Chapter 3
QUEUE

Data Structure & Algorithm

Compiled by :

Ankit Bhattarai
Email: ankit.bca@kathford.edu.np

Syllabus

Unit	Contents	Hours	Remarks
1.	Introduction to Data Structure	2	
2.	The Stack	3	
3.	Queue	3	
4.	List	2	
5.	Linked Lists	5	
6.	Recursion	4	
7.	Trees	5	
8.	Sorting	5	
9.	Searching	5	
10.	Graphs	5	
11.	Algorithms	5	

Credit : 3

Outlines

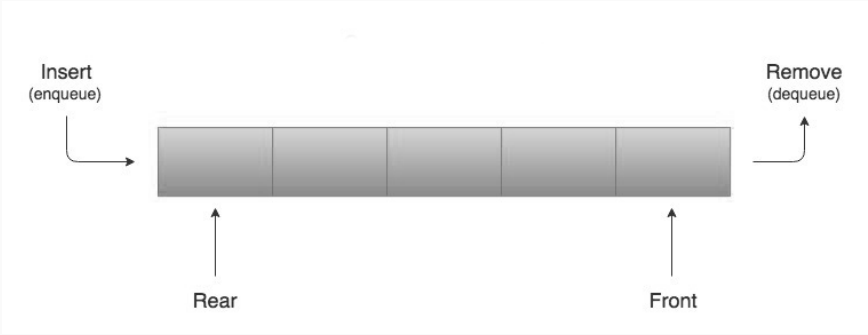
- Introduction
- Queue as an ADT
- Primitive operations in Queue
- Linear and Circular Queue and their applications
- Enqueue and Dequeue
- Priority Queue

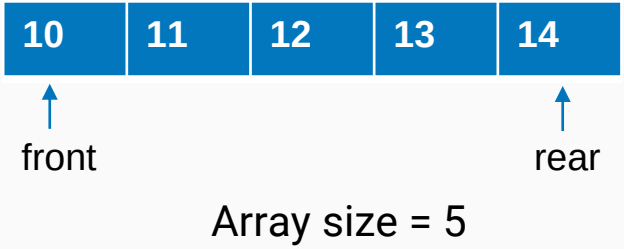
Introduction to QUEUE

- A queue is a linear data structure . It is an ordered collection of items in which items are inserted at one end called **rear** and existing items are deleted at other end called **front**.
- It is also called the First In First Out type of data structure (FIFO), since the first item removed is the first item inserted.
- Scenario in real world:

✓ Printing of files in a network of computers.

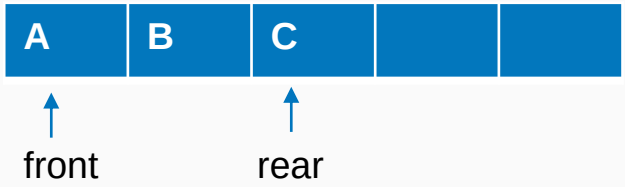
✓ Queues at theatres (movie tickets).





Example of QUEUE

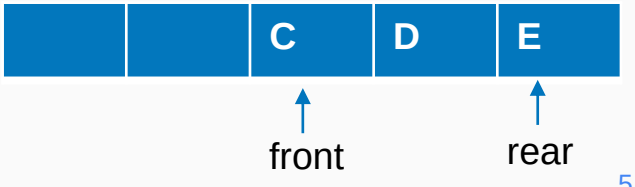
- A queue containing three elements A, B and C with two ends.



Array size = 5

- If D and E are to be inserted then the queue will look like:

- If we want to remove B, it is not possible until A is in the queue, Because B is not the front. So we have to remove A & then B.



QUEUE Applications

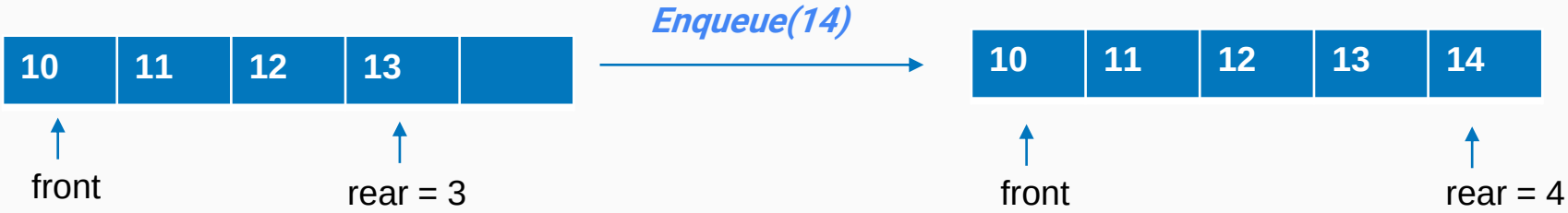
- Queue is most often used in a scenario where there is a shared resource that is supposed to serve some request, but the resource can handle only one request at a time.

- **Applications of Queue:**

1. Print queue: Jobs sent to the printer.
2. Operating System maintains queue in allocating the process to each unit by storing them in buffer.
3. Queue acts as a auxiliary data structure for various algorithms and it is component of other data structures.

Primitive Operations in QUEUE

- **Enqueue** (Insert operation)
  - Inserts an item at the rear of the queue. Other names - Add, Insert
  - The rear of the queue will be now occupied with an item currently added in the queue.
  - Rear count will be incremented by one after addition of new data item.
  - **rear = rear + 1**



Primitive Operations in QUEUE

- **Dequeue** (Delete operation)
  - Deletes an item from the front of the queue. Other names – Delete, Remove
  - Now the front will be incremented by one each time when an item is removed from the queue.
  - front = front + 1

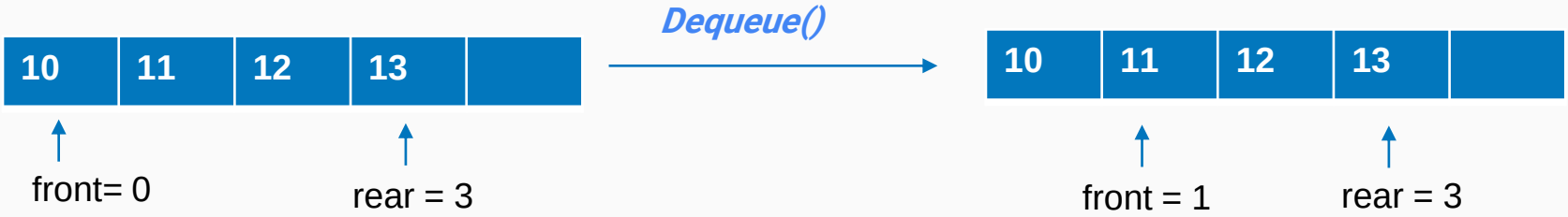
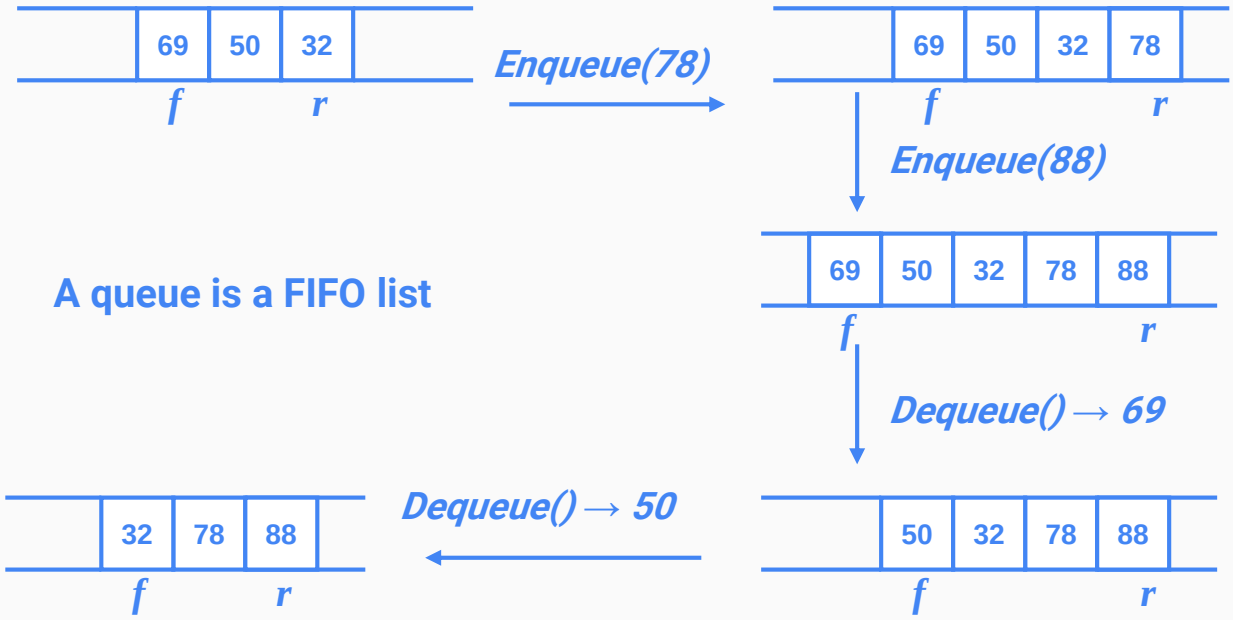


Illustration of Queue Operation



A queue is a FIFO list

QUEUE as an ADT

Values: A queue of elements of type T is a finite sequence of elements of T together with the operations

Operations:

1. **MakeEmpty(Q)** : Create an empty queue Q
2. **Empty(Q)** : Determine if the queue Q is empty or not
3. **Enqueue(Q, x)** : Insert element x at the end of the queue Q
4. **Dequeue(Q)** : If the queue Q is not empty, remove the element at the front of the queue
5. **Front(Q)** : Retrieve the element at the front of the queue Q, without deleting it
6. **Full(Q)** : Checks if the queue is full or not.

Implementation of Queue

1. **Static Implementation Using Array**

- Linear array: A linear array, is a list of finite numbers of elements stored in the memory. In a linear array, we can store only homogeneous data elements.

- Circular array : An array is called circular if we consider the first element as next of the last element.

2. **Dynamic Implementation using Linked List** (Chapter 5)

Types of Queue

i. Linear Queue
ii. Circular Queue
iii. Priority Queue
iv. DEQUE (Double Ended Queue)

i. Linear Queue:

A linear queue is a linear data structure that serves the request first, which has been arrived first. It consists of data elements which are connected in a linear fashion.

Algorithm for enqueue operation in Linear queue

Step 1: Start

Step 2: Check if the queue is full or not

if the queue is full write " Queue overflow or queue is full"

and go to step 5

Step 3: If the queue is not full increment

rear ← rear + 1

to the next empty space.

Step 4: Add the data item in the queue where the rear is pointing.

queue[rear] ← item

Step 5: Stop

Algorithm for dequeue operation in Linear queue

Step 1: Start

Step 2: Check if the queue is empty or not

if the queue is full write “ Queue underflow or queue is empty”

and go to step 5

Step 3: If the queue is not empty access the data item where front is pointing

queue[front] = item

Step 4: Increment the front pointer to the next data items.

front = front + 1

Step 5: Stop

THANK YOU

Any Queries ?

ankit.bca@kathford.edu.np