Sorting

- arrangement of data/element in ascending or descending order of their values
- Sorting Algorithms
  - Basic
    1. selection sort
    2. bubble sort
    3. insertion sort
  - Advanced
    - merge sort
    - quick sort
    - heap sort

Selection sort

- Algorithm
  - Find the minimum element in an array A[i -> n-1] and place it at beginning
    - where n – size of array and i – 0, 1, 2, ...n-2
  - Repeat the above procedure n – 1 times where n is size of array
- Algorithm
  1. select one element from array(from left)
  2. compare selected element with all other elements
  3. if selected element is greater than other element then swap both
  4. if selected element is less or equal than other element then do nothing
  5. repeat step 1 to 4, N-1 times
- No of passes = N - 1
- Time Complexity
  - Best/Average/Worst case - O(n^2)

Bubble sort

- Algorithm
  - Find the maximum element from two consecutive elements of an array A[i -> n-i-1] and place it at second location
    - where n – size of array and i – 0, 1, 2, ...n-2
  - Repeat the above procedure n – 1 times where n is size of array
- Algorithm
  1. Compare two consecutive elements
  2. if left element is greater than right element, then swap both
  3. repeat step 1 and 2 for all consecutive elements of array
  4. repeat step 1 to 3, N-1 times

- No of passes = N - 1
- Time Complexity
  - Best case - O(n)
  - Average/Worst case - O(n^2)

Insertion sort

- Algorithm
  1. Take backup of one element from array
  2. Compare it with all its left neighbours
  3. If left neighbour is greater than backuped element, move it one position ahead
  4. If left neighbour is less than backuped element, insert backuped element at next position
  5. repeat step 1 to 4, N-1 times
- No of passes = N - 1
- Time Complexity
  - Best case - O(n)
  - Average/Worst case - O(n^2)

Queue

- Queue is First-In-First-Out structure.
- Queue Operations:
  - enqueue()
  - dequeue()
  - peek()
  - is_empty()
  - is_full()
- Types of queue:
  - Linear Queue
  - Circular Queue
  - Deque
  - Priority Queue
- In linear queue (using array) when rear reaches last index, further elements cannot be added, even If space is available due to deletion of elements from front. Thus space utilization is poor.
- Circular queue allows adding elements at the start of array if rear reaches last index and space is free at the start of the array.
- Thus rear and front can be incremented in circular fashion i.e. 0, 1, 2, 3, ..., n-1, 0, 1, ...n-1. So they are said to be circular queue.

- However queue full and empty conditions become tricky.
- Time Complexities
    - Push - O(n)
    - Pop - O(n)
    - Peek - O(n)
- Applications
    - Jobs submitted to printer
    - In Network setups – file access of file server machine is given to First come First serve basis
    - Calls are placed on a queue when all operators are busy
    - Used in advanced data structures to give efficiency.
    - Process waiting queues in OS
    - etc