**1)**

section .bss

num1 resb 9

num2 resb 9

section .data

sys\_out equ 4

sys\_in equ 3

stdin equ 2

stdout equ 1

mss1 db 'ENTER FIRST NUMBER: '

ml1 equ $-mss1

mss2 db 'ENTER THE SECOND NUMBER: '

ml2 equ $-mss2

mss3 db 'LARGEST: '

ml3 equ $-mss3

section .text

global \_start:

num1Largest:

mov eax, sys\_out

mov ebx, stdout

mov ecx, mss3

mov edx, ml3

int 80h

mov eax, sys\_out

mov ebx, stdout

mov ecx, num1

mov edx, 9

int 80h

jmp exit

num2Largest:

mov eax, sys\_out

mov ebx, stdout

mov ecx, mss3

mov edx, ml3

int 80h

mov eax, sys\_out

mov ebx, stdout

mov ecx, num2

mov edx, 9

int 80h

jmp exit

\_start:

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* INPUT

mov eax, sys\_out ;NUM1

mov ebx, stdout

mov ecx, mss1

mov edx, ml1

int 80h

mov eax, sys\_in

mov ebx, stdin

mov ecx, num1

mov edx, 9

int 80h

mov eax, sys\_out ;NUM2

mov ebx, stdout

mov ecx, mss2

mov edx, ml2

int 80h

mov eax, sys\_in

mov ebx, stdin

mov ecx, num2

mov edx, 9

int 80h

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

mov al,[num1]

sub al,'0'

mov bl,[num2]

sub bl,'0'

cmp al,bl

jg num1Largest

jl num2Largest

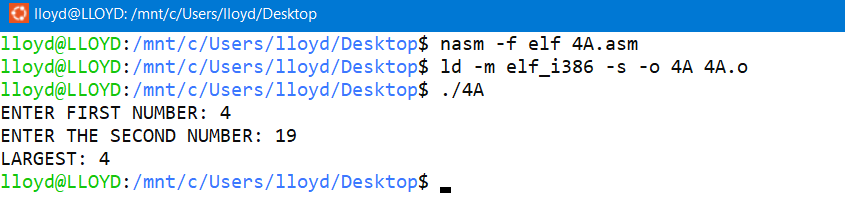
exit:

mov eax, 1

mov ebx, 0

int 80h

**OUTPUT**

****

**2)**

section .bss

num1 resb 9

num2 resb 9

num3 resb 9

section .data

sys\_out equ 4

sys\_in equ 3

stdin equ 2

stdout equ 1

mss1 db 'ENTER FIRST NUMBER: '

ml1 equ $-mss1

mss2 db 'ENTER THE SECOND NUMBER: '

ml2 equ $-mss2

mss3 db 'ENTER THE THIRD NUMBER: '

ml3 equ $-mss3

mss4 db 'LARGEST: '

ml4 equ $-mss4

section .text

global \_start:

ThirdCheck1:

mov al,[num1]

sub al,'0'

mov bl,[num3]

sub bl,'0'

cmp al,bl

jg num1Largest

jl num3Largest

jmp exit

ThirdCheck2:

mov al,[num2]

sub al,'0'

mov bl,[num3]

sub bl,'0'

cmp al,bl

jl num3Largest

jg num2Largest

jmp exit

num1Largest:

mov eax, sys\_out

mov ebx, stdout

mov ecx, mss4

mov edx, ml4

int 80h

mov eax, sys\_out

mov ebx, stdout

mov ecx, num1

mov edx, 9

int 80h

jmp exit

num2Largest:

mov eax, sys\_out

mov ebx, stdout

mov ecx, mss4

mov edx, ml4

int 80h

mov eax, sys\_out

mov ebx, stdout

mov ecx, num2

mov edx, 9

int 80h

jmp exit

num3Largest:

mov eax, sys\_out

mov ebx, stdout

mov ecx, mss4

mov edx, ml4

int 80h

mov eax, sys\_out

mov ebx, stdout

mov ecx, num3

mov edx, 9

int 80h

jmp exit

\_start:

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* INPUT

mov eax, sys\_out ;NUM1

mov ebx, stdout

mov ecx, mss1

mov edx, ml1

int 80h

mov eax, sys\_in

mov ebx, stdin

mov ecx, num1

mov edx, 9

int 80h

mov eax, sys\_out ;NUM2

mov ebx, stdout

mov ecx, mss2

mov edx, ml2

int 80h

mov eax, sys\_in

mov ebx, stdin

mov ecx, num2

mov edx, 9

int 80h

mov eax, sys\_out ;NUM3

mov ebx, stdout

mov ecx, mss3

mov edx, ml3

int 80h

mov eax, sys\_in

mov ebx, stdin

mov ecx, num3

mov edx, 9

int 80h

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

mov al,[num1]

sub al,'0'

mov bl,[num2]

sub bl,'0'

cmp al,bl

jg ThirdCheck1

jl ThirdCheck2

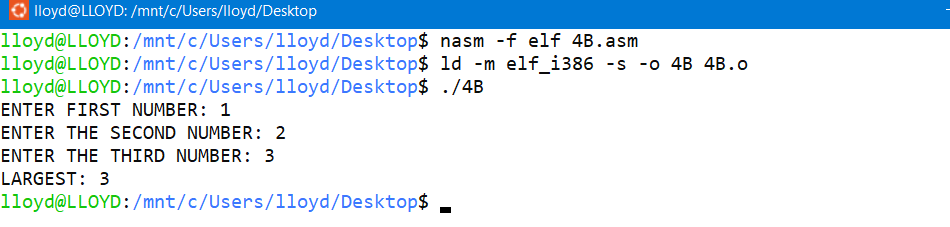
exit:

