K.R. MANGALAM UNIVERSITY
THE COMPLETE WORLD OF EDUCATION

Data Structure

ENCS205

School of Engineering & Technology (SOET)
K.R. MANGALAM University

UNIT-2
Session 28: Implementing Queues (Array & Linked List)

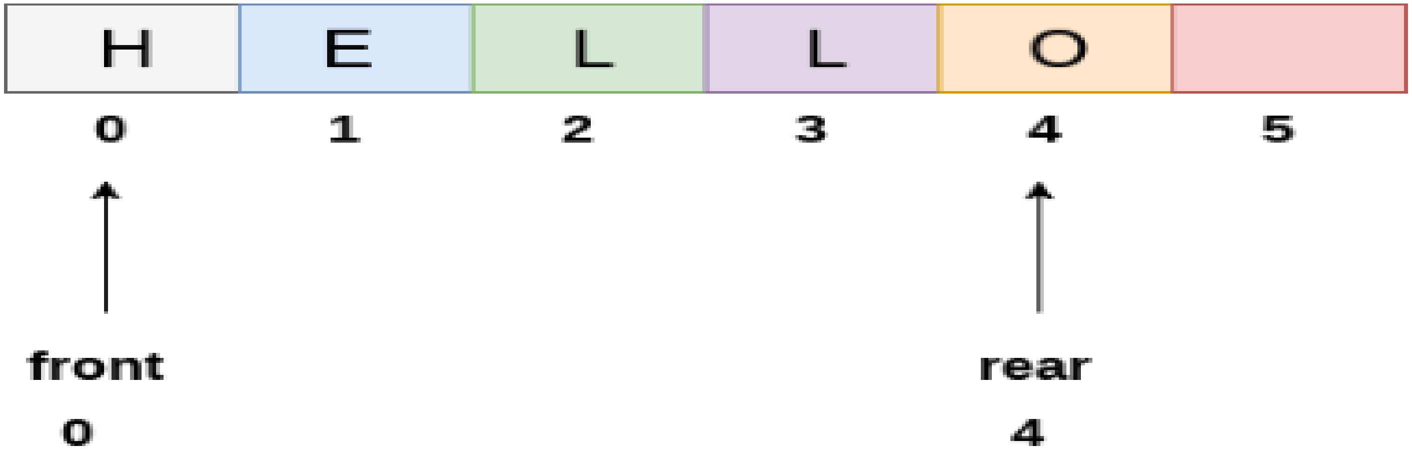
Session 28 Outlook

- Implementation of Queues using array
- Operations on Queues using arrays
- Implementation of Queues using linked list
- Operations on Queues using linked list

Objective

- Students should be able to recall Queues terminology.
- Student should be able to Understand Queues structure vs. other data structures. Differentiate implementations and expression representations.
- Apply queue structures for process scheduling problems.
- Students should Evaluate and optimize queue performance

Array based implementation of linear Queues

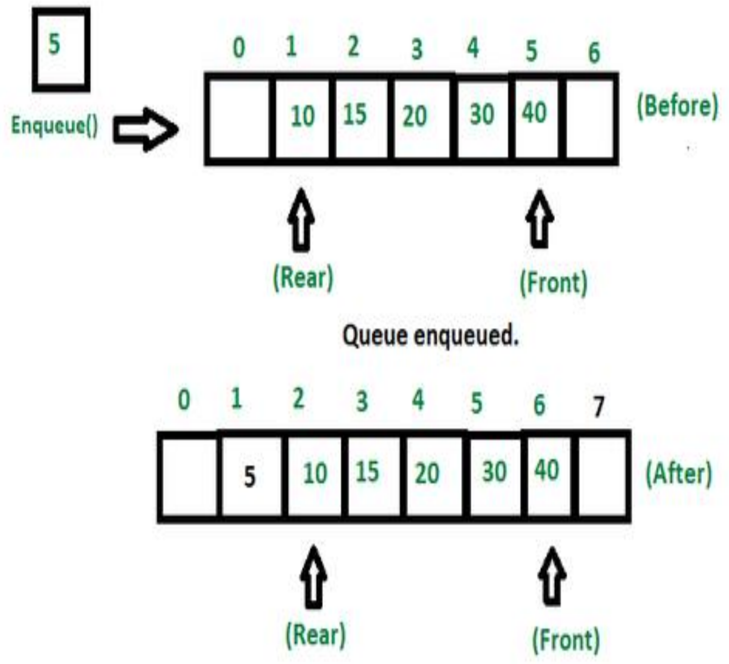


https://www.javatpoint.com/array-representation-of-queue

Enqueue Operations

Enqueue (Insert): Algorithm

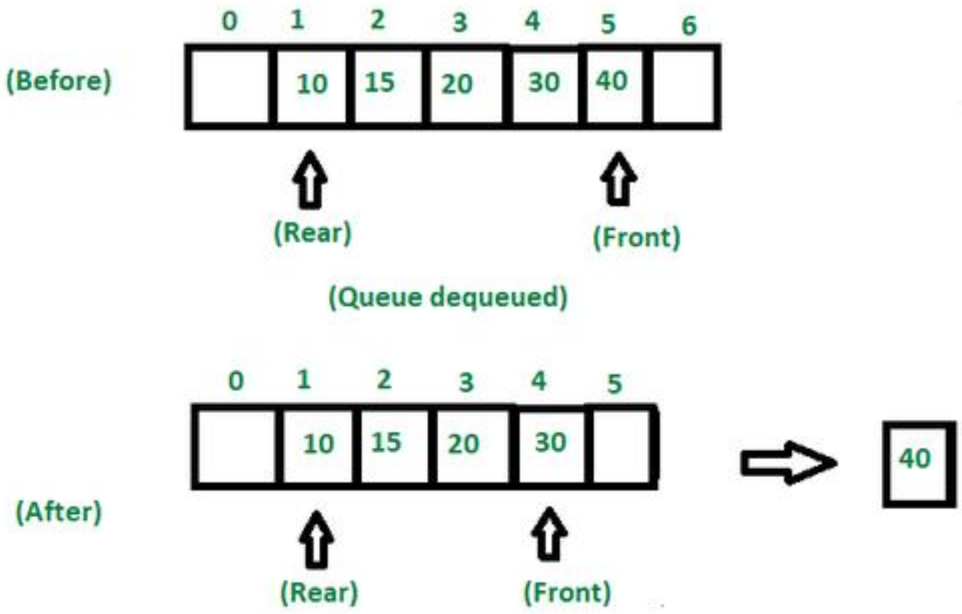
```
Step 1: IF REAR = MAX - 1
            Write OVERFLOW
            Go to step
        [END OF IF]
Step 2: IF FRONT = -1 and REAR = -1
            SET FRONT = REAR = 0
        ELSE
            SET REAR = REAR + 1
        [END OF IF]
Step 3: Set QUEUE[REAR] = NUM
Step 4: EXIT
```



