**Code:**

section .data

num db 'Enter a number',10

nlen equ $-num

result db 'The sum is:'

rlen equ $-result

SYS\_WRITE equ 4

SYS\_READ equ 3

STDOUT equ 1

STDIN equ 2

SYS\_EXIT equ 1

RETURN\_CODE\_SUCCESS equ 0

section .bss

num1 resb 4

num2 resb 4

res resb 4

section .text

global \_start

\_start:

mov eax,SYS\_WRITE

mov ebx,STDOUT

mov ecx,num

mov edx,nlen

int 0x80

mov eax,SYS\_READ

mov ebx,STDIN

mov ecx,num1

mov edx,2

int 0x80

mov eax,SYS\_WRITE

mov ebx,STDOUT

mov ecx,num

mov edx,nlen

int 0x80

mov eax,SYS\_READ

mov ebx,STDIN

mov ecx,num2

mov edx,2

int 0x80

mov eax,[num1]

sub eax,'0'

mov ebx,[num2]

sub ebx,'0'

add eax,ebx

add eax,'0'

mov [res],eax

mov eax,SYS\_WRITE

mov ebx,STDOUT

mov ecx,result

mov edx,rlen

int 0x80

mov eax,SYS\_WRITE

mov ebx,STDOUT

mov ecx,res

mov edx,1

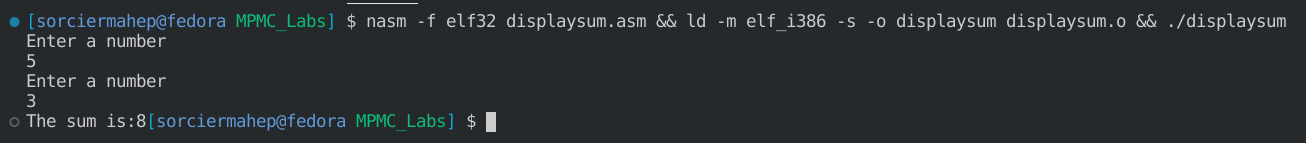
int 0x80

mov eax,SYS\_EXIT

mov ebx,RETURN\_CODE\_SUCCESS

int 80h

**Output:**

****

**Code:**

section .data

input db 10,'enter a number : '

ilen equ $-input

output\_s db 10,'the sum is : '

olens equ $-output\_s

output\_d db 10,'the dif is : '

olend equ $-output\_d

output\_p db 10,'the mul is : '

olenp equ $-output\_p

output\_q db 10,'the quo is : '

olenq equ $-output\_q

output\_r db 10,'the rem is : '

olenr equ $-output\_r

section .bss

num1 resb 4

num2 resb 4

sum resb 4

diff resb 4

prod resb 4

quo resb 4

rem resb 4

section .text

global \_start

\_start:

mov eax, 4

mov ebx, 1

mov ecx, input

mov edx , ilen

int 0x80

mov eax,3

mov ebx,2

mov ecx,num1

mov edx,2

int 0x80

mov eax, 4

mov ebx, 1

mov ecx, input

mov edx , ilen

int 0x80

mov eax,3

mov ebx,2

mov ecx,num2

mov edx,2

int 0x80

mov eax,[num1]

sub eax,'0'

mov ebx,[num2]

sub ebx,'0'

add eax,ebx

add eax,'0'

mov [sum],eax

mov eax,4

mov ebx,1

mov ecx,output\_s

mov edx,olens

int 0x80

mov eax,4

mov ebx,1

mov ecx,sum

mov edx,2

int 0x80

mov eax,[num1]

sub eax,'0'

mov ebx,[num2]

sub ebx,'0'

sub eax,ebx

add eax,'0'

mov [diff],eax

mov eax,4

mov ebx,1

mov ecx,output\_d

mov edx,olend

int 0x80

mov eax,4

mov ebx,1

mov ecx,diff

mov edx,2

int 0x80

mov al,[num1]

sub al,'0'

mov bl,[num2]

sub bl,'0'

mul bl

add al,'0'

mov [prod],al

mov eax,4

mov ebx,1

mov ecx,output\_p

mov edx,olenp

int 0x80

mov eax,4

mov ebx,1

mov ecx,prod

mov edx,2

int 0x80

mov al,[num1]

sub al,'0'

mov bl,[num2]

sub bl,'0'

div bl

add al,'0'

add ah,'0'

