**Title: Loops and Macros in 80386**

**Theory:**

**Loop: General syntax**

**Macros: What are macros. advantages of macros**

**Experiment 3a: To write program in 80386 using loop**

**Finding the sum of numbers using loop**

section .text

global \_start ;must be declared for linker (ld)

\_start:

mov eax,3 ;number bytes to be summed

mov ebx,0 ;EBX will store the sum

mov ecx, x ;ECX will point to the current element to be summed

top: add ebx, [ecx]

add ecx,1 ;move pointer to next element

dec eax =2 ;decrement counter

jnz top ;if counter not 0, then loop again

done:

add ebx, '0'

mov [sum], ebx ;done, store result in "sum"

display:

mov edx,1 ;message length

mov ecx, sum ;message to write

mov ebx, 1 ;file descriptor (stdout)

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

int 0x80 ;call kernel

mov eax, 1 ;system call number (sys\_exit)

int 0x80 ;call kernel

section .data

global x

x:

db 2

db 4

db 3

sum:

db 0

**Experiment 3b: Write the program to demonstrate the use of macros**

**%macro print 2**

**mov edx,%2**

**mov ecx,%1**

**mov ebx,1**

**mov eax,4**

**int 0x80**

**%endmacro**

**segment .text**

**global \_start**

**\_start:**

**print msg,len**

**mov eax,1**

**int 0x80**

**segment .data**

**msg db 'Hello,World!'**

**len equ $ -msg**

**Experiment 3c:Write the 80386 program**  **to find the largest of the given three numbers (use macro)**

Write a program which illustrates the programming constructs of higher level language in 80386 assembly coding (to find the largest of the given three numbers)

section .text

global \_start ;must be declared for using gcc

\_start: ;tell linker entry point

mov ecx, [num1]

cmp ecx, [num2]

jg check\_third\_num

mov ecx, [num2]

check\_third\_num:

cmp ecx, [num3]

jg \_exit

mov ecx, [num3]

\_exit:

mov [largest], ecx

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 '47'

num2 dd '22'

num3 dd '31'

segment .bss

largest resb 2

output :

The largest digit is: