SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A' Grade

Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING-IOT Including CS&BCT

COURSE NAME : 19ITT201 - DATA STRUCTURES

II YEAR / III SEMESTER

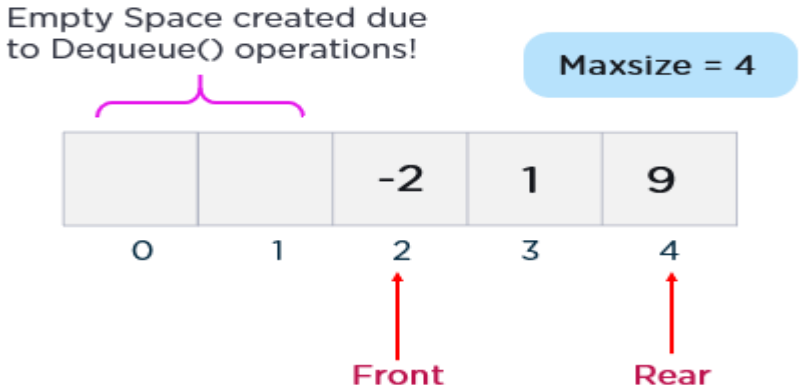
Unit 1- LINEAR STRUCTURES AND TREES

Topic 9 : Circular Queue Based Implementation

Problem

Why Was the Concept of Circular Queue Introduced?

Implementation of a linear queue brings the drawback of memory wastage. However, memory is a crucial resource that you should always protect by analyzing all the implications while designing algorithms or solutions. In the case of a linear queue, when the rear pointer reaches the MaxSize of a queue, there might be a possibility that after a certain number of dequeue() operations, it will create an empty space at the start of a queue.



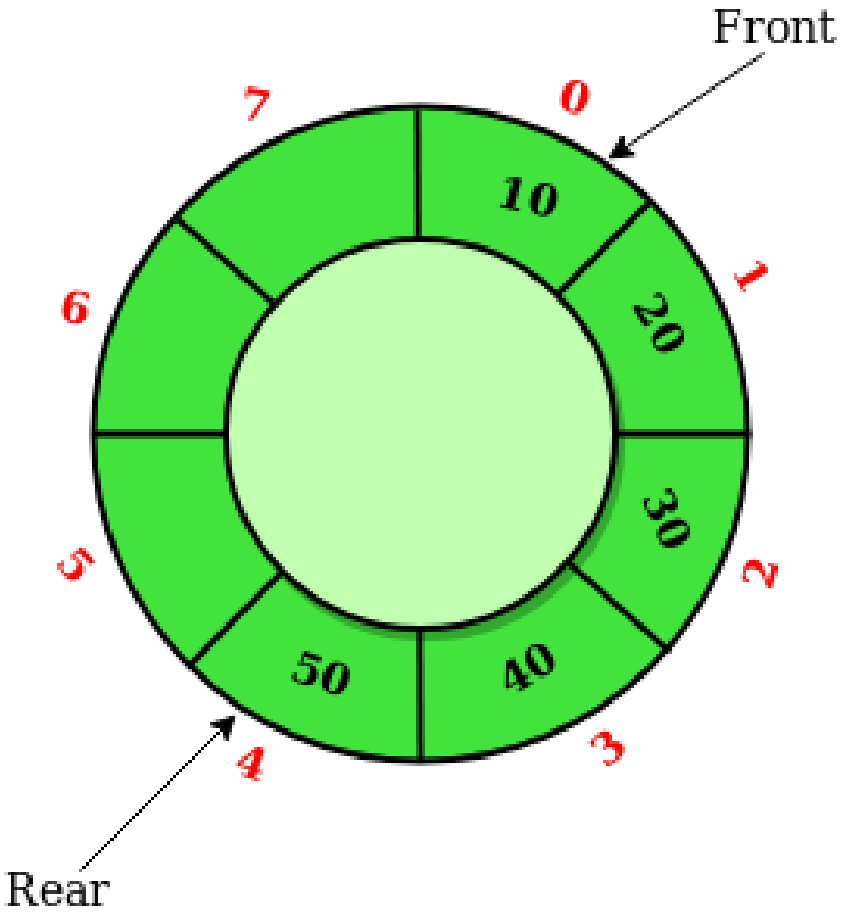
Additionally, this newly created empty space can never be re-utilized as the rear pointer reaches the end of a queue. Hence, experts introduced the concept of the circular queue to overcome this limitation.