https://www.geeksforgeeks.org/basic-operations-for-queue-in-data-structure/

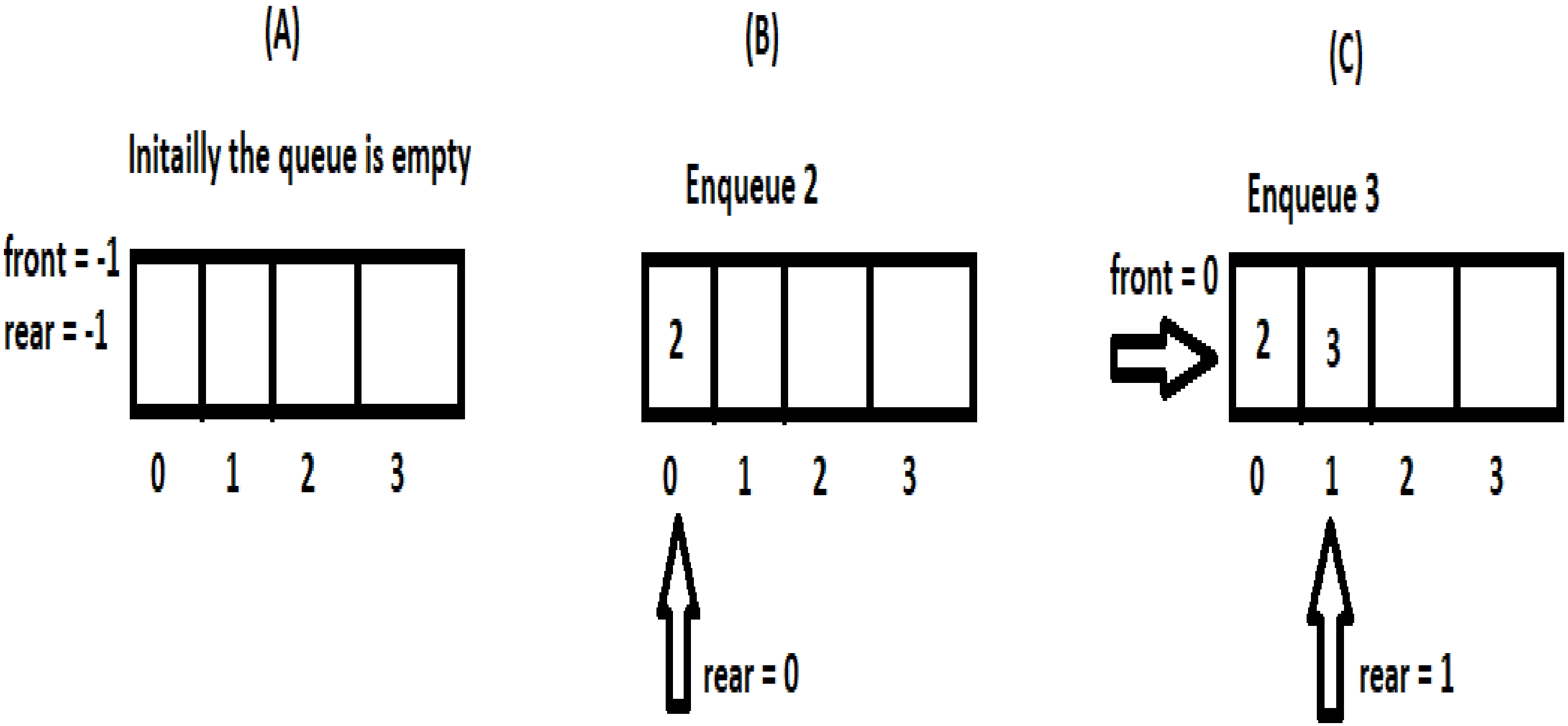
Dequeue Operation

```
Step 1: IF FRONT = -1 or FRONT > REAR
Write UNDERFLOW
ELSE
SET VAL = QUEUE[FRONT]
SET FRONT = FRONT + 1
[END OF IF]
Step 2: EXIT
```



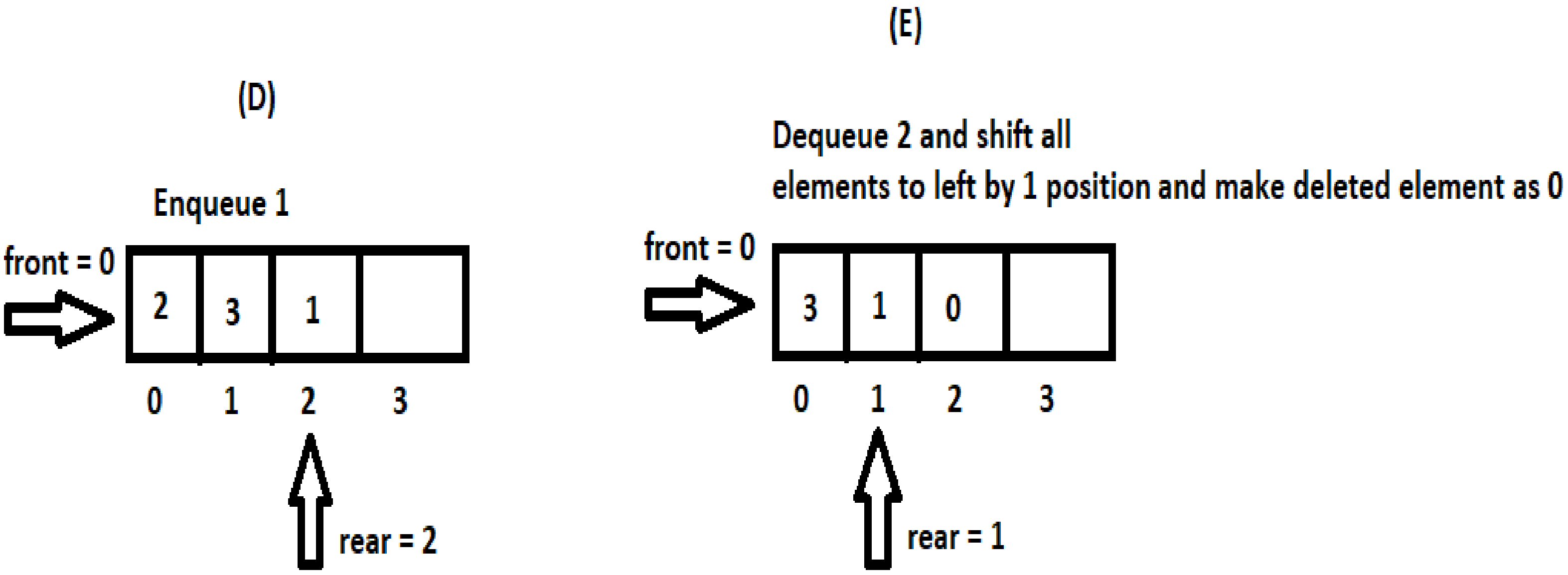
https://www.geeksforgeeks.org/basic-operations-for-queue-in-data-structure/

