32 Bit Code

%macro scall 4 ;macro declaration with 4 parameters

mov eax,%1 ;1st parameter has been moved to eax

mov ebx,%2 ;2nd parameter has been moved to ebx

mov ecx,%3 ;3rd parameter has been moved to ecx

mov edx,%4 ;4th parameter has been moved to edx

int 80h ;Call the Kernel

%endmacro ;end of macro

section .data ;.data begins here

msg1 db 10d,13d,"Enter 10 Hexadecimal Numbers: " ;variable initialised with string

l1 equ $-msg1 ;l1 stores length of string msg1

msg2 db 10d,13d,"Entered Numbers are: " ;variable initialised with string

l2 equ $-msg2 ;l2 stores length of string msg2

section .bss ;.bss begins here

num resd 10 ;array with 10 elements of double word type

section .text ;.text begins here

global \_start ;moving to \_start label

\_start: ;\_start label

mov esi,num ;esi points to base address of num

mov edi,10 ;stores the count of numbers

scall 4,1,msg1,l1 ;macro call to display msg1

up1: scall 3,0,esi,9 ;macro call to take input at esi

add esi,9 ;to make esi point to next address

dec edi ;decrement edi

jnz up1 ;jump to up1 if not zero

mov esi,num ;esi points to base address of num

mov edi,10 ;stores the count of numbers

scall 4,1,msg2,l2 ;macro call to display msg2

up2: scall 4,1,esi,9 ;macro call to display esi address contents

add esi,9 ;to make esi point to next address

dec edi ;decrement edi

jnz up2 ;jump to up2 if not zero

mov eax,1 ;sys\_exit

mov ebx,0 ;Sucessful Termination

int 80h ;Call the Kernel

64 Bit Code

%macro scall 4 ;macro declaration with 4 parameters

mov rax,%1 ;1st parameter has been moved to rax

mov rdi,%2 ;2nd parameter has been moved to rdi

mov rsi,%3 ;3rd parameter has been moved to rsi

mov rdx,%4 ;4th parameter has been moved to rdx

syscall ;Call the Kernal

%endmacro ;end of macro

section .data ;.data begins here

msg1 db "Enter the ten 64 bit numbers:" ,10d,13d ;msg1 variable initialised with string

l1 equ $-msg1 ;l1 stores length of string msg1

msg2 db "The ten 64 bit numbers are:" ,10d,13d ;msg2 variable initialised with string

l2 equ $-msg2 ;l2 stores length of string msg2

section .bss ;.bss begins here

num resq 10 ;array of 10 elements of type quad word

cnt resb 1 ;variable of type byte

section .text ;.text begins here

global \_start ;moving to \_start label

\_start: ;\_start label

mov rbx,num ;rbx points to base address of num

mov byte[cnt],10 ;type casting and storing 10 in cnt

scall 1,1,msg1,l1 ;macro call to display msg1

up1: scall 0,0,rbx,48 ;macro call to input at rbx

add rbx,48 ;make rbx point at next address

dec byte[cnt] ;decrement cnt

jnz up1 ;jump to up1 if not zero

mov rbx,num ;rbx points to base address of num

mov byte[cnt],10 ;type casting and storing 10 in cnt

scall 1,1,msg2,l2 ;macro call to display msg2

up2: scall 1,1,rbx,48 ;macro call to display rbx address contents

add rbx,48 ;make rbx point at next address

dec byte[cnt] ;decrement cnt

jnz up2 ;jump to up2 if not zero

mov rax,60 ;sys\_exit function

mov rbx,0 ;Sucessful Termination

syscall ;Call the Kernel

Problem Statement:Write X86 ALP to find, a) Number of Blank spaces b) Number of lines c) Occurrence of a particular character. Accept the data from the text file. The text file has to be accessed during Program\_1 execution and write FAR PROCEDURES in Program\_2 for the rest of the processing. Use of PUBLIC and EXTERN directives is mandatory.

A5\_file1.asm

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

extern far\_proc ; [ FAR PROCRDURE

; USING EXTERN DIRECTIVE ]

global filehandle, char, buf, abuf\_len

%include "macro.asm"

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

section .data

nline db 10

nline\_len equ $-nline

ano db 10,10,10,10,"ML assignment 05 :- String Operation using Far Procedure"

db 10,"---------------------------------------------------",10

ano\_len equ $-ano

filemsg db 10,"Enter filename for string operation : "

filemsg\_len equ $-filemsg

charmsg db 10,"Enter character to search : "

charmsg\_len equ $-charmsg

errmsg db 10,"ERROR in opening File...",10

errmsg\_len equ $-errmsg

exitmsg db 10,10,"Exit from program...",10,10

exitmsg\_len equ $-exitmsg

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

section .bss

buf resb 4096

buf\_len equ $-buf ; buffer initial length

filename resb 50

char resb 2

filehandle resq 1

abuf\_len resq 1 ; actual buffer length

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

section .text

global \_start

\_start:

display ano,ano\_len ;assignment no.

