Data Structures and Algorithms

Rao Muhammad Umer
Lecturer,
CS and IT Department,
The University of Lahore.
Web: raoumer.github.io

outline

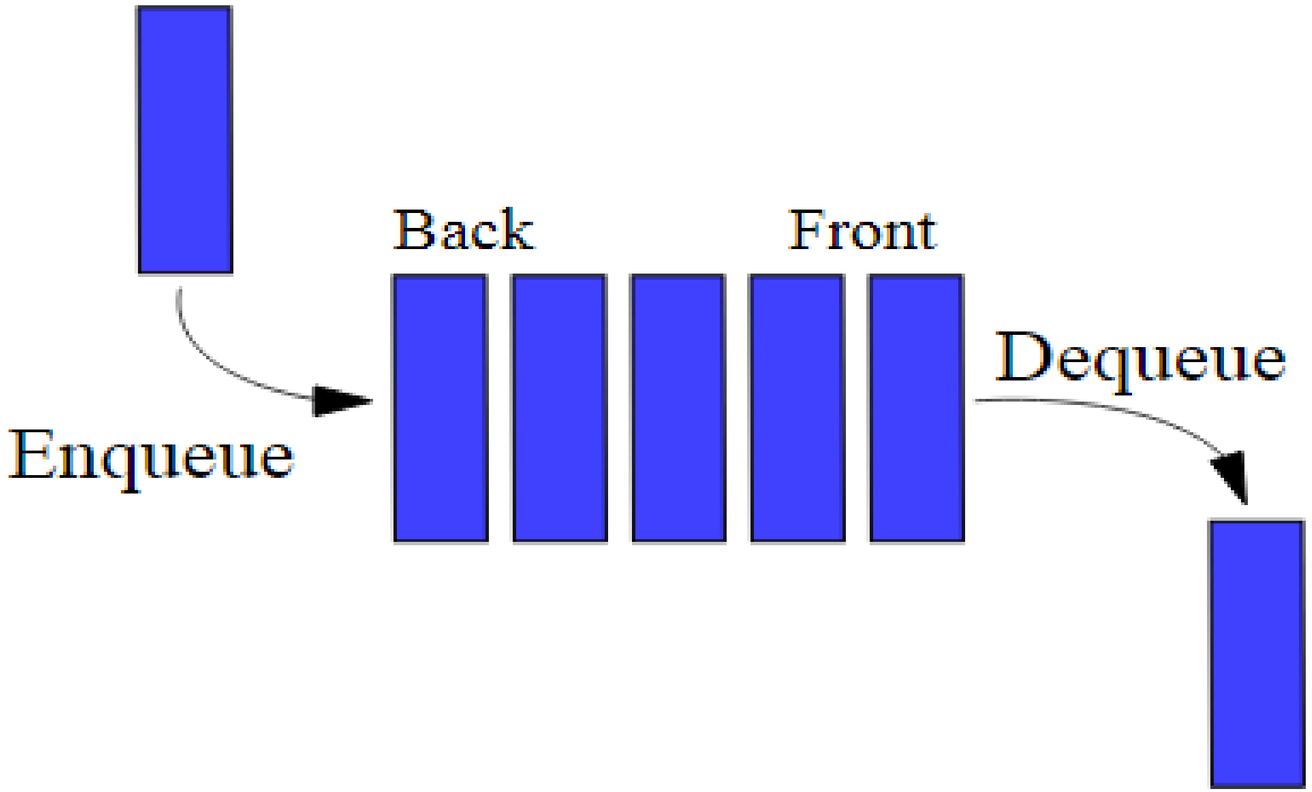
Queues

- What are Queues?
- Operations on Queue
  - Insertion
  - Deletion
- Queue as an Array
- Queue as a Linked List
- Circular Queues
- Deque (Double-ended Queue)
- Priority Queue
- Applications of Queues

Queue

- **Queue** is a linear data structure that permits insertion of new element at one end and deletion of an element at the other end.
- Deletion of element from front
- Insertion of element from rear
- ADT (Abstract Data Type)
- Example:
  - A queue of persons paying bill in bank
  - A queue of people waiting at a bus stop
- Also known as **FIFO (First In First Out)**.