Implementation



https://hetalrachh.home.blog/2020/02/24/implementing-queue-using-array-in-java/

Implementation



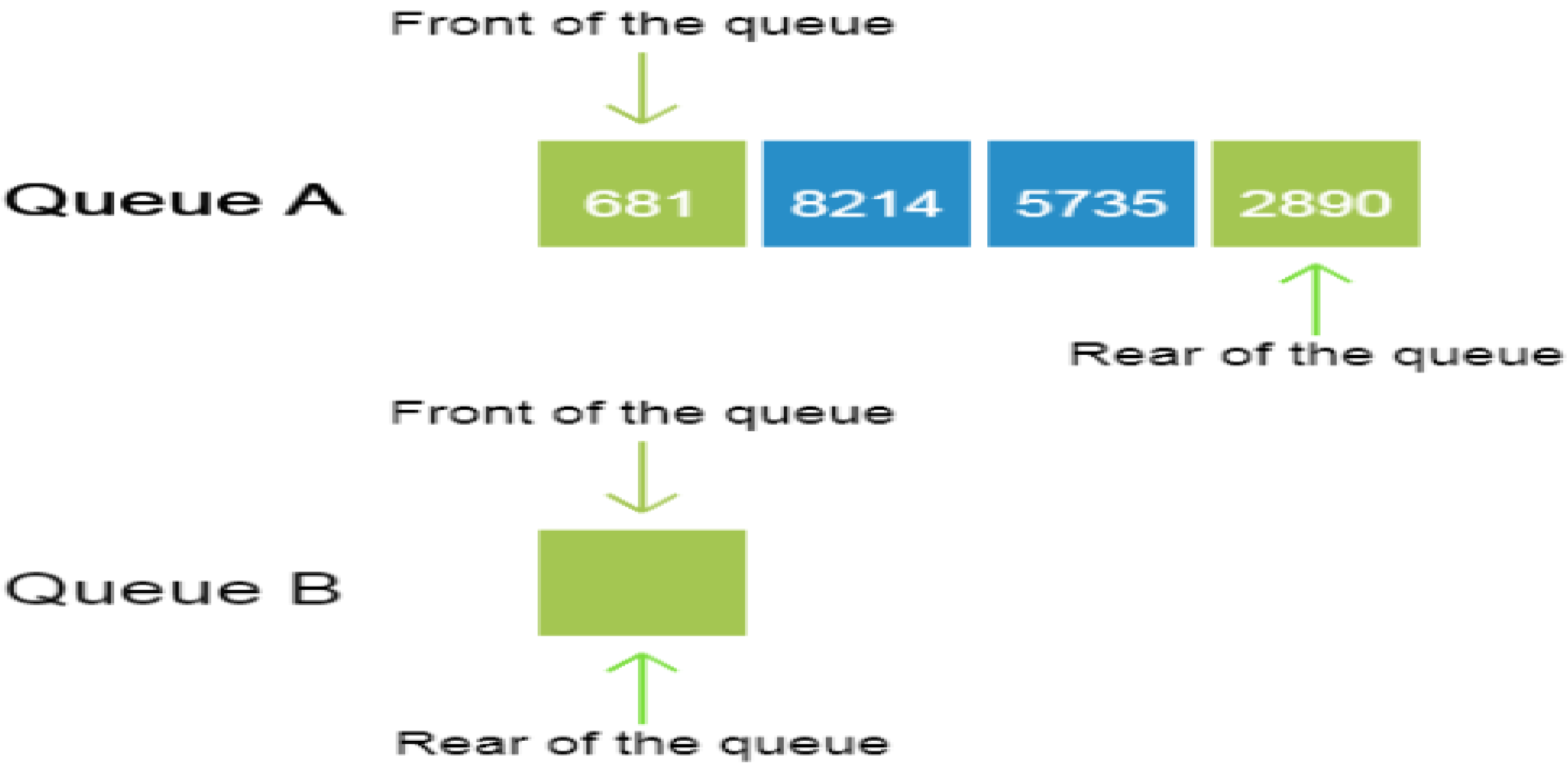
Dequeue 2 and shift all elements to left by 1 position and make deleted element as 0

