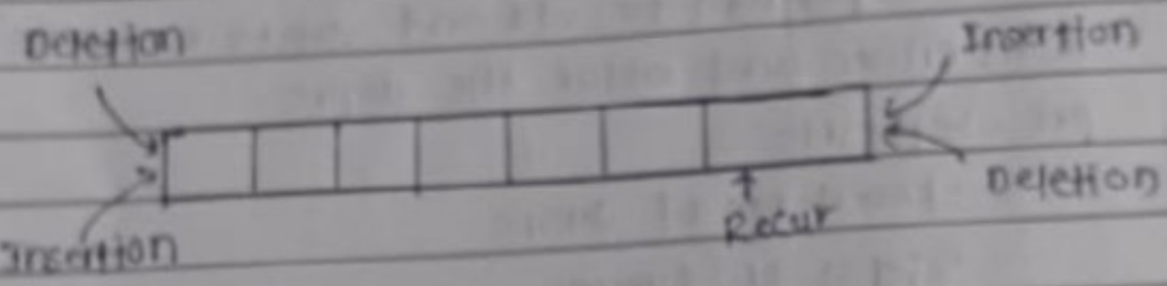
Dr. D. Y. PATIL

Theory:

Double ended queue

A double ended queue is an abstract data type similar to an simple queue. It allows you to insert and delete from both sides means items can be added or deleted from the format or rear end.



Algorithm for insertion and at rear end

Step 1 [Check For over flow]

If (rear == MAX)

print ("Queue' is overflows');

return;

Step 2: [Insert element]

else

rear = rear + 2

q [rear] = no';

[Set rear and front pointer]

If rear = 0

rear = 1;

If front = 0

front = 1;

Algorithm for Deletion from front end

Step 1 [check for front pointer]

If front = 0

print ("Queue is underflow");

return

Step 2 [perform deletion]

else

no = q[front];

print ("Deleted element is", no);

[set front and rear pointer]

If front = rear

Front = 0;

rear = 0;

else

Front = front + 1;

Step 3: Return

Implementation of deletion from front end void

delete_item_front()

{

int num;

if (front == 0)

{

printf ("\n Queue is underflow \n");

return;

}

Dr. D. Y. PATIL
EDUCATIONAL FEDERATION

```
else
{
num= q[front];
printf ("\nDeleted item is %d ", num);
  if (front= =rear)
  {
  front=0;
  rear=0;
  }
}
```

Implementation of Deletion from rear end

```
  void delete-item-rear()
  {
  int num;
  if (rear==0)
  {
  print f ("\n cannot delete item at rear end \n");
  return;
  }
  else
  {
  num = q [rear];
  if (front = = rear)
  {
```

Dr. D. Y. PATIL
EDUCATIONAL FEDERATION

Step3: Return

Implementation of Insertion at rear and

```
void add_item_rear()
{
int num;
printF ("\n Enter Item to insert: 1):
scanF ("%d", &num);
if (rear == MAX)
{
@rintF ("\n Queue is overflow");
  return;
}
else
{
rear++
q[rear] =num;
if (rear==0)
rear =1;
if (front == =0)
front = 1;
}
}
```

Algorithm for insertion and front end

step1 [check for the front position]

if (front <=1).

print(" cannot add item at front end ");
return;
Step2: [insert at front]
else
front = front - 1;
q[front] = no;
Step3: Return

Implementation of Insertion of front end

```
void add_item_front()
{
int num;
printf("\n Enter item to insert:");
scanf("%d", &num);
if (front <= 1)
{
printf("\n cannot add item at front end");
return;
}
else
{
front--;
q[front] = num;
}
```

```
Front = 0;
rear = 0;
}
else
{
rear---;
print F ("\n Deleted Item is %d", num);
}
}
```

12.7

Flowchart :