13.1

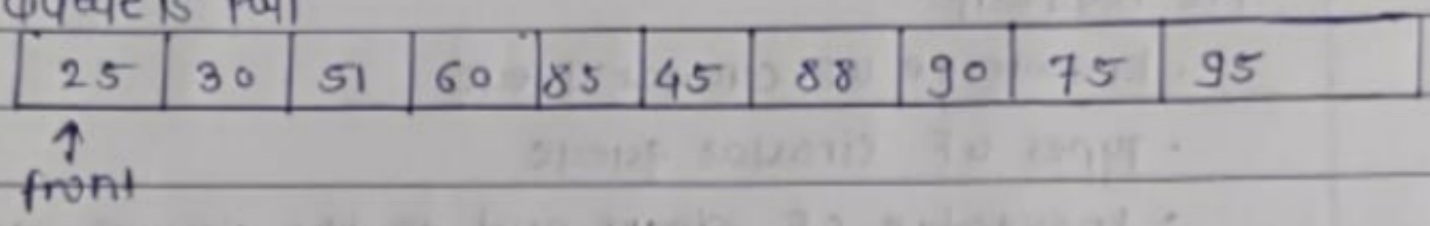
Dr. D. Y. PATIL
EDUCATIONAL FEDERATION

Theory:

Why circular Queue?

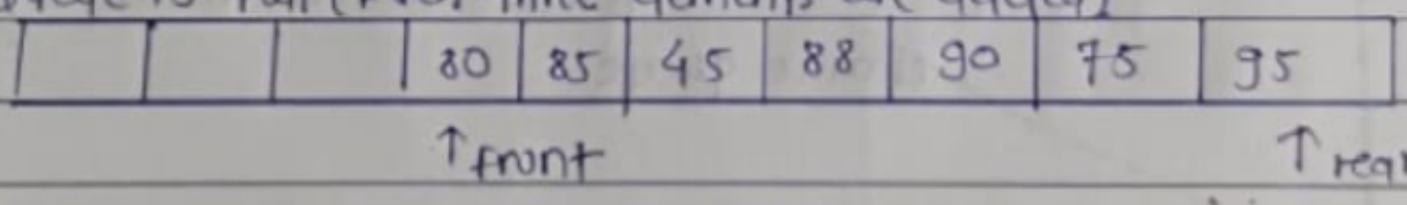
In a normal queue data structure, we can insert elements until queue become full but once if queue becomes full, we can't insert the next elements will all the elements are deleted from the queue. For example consider the queue below After inserting all the elements into the queue

Queue is full



Now consider the following situation after deleting three elements from the queue.

Queue is Full (Even three elements are deleted)



This situation also says that Queue is Full and we cant insert the new element because rear is still at last position. In above situation, even though are have empty positions in the queue of them insert new element this is the major problem in normal queue data structure to overcome this problem are use circular queue data structure

13.2

Dr. D. Y. PATIL
EDUCATIONAL FEDERATION

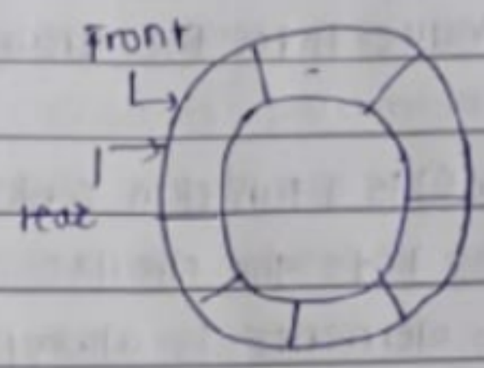
• What is circular queue?

In q. n

A circular queue can be defined as follows

circular queue is a linear data structure in which the operations are performed based on FIFO (First In First Out) principle and the last position is connected back to the first position the make a circle.

Graphical representation of a circular queue is as follows



Implementation of circular queue.

to implement a circular deque data structure using array. we first perform the following steps before we implement actual operations

Step 1: Include all the header files which are used in the program and define a constant 'SIZE' with specific value

13.3

Dr. D. Y. PATIL
EDUCATIONAL FEDERATION

Algorithm:

1. set front and rear to -1;
2. set the maximum size of the circular queue
3. Creates an array 'cq' of size MAX. SIZE
4. End.

Display:

1: If front is -1
1.1: print "queue is empty".
1.2: Return
2. set to front
3. Repeat until i is not equal to rears
   3.1 · print eq[i]
   3.2 Increment i by 1
   3.3 : set i to (i+1)%MAX-SIZE).
4. print eq[rear]
5. End.

13.4

Flowchart: