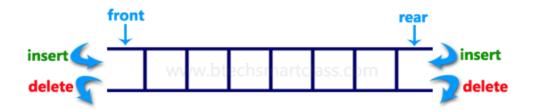
UNIT – II LINEAR DATA STRUCTURES

Lecture NO. 9

Dequeue (Double Ended Queue)

It is a linear data structure. This queue is called double ended queue. In this type of queue insertion and deletion of elements can be take place from both ends.



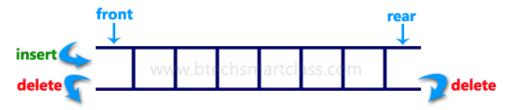
In this queue, insertion and deletion of elements from Rear and Front can take place on the principal of Right and Left pointer.

There are two types of dequeue.

1) input restricted queue

In this type of queue, insertion can take place only from one end and deletion can take place from both ends.

Input Restricted Double Ended Queue



2) Output restricted queue

In this type of queue, insertion can take place from both ends and deletion can take place from one end.



Principal for Insertion

When the elements are inserted from Rear/ Right then Rear pointer is incremented and when elements are inserted from Front/ Left then Front pointer decremented.

Double Ended Queue Algorithms

ALGORITHM FOR THE INSERTION OF ELEMENT TO RIGHT IN DEQUEUE

Algorithm DQInsertRight (DQ, N, item, Rear, Front)

[This algorithm is used to insert the element to the right of DQ, where

DQ = Array

N = Final Limit

Item = which is to be inserted

Front, Rear = Deletion and Insertion pointers]

Step-1 [Check for overflow]

```
If ((Front = 1) && (Rear = N) or (Front = Rear + 1) then
```

Write ("Overflow")

Goto step-4

End if

Step-2 [Reset rear pointer]

IF (F=0 && R=0)

F=R=1

Else if (Rear = N) then

Rear = 1

Else

```
Rear = Rear + 1
        End if
Step-4 [Insert the element]
        DQ [Rear] = item
        Goto step-1
Step-5 [Finished]
Exit
ALGORITHM FOR THE INSERTION OF ELEMENT TO LEFT IN DEQUEUE
Algorithm DQInsertLeft (DQ, N, item, Rear, Front)
[This algorithm is used to insert the element to the left of DQ, where
        DQ = Array
        N = Final Limit
        Item = which is to be inserted
        Front, Rear = Deletion and Insertion pointers ]
Step-1 [Check for overflow]
        IF ((Front = 1) && (Rear = N) or (Front = Rear + 1) then
        Write ("Overflow")
        Goto step-4
        End if
Step-2 [Reset the front pointer]
        IF (F=0 && R=0)
        F=R=1
        Else IF (F = 1) then
               F = N
        Else
               F = F-1
        End if
Step-4 [Insert the element]
        DQ[F] = item
```

```
Goto step-1
Step-5 [Finished]
       Exit
ALGORITHM FOR THE Deletion OF ELEMENT TO FRONT IN DEQUEUE
Algorithm DQDelFront (DQ, N, item, Rear, Front)
[This algorithm is used to delete the element to the front of DQ, where
       DQ = Array
       N = Final Limit
       Item = which is to be inserted
       Front, Rear = Deletion and Insertion pointers]
Step-1 [Check for underflow]
       IF (F=0 && R=0)
       Write ("Underflow")
       Goto step-4
       End if
Step-2 [Check the front pointer and delete the item]
       IF (F=R)
       X = Dq(F)
       F=R=0 goto step-1
       Else IF (F=N)
       X = Dq(F)
       F=1 goto step-1
       Else
       X=Dq(F)
       F= F+1 goto step-1
Step-3 [Finished]
```

Exit

```
Algorithm DQDelRight (DQ, N, item, Rear, Front)
[This algorithm is used to delete the element to the right of DQ, where
        DQ = Array
        N = Final Limit
        Item = which is to be inserted
        Front, Rear = Deletion and Insertion pointers]
Step-1 [Check for underflow]
IF (F=0 && R=0)
        Write ("Underflow")
        Goto step-4
        End if
Step-2 [Check the Rear pointer and delete the item]
        IF (F=R)
       X = Dq(R)
        F=R=0 goto Step-1
        Else IF (R=1)
       X=Dq[R]
        R=N, Goto Step-1
        Else
       X=Dq[R]
        R= R-1, Goto Step-1
Step-4 [Finished]
        Exit
```

Algorithm DQInsertRight	Algorithm DQInsertLeft
Step-1 [Check for overflow]	Step-1 [Check for overflow]
If ((Front = 1) && (Rear = N) or (Front =	IF ((Front = 1) && (Rear = N) or (Front =
Rear + 1) then	Rear + 1) then
Write ("Overflow")	Write ("Overflow")
Goto step-4	Goto step-4
End if	End if
Step-2 [Reset rear pointer]	Step-2 [Reset the front pointer]
IF (F=0 && R=0)	IF (F=0 && R=0)
F=R=1	F=R=1
Else if (Rear = N) then	Else IF (F = 1) then
Rear = 1	F = N
Else	Else
Rear = Rear + 1	F = F-1
End if	End if
Step-4 [Insert the element]	Step-4 [Insert the element]
DQ [Rear] = item	DQ [F] = item
Goto step-1	Goto step-1
Step-5 [Finished]	Step-5 [Finished]
Exit	Exit
Algorithm DQDelFront	Algorithm DQDelRight
Step-1 [Check for underflow]	Step-1 [Check for underflow]
IF (F=0 && R=0)	IF (F=0 && R=0)
Write ("Underflow")	Write ("Underflow")
Goto step-4	Goto step-4
End if	End if
Step-2 [Check the front pointer and delete the	Step-2 [Check the Rear pointer and delete the
item]	item]
IF (F=R)	IF (F=R)
X = Dq(F)	X = Dq(R)
F=R=0 goto step-1	F=R=0 goto Step-1
Else IF (F=N)	Else IF (R=1)
X= Dq(F)	X=Dq[R]
F=1 goto step-1	R=N, Goto Step-1
Else	Else
X=Dq(F)	X=Dq[R]
F= F+1 goto step-1	R= R-1, Goto Step-1
Step-3 [Finished]	Step-4 [Finished]
Exit	Exit
	I EXII