Circular Queue Based Implementation

•Circular Queue is a linear data structure in which the operations are performed based on FIFO (First In First Out) principle and the last position is connected back to the first position to make a circle.

•It is also called **'Ring Buffer'**.

Circular Queue Based Implementation –Cont..



Circular Queue Based Implementation –Cont..

Basic Operations

- **Front:** Get the front item from queue.
- **Rear:** Get the last item from queue.
- **enQueue(value)** This function is used to insert an element into the circular queue. In a circular queue, the new element is always inserted at Rear position.
- **deQueue()** This function is used to delete an element from the circular queue. In a circular queue, the element is always deleted from front position.

Circular Queue Based Implementation –Cont..

Enqueue(x) Operation

You should follow the following steps to insert (enqueue) a data element into a circular queue -

Step 1: Check if the queue is full (Rear + 1 % Maxsize = Front)

Step 2: If the queue is full, there will be an Overflow error

Step 3: Check if the queue is empty, and set both Front and Rear to 0

Step 4: If Rear = Maxsize - 1 & Front != 0 (rear pointer is at the end of the queue and front is not at 0th index), then set Rear = 0

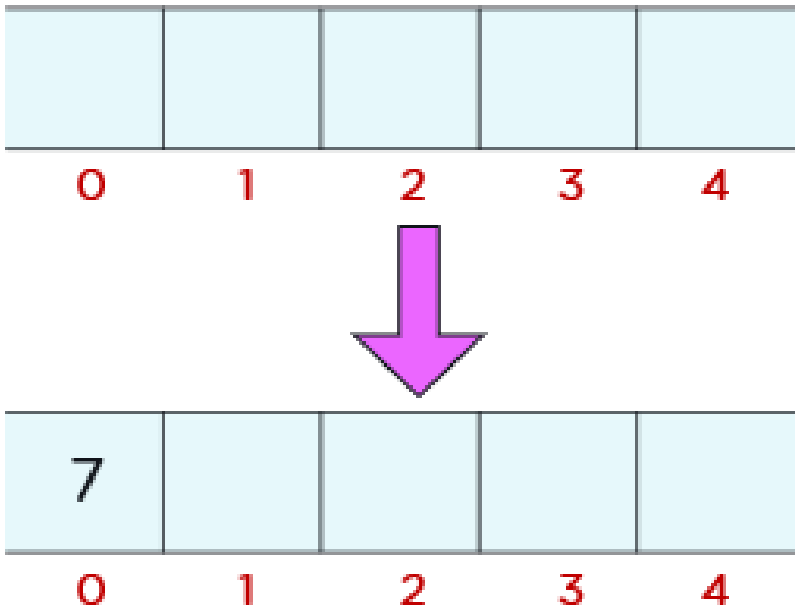
Step 5: Otherwise, set Rear = (Rear + 1) % Maxsize

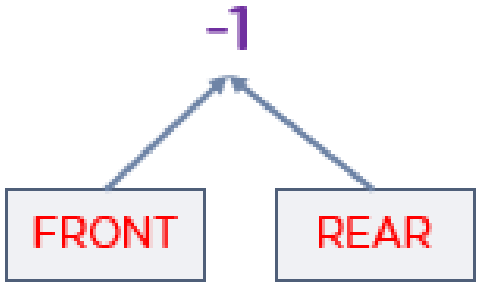
Step 6: Insert the element into the queue (Queue[Rear] = x)

Step 7: Exit

Circular Queue Based Implementation –Cont..