mov eax, 1

mov ebx, 0

int 80h

**OUTPUT**

****

**3)**

section .bss

num1 resb 9

num2 resb 9

section .data

sys\_out equ 4

sys\_in equ 3

stdin equ 2

stdout equ 1

mss1 db 'ENTER FIRST NUMBER: '

ml1 equ $-mss1

mss2 db 'ENTER THE SECOND NUMBER: '

ml2 equ $-mss2

mss3 db 'SMALLEST: '

ml3 equ $-mss3

section .text

global \_start:

num1Smallest:

mov eax, sys\_out

mov ebx, stdout

mov ecx, mss3

mov edx, ml3

int 80h

mov eax, sys\_out

mov ebx, stdout

mov ecx, num1

mov edx, 9

int 80h

jmp exit

num2Smallest:

mov eax, sys\_out

mov ebx, stdout

mov ecx, mss3

mov edx, ml3

int 80h

mov eax, sys\_out

mov ebx, stdout

mov ecx, num2

mov edx, 9

int 80h

jmp exit

\_start:

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* INPUT

mov eax, sys\_out ;NUM1

mov ebx, stdout

mov ecx, mss1

mov edx, ml1

int 80h

mov eax, sys\_in

mov ebx, stdin

mov ecx, num1

mov edx, 9

int 80h

mov eax, sys\_out ;NUM2

mov ebx, stdout

mov ecx, mss2

mov edx, ml2

int 80h

mov eax, sys\_in

mov ebx, stdin

mov ecx, num2

mov edx, 9

int 80h

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

mov al,[num1]

sub al,'0'

mov bl,[num2]

sub bl,'0'

cmp al,bl

jl num1Smallest

jg num2Smallest

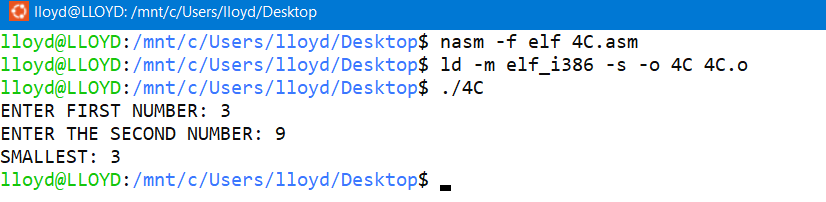
exit:

mov eax, 1

mov ebx, 0

int 80h

**OUTPUT**

****

**4)**

section .bss

num1 resb 9

section .data

sys\_out equ 4

sys\_in equ 3

stdin equ 2

stdout equ 1

mss1 db 'ENTER A NUMBER: '

ml1 equ $-mss1

mss2 db 'NUMBER YOU ENTERED IS SMALLER THAN 5'

ml2 equ $-mss2

mss3 db 'NUMBER YOU ENTERED IS NOT SMALLER THAN 5'

ml3 equ $-mss3

mss4 db 'NUMBER YOU ENTERED IS EQUAL TO 5'

ml4 equ $-mss4

section .text

global \_start:

case1:

mov eax, sys\_out

mov ebx, stdout

mov ecx, mss2

mov edx, ml2

int 80h

jmp exit

case2:

mov eax, sys\_out

mov ebx, stdout

mov ecx, mss3

mov edx, ml3

int 80h

jmp exit

case3:

mov eax, sys\_out

mov ebx, stdout

mov ecx, mss4

mov edx, ml4

int 80h

jmp exit

\_start:

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* INPUT

mov eax, sys\_out ;NUM1

mov ebx, stdout

mov ecx, mss1

mov edx, ml1

int 80h

mov eax, sys\_in

mov ebx, stdin

mov ecx, num1

mov edx, 9

int 80h

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

mov al,[num1]

sub al,'0'

mov bl,'5'

sub bl,'0'

cmp al,bl

jl case1

jg case2

je case3

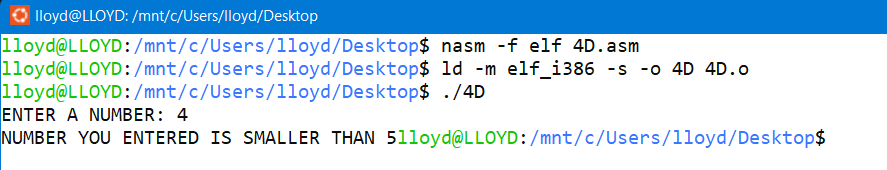
exit:

mov eax, 1

mov ebx, 0

int 80h

**OUTPUT**

****

**5)**

section .bss

num1 resb 9

temp resb 9

section .data

sys\_out equ 4

sys\_in equ 3

stdin equ 2

stdout equ 1

mss1 db 'ENTER A NUMBER: '

ml1 equ $-mss1

mss2 db 'NUMBER YOU ENTERED IS ODD'

ml2 equ $-mss2

mss3 db 'NUMBER YOU ENTERED IS EVEN'

ml3 equ $-mss3

section .text

global \_start:

odd:

mov eax, sys\_out

mov ebx, stdout

mov ecx, mss2

mov edx, ml2

int 80h

jmp exit

even:

mov eax, sys\_out

mov ebx, stdout

mov ecx, mss3

mov edx, ml3

int 80h

jmp exit

jmp exit

\_start:

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* INPUT

mov eax, sys\_out ;NUM1

mov ebx, stdout

mov ecx, mss1

mov edx, ml1

int 80h

mov eax, sys\_in

mov ebx, stdin

mov ecx, num1

mov edx, 9

int 80h

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

mov al,[num1]

sub al,'0'

mov bl,'2'

sub bl,'0'

div bl

add ah,'0'

cmp ah,'0'

je even

jmp odd ;else its odd

exit:

mov eax, 1

mov ebx, 0

int 80h

**OUTPUT**