Implementing queue using array

https://hetalrachh.home.blog/2020/02/24/implementing-queue-using-array-in-java/

Practice

Question: A set of numbers are given in Queue A. With the help of queue B, deque the elements in DESCENDING order.



Code for Queue Operations Using Array

```
#include <iostream>
using namespace std;
int queue[100], n = 100, front = - 1,
rear = - 1;
void Insert() {
   int val;
   if (rear == n - 1)
   cout<<"Queue Overflow"<<endl;
   else {
      if (front == - 1)
      front = 0;
      cout<<"Insert the element in
queue : "<<endl;
      cin>>val;
      rear++;
      queue[rear] = val;
   }
}
void Delete()
 {    if (front == - 1 | | front > rear) {
   cout<<"Queue Underflow ";       return ;
} else {
  cout<<"Element deleted from queue is : "<< queue[front] <<endl;
  front++;;    } }
```

```
void Display()
{    if (front == - 1)    cout<<"Queue is empty"<<endl;
 else {
 cout<<"Queue elements are : ";
    for (int i = front; i <= rear; i++)
  cout<<queue[i]<<" ";
 cout<<endl;    } }
int main()
{    int ch;    cout<<“
1) Insert element to queue"<<endl;    cout<<“
2) Delete element from queue"<<endl;    cout<<
3) Display all the elements of queue"<<endl;
 cout<<“
4) Exit"<<endl;
```

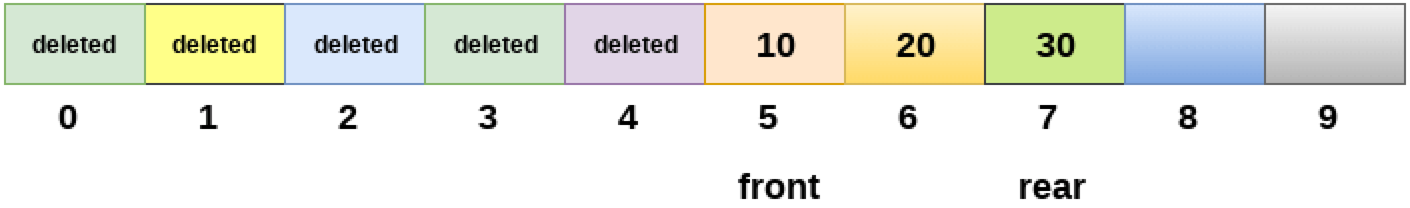
```
do {
 cout<<"Enter your choice : "<<endl;      cin>>ch;
switch (ch)
{
 case 1: Insert();
    break;
case 2: Delete();
 break;
  case 3: Display();
  break;
case 4: cout<<"Exit"<<endl;
break;
 default: cout<<"Invalid choice"<<endl;      }   }
while(ch!=4);    return 0; }
```

Output

```
1)  Insert element to queue
2)  Delete element from queue
3)   Display all the elements of queue
4)   Exit Enter your choice : 1
          Insert the element in queue : 4
          Enter your choice : 1
           Insert the element in queue : 3
           Enter your choice : 1
          Insert the element in queue : 5
          Enter your choice : 2
          Element deleted from queue is : 4
           Enter your choice : 3
          Queue elements are : 3 5
           Enter your choice : 7 Invalid
```