display filemsg,filemsg\_len

accept filename,50

dec rax

mov byte[filename + rax],0 ; blank char/null char

display charmsg,charmsg\_len

accept char,2

fopen filename ; on succes returns handle

cmp rax,-1H ; on failure returns -1

jle Error

mov [filehandle],rax

fread [filehandle],buf, buf\_len

mov [abuf\_len],rax

call far\_proc

jmp Exit

Error: display errmsg, errmsg\_len

Exit: display exitmsg,exitmsg\_len

display nline,nline\_len

mov rax,60

mov rdi,0

syscall

A5\_file2.asm

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

global far\_proc

extern filehandle, char, buf, abuf\_len

%include "macro.asm"

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

section .data

nline db 10,10

nline\_len: equ $-nline

smsg db 10,"No. of spaces are : "

smsg\_len: equ $-smsg

nmsg db 10,"No. of lines are : "

nmsg\_len: equ $-nmsg

cmsg db 10,"No. of character occurances are : "

cmsg\_len: equ $-cmsg

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

section .bss

scount resq 1

ncount resq 1

ccount resq 1

dispbuff resb 4

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

section .text

; global \_main

;\_main:

far\_proc: ;FAR Procedure

mov rax,0

mov rbx,0

mov rcx,0

mov rsi,0

mov bl,[char]

mov rsi,buf

mov rcx,[abuf\_len]

again: mov al,[rsi]

case\_s: cmp al,20h ;space : 32 (20H)

jne case\_n

inc qword[scount]

jmp next

case\_n: cmp al,0Ah ;newline : 10(0AH)

jne case\_c

inc qword[ncount]

jmp next

case\_c: cmp al,bl ;character

jne next

inc qword[ccount]

next: inc rsi

dec rcx ;

jnz again ;loop again

display smsg,smsg\_len

mov rbx,[scount]

call display16\_proc

display nmsg,nmsg\_len

mov rbx,[ncount]

call display16\_proc

display cmsg,cmsg\_len

mov rbx,[ccount]

call display16\_proc

fclose [filehandle]

ret

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

display16\_proc:

mov rdi,dispbuff ;point esi to buffer

mov rcx,4 ;load number of digits to display

dispup1:

rol bx,4 ;rotate number left by four bits

mov dl,bl ;move lower byte in dl

and dl,0fh ;mask upper digit of byte in dl

add dl,30h ;add 30h to calculate ASCII code

cmp dl,39h ;compare with 39h

jbe dispskip1 ;if less than 39h akip adding 07 more

add dl,07h ;else add 07

dispskip1:

mov [rdi],dl ;store ASCII code in buffer

inc rdi ;point to next byte

loop dispup1 ;decrement the count of digits to display

;if not zero jump to repeat

display dispbuff,4 ;

ret

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

macro.asm

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

;macro.asm

;macros as per 64 bit conventions

%macro accept 2

mov rax,0 ;read

mov rdi,0 ;stdin/keyboard

mov rsi,%1 ;buf

mov rdx,%2 ;buf\_len

syscall

%endmacro

%macro display 2

mov rax,1 ;print

mov rdi,1 ;stdout/screen

mov rsi,%1 ;msg

mov rdx,%2 ;msg\_len

syscall

%endmacro

%macro fopen 1

mov rax,2 ;open

mov rdi,%1 ;filename

mov rsi,2 ;mode RW

mov rdx,0777o ;File permissions

syscall

%endmacro

%macro fread 3

mov rax,0 ;read

mov rdi,%1 ;filehandle

mov rsi,%2 ;buf

mov rdx,%3 ;buf\_len

syscall

%endmacro

%macro fwrite 3

mov rax,1 ;write/print

mov rdi,%1 ;filehandle

mov rsi,%2 ;buf

mov rdx,%3 ;buf\_len

syscall

%endmacro

%macro fclose 1

mov rax,3 ;close

mov rdi,%1 ;file handle

syscall

%endmacro

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

myfile.txt

"Welcome!!!"

Computer Engineering

Sinhgad Institute of Technology & Science Narhe,Pune

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Output\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;[root@localhost A5\_Far]# nasm -f elf64 A5\_file1.asm

;[root@localhost A5\_Far]# nasm -f elf64 A5\_file2.asm

;[root@localhost A5\_Far]# ld -o A5\_file1 A5\_file1.o A5\_file2.o

;[root@localhost A5\_Far]# ./A5\_file1

;ML assignment 05 :- String Operation using Far Procedure

;---------------------------------------------------

;Enter filename for string operation : myfile.txt

;Enter character to search : e

;No. of spaces are : 0007

;No. of lines are : 0003

;No. of character occurances are : 000B

;Exit from program...

;[root@localhost A5\_Far]#

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

Note:All files are save within single folder.