writenum

print space

mov val1, ax

mov ax, bx

mov cx, val1

sub ax, cx

call writenum

print char2

print space

mov ax, val1

sub ax,2

jnz loop1

print char1

call writenum

print space

mov ax, 100

call writenum

print char2

exit:

mov ah, 4ch

int 21h

main endp

writenum proc near

; this procedure will display a decimal number

; input : AX

; output : none

push ax

push bx

push cx

push dx

xor cx, cx

mov bx, 0ah

@output:

xor dx, dx

div bx ; divide AX by BX

push dx ; push remainder onto the STACK

inc cx

or ax, ax

jne @output

mov ah, 02h ; output

@display:

pop dx ; pop a value(remainder) from STACK to DX

or dl, 48 ; convert decimal to ascii code

int 21h

loop @display

pop dx

pop cx

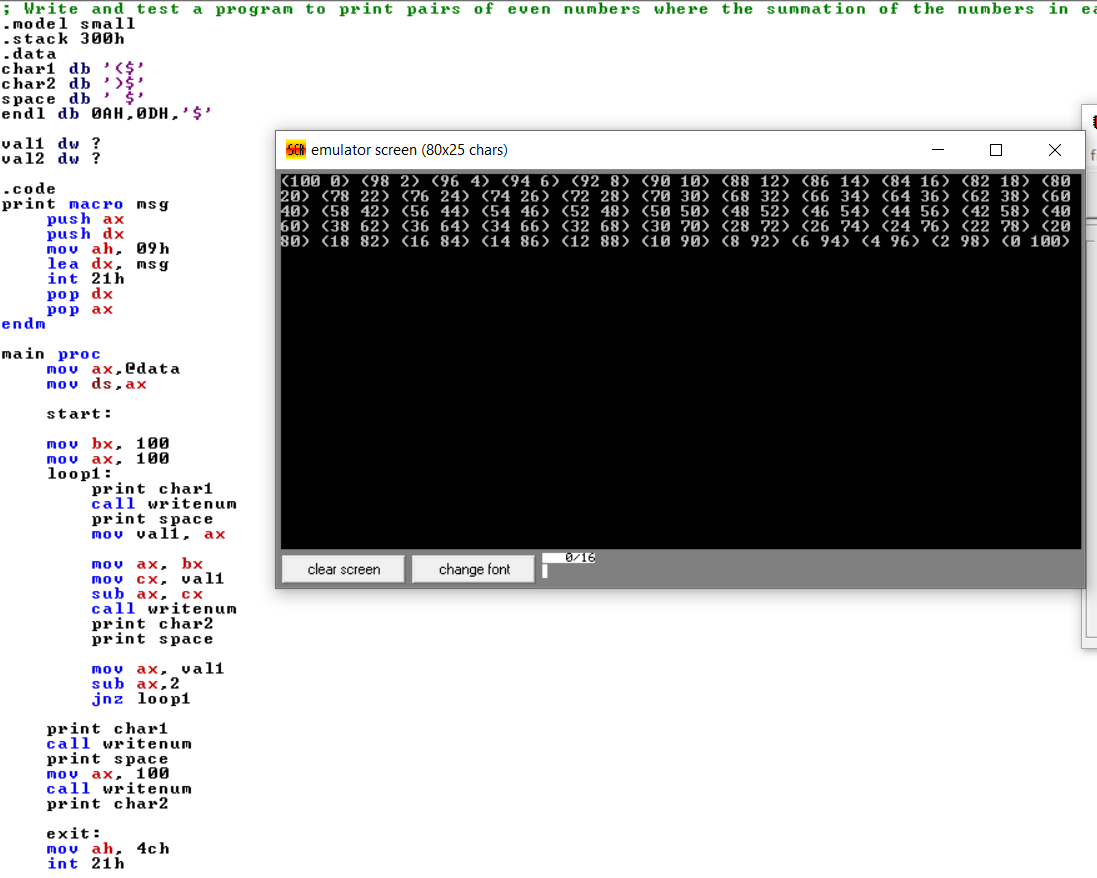
pop bx

pop ax

ret

writenum endp

end main



**Q4:**

; Write and test a MASM program to multiply two 32 bit numbers.

.model small

.stack 300h

.data

msg1 db 0AH,0DH,'ENTER 1ST NUMBER: $'

msg2 db 0AH,0DH,'ENTER 2ND NUMBER: $'

msg3 db 0AH,0DH,'THE RESULT AFTER MULTIPLYING IS: $'

space db ' $'

endl db 0AH,0DH,'$'

val1 dw ?

val2 dw ?

.code

print macro msg

push ax

push dx

mov ah, 09h

lea dx, msg

int 21h

pop dx

pop ax

endm

main proc

mov ax,@data

mov ds,ax

start:

print msg1

call readnum

mov val1, ax

print msg2

call readnum

mov val2, ax

print msg3

mul val1

call writenum

exit:

mov ah, 4ch

int 21h

main endp

readnum proc near

; this procedure will take a number as input from user and store in AX

; input : none

; output : AX

push bx

push cx

mov cx,0ah

mov bx,00h

loopnum:

mov ah,01h

int 21h

cmp al,'0'

jb skip

cmp al,'9'

ja skip

sub al,'0'

push ax

mov ax,bx

mul cx

mov bx,ax

pop ax

mov ah,00h

add bx,ax

jmp loopnum

skip:

mov ax,bx

pop cx

pop bx

ret

readnum endp

writenum proc near

; this procedure will display a decimal number

; input : AX

; output : none

push ax

push bx

push cx

push dx

xor cx, cx

mov bx, 0ah

@output:

xor dx, dx

div bx ; divide AX by BX

push dx ; push remainder onto the STACK

inc cx

or ax, ax

jne @output

mov ah, 02h ; set output function

@display:

pop dx ; pop a value(remainder) from STACK to DX

or dl, 30h ; convert decimal to ascii code

int 21h

loop @display

pop dx

pop cx

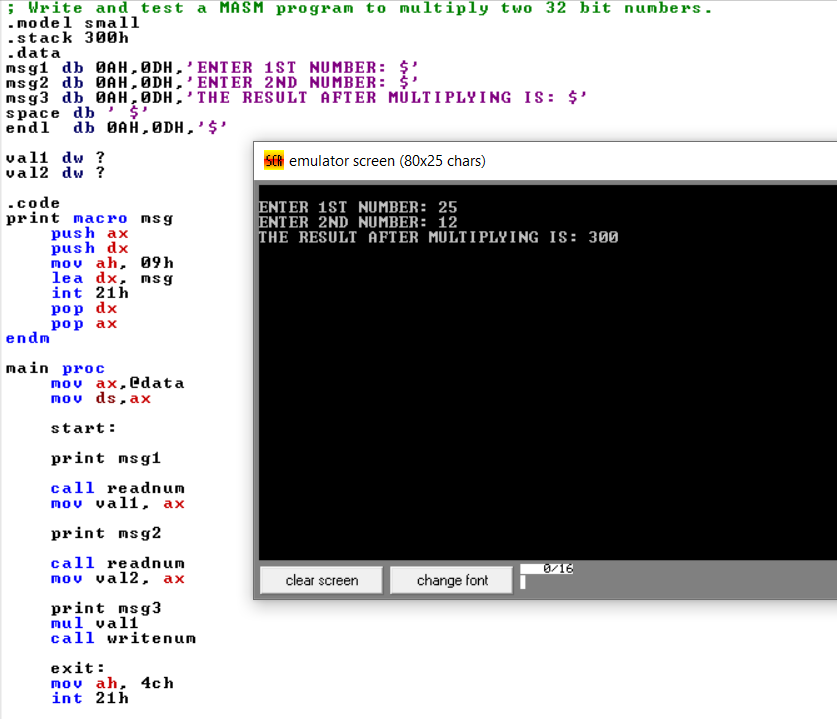
pop bx

pop ax

ret

writenum endp

end main



**Q6:**

; Write and test a MASM program to Print Fibonacci series up to 10 terms.

.model small

.stack 100h

.data

x db 0ah, 0dh, "$"

.code

main proc

mov ax, @data

mov ds, ax

mov al, 0

mov bl, 1

call display\_number

mov al, bl

call display\_number

mov al, 0

mov ch, 02h

l1:

mov cl, bl

add bl, al

mov al, bl

call display\_number

mov al, cl

inc ch

cmp ch, 10

jne l1

mov ah, 4ch

int 21h

main endp

display\_number proc

push bx

mov bl, 10

mov bh, 00h

l2:

mov ah, 00h

div bl

push ax

inc bh

cmp al, 0

jne l2

l3:

