

section .data

num1 db 4 ; first number

num2 db 3 ; second number

result db 0 ; will store sum

msg db "Result: ", 0

newline db 10,0 ; newline char

section .text

global \_start

\_start:

; load numbers

mov al, [num1]

add al, [num2]

mov [result], al

; convert result (0–9) to ASCII

add byte [result], '0'

; print "Result: "

mov eax, 4 ; sys\_write

mov ebx, 1 ; stdout

mov ecx, msg

mov edx, 8 ; length of "Result: "

int 0x80

; print result

mov eax, 4

mov ebx, 1

mov ecx, result

mov edx, 1

int 0x80

; print newline

mov eax, 4

mov ebx, 1

mov ecx, newline

mov edx, 1

int 0x80

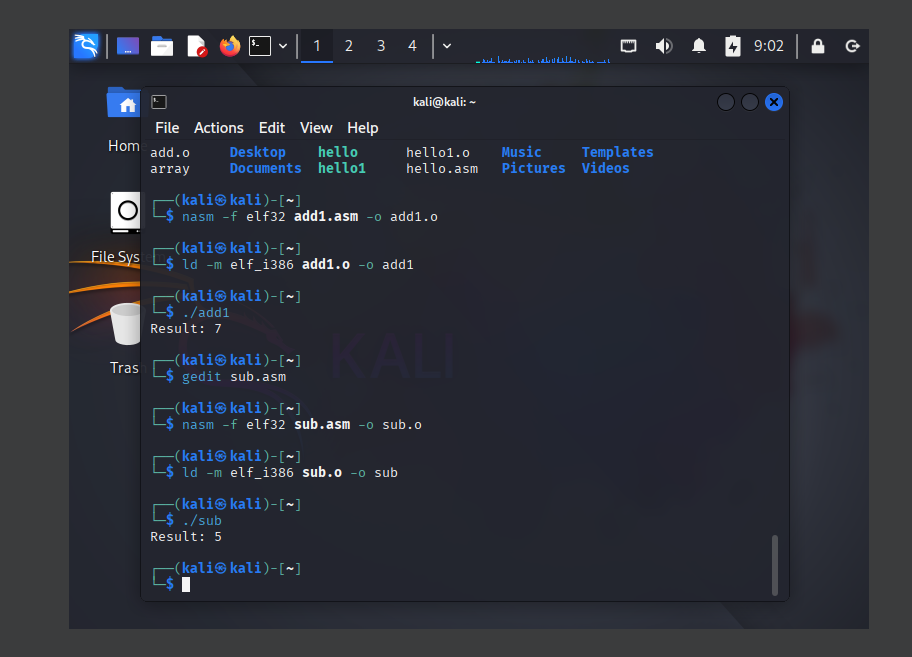
; exit

mov eax, 1

xor ebx, ebx

int 0x80

Subtraction



section .data

num1 db 9 ; first number

num2 db 4 ; second number

result db 0 ; will store difference

msg db "Result: ", 0

newline db 10,0 ; newline char

section .text

global \_start

\_start:

; load numbers

mov al, [num1]

sub al, [num2] ; subtract num2 from num1

mov [result], al

; convert result (0–9) to ASCII

add byte [result], '0'

; print "Result: "

mov eax, 4 ; sys\_write

mov ebx, 1 ; stdout

mov ecx, msg

mov edx, 8 ; length of "Result: "