1. Insertion when Queue is Empty:

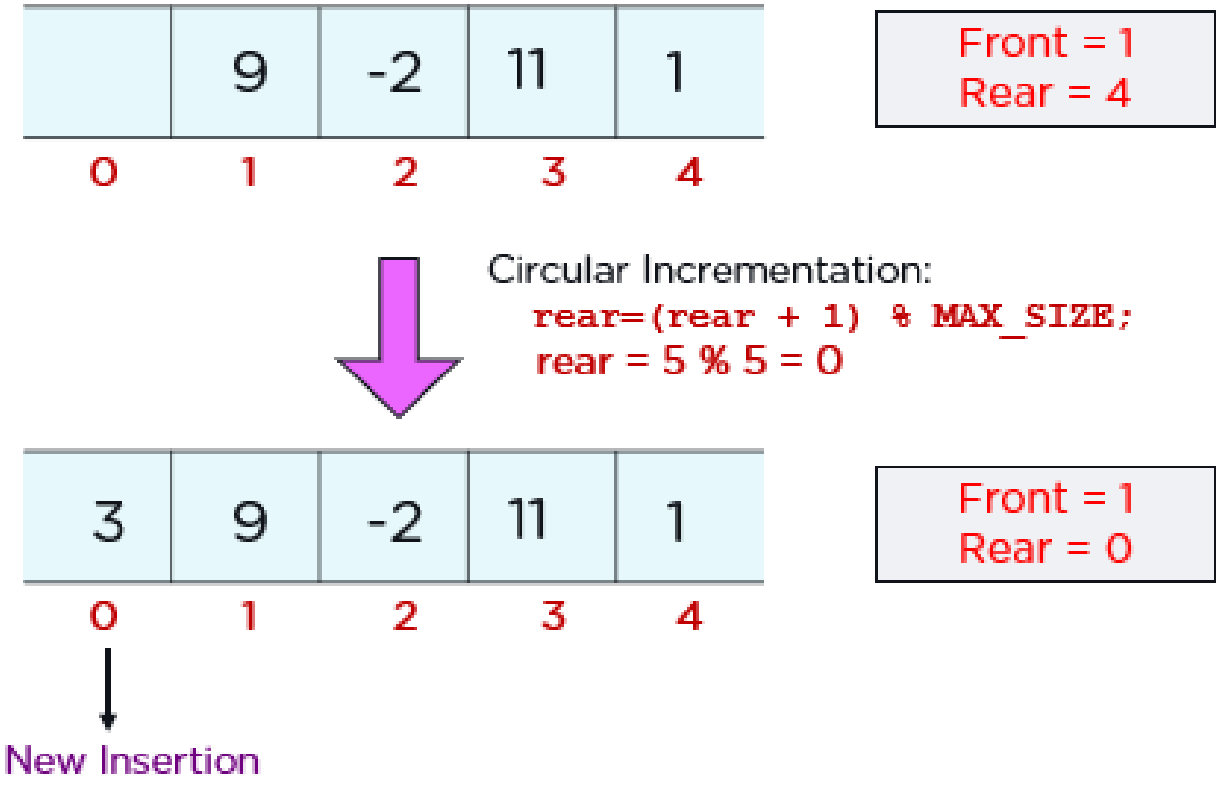




Front = 0
Rear = 0

Circular Queue Based Implementation –Cont..

2. Insertion when queue is completely filled but there is space at the beginning of the queue:



Circular Queue Based Implementation –Cont..

A structure to represent a queue

```
struct CQueue {
    int cQueue[SIZE], front = -1, rear = -1;
};
```

- void enQueue(int value)
- {
- if((front == 0 && rear == SIZE - 1) || (front == rear+1))
- printf("\nCircular Queue is Full! Insertion not possible!!!\n");
- else{ if(rear == SIZE-1 && front != 0) rear = -1;
- cQueue[++rear] = value; printf("\nInsertion Success!!!\n"); if(front == -1) front = 0; } }

Circular Queue Based Implementation - Cont..

Dequeue() Operation

Obtaining data from the queue comprises two subtasks: access the data where the front is pointing and remove the data after access. You should take the following steps to remove data from a circular queue -

Step 1: Check if the queue is empty (Front = -1 & Rear = -1)

Step 2: If the queue is empty, Underflow error

Step 3: Set Element = Queue[Front]

Step 4: If there is only one element in a queue, set both Front and Rear to -1 (IF Front = Rear, set Front = Rear = -1)