pop dx

mov dl, dh

mov dh, 0

add dl, 48

mov ah, 02h

int 21h

dec bh

cmp bh, 0

jne l3

lea dx, x

mov ah, 09h

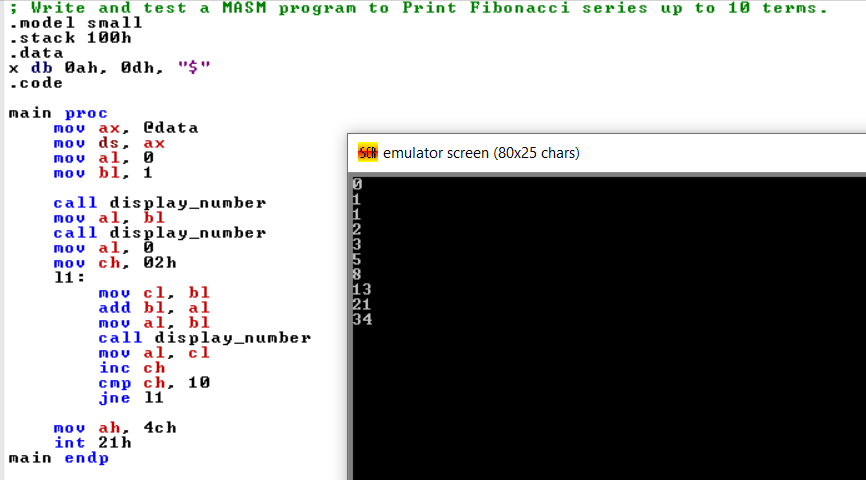
int 21h

pop bx

ret

display\_number endp

end



**Q10:**

; Write and test a MASM program to print prime numbers between 1 to 100.

.model small

.stack 100

.data

res db 3 dup(0)

msg db "Primes from 1 to 100 : ",13,10,"$"

.code

main proc

mov ax,@data

mov ds,ax

lea dx,msg

mov ah,9

int 21h

mov dl,1

mov cx,25

l1:

mov bl, 02

add dl, 01h

cmp dl, 02h

je print

cmp dl, 03h

je print

cmp dl, 04h

jge Logic

logic:

mov ah, 00

mov al, dl

div bl

cmp ah, 00

je l1

add bl, 01h

cmp bl, al

jle Logic

jmp print

print:

mov al, dl

mov ah,00

call output

loop l1

exit:

mov ah, 4ch

int 21h

ret

main endp

output proc ;data is in ax as always

push ax

push bx

push cx

push dx

mov cx,0

mov bx,10

mov si,offset res

loop1:

mov dx,0

div bx

add dl,30h

push dx

inc cx

cmp ax,9

jg loop1

add al,30h

mov [si],al

loop2:

pop ax

inc si

mov [si],al

loop loop2

mov dl,res[0]

mov ah,2

int 21h

mov dl,res[1]

int 21h

mov dl,32

int 21h

pop dx

pop cx

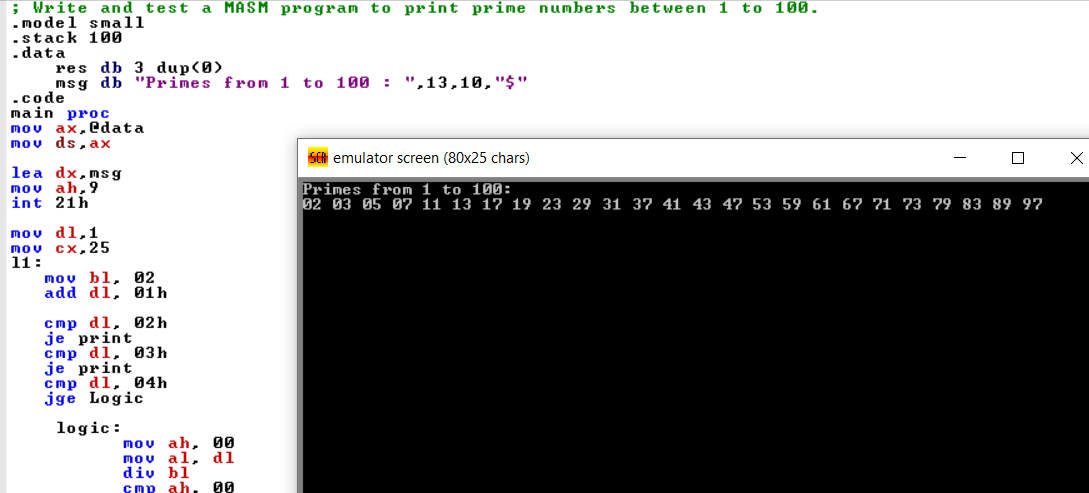
pop bx

pop ax

ret

output endp

end main



-----------------------------