mov [quo],al

mov [rem],ah

mov eax,4

mov ebx,1

mov ecx,output\_q

mov edx,olenq

int 0x80

mov eax,4

mov ebx,1

mov ecx,quo

mov edx,2

int 0x80

mov eax,4

mov ebx,1

mov ecx,output\_r

mov edx,olenr

int 0x80

mov eax,4

mov ebx,1

mov ecx,rem

mov edx,2

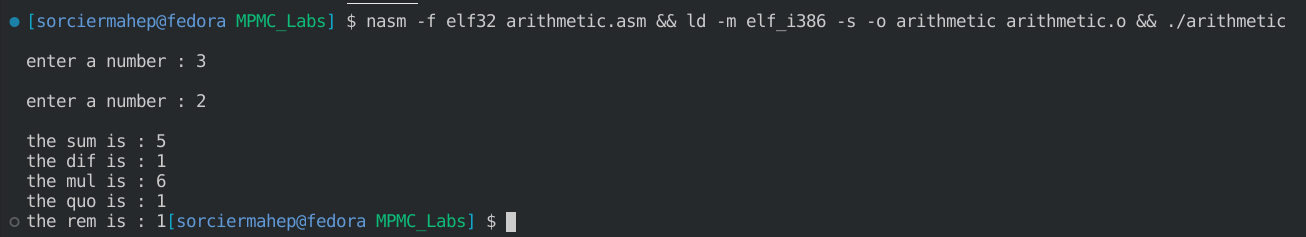
int 0x80

mov eax, 1

mov ebx, 0

int 0x80

**Output:**



**Code:**

section .data

askuserrect db "Enter the length of a rectangle: ", 10

askuserrectlen equ $-askuserrect

askUserBreadth db "Enter the breadth of a rectangle: ", 10

askBreadthLen equ $-askUserBreadth

displayarearect db "The area is : ", 9

displayarearectlen equ $-displayarearect

displayperirect db "The perimeter is: ", 9

displayperirectlen equ $-displayperirect

askusertrig db "Enter the side of a triangle: ", 10

askusertriglen equ $-askusertrig

askUserHeight db "Enter the height of a triangle: ", 10

askHeightLen equ $-askUserHeight

displayareatrig db "The area is : ", 9

displayareatriglen equ $-displayareatrig

displayperitrig db "The perimeter is: ", 9

displayperitriglen equ $-displayperitrig

newLineMsg db 0xA, 0xD

newLineLen equ $-newLineMsg

section .bss

length resb 2

breadth resb 2

inter resb 2

area resb 2

perimeter resb 2

side1 resb 2

side2 resb 2

side3 resb 2

height resb 2

section .text

global \_start

\_start:

;-----Rectangle Calculation

mov eax, 4

mov ebx, 1

mov ecx, askuserrect

mov edx, askuserrectlen

int 0x80

mov eax, 3

mov ebx, 2

mov ecx, length

mov edx, 2

int 0x80

mov eax, 4

mov ebx, 1

mov ecx, askUserBreadth

mov edx, askBreadthLen

int 0x80

mov eax, 3

mov ebx, 2

mov ecx, breadth

mov edx, 2

int 0x80

mov al, [length]

sub al, '0'

mov bl, [breadth]

sub bl, '0'

mul bl

add al, '0'

mov [area], al

mov eax, 4

mov ebx, 1

mov ecx, displayarearect

mov edx, displayarearectlen

int 0x80

mov eax, 4

mov ebx, 1

mov ecx, area

mov edx, 2

int 0x80

mov eax, 4

mov ebx, 1

mov ecx, newLineMsg

mov edx, newLineLen

int 0x80

mov eax, [length]

sub eax, '0'

mov ebx, [breadth]

sub ebx, '0'

add eax, ebx

add eax, '0'

mov [inter], eax

mov al, [inter]

sub al, '0'

mov bl, '2'

sub bl, '0'

mul bl

add al, '0'

mov [perimeter], al

mov eax, 4

mov ebx, 1

mov ecx, displayperirect

mov edx, displayperirectlen

int 0x80

mov eax, 4

mov ebx, 1

mov ecx, perimeter

mov edx, 2

int 0x80

mov eax, 4

mov ebx, 1

mov ecx, newLineMsg

mov edx, newLineLen

int 0x80

;-----Triangle calculation

mov eax, 4

mov ebx, 1

mov ecx, askusertrig

mov edx, askusertriglen

int 0x80

mov eax, 3

mov ebx, 2

mov ecx, side1

mov edx, 2

int 0x80

mov eax, 3

mov ebx, 2

mov ecx, side2

mov edx, 2

int 0x80

mov eax, 3

mov ebx, 2

mov ecx, side3

mov edx, 2

int 0x80

mov eax, 4

mov ebx, 1

mov ecx, askUserHeight

mov edx, askHeightLen

int 0x80

mov eax, 3

mov ebx, 2

mov ecx, height

mov edx, 2

int 0x80

mov al, [side1]

