**NAME :: IBRAR BABAR**

**Roll no. 19P-0104**

**BS(CS) :: Section B**

**TASK#1**

[org 0x100]

jmp start

num1: db 11100111B

start:

mov ax,0

mov bx,0

mov cx,0

mov dx,0

mov al,[num1]

l1:

shr al,1

add cx,1

cmp ax,0

jne l1

xor ax,ax

mov al,[num1]

l2:

shl al,1

add dx,1

cmp ax,0

jne l2

mov ax,0

cmp cx,dx

jne skip

mov ax,1

skip:

mov ax,0x4c00

int 0x21

TASK#2

[org 0x100]

jmp start

num: dw 4717,7210,3100

first: dw 0

second: dw 0

third: dw 0

greatest: dw 0

greatest\_num: dw 0

start:

xor ax,ax

xor bx,bx

xor cx,cx

xor dx,dx

mov ax ,[num]

mov bl,10

li:

mov dx, 0

div bx

add cx,dx

cmp ax,0

jne li

mov [first],cx

xor ax,ax

xor bx,bx

xor cx,cx

xor dx,dx

mov ax ,[num+2]

mov bl,10

l2:

mov dx, 0

div bx

add cx,dx

cmp ax,0

jne l2

mov [second],cx

xor ax,ax

xor bx,bx

xor cx,cx

xor dx,dx

mov ax ,[num+4]

mov bl,10

l3:

mov dx, 0

div bx

add cx,dx

cmp ax,0

jne l3

mov [third],cx

xor ax,ax

xor bx,bx

xor cx,cx

xor dx,dx

mov ax,[first]

mov bx,[second]

mov cx,ax ;if ax is greater

cmp ax,bx

ja check\_next

mov cx,bx ;else bx is greater

check\_next:

mov dx,[third]

mov [greatest],cx ;if cx is greatest

cmp cx,dx

ja skip

mov [greatest],dx ;else dx is greatest

skip:

xor cx,cx

mov cx,[greatest]

mov ax,[num]

mov [greatest\_num],ax

mov ax,0x4c00

int 0x21

============================================

TASK#3

[org 0x100]

jmp start

num: dw 14

start:

xor ax,ax

xor bx,bx

xor cx,cx

xor dx,dx

mov ax,[num]

mov bl, 2

l2:

mov dx, 0

div bx

add cx,1

cmp ah,0

je odd

cmp ax,0

jne l2

odd:

dec ax

add cx,1

cmp ax,0

jne l2

mov ax,0x4c00

int 0x21

TASK#4

[org 0x100]

jmp start

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

sum: dw 0

multi: dw 0

divee: dw 0

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

division:

push bp

mov bp,sp

div bx

mov [divee],ax

pop bp

ret 4

multiplication:

push bp

mov bp,sp

push cx

mov cx,8

checkbit:

shr bx,1

jnc skip

add [multi],ax

skip:

shl ax,1

dec cx

jne checkbit

xor ax,ax

xor bx,bx

xor bx,bx

mov ax,[multi]

mov bx,[bp+4]

call division

pop cx

pop bp

ret 6

add:

push bp

mov bp,sp

push dx

mov dx,0

add ax,bx

mov [sum],ax

mov ax,[sum]

mov bx,[bp+6]

mov dx,[bp+4]

call multilication

start:

mov ax,2

mov bx,3

call add

mov ax,0x4c00

int 0x21