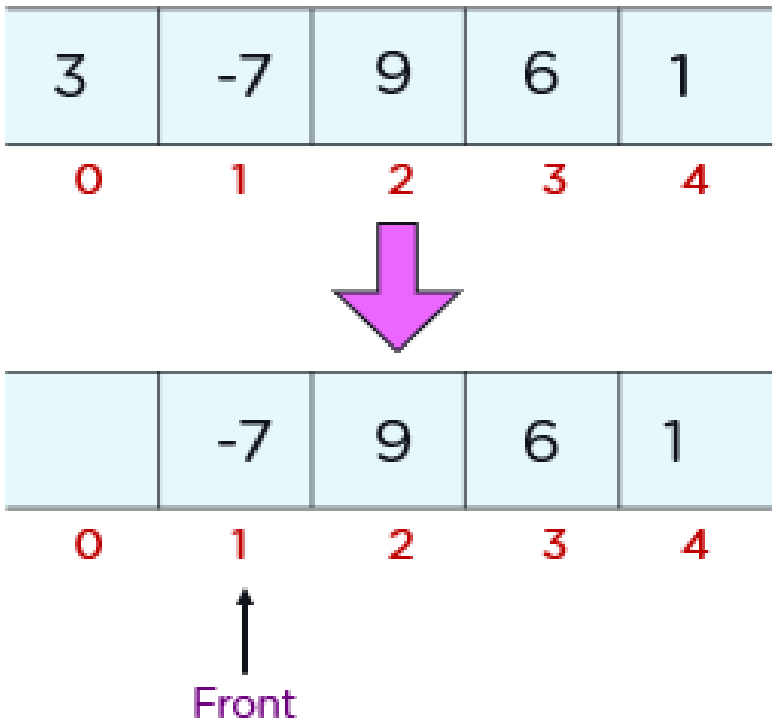
Step 5: And if Front = Maxsize -1 set Front = 0

Step 6: Otherwise, set Front = Front + 1

Step 7: Exit

Circular Queue Based Implementation - Cont..

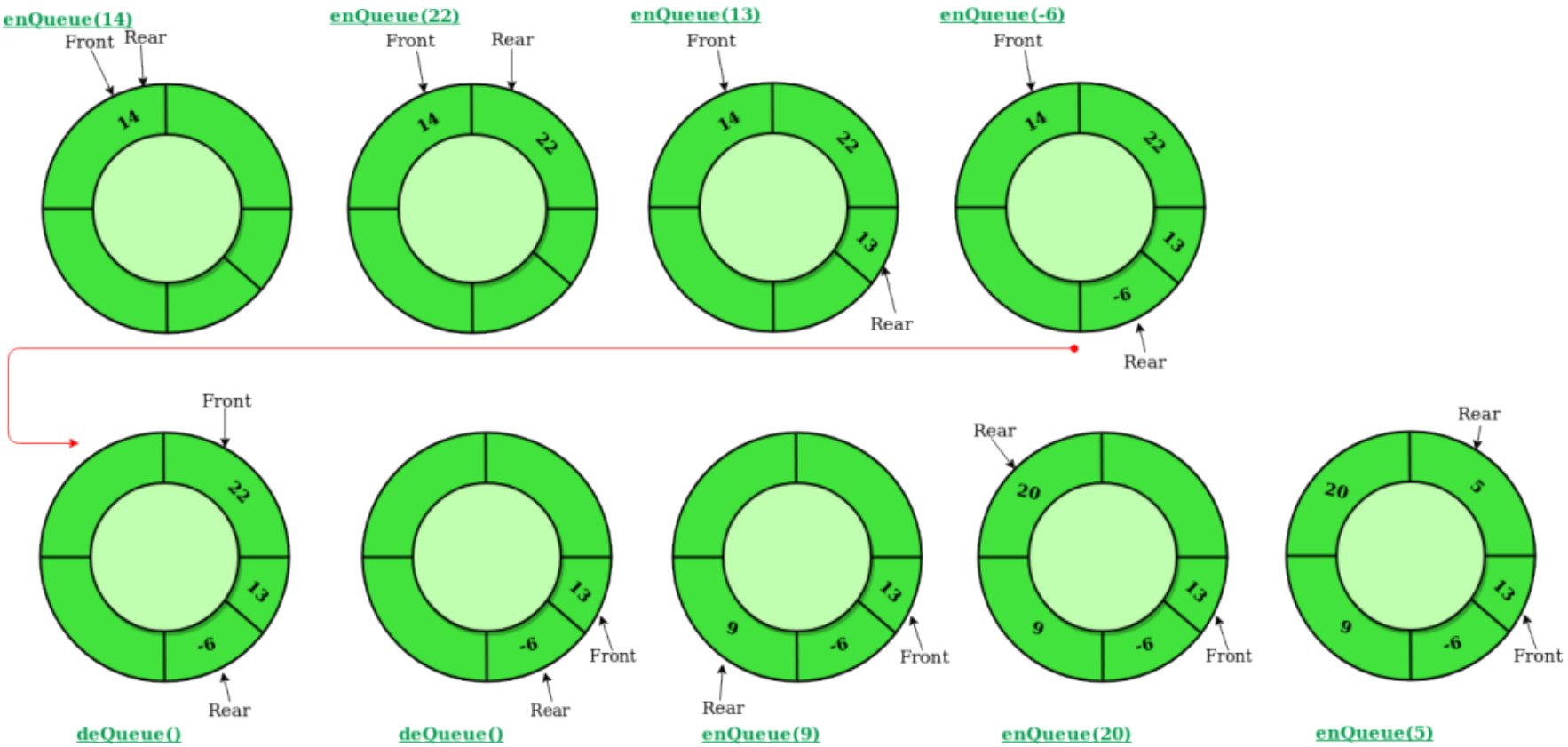
1. Deletion when rear at the end of queue and front at the beginning of the queue



