NO.

characteristics of an algm

unambiguity - instrns should be clear & straighfwd

Finiteness - terminate after finite number of steps

Definiteness - steps defined precisely.

I/P -

o/P -

effectiveness - sufficient basic & infinite length of time

a) $n(n+1)$ & $2000n^2$

n^2+n $\quad$ $2000n^2$ same order.

b) $100n^2$ & $0.01n^3$

$\quad$ 2 $\quad$ 3

$\boxed{100n^2 < 0.01n^3}$

3) $\log_2 n$ & $\ln n$

$$\log_2 n = \frac{\log_e n}{\log_e 2}$$

$$= \frac{\ln n}{\log_e 2}$$

i.e., $\ln n = \log_e 2 * \log_2 n$ $\boxed{\text{same order}}$

so. $\ln n \approx \log_2 n$

4) $\log_2^2 n$ & $\log_2 n^2$

$\log_2 n * \log_2 n$ $\quad$ $2 \log_2 n$

$\log_2 n * \log_2 n >> 2 \log_2 n$ for very large value of n

$$\boxed{\log_2^2 n > \log_2 n^2}$$

5) 2^{n-1} & 2^n

$\frac{2^n}{2}$ & 2^n

$\boxed{2^n \to 2^n}$ same

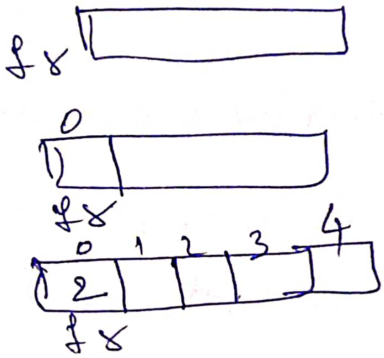
6) $(n-2)!$ | $5 \log (n+100)^{10}$ | 2^{2n} | n^4

$(n-2)!$ | $\log n^{10}$

$\log n^{10} < n^4 < 2^{2n} < (n-2)!$

Circular Queue

```
# define N 5
int Queue[N]
int front=-1, rear=-1,
void enqueue(int x)
{
  if (f==-1 && r==-1)
  {
     f=r=0
     Queue[r]=x,
  }
  else if ((rear+1)%N)==front)
  {
     Queue is full
  }
  else
  {
  rear=(rear+1)%N
     Queue[rear]=x,
```

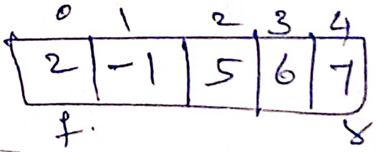


2 5 (2
4
T

(rear+1)%N==f
(1+1)
2%5=
$\boxed{2==2}$
full

r=(0+1)%5
1==0

rear=1

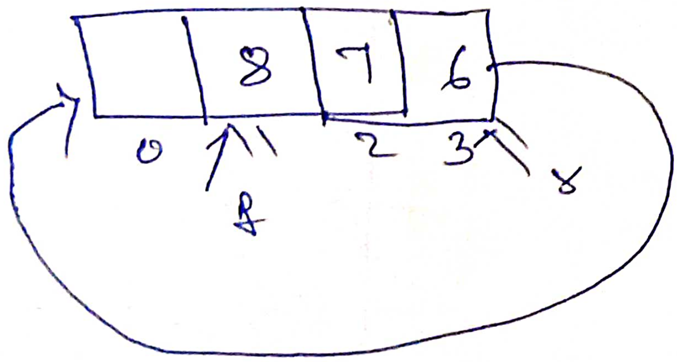



```
void dequeue(int x)
{
   if (f==-1 && r==-1)
   {
     Queue is empty
   }
else if
```

Enqueue & Dequeue

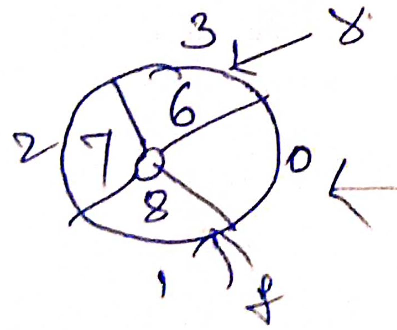
both insertion & deletion can happen in both end.

circular Queue



wrap around the queue.

