a. Write an assembly language program to find the largest of two numbers

exit equ 1

read equ 3

write equ 4

stdin equ 0

stdout equ 1

section .data

prompt1 db "Enter two numbers : ",10

prompt1len equ $-prompt1

prompt2 db "Largest of the numbers is : ",8

prompt2len equ $-prompt2

newline db 10,0

section .bss

num1: resb 1

num2: resb 1

result: resb 1

section .text

global \_start

\_start:

mov eax, write

mov ebx, stdout

mov ecx, prompt1

mov edx, prompt1len

int 80h

mov eax, read

mov ebx, stdin

mov ecx, num1

mov edx, 2

int 80h

mov eax, read

mov ebx, stdin

mov ecx, num2

mov edx, 2

int 80h

mov al, [num1]

mov bl, [num2]

cmp al, bl

jg largest

mov [result], bl

jmp end

largest:

mov [result], al

end:

mov eax, write

mov ebx, stdout

mov ecx, prompt2

mov edx, prompt2len

int 80h

mov eax, write

mov ebx, stdout

mov ecx, result

mov edx, 1

int 80h

mov eax, write

mov ebx, stdout

mov ecx, newline

mov edx, 1

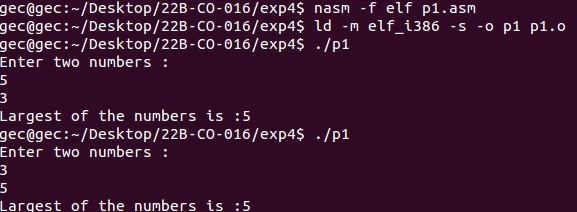
int 80h

mov eax, exit

mov ebx, 0

int 80h

**Outout:**



b. write an assembly language program to find the largest of three numbers

exit equ 1

read equ 3

write equ 4

stdin equ 0

stdout equ 1

section .data

prompt db "Enter 3 numbers: ", 10

prompt\_len equ $ - prompt

prompt2 db "Largest of 3 numbers entered : ", 0

prompt2\_len equ $ - prompt2

newline db 10,0

section .bss

num1: resb 1

num2: resb 1

num3: resb 1

result: resb 1

section .text

global \_start

\_start:

mov eax, write

mov ebx, stdout

mov ecx, prompt

mov edx, prompt\_len

int 80h

mov eax, read

mov ebx, stdin

mov ecx, num1

mov edx, 2

int 80h

mov eax, read

mov ebx, stdin

mov ecx, num2

mov edx, 2

int 80h

mov eax, read

mov ebx, stdin

mov ecx, num3

mov edx, 2

int 80h

mov al, [num1]

mov bl, [num2]

cmp al, bl

jg check\_num1\_num3

mov al, [num2]

mov bl, [num3]

cmp al, bl

jg check\_num2\_num3

mov [result], bl

jmp end

check\_num1\_num3:

mov al, [num1]

mov bl, [num3]

cmp al, bl

jg move\_num1\_result

mov [result], bl

jmp end

check\_num2\_num3:

mov [result], al

jmp end

move\_num1\_result:

mov [result], al

end:

mov eax, write

mov ebx, stdout

mov ecx, prompt2

mov edx, prompt2\_len

int 80h

mov eax, write

mov ebx, stdout

mov ecx, result

mov edx, 1

int 80h

mov eax, write

mov ebx, stdout

mov ecx, newline

mov edx, 1

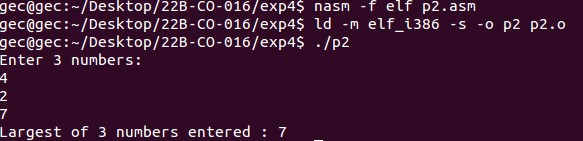
int 80h

mov eax, exit

mov ebx, 0

int 80h

**Output:**



c. Write an assembly language program to find the smallest of three numbers.