Dequeue operation

```
void deQueue()
 {
if(front == -1 && rear == -1)
printf("\nCircular Queue is Empty! Deletion is not possible!!!\n"); else
{ printf("\nDeleted element : %d\n",cQueue[front++]);
if(front == SIZE)
 front = 0;
if(front-1 == rear) front = rear = -1;
 }
}
```

Queue Based Implementation –Cont..



Circular Queue Based Implementation –Cont..

1. Buffer in Computer Systems: Computer systems supply a holding area for maintaining communication between two processes, two programs, or even two systems. This memory area is also known as a ring buffer.
2. CPU Scheduling: In the Round-Robin Scheduling Algorithm, a circular queue is utilized to maintain processes that are in a ready state.
3. Traffic System: Circular queue is also utilized in traffic systems that are controlled by computers. Each traffic light turns ON in a circular incrementation pattern in a constant interval of time.

Activity

MCQ

1. What is a dequeue?

   a) A queue with insert/delete defined for both front and rear ends of the queue

   b) A queue implemented with a doubly linked list

   c) A queue implemented with both singly and doubly linked lists

   d) A queue with insert/delete defined for front side of the queue

2. Circular Queue is also called _____

   a) Square Buffer

   b) Ring Buffer

   c) Rectangle Buffer

   d) Curve Buffer

Advantages

1. Doesn't use dynamic memory → No memory leaks
2. Conserves memory as we only store up to our capacity (opposed to a queue which could continue to grow if input outpaces output.)
3. Simple Implementation → easy to trust and test
4. Never has to reorganize / copy data around
5. All operations occur in constant time O(1)

Disadvantages

1. It's hard to tell an empty queue from a full queue without retaining additional information.
2. Circular queue is you can only store queue.length elements. If you are using it as a buffer, you are limiting your history depth.

Assessment 1

1. List out the advantages of Circular queue based implementation

a)_____

b)_____

c)_____

d)_____

2. Identify the disadvantages of Circular queue based implementation

a)_____

b)_____

c)_____

d)_____

REFERENCES

1. M. A. Weiss, “Data Structures and Algorithm Analysis in C”, Pearson Education, 8th Edition, 2007. [Unit I, II, III, IV,V]

2. A. V. Aho, J. E. Hopcroft and J. D. Ullman, “Data Structures and Algorithms”, Pearson Education, 2nd Edition, 2007 [Unit IV].

3. A.M.Tenenbaum, Y. Langsam and M. J. Augenstein, “Data Structures using C”,PearsonEducation, 1st Edition, 2003.(UNIT I,II,V)

4.https://www.geeksforgeeks.org/queue-set-1introduction-and-array-implementation/

THANK YOU