GROUP A= Assignment No.3

section .data

menumsg db 10,"\*\*\*\*\*\*\*\*\*\*\*\*MENU\*\*\*\*\*\*\*\*\*\*\*\*\*"

db 10,"1.HEX to BCD"

db 10,"2.BCD to HEX"

db 10,"3.EXIT"

db 10,"Enter U r Choice:"

menul equ $-menumsg

h2b db 10,"HEX to BCD"

db 10,"Enter 4 digit Hex Number:"

h2bl equ $-h2b

b2h db 10,"BCD to HEX"

db 10,"Enter 5 digit BCD Number:"

b2hl equ $-b2h

emsg db 10,"U Entered Invalid Data..."

emsgl equ $-emsg

bmsg db 10,13,"Equivalent BCD number is:"

bmsgl equ $-bmsg

b1msg db 10,13,"Equivalent HEX number is:"

b1msgl equ $-b1msg

dmsg db 10,"Do u Want to cnt...."

dmsgl equ $-dmsg

section .bss

choice resb 2

buf resb 6

bufl equ $-buf

digitcount resb 1

ans resw 1

char\_ans resb 4

fact resw 1

%macro scall 4

mov rax,%1

mov rdi,%2

mov rsi,%3

mov rdx,%4

syscall

%endmacro

%macro exit 0

mov rax,60

xor rdi,rdi

syscall

%endm

section .text

global \_start

\_start:

menu:

scall 01,01,menumsg,menul

scall 0,0,choice,2

mov al,[choice]

cmp al,'1'

jne case2

call h2bproc

jmp cnt

case2:

cmp al,'2'

jne case3

call b2cproc

jmp cnt

cnt:

scall 01,01,dmsg,dmsgl

scall 0,0,choice,2

mov al,[choice]

cmp al,'y'

jz menu

case3:

cmp al,'3'

jne err

exit

err:

scall 01,01,emsg,emsgl

jmp menu

exit

h2bproc:

scall 01,01,h2b,h2bl

call accept\_16

mov ax,bx

mov rbx,0A

back:

xor rdx,rdx

div rbx

push dx

inc byte[digitcount]

cmp rax,0h

jne back

scall 01,01,bmsg,bmsgl

print\_bcd:

pop dx

add dl,30h

mov [char\_ans],dl

scall 01,01,char\_ans,1

dec byte[digitcount]

jnz print\_bcd

ret

accept\_16:

scall 0,0,buf,5

xor bx,bx

mov rcx,4

mov rsi,buf

next\_digit:

shl bx,04

mov al,[rsi]

cmp al,39h

jbe l1

sub al,07h

l1: sub al,30h

add bx,ax

inc rsi

loop next\_digit

ret

b2cproc:

scall 01,01,b2h,b2hl

scall 0,0,buf,5

mov rsi,buf+4

mov word[fact],1

mov rcx,5

xor rbx,rbx

back1:

xor rax,rax

mov al,[rsi]

sub al,30h

mul word[fact]

add bx,ax

mov ax,10

mul word[fact]

mov word[fact],ax

dec rsi

loop back1

call display\_16

ret

display\_16:

scall 01,01, b1msg,b1msgl

mov rcx,4

mov rsi,char\_ans

back2:

rol bx,04

mov dl,bl

and dl,0fh

cmp dl,09h

jbe l2

add dl,07

l2:add dl,30h

mov [rsi],dl

inc rsi

loop back2

scall 01,01,char\_ans,4

ret