exit equ 1

read equ 3

write equ 4

stdin equ 0

stdout equ 1

section .data

prompt db "Enter 3 numbers : ", 10

prompt\_len equ $ - prompt

prompt2 db "smallest of 3 numbers entered : ", 0

prompt2\_len equ $ - prompt2

newline db 10,0

section .bss

num1: resb 1

num2: resb 1

num3: resb 1

result: resb 1

section .text

global \_start

\_start:

mov eax, write

mov ebx, stdout

mov ecx, prompt

mov edx, prompt\_len

int 80h

mov eax, read

mov ebx, stdin

mov ecx, num1

mov edx, 2

int 80h

mov eax, read

mov ebx, stdin

mov ecx, num2

mov edx, 2

int 80h

mov eax, read

mov ebx, stdin

mov ecx, num3

mov edx, 2

int 80h

mov al, [num1]

mov bl, [num2]

cmp al, bl

jl check\_num1\_num3

mov al, [num2]

mov bl, [num3]

cmp al, bl

jl check\_num2\_num3

mov [result], bl

jmp end

check\_num1\_num3:

mov al, [num1]

mov bl, [num3]

cmp al, bl

jl move\_num1\_result

mov [result], bl

jmp end

check\_num2\_num3:

mov al, [num2]

mov bl, [num3]

cmp al, bl

jl move\_num2\_result

mov [result], bl

jmp end

move\_num1\_result:

mov [result], al

jmp end

move\_num2\_result:

mov [result], al

jmp end

end:

mov eax, write

mov ebx, stdout

mov ecx, prompt2

mov edx, prompt2\_len

int 80h

mov eax, write

mov ebx, stdout

mov ecx, result

mov edx, 1

int 80h

mov eax, write

mov ebx, stdout

mov ecx, newline

mov edx, 1

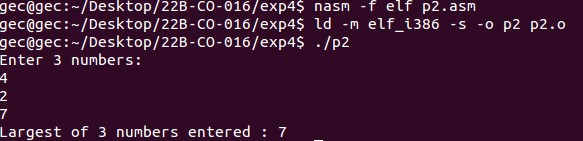
int 80h

mov eax, exit

mov ebx,0

int 80h

**Output:**



d. Write an assembly language program to check if the number is greater than 5 or lesser that 5.

exit equ 1

read equ 3

write equ 4

stdin equ 0

stdout equ 1

section .data

prompt1 db "Enter a numbers : ",10

prompt1len equ $-prompt1

prompt2 db "greater than 5 ",10

prompt2len equ $-prompt2

prompt3 db "less than 5 ",10

prompt3len equ $-prompt3

newline db 10,0

section .bss

num1: resb 1

section .text

global \_start

\_start:

mov eax, write

mov ebx, stdout

mov ecx, prompt1

mov edx, prompt1len

int 80h

mov eax, read

mov ebx, stdin

mov ecx, num1

mov edx, 2

int 80h

mov al, [num1]

mov bl, "5"

cmp al, bl

jg larger

mov eax, write

mov ebx, stdout

mov ecx, prompt3

mov edx, prompt3len

int 80h

jmp end

larger:

mov eax, write

mov ebx, stdout

mov ecx, prompt2

mov edx, prompt2len

int 80h

end:

mov eax, write

mov ebx, stdout

mov ecx, newline

mov edx, 1

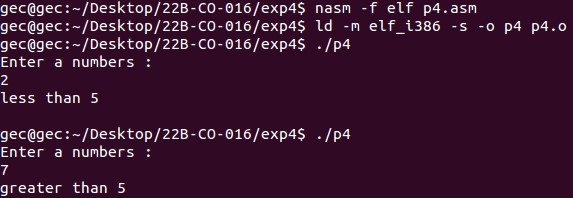
int 80h

mov eax, exit

mov ebx, 0

int 80h

**Output:**



e. Write an assembly language program to check whether the entered number us even or odd

vexit equ 1

read equ 3

write equ 4

stdin equ 0

stdout equ 1

section .data

prompt db "Enter a number: ", 0

promptlen equ $ - prompt

prompt2 db "Odd ", 10

prompt2len equ $ - prompt2

prompt3 db "Even ", 10

prompt3len equ $ - prompt3

section .bss

num: resb 1

section .text

global \_start

\_start:

mov eax, write

mov ebx, stdout

mov ecx, prompt

mov edx, promptlen

int 80h

mov eax, 3

mov ebx, 0

mov ecx, num

mov edx, 2

int 80h

mov al, [num]

and al, 1

jz even

mov eax, write

mov ebx, stdout

mov ecx, prompt2

mov edx, prompt2len

int 80h

jmp end

even:

mov eax, write

mov ebx, stdout

mov ecx, prompt3

mov edx, prompt3len

int 80h

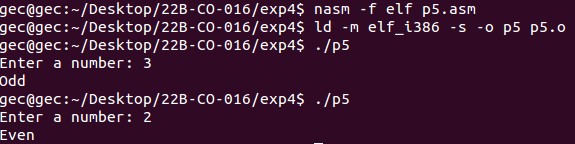
end:

mov eax, 1

xor ebx, ebx

int 0x80

**Output:**



f. Write an assembly language program that compares two strings and checks it they are same strings.

exit equ 1

read equ 3

write equ 4

stdin equ 0

stdout equ 1

section .data

prompt db "Enter 2 strings : ", 10

promptlen equ $ - prompt

prompt2 db "Equal strings", 10

prompt2len equ $ - prompt2

prompt3 db "Not Equal Strings ", 10

prompt3len equ $ - prompt3

section .bss

str1: resb 20

str2: resb 20

section .text

global \_start

\_start:

mov eax, write

mov ebx, stdout

mov ecx, prompt

mov edx, promptlen

int 80h

mov eax, read

mov ebx, stdin

mov ecx, str1

mov edx, 20

int 80h

mov eax, read

mov ebx, stdin

mov ecx, str2

mov edx, 20

int 80h

mov esi, str1

mov edi, str2

compare\_string:

mov al, [esi]

mov bl, [edi]

cmp al, bl

jne string\_not\_equal

cmp al, 0

je string\_equal

inc esi

inc edi

jmp compare\_string

string\_equal:

mov eax, write

mov ebx, stdout

mov ecx, prompt2

mov edx, prompt2len

int 80h

jmp end

string\_not\_equal:

mov eax, write

mov ebx, stdout

mov ecx, prompt3

mov edx, prompt3len

int 80h

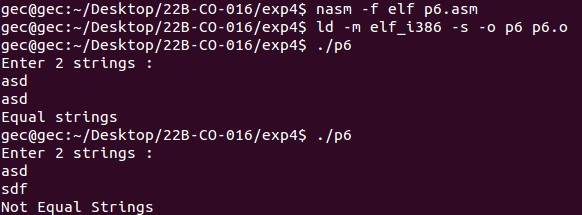
end:

mov eax, 1

xor ebx, ebx

int 0x80

**Output:**



g. Write an assembly language program to check if a number is a multiple of 3.

exit equ 1

read equ 3

write equ 4

stdin equ 0

stdout equ 1

section .data

prompt1 db "Enter a numbers : ",10

prompt1len equ $-prompt1

prompt2 db "Is multiple of 3 ",10

prompt2len equ $-prompt2

prompt3 db "Is Not multiple of 3 ",10

prompt3len equ $-prompt3

section .bss

num: resb 1

section .text

global \_start

\_start:

mov eax, write

mov ebx, stdout

mov ecx, prompt1

mov edx, prompt1len

int 80h

mov eax, read

mov ebx, stdin

mov ecx, num

mov edx, 2

int 80h

mov al, [num]

sub al, '0'

mov bl, '3'

sub bl, '0'

div bl

cmp ah, 0

je is\_multipal

mov eax, write

mov ebx, stdout

mov ecx, prompt3

mov edx, prompt3len

int 80h

jmp end

is\_multipal:

mov eax, write

mov ebx, stdout

mov ecx, prompt2

mov edx, prompt2len

int 80h

end:

mov eax, exit

mov ebx, 0

int 80h

**Output:**

