MP Sessional
LAB 3

1. **Basic Loop Syntax –**

```
MOV CX, count       ; Set loop counter
label:              ; Loop start label
  ; Your code here
LOOP label          ; Decrease CX and loop if CX ≠ 0
```

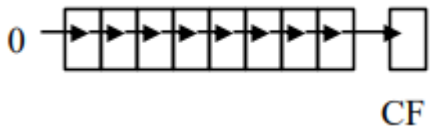
2. **Print from 0 to 9 using loop –**

3. **Logical Instructions –**

Instruction	Meaning	Example
`AND`	Bitwise AND	`AND AX, BX`
`OR`	Bitwise OR	`OR AL, BL`
`XOR`	Bitwise Exclusive OR	`XOR CX, DX`
`NOT`	Bitwise Invert (1's complement)	`NOT AX`

4. **Shift Instructions –**

Logical Shift :
A logical shift fills the newly created bit position with zero. If we do a single logical right shift on 11001111, it becomes 011001111.



Arithmetic Shift :
An arithmetic shift is filled with a copy of the original number's sign bit. If we do a single arithmetic right shift on 11001111, it becomes 11100111.

MP Sessional
LAB 3

i) SHL/SHR :

The instruction format is:
SHL destination, bits_shifted(immediate value – (1-15) for 16bit register)

Instruction formats include:
SHL reg, imm8
SHL reg, CL

Ex-
SHL/SHR AX, 1 ; Shift AX left by 1 bit
SHL/SHR BX, 4 ; Shift BX left by 4 bits
MOV CL, 3
SHL/SHR DX, CL ; Shift DX left by value in CL (3 bits)

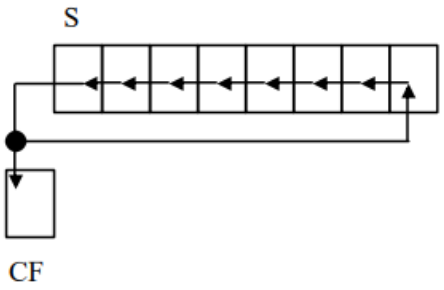
5. Rotate Instruction:

i) ROL

- The **ROL** instruction shifts each bit to the left, with the highest bit copied in the Carry flag and ***into the lowest bit***.
- The instruction format is:

  ROL *destination*, *bits_shifted*
- Instruction formats include:

  ROL *reg*, *imm8*
  ROL *mem*, *imm8*
  ROL *reg*, *CL*
  ROL *mem*, *CL*



Ex-
The following instruction sequence shifts the AL three times (once each) to the left, with the highest bit copied into the Carry flag and into the lowest bit:

```
mov al, 40h ;   AL = 01000000b
rol al, 1 ;     AL = 10000000b, CF = 0
rol al, 1 ;     AL = 00000001b, CF = 1
rol al, 1 ;     AL = 00000010b, CF = 0
```

MP Sessional
LAB 3

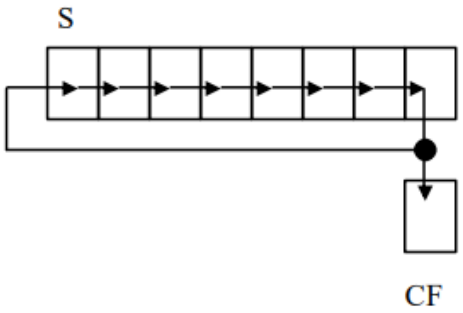
ii) ROR

- The **ROR** instruction shifts each bit to the right, with the lowest bit copied in the Carry flag and ***into the highest bit***.
- The instruction format is:

ROR *destination*, *bits_shifted*

- Instruction formats include:

ROR	*reg*, *imm8*
ROR	*mem*, *imm8*
ROR	*reg*, *CL*
ROR	*mem*, *CL*



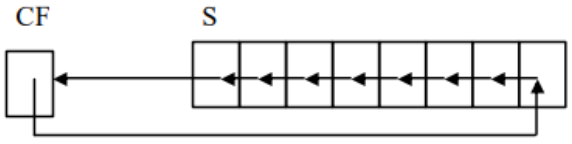
The following instruction sequence shifts the AL three times (once each) to the right, with the lowest bit copied into the Carry flag and into the highest bit:

```
mov al, 01h ;   AL = 00000001b
ror al, 1 ;     AL = 10000000b, CF = 1
ror al, 1 ;     AL = 01000000b, CF = 0
ror al, 1 ;     AL = 00100000b, CF = 0
```

iii) RCL :

- The **RCL** (**R**otate and **C**arry **L**eft) instruction shifts each bit to the left, copies the Carry flag to the least significant bit and copies the most significant bit into the Carry flag.
- In this examples, the lowest bit is copied into the Carry flag and into the highest bit of the result:

```
clc                     ; CF = 0
mov     bl, 88h         ; CF = 0 BL = 10001000b
rcl     bl, 1           ; CF = 1 AL = 00010000b
rcl     bl, 1           ; CF = 0 AL = 00100001b
```

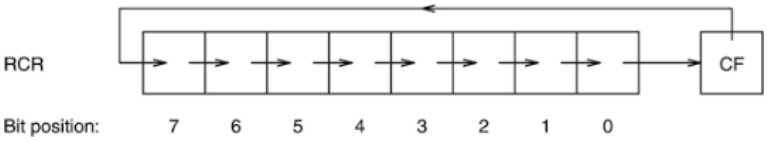


MP Sessional
LAB 3

iv) RCR :

- The **RCR** (**R**otate and **C**arry **R**ight) instruction shifts each bit to the right, copies the Carry flag to the most significant bit and copies the least significant bit into the Carry flag.
- In this examples, the lowest bit is copied into the Carry flag and into the highest bit of the result:

```
stc                     ; CF = 1
mov     ah, 10h         ; CF = 1 AH = 00010000b
rcr     ah, 1           ; CF = 0 AL = 00001000b
rcr     ah, 1           ; CF = 0 AL = 00000100b
```



MP Sessional
LAB 3

LAB TASKS

1. Find factorial of a number n by taking input using loop.
2. Write a program to perform AND, OR, XOR between two 8-bit numbers and show the result as output.
3. Multiply and divide a number using Shift operations.
4. Check if a number is **even** or **odd** using SHR.

    Check the **Carry Flag (CF)**:

    If CF = 1, number was **odd**.

    If CF = 0, number was **even**.