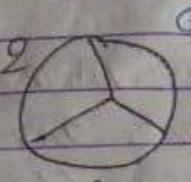
5/2020

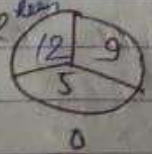
Circular Queue



Queue Empty
Front = -1
Rear = -1

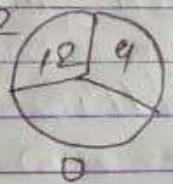
After 1st Addition
Front = 0
Rear = 0

After 2nd Addition
Front = 0
Rear = 1



After 3rd Addition
Front = 0
Rear = 2
Queue Full

Formula
i) if front = 0 and rear = last than Que is Full.



After 1st Deletion
Front = 1
Rear = 2

After 2nd Delete
Front = 2
Rear = 2

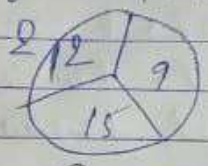
After 3rd Deletion
Queue Empty
Front = Rear = 2

Formula.

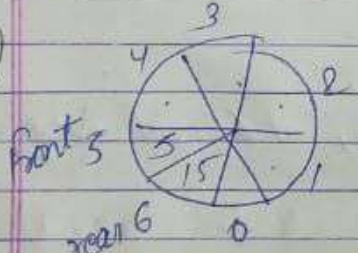
If rear + 1 = front then Queue full.

After One deletion we can add a new element in the given Que. (in blank space)

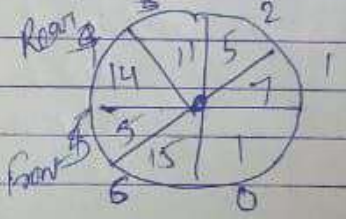
After 4th addition



(i)



(ii) After 4 new addition



After 5th 4th Deletion front reach on 5 number

```
# include <stdio.h>
# include <conio.h>
int main ( )
{
 void addque (int *a, int *front, int *rear);
 void delque (int *a, int *front, int *rear);

int a[5];
int front, rear;
  front = rear = -1;
  addque (a, &front, &rear);
  addque (a, &front, &rear);
  addque (a, &front, &rear);
  delque ( a, &front, &rear);
   getch ( );
   return ( );
    }

void addque (int *a, int *front, int *rear)
{
 if ( *front == 0 && *rear == 4)
  || (*rear + 1 == *front)
  {
   printf ("\n overflow");
        return;
        }
```

```
(*rear)++;
printf("\n Enter value to add");
scanf("%d", &q[*rear]);
if (*rear == 4)
    *rear == 0;
else
    (*rear)++;
if (*front == -1)
    front = 0;
}

void delque (int *c, int *front, int *rear)
{
if (*front == -1)
{
printf("Underflow");
return;
}
Printf ("deleted value is %d" q[*front];
    q[*front] = 0;
    if (*front == 4)
    *front = 0;
    else if (*front == *rear)
  * front = *rear == -1
        else (*front)++;
}
```

Linear que =>

```
#include<stdio.h>
#include<conio.h>
int main()
{
void addque(int *, int *front, int *rear);
void delque(int *, int *front, int *rear);
int s[5];
int front, rear;
front = rear = -1;
addque(s, &front, &rear);
addque(s, &front, &rear);
addque(s, &front, &rear);
delque(s, &front, &rear);
delque(s, &front, &rear);
    getch();
    return 0;
    }
void adelque(int *s, int *front, int *rear)
{
  if(*rear==4)
  {
    Printf("\n Overflow");
    return;
  }
    Printf("Enter Value to add");
    *rear++;
```

```
Scanf ("%d, &a[*rear]);
  if (*front == -1)
     front = 0;
  }

Void delque (int *s, int *front, int *rear)
  {
    if (*front == -1)
    {
     Printf ("Underflow");
     return;
    }
Printf ("deleted value is %d", a(*front));
  a[*front] = 0;
  if (*front == *rear)
     *front == *rear = -1;
     else
     (*front)++;
  }
```

Dec 10/2020

Deque

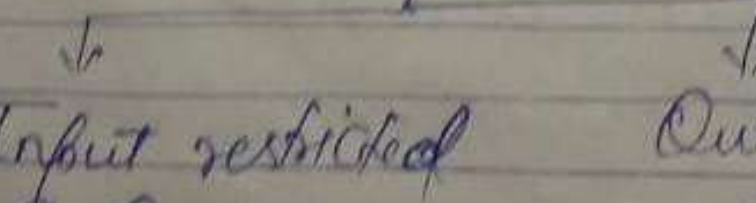
Double Ended Queue.

Deque is a linear Data Structure which adds a new element at the rear end and deletes existing element from the front ends. But As it's name satisfied Specified, this type of que has both ends working as front ends and rear end. This que is also follows the principal of FIFO.

Deque.

It means we can say, we can add from both ends as well as delete from both ends.

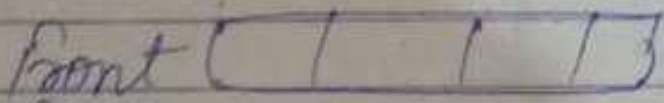
Deque



i) Input restricted Deque ⇒

In this type of Deque deletion can be done from both side but addition can be done from only one side. This is because it has front at both end but rear at only one end.

(ii) Output Restricted Deque ⇒



This type of Deque has rear at both ends but front at only one end. That is why additions can be done from both ends ~~deletion~~ but Deletion from only one end.

DeQueue Programs.

```
#include <stdio.h>
#include <conio.h>
int main()
{
  void addbeg(int *a, int *front, *rear);
  void delbeg(int *a, int *front, *rear);
  void addlast(int *a, int *front, *rear);
  void dellast(int *a, int *front, *rear);
  int a[5] = {0,0,0,0,0};
  int front, rear;
  front = rear = -1;
  addbeg(a, &front, &rear);
  addbeg(a, &front, &rear);
  addbeg(a, &front, &rear);
  delbeg(a, &front, &rear);
  dellast(a, &front, &rear);
  getch();
  return 0;
}
void addbeg(int *a, int *front, int *rear)
{
  int i, ctr, m;
  if(*front == 0 && *rear == 4)
  {
    printf("Overflow");
```

return;
}
if (*front == -1 && *rear == -1) // Add first node
{
*front = rear = 0;
Printf ("Enter new value");
Scanf ("%d", &a[*front]);
}
else if (*front > 0) // space in the begining
{
(*front)--;
Printf ("Enter new value");
Scanf ("%d", &a(*front]);
}
else if (*front == 0 && rear < 4) // NO space in the beginging but in Last
{
Count the number of filled up node
ctr = 0
for (i=0; i<5; i++)
{
if (a[i] != 0)
ctr++;
}

// Right Shift begins

```
m = *rear + 1;
for (i=1; i<=ch; i++)
  {
    a[m] = a[m-1];
    m--;
  }
Printf ("Enter new value");
Scanf ("%d", &a[m]);
    (*rear)++;
  }
}
```

To be continued. Other functions on next day