TUTORIAL - 5

A.1
```
Queue {
    front
    rear
    jobs[10][100]
    }
```

- main()
1. Initialise a Queue ~~as Q~~ pointer variable as Q.
2. Set Q→front = -1 and Q→rear = -1
3. Repeat step 3.1 to 3.6 while x = 1
- 3.1 Display " 1. Add job
  2. Complete job
  3. Display jobs
  0. Exit
- 3.2. Input x
- 3.3. If x is equal to 1
- 3.3.1 enqueue (Q)
- 3.4. If x is equal to 2
- 3.4.1 dequeue (Q)
- 3.5 If x is equal to 3
- 3.5.1 display (Q)
- 3.6 Else put x = 0 and break

- enqueue (Queue Q)
1. If rear is equal to 9
   Display "Overflow Error" and return
2. If front is equal to -1
   front = 0
   rear = 0
3. Display "Enter value : Q→job
4. Input val
5.

- enqueue(Queue Q)

1. If Q→rear is 9
1.1 Display "Overflow Error" and return
2. Display "Enter value:"
3. Input 'val'
4. If Q→front is -1
4.1 Q→front = 0
5. Q→jobs[rear+1] = val
6. Q→rear = Q→rear + 1

- dequeue(Queue Q)

1. If Q→front is -1
1.1 Display "Underflow Error" and return
2. Repeat step 2.1 for p = Q→front till p < Q→rear
2.1 Q→jobs[p] = Q→jobs[p+1]
3. Q→rear = Q→rear - 1
4. If Q→front > Q→rear
4.1 Q→front = -1
4.2 Q→rear = -1
~~4. Display "Empty"~~
5. Return

- display(Queue Q)

1. If Q→front is -1
1.1 Display "Empty Queue" and return
2. Repeat step 2.1 for p = front till p <= rear, step +1
2.1 Display Q→jobs[p]
3. Return

A.2

```
Queue {
    front
    rear
    arr[10]
    }
```

— main()

1. Initialise a Queue pointer variable 'Q'
2. Q → front = -1
3. Q → rear = -1
4. Display "Enter the no. of elements to insert - "
5. Input 'num'
6. Repeat step for i = 1 till i <= num, step +1
   Display "Enter value - "
   Input val
   enqueue(Q, val)
7. display(Q)
8. Repeat step while x = 1
   8.1 Input 'action'
   8.2 If action is 'POP'
   8.2.1 ~~dequeue~~ delete(Q) and display(Q)
   8.3 If action is 'PUSH'
   8.3.1 Input val
   8.3.2 enqueue(Q, val) and display(Q)
   8.4 If action is 'EXIT'
   8.4.1 x = 0
   8.4.2 break

→ P.T.O

- enqueue (Queue Q, Integer val)

1. If Q→rear is greater than equal to 9
   Display "Overflow Error"
   Return
2. Q→arr[Q→rear + 1] = val
3. Q→rear = Q→rear + 1
4. If Q→front is -1
   Q→front = Q→front + 1
5. Return

- dequeue (Queue Q)

1. If Q→front is -1
   Display "Underflow Error"
   Return 0
2. Repeat step for p = Q→front till p < Q→rear, step
   Q→arr[p] = Q→arr[p + 1]
3. Q→rear = Q→rear - 1
4. If Q→rear < Q→front
   Q→front = -1
   Q→rear = -1
5. Return 1

- delete (Queue Q)

1. Repeat step for p = Q→front till p < Q→rear, step +1
   val = Q→arr[front]
   x = dequeue (Q)
   enqueue (Q, val)
   enqueue (Q, val)
2. dequeue (Q)

- display (Queue Q)

1. Repeat step for p = Q→front till p<= Q→rear , step H
1.1 Display Q→arr[p]
2. If Q→front = -1
2.1 Display "Empty"
3. Return