COMPUTER SCIENCE AND DA

Data Structures through Python

Queues and Hash Tables

Lecture No. 04

By- Kashif Sir

Topics to be covered

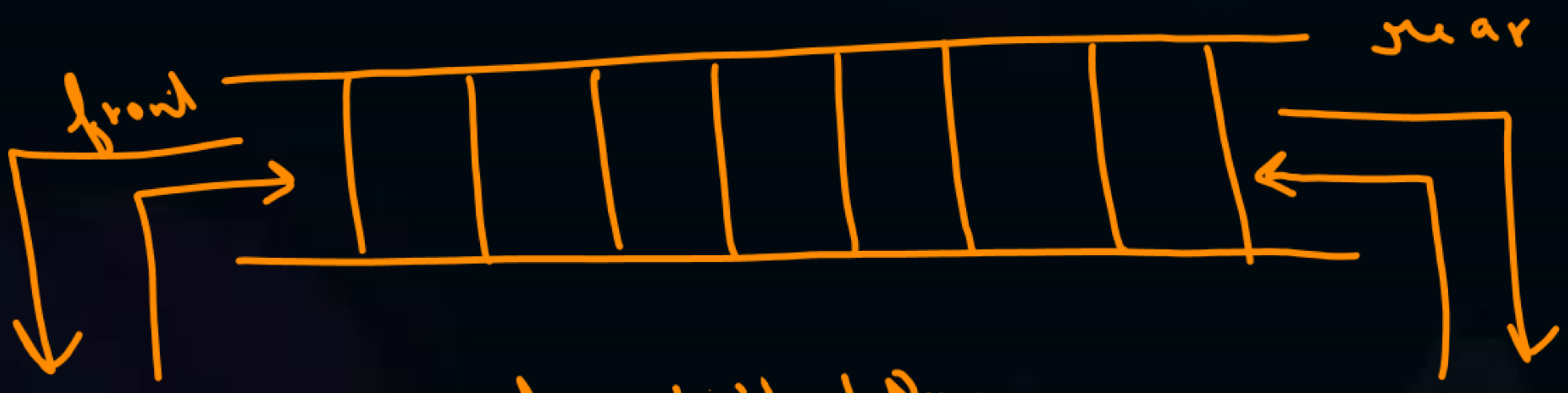
→ Circular Queue

Agenda → Double Ended Queue

⇒ Priori Queue

Double Ended Queue (Deque)

In Double Ended Queue, Insertion & Deletion can be done at both end.



Forms of Double Ended Queue

→ Input restricited Queue → Insertion : rear
Deletion : rear, front

→ Output restricted Queue — Deletion : front
Insertion : rear, front.

Consider a Deque D with maxsize = 7

$\underbrace{f = r}$ = None

			↓r	↓f		f
10	20	50	60	70	40	30
0	1	2	3	4	5	6

Enqueue_rear(D, value)
Enqueue_front(D, value)

Dequeue_rear(D)

Dequeue_front(D)

Enqueue_rear(D, 10) ⇒ f = 0, r = 0
D[r] = D[0] = 10

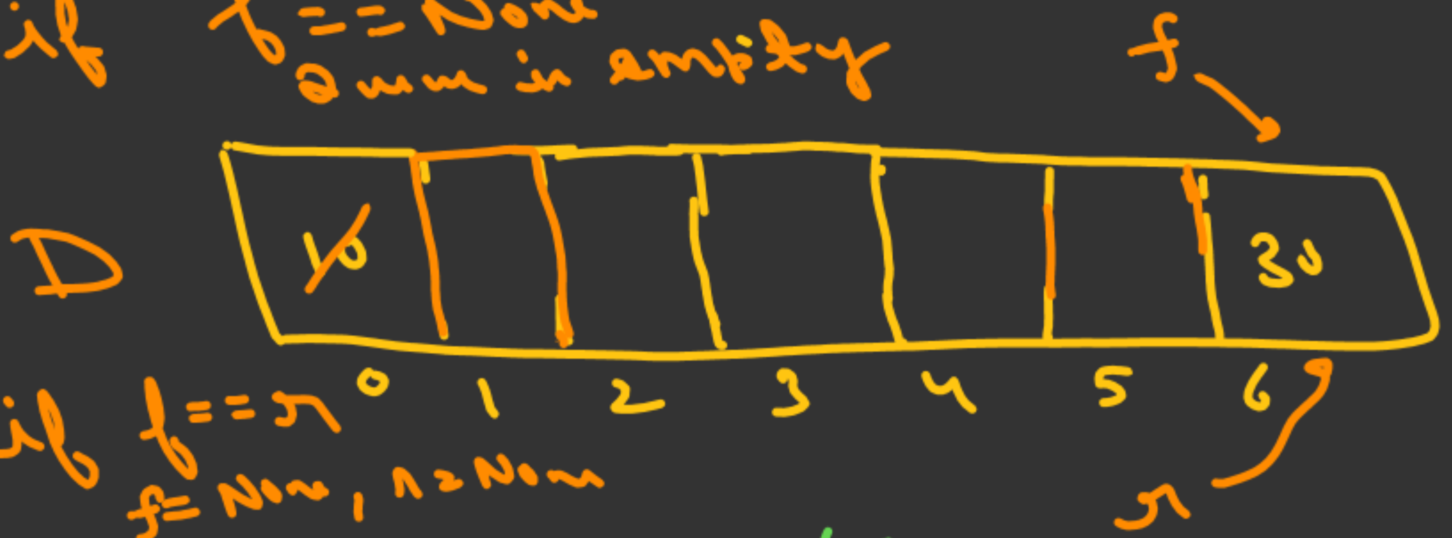
Enqueue_rear(D, 20) ⇒ r = (r + 1) % maxsize
r = 1, D[r] = D[1] = 20