(1)



$f < r$

~~rear = N~~

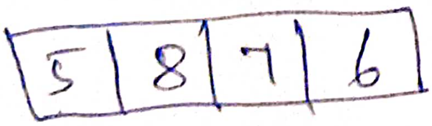
$(rear+1) \% N$

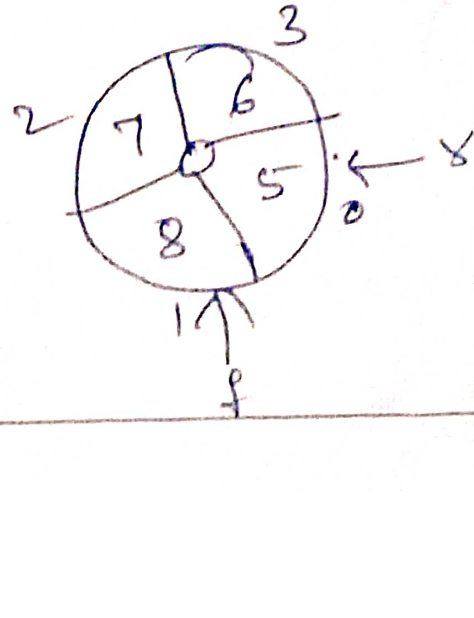
$(4+1) \% 5$

$5 \% 5$

$= 0$

(2)





$f > r$

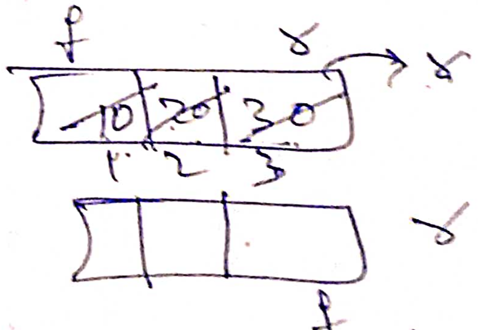
```
algo queue-delete (queue[], front, rear, max, item)
if (front==0)
  { print underflow;
  }
else
  { item Queue[front]
    if (f == rear)
      { f=0
      }
    else if (f==maxq)
      { f=1
      }
    else {
        f=f+1
      }
  } exit
```

Action	F	R	Queue
Empty	0	0	1: _, 2: _, 3: _, 4: _, 5: _
insert A, B, C	1	3	A, B, C, _, _
Delete A.	2	3	_, B, C, _, _
insert DE	2	5	_, B, C, D, E
Delete BC	4	5	_, _, _, D, E
insert F	4	1	F, _, _, D, E
insert G H	4	3	F, G, H, D, E
Delete DE	1	3	F, G, H, _, _
Insert I	1	4	F, G, H, I, _
Delete F	2	4	_, G, H, I, _

(*) Dequeue

```
{
  if isEmpty()
       return
  else if front == rear
     front ← rear ← -1
  else
       front ← front + 1
}
```

(*) f < r



r

f < r

f < r.

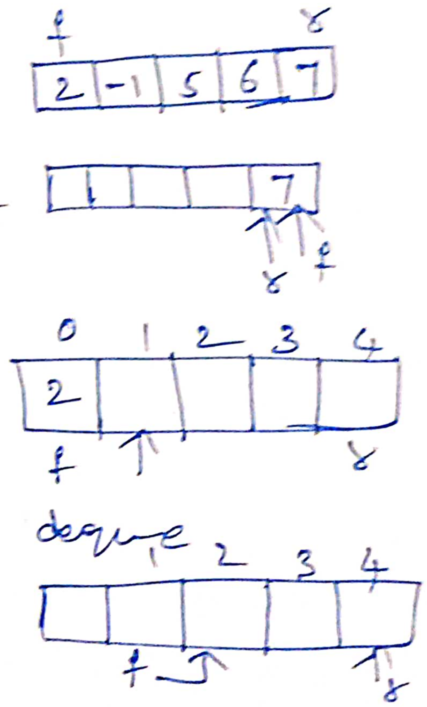
r = 3

f > r

Peek → head of the Queue

cular dequeue

```
void dequeue()
{
if (f==-1 && r==-1)
  {
   Q is underflow
  }
else if (f==r)
  {
   f=r=-1      // one element
else
   f=(f+1)%N
    (0+1)%5
      1%5
      =1
    (1+1)%5
      2%5      5)2(
                 4
                 1.5
```



display

```
    while (i!=rear)
    {
    print ("%d, q[i])
        i=(i+1)%N;
    }
```

$(x-5)!$, $2\log(x+200)^{20}$, 4^{2n}, $0.001x^4+5x^3+2$

$(x-5)!$, $\log x^{20}$, 4^{2n}, x^4

$\log x^{20} < x^4 < 4^{2n} < (n-2)!$

2^n

```
if (n==0) return 1
return F(n-1)+F(n-1)
```

$$F(n) = \begin{cases} 0 & \text{if } n=0 \\ 1+2f(n-1) & \text{otherwise.} \end{cases}$$

$f(n) = 1+2f(n-1)$

$= 1+2(1+2f(n-2))$

$= 1+2+2^2 f(n-2)$

$= 1+2+2^2(1+2f(n-3))$

$\vdots$

$= 1+2+2^2+2^3+2^4 \dots + 2^i . f(n-i)$

$= 1+2+2^2+2^3+2^4 \dots 2^{n-1}$

$= \dfrac{a(r^n-1)}{r-1} = \dfrac{1(2^n-1)}{2-1}$ $\boxed{= 2^n - 1}$

$f(n) = 1+2f(n-1)$