Pictorial Representation of Queue



https://en.wikipedia.org/wiki/Queue_(abstract_data_type)

Operations on Queue

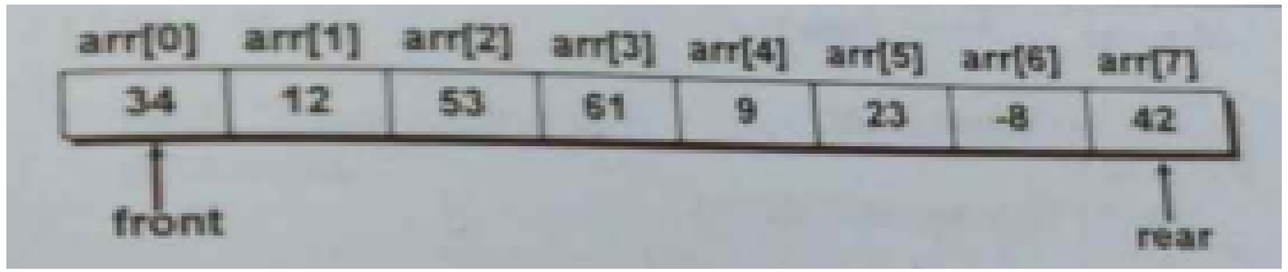
- **Insert:** Allows adding an element at the rear of the queue.
- **Delete:** Allows to remove an element from the front of the queue.

- See Animation for working of a queue

Representation of Queue

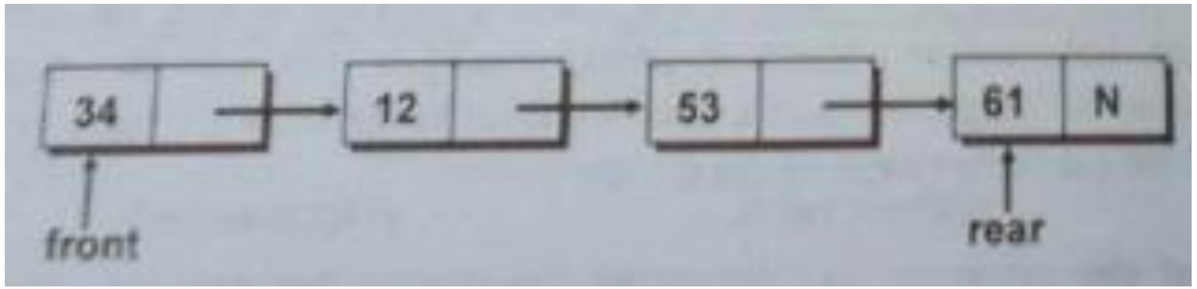
- Queue as an Array
- Queue as a Linked List

Queue as an Array



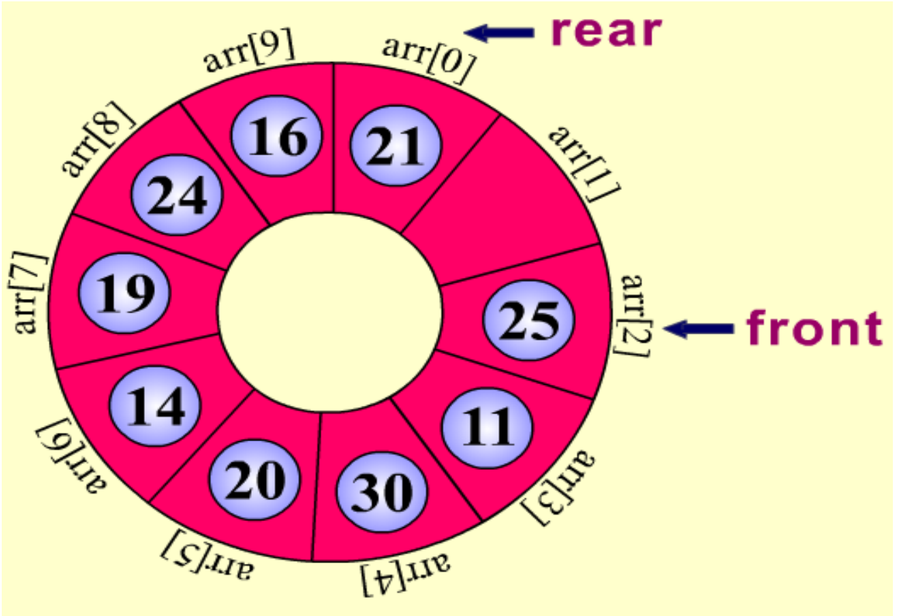
- See source code in C++ for Queue as an Array

Queue as a Linked List



- See source code in C++ for Queue as a Linked List

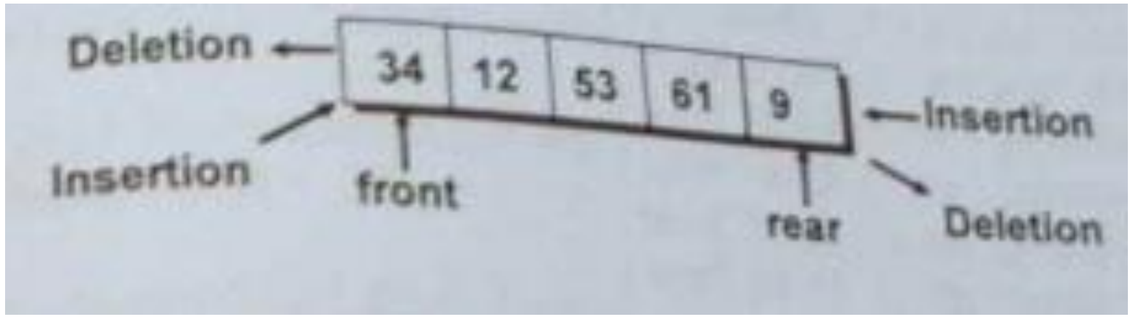
Circular Queue



- See Animation of working of circular queue
- See source code in C++ for circular queue

Deque (Double-ended Queue)

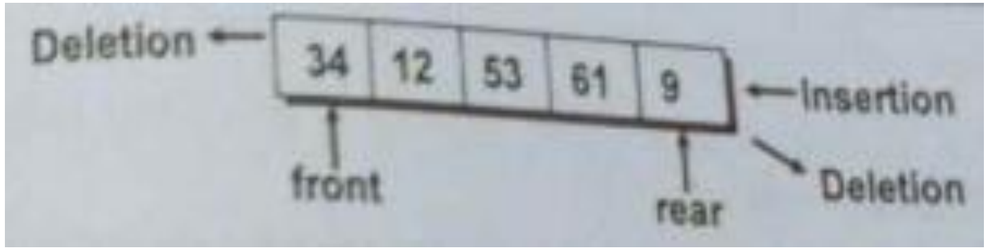
- Insertion and deletion at either the front or rear end
- Generalization of both a stack and a queue



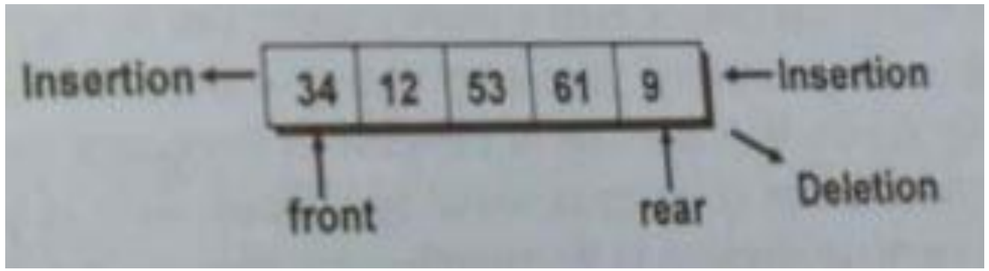
- See Animation of working of deque
- See source code in C++ for deque

Deque

- Two Types of Deque:
  - **Input-restricted deque**
    - restricts the insertion of elements at one end only, but deletion of elements can be done at both the ends.
    - No **addqatbeg()** function
    - See Animation of working of input-restricted deque



  - **Output-restricted deque**
    - restricts the deletion of elements at one end only, and allows insertion to be done at both the ends of a deque.
    - No **delqatbeg()** function
    - See Animation of working of output-restricted deque



Priority Queue

- See Animation of working of priority queue
- See source code in C++ for priority queue

Applications of Queues

- **Bandwidth management**:
  - A router is connected to a line with limited bandwidth. If there is insufficient bandwidth, the router maintains a queue for incoming data such that the most important data will get forwarded first as bandwidth becomes available.

- **Printing**:
  - A shared server has a list of print jobs to print. It wants to print them in chronological order, but each print job also has a *priority*, and higher-priority jobs always print before lower-priority jobs

- **Algorithms**:
  - Writing a ghost AI algorithm for Pac-Man. It needs to search for the best path to find Pac-Man; it will enqueue all possible paths with priorities (based on guesses about which one will succeed), and try them in order.

Acknowledgement

- Mostly Slides taken from Book: **“Data Structures through C++”** by Yashavant P. Kanetkar
- https://en.wikibooks.org/wiki/Data_Structures/Stacks_and_Queues