Drawback of array implementation

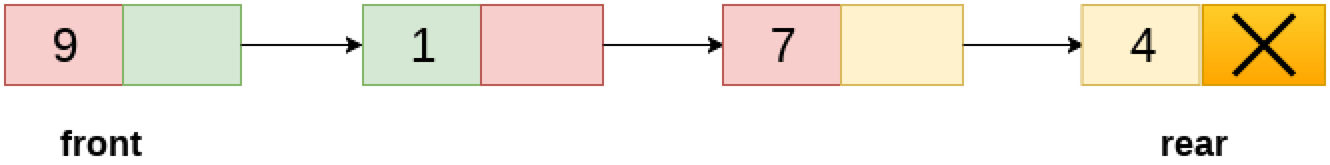
1) Memory wastage
2) Deciding the array size



deleted	deleted	deleted	deleted	deleted	10	20	30		
0	1	2	3	4	5	6	7	8	9
					front		rear		

Implementation of Queue using Linked List

- In a linked queue, each node of the queue consists of two parts i.e. data part and the link part.
- Each element of the queue points to its immediate next element in the memory.



Linked Queue

Operation on Linked Queue

Insert operation

- The new element will be the last element of the queue.
- Firstly, allocate the memory for the new node ptr

```
ptr = (struct node *) malloc (sizeof(struct node));
```

There can be the two scenario of inserting this new node ptr into the linked queue.

1. **Queues is empty**

```
ptr -> data = item;
        if(front == NULL)
        {
            front = ptr;
            rear = ptr;
            front -> next = NULL;
            rear -> next = NULL;
        }
```

Operation on Linked Queue

2. The queue contains more than one element..

```
rear -> next = ptr;
        rear = ptr;
        rear->next = NULL;
```

Algorithm for insert Operation on Linked Queue

```
Step 1: Allocate the space for the new node PTR
Step 2: SET PTR -> DATA = VAL
Step 3: IF FRONT = NULL
            SET FRONT = REAR = PTR
            SET FRONT -> NEXT = REAR -> NEXT = NULL
        ELSE
            SET REAR -> NEXT = PTR
            SET REAR = PTR
            SET REAR -> NEXT = NULL
        [END OF IF]
Step 4: END
```

Delete Operation on Linked Queue

- Deletion operation removes the element that is first inserted among all the queue elements.

```
ptr = front;
        front = front -> next;
        free(ptr);
```

Algorithm for Delete Operation on Linked Queue

```
Step 1: IF FRONT = NULL
Write " Underflow "
Go to Step 5
[END OF IF]
Step 2: SET PTR = FRONT
Step 3: SET FRONT = FRONT -> NEXT
Step 4: FREE PTR
Step 5: END
```

Which operation adds an element to the end of a Queue?

A. Enqueue

B. Dequeue

C. Peek

D. Push

Which operation removes an element from the front of a Queue?

A. Enqueue

B. Dequeue

C. Peek

D. Push

What happens when attempting to enqueue an element into a full Queue in an array-based implementation?

A. The element is added to the end of the Queue, expanding the size of the array.

B. The element is added to the end of the Queue, and the front element is removed.

C. The operation fails, as the Queue is full.

D. The element is added to the beginning of the Queue, shifting all elements one position to the right.

Which of the following data structures is typically used for the underlying implementation of Queues?

A. Arrays

B. Linked Lists

C. Stacks

D. Trees

What are the two primary operations performed on a Queue?

A. Insertion and Deletion

B. Searching and Sorting

C. Push and Pop

D. Enqueue and Dequeue

In an array-based implementation of a linear Queue, where is the front pointer initially positioned?

A. At the beginning of the array

B. At the end of the array

C. At a random position within the array

D. Not applicable, as arrays cannot be used to implement Queues

(Answers)

1. A. Enqueue
2. B. Dequeue
3. C. Peek
4. A. O(1) for both enqueue and dequeue
5. C. The operation fails, as the Queue is full.
6. B. Linked Lists
7. D. Enqueue and Dequeue
8. A. At the beginning of the array

Review

Basic Introduction of Queues: Queues are a fundamental data structure that follows the First In, First Out (FIFO) principle, similar to waiting in line. They manage data sequentially, with elements added to the rear and removed from the front.

Operations of Queues:

Enqueue: Insertion of an element at the rear.

Dequeue: Removal of an element from the front.

Peek: Viewing the front element without removal.

isEmpty, isFull, Size: Checking queue status.

Types of Queues

Implementations of Queues:

Array-based Queue: Simple, fixed size.

Linked List Queue: Dynamic size, efficient insertion/deletion.

THANK
YOU