sub al, '0'

mov bl, [height]

sub bl, '0'

mul bl

add al, '0'

mov [inter], al

mov al, [inter]

sub al, '0'

mov bl, '2'

sub bl, '0'

div bl

add al, '0'

mov [area], al

mov eax, 4

mov ebx, 1

mov ecx, displayareatrig

mov edx, displayareatriglen

int 0x80

mov eax, 4

mov ebx, 1

mov ecx, area

mov edx, 2

int 0x80

mov eax, 4

mov ebx, 1

mov ecx, newLineMsg

mov edx, newLineLen

int 0x80

mov eax, [side1]

sub eax, '0'

mov ebx, [side2]

sub ebx, '0'

add eax, ebx

add eax, '0'

mov [inter], eax

mov eax, [side3]

sub eax, '0'

mov ebx, [inter]

sub ebx, '0'

add eax, ebx

add eax, '0'

mov [perimeter], eax

mov eax, 4

mov ebx, 1

mov ecx, displayperitrig

mov edx, displayperitriglen

int 0x80

mov eax, 4

mov ebx, 1

mov ecx, perimeter

mov edx, 2

int 0x80

mov eax, 4

mov ebx, 1

mov ecx, newLineMsg

mov edx, newLineLen

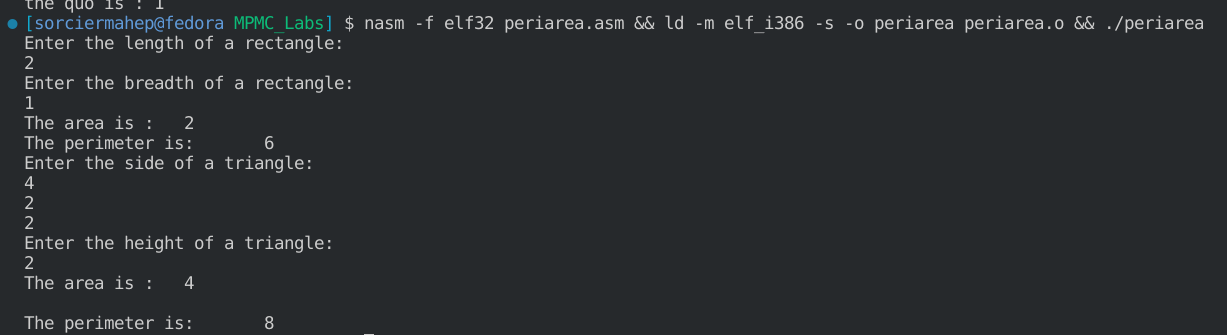
int 0x80

mov eax, 1

xor ebx, ebx

int 0x80

**Output:**

****

**Code:**

section .data

SYS\_WRITE equ 4

STDOUT equ 1

SYS\_READ equ 3

STDIN equ 2

SYS\_EXIT equ 1

RETURN\_CODE\_SUCCESS equ 0

input db 'Enter a number:'

input\_len equ $-input

section .bss

num resb 2

section .text

global \_start

\_start:

mov eax,SYS\_WRITE

mov ebx,STDOUT

mov ecx,input

mov edx,input\_len

int 0x80

mov eax,SYS\_READ

mov ebx,STDIN

mov ecx,num

mov edx,2

int 0x80

inc byte [num]

mov eax,SYS\_WRITE

mov ebx,STDOUT

mov ecx,num

mov edx,2

int 0x80

inc byte [num]

mov eax,SYS\_WRITE

mov ebx,STDOUT

mov ecx,num

mov edx,2

int 0x80

inc byte [num]

mov eax,SYS\_WRITE

mov ebx,STDOUT

mov ecx,num

mov edx,2

int 0x80

inc byte [num]

mov eax,SYS\_WRITE

mov ebx,STDOUT

mov ecx,num

mov edx,2

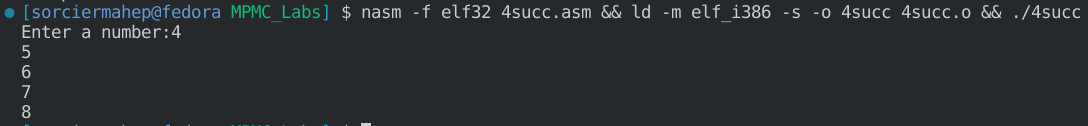
int 0x80

mov eax,SYS\_EXIT

mov ebx,RETURN\_CODE\_SUCCESS

int 0x80

**Output:**

****