section .text

global \_start ;must be declared for using gcc

\_start : ;tell linker entry point

mov ecx, [num1] ;num1 value move to register ecx

cmp ecx, [num2] ;comparing the ecx and num2 values

jg check\_third\_num ;jump greater (jg) when ecx has higher than num2

mov ecx, [num2] ;when condition fails num2 is higher so mov num2 to register ecx

check\_third\_num: ;comparing the ecx and num3

cmp ecx, [num3] ;here comparing

jg \_exit ;when ecx has higher value goto \_exit otherwise next

mov ecx, [num3] ;when condition fails now move num3 to ecx sothat ecx has higher value

\_exit: ;ecx has higher value

mov [largest], ecx ;here move the ecx value to largest

;printing the largest value

mov ecx,msg

mov edx, len

mov ebx,1 ;file descriptor (stdout)

mov eax,4 ;system call number (sys\_write)

int 0x80 ;call kernel

mov ecx,largest

mov edx, 2

mov ebx,1 ;file descriptor (stdout)

mov eax,4 ;system call number (sys\_write)

int 0x80 ;call kernel

mov eax, 1

int 80h

section .data

msg db "The largest digit is: ", 0xA,0xD

len equ $- msg

num1 dd '31'

num2 dd '51'

num3 dd '41'

segment .bss

largest resb 2