int 0x80

; print result

mov eax, 4

mov ebx, 1

mov ecx, result

mov edx, 1

int 0x80

; print newline

mov eax, 4

mov ebx, 1

mov ecx, newline

mov edx, 1

int 0x80

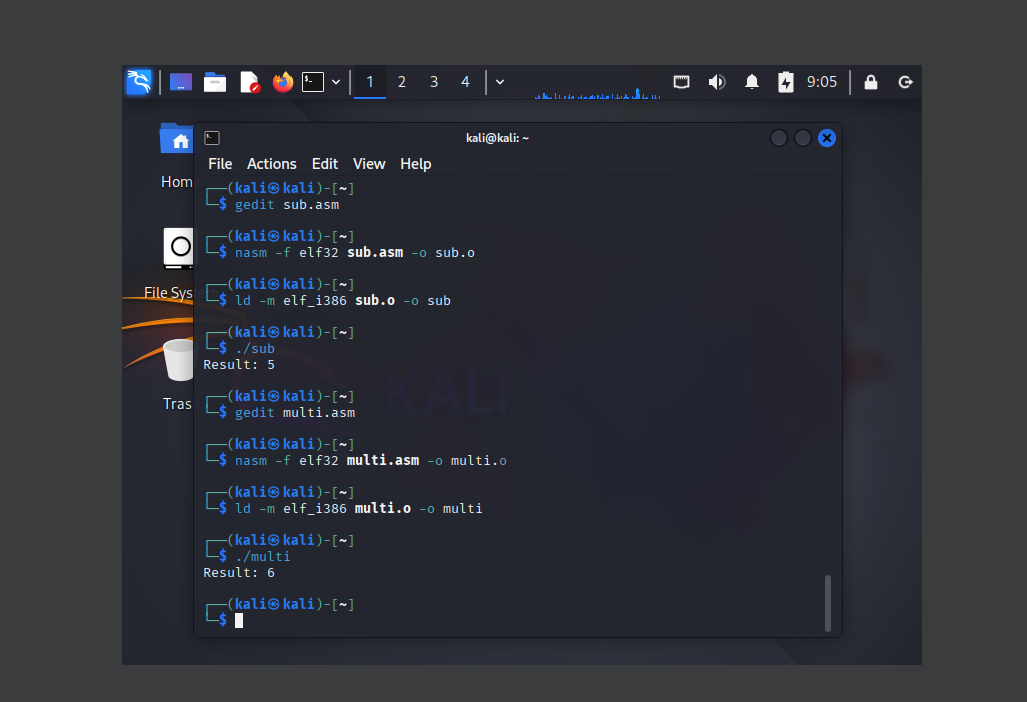
; exit

mov eax, 1

xor ebx, ebx

int 0x80

Mutiplication-



section .data

num1 db 3 ; first number

num2 db 2 ; second number

result db 0 ; will store product

msg db "Result: ", 0

newline db 10,0

section .text

global \_start

\_start:

; load numbers into registers

mov al, [num1]

mov bl, [num2]

; multiply AL \* BL → result in AL

mul bl ; unsigned multiply AL \* BL → AX

mov [result], al ; store only low byte (works for 0–9)

; convert result to ASCII

add byte [result], '0'

; print "Result: "

mov eax, 4

mov ebx, 1

mov ecx, msg

mov edx, 8

int 0x80

; print result

mov eax, 4

mov ebx, 1

mov ecx, result

mov edx, 1

int 0x80

; print newline

mov eax, 4

mov ebx, 1

mov ecx, newline

mov edx, 1

int 0x80

; exit

mov eax, 1

xor ebx, ebx

int 0x80

Division-

section .data

num1 db 8 ; dividend

num2 db 2 ; divisor

result db 0 ; will store quotient

msg db "Result: ", 0

newline db 10,0

section .text

global \_start

\_start:

; load dividend into AL

mov al, [num1]

mov bl, [num2]

; clear AH for division

xor ah, ah

; divide AX by BL → quotient in AL, remainder in AH

div bl

mov [result], al

; convert result (0–9) to ASCII

add byte [result], '0'

; print "Result: "

mov eax, 4

mov ebx, 1

mov ecx, msg

mov edx, 8

int 0x80

; print result

mov eax, 4

mov ebx, 1

mov ecx, result

mov edx, 1

int 0x80

; print newline

mov eax, 4

mov ebx, 1

mov ecx, newline

mov edx, 1

int 0x80

; exit

mov eax, 1

xor ebx, ebx

int 0x80