Enqueue_front(D, 30) ⇒ if f == 0 :
f = maxsize - 1
else
f = f - 1
f = 7 - 1 = 6, D[f] = D[6] = 30

Enqueue_front(D, 40)
f = f - 1 = 6 - 1 = 5
D[f] = D[5] = 40

Enqueue_rear(D, 50)
r = (r + 1) % maxsize
= 2
D[r] = D[2] = 50

if $f == None$
queue is empty



if $f == r$
$f = None$, $r = None$

Dequeue_rear(D)
↳ del_val = $D[r] = D[3] = 60$
$r = r-1 = 3-1 = 2$

Dequeue_rear(D)
↳ del_val = $D[r] = D[2] = 50$
$r = r-1 = 2-1 = 1$

Dequeue_rear(D)

```
⟹ del_val = D[r]
   if r == 0 :
        r = maxsize-1
   else
        r = r-1
```

Dequeue_front(D)
↳ del_val = $D[f] = D[4] = 70$
$f = (f+1) \%$ maxsize
$f = (4+1) \% 7 = 5$

Dequeue_front(D)
↳ del_val = $D[f] = D[5] = 40$
$f = (f+1) \%$ maxsize
$= (5+1) \%$ maxsize
$= 6$

```
def Dequeue-rear(D):
    if r == None:
        print("Queue is empty")
        return
    del-val = D[r]
    if f == r:
        f = None
        r = None
    else
        if r == 0
            r = maxsize-1
        else
            r = r-1
    return del-val
```

```
def Dequeue-front(D)
    if f == None:
        print('Queue is empty')
        return
    ret-val = D[f]
    if f == r:
        f = None
        r = None
    else
        f = (f+1) % maxsize
    return ret-val
```

Que) Consider an empty DEQUE of size 7

10 _ _ _ _ _ _   Enqueue-rear (D, 10)

10 20 _ _ _ _ _   Enqueue-rear (D, 20)

_ 20 _ _ _ _ _   Dequeue-front (D)

30 20 _ _ _ _ _   Enqueue-front (D, 30)

30 20 _ _ _ _ 40   Enqueue-front (D, 40)

30 20 _ _ _ 50 40   Enqueue-front (D, 50)

30 20 60 _ _ 50 40   Enqueue-rear (D, 60)

Dequeue-front (D)   30 20 60 _ _ _ 40

Dequeue-front (D)   30 20 60 _ _ _ _

Dequeue-rear (D)   30 20 _ _ _ _ _

Enqueue-front (D, 70)   30 20 _ _ _ _ 70

Enqueue-rear (D, 80)   30 20 80 _ _ _ 70

Final DEQUE will be:

0	1	2	3	4	5	6
30	20	80				70

rear

front

Enqueue

```
rear = (rear + 1) % maxsize
D[rear] = value
```

```
if front == 0
    front = maxsize - 1
else
    front = fron - 1
D[front] = value
```

Dequeue

```
del-val = D[rear]
if rear == 0
    rear = maxsize - 1
else
    rear = rear - 1
```

```
del-val = D[front)
front = (front + 1) %
              maxsize
```

```
def Enqueue_rear(D, value)
    if f==((r+1)%maxsize):
        print("Queue is full")
        return

    if r==None
        r=0
        f=0

    else
        r=(r+1)%maxsize

    D[r]=value
```

```
def Enqueue_front(D, value):
    if f==((r+1)%maxsize)
        print("Queue is full")
        return

    if f==None
        f=0
        r=0
    else
        if f==0
            f=maxsize-1
        else
            f=f-1
    D[f]=value
```

THANK - YOU