1. Write a LEX program to recognize tokens from the given expression.
2. Write a YACC program to check validity of expression and evaluate the expression.
3. Design suitable Data structures and implement Pass-I of two pass assembler to generate intermediate code. Sample input is following assembly language program.

```
        START 200
              MOVER AREG, =’5’
              MOVEM AREG, X
        L1    MOVER BREG, =’2’
              ORIGIN L1+3
              LTORG
NEXT          ADD AREG, =’1’
              SUB BREG, =’2’
              BC LT, BACK
              LTORG
BACK          EQU L1
              ORIGIN NEXT+5
              MULT CREG, =’4’
              STOP
              X DS 1
              END
```

4. Design Pass II of a two pass assembler for following intermediate code and data structures to generate machine code.

Symbol table

Symbol	address
X	214
L1	202
NEXT	207
BACK	202

LITERAL TABLE

LITERAL	ADRESS
='5'	205
='2'	206
='1'	210
='2'	211
='4'	215

Intermediate code

(AD,01) (C,200)
(IS,04) 1 (L,1)
(IS,05) 1 (S,1)
(IS,04) 2(L,2)
(AD,03) (S,2)+3
(AD,05)

(L,1)

(L,2)
(IS,01) 1 (L,3)
(IS, 02) 2 (L,4)
(IS,07) 1(S,4)
(AD,05)
(L,3)
(L,4)
(AD,04) (S,2)
(IS,03) 3 (L,5)
(IS,00)
(DL,02) (C,1)
(AD,02)

5. Design suitable data structures and implement Pass-I of a two-pass macro processor. Display contents of MNT, MDT, and ALA.

```
START
MACRO
INCR &ARG1, &ARG2
ADD AREG, &AREG2
MEND
MACRO
DECR &ARG3, &ARG4
SUB AREG, &AREG3
MOVER CREG, &ARG4
MEND
INCR N1, N2
DECR N3,N4
END
```

6. Design Pass-II of a two-pass macroprocessor. The MNT, MDT and intermediate code are inputs for Pass II microprocessor.

Macro Definition table(MDT)

INCR &ARG1, &ARG2

MOVER AREG, ARG1

ADD AREG, ARG2

MEND

DECR &ARG3, &ARG4

MOVER AREG, ARG3

SUB AREG, ARG4

MEND

Macro Name Table

INCR 0

DECR 4

Argument List Array

MACRO NAME: INCR

&ARG1

&ARG2

MACRO NAME: DECR

&ARG3

&ARG4

OUTPUT

START

MOVER AREG, N1

ADD AREG, N2

MOVER AREG, N3

SUB AREG, N4

END

7. Write a program to solve Classical Problems of Synchronization using Mutex.

8. Write a program to solve Classical Problems of Synchronization using Semaphore.

9. Write a program to simulate CPU Scheduling Algorithm: FCFS
   Sample Input:

Process ID	Arrival Time	Service Time
P1	0	3
P2	2	6
P3	4	4
P4	6	5
P5	8	2

10. Write a program to simulate CPU Scheduling Algorithm: SJF (Preemptive)
    Sample Input:

Process	Arrival time	Burst Time
P1	0	6

P2	1	4
P3	4	8
P4	3	3

11. Write a program to simulate CPU Scheduling Algorithm: Priority (Non-Preemptive)

    Sample Input:

    (Consider Low no. as high priority)

Processes	Arrival time	Burst Time	Priority
P1	0	8	1
P2	0	6	2
P3	2	1	3
P4	3	2	0

12. Write a program to simulate CPU Scheduling Algorithm: Round Robin (Preemptive).

    Sample Input:

    Time Quantum = 2

Process	Arrival time	Burst Time
P1	0	6
P2	1	4
P3	4	8
P4	3	3

13. Write a program to simulate Memory placement strategy – best fit,

    Sample Input

    memory partitions of 100K, 500K, 200K, 300K and 600K (in order)

Place processes of size 212K, 417K, 112K, 426K (in order)

14. Write a program to simulate Memory placement strategy – first fit.
    Sample Input
    memory partitions of 100K, 500K, 200K, 300K and 600K (in order)
    Place processes of size 212K, 417K, 112K, 426K (in order)

15. Write a program to simulate Memory placement strategy – next fit
    Sample Input
    memory partitions of 100K, 500K, 200K, 300K and 600K (in order)
    Place processes of size 212K, 417K, 112K, 426K (in order)

16. Write a program to simulate Memory placement strategy – worst fit.
    Sample Input
    memory partitions of 100K, 500K, 200K, 300K and 600K (in order)
    Place processes of size 212K, 417K, 112K, 426K (in order)

17. Write a program to simulate Page replacement algorithm. FIFO

    Sample Input:
    2, 3, 2, 1, 5, 2, 4, 5, 3, 2, 5, 2 and no. of frames is 3

18. Write a program to simulate Page replacement algorithm. Optimal
    Sample Input:

    2, 3, 2, 1, 5, 2, 4, 5, 3, 2, 5, 2 and no. of frames is 3

19. Write a program to simulate Page replacement algorithm. LRU

Sample Input:

2, 3, 2, 1, 5, 2, 4, 5, 3, 2, 5, 2 and no. of frames is 3