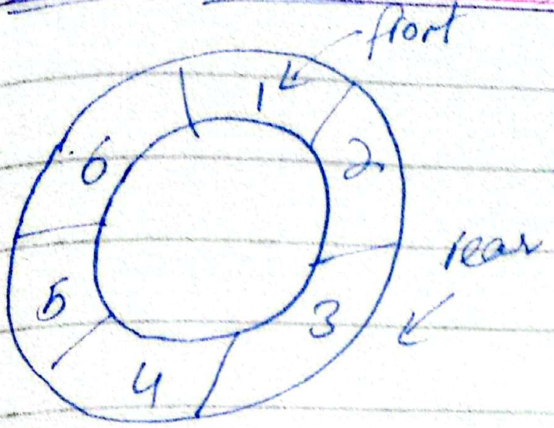
# Circular Queue (FIFO) — with fixed capacity



push → rear
pop → front
front → front

→ Implementation using Array

```
class CircularQueue{
 int *arr
 int currsize, cap
 int f, r
public:
CircularQueue(int size){
  cap=size
arr= new int(cap)
  f=0 ; r=-1
① void push()
② void pop()
③ int front()
④ bool empty()
③ int front(){
   if(empty()){
     cout<<"CQ is Empty";
        return -1 }
   return arr[f];
```

```
① void push(int data){
   if (currsize==cap){
    cout<< "CQ is Full"
         return; }
    r= (r+1)% cap;
     arr[r] = data;
      currsize++
   }
② void pop(){
   if(empty()){
    cout<<"CQ is Empty"
          return;}
    f =(f+1)% cap
      currsize--
   }
④ bool empty(){
    return currsize==0;
```