21-14632-1
Tanikul Islam Nishat
Date 22/01/24
sec: D

1. Write the signed and unsigned conditional statement

Ans:

Signed conditional Jumps:

JG or JNLE : Jump if greater
JGE or JNL : Jump if greater or equal
JL or JNGE : Jump if less
JLE or JNG : Jump if less or equal
JE or JZ : Jump if equal
JNE or JNZ : Jump if not equal

Unsigned conditional Jumps:

JA or JNBE : Jump if above
JAE or JNB : Jump if above equal
JB or JNAE : Jump if below
JBE or JNA : Jump if below or equal
JE or JZ : Jump if equal
JNE or JNZ : Jump if not equal

2. Find the max value among 3 numbers.

```
.model small
.stack 100h
.data
msg db 13,10 "Enter three numbers:$"
max msg db 13,10 "Maximum number is:$"
num1 db ?
num2 db ?
num3 db ?
max_num db ?
newline db 13,10, '$'
.code
main proc
mov ax, @data
mov ds, ax
mov ah,9
lea dx, msg
int 21h

mov ah, 1
int 21h
sub al, '0'
mov num1, al

int 21h
sub al, '0'
mov num2, al

int 21h
sub al, '0'
mov num3, al

mov al, num1
cmp al, num2
jge check_num3
mov al, num2

check_num3:
cmp al, num3
jge max found
mov al, num3
max found:
mov max_num, al

mov ah, 9
lea dx, max_msg
int 21h

mov dl, max_num
add dl, '0'
mov ah, 2
int 21h

mov dx, offset newline
mov ah, 9
int 21h

mov ah, 4ch
int 21h

main endp
end main.
```

3. Print 1,2,3,4,5 using while loop and Assembly language.

```
.model small
.stack 100h
.data
msg db '0', '$'
.code
main proc
mov ax, @data
mov ds, ax
mov cl, 1
print_loop:
cmp cl, 6
jge end_loop
mov al, cl
add al, '0'
mov msg, al
mov ah, 09h
lea dx, msg
int 21h
inc cl
jmp print_loop
end_loop:
mov ax, 4Ch
int 21h
main endp
end main
```

Output:

1 2 3 4 5 6

4. Find the remainder and quotient after divided your ids last 2 digit by 3.

Ans:

```
.model small
.stack 100h
.data
id db 32
divisor db 3
remainder db ?
quotient db ?
.code
main proc
mov ax, @data
mov ds, ax

mov al, id
and al, 0FH
mov bl, al

mov al, bl
div divisor

mov quotient, al
mov remainder, ah

mov ah, 02h
add quotient, '0'
mov dl, quotient
int 21h

mov ah, 02h
add remainder, '0'
mov dl, remainder
int 21h

mov ah, 4Ch
int 21h
main endp
end main
```

Output: 2