

# Pre-Placements checklist

## Data Structures:

1. Array
  - a. Kaden's Algorithm
  - b. N/2, N/3 greatest Number
  - c. Merge overlapping intervals
  - d. Rotate matrix
  - e. Buy / Sell stocks - I, II, III:  
<https://leetcode.com/problems/best-time-to-buy-and-sell-stock/>
2. String
  - a. Pattern matching algorithms (KMP + Rabin Karp)
  - b. Using StringBuilder class -> Add, Multiply Strings
  - c. String compression algorithm
3. LinkedList
  - a. Implementation of Linkedlist
  - b. Detect cycle in a linkedlist - Floyd Algo
  - c. Reverse a linkedlist + reverse in groups
4. Stack
  - a. Implementation of Stack
  - b. Balance parenthesis
  - c. Trapping rain water
  - d. Implement min stack
5. Queue
  - a. Implementation of Queue + Deque
  - b. Sliding window maximum
  - c. Implement BFS
  - d. Implement Level order in Binary tree
6. PriorityQueue or Heap

- a. Implementation of Heap Data structure
  - b. Connect n ropes with min cost:  
<https://www.geeksforgeeks.org/connect-n-ropes-minimum-cost/>
  - c. Median of running stream:  
<https://www.geeksforgeeks.org/median-of-stream-of-running-integers-using-stl/>
  - d. LRU and LFU cache
7. Set & Map
- a. Internal working of HashMap
  - b. 4-sum
  - c. Longest substring without repeat:  
<https://www.interviewbit.com/problems/longest-substring-without-repeat/>
8. Binary Tree
- a. Implementation: insert, delete, traverse:  
<https://youtu.be/QhIM-G7FAow>
  - b. Print top level, left level, right level, level order, zig-zag traversal of Binary tree
  - c. Invert a binary tree:  
<https://leetcode.com/problems/invert-binary-tree/>
  - d. Lowest common ancestor
9. Binary Search Tree
- a. Implementation
  - b. Check if a tree is BST or not
  - c. AVL tree and rotation
10. Graph
- a. Implementation, BFS and DFS traversals
  - b. Topological sorting
  - c. Bellman ford Algorithm
  - d. Dijkstra's Algorithm
  - e. Prim's Algorithm

- f. Kruskal's Algorithm
  - g. Unique Islands Problem:  
<https://www.geeksforgeeks.org/find-the-number-of-distinct-islands-in-a-2d-matrix/>
  - 11. Trie
    - a. Implementation
  - 12. Segment Trees : More important in CP
    - a. Implementation
- 

## Algorithms:

- 1. Two pointers Algorithm
  - a. 3-Sum
  - b. Container with most water
  - c. Sort the array containing only 0, 1 and 2
- 2. Math
  - a. Fast Power: <https://www.youtube.com/watch?v=dyrRM8dTEus>
  - b. Euclid GCD
  - c. Sieve of Eratosthenes
- 3. Recursion + Backtracking
  - a. Sudoku solver
  - b. N-Queens Problem
  - c. Permutation and Combinations (Bruteforce)
- 4. Bits Manipulation + Mathematics
  - a. Find one non-repeating number, find two
  - b. Count 1 bits in a number
- 5. Divide & Conquer
  - a. Merge Sort
  - b. Median of two sorted arrays

## 6. Binary Searching

- a. Find upper and lower bound using Binary search
- b. Allocate books:

<https://www.interviewbit.com/problems/allocate-books/>

## 7. Greedy Programming

- a. Candy distribution:

<https://www.interviewbit.com/problems/distribute-candy/>

- b. Gas station: <https://www.interviewbit.com/problems/gas-station/>
- c. Fractional Knapsack

## 8. Dynamic Programming

- a. 0/1 Knapsack: <https://www.youtube.com/watch?v=y6kpGJBI7t0>
- b. Longest increasing subsequence
- c. Matrix chain multiplication
- d. Coin change problem

---

# Operating System:

- 1. Basics of Threads
  - 2. Process scheduling algorithms
  - 3. Critical section Problem
  - 4. Deadlock
  - 5. Memory management
    - a. Paging
    - b. Segmentation
  - 6. Page replacement algorithms
  - 7. Disk scheduling algorithms
-

## DBMS:

1. Types of Keys: Candidate, Super, Foreign keys
  2. Normal Forms
  3. Joins
  4. SQL queries
  5. ACID properties
  6. Indexing: B trees, B+ trees concepts
- 

## System design:

1. Low level design
  - a. Class, ER diagrams
  - b. OOPS concepts
  - c. Design Elevator system, Parking Lot, MakeMyTrip System
2. High level design
  - a. Scaling
  - b. Distributed systems
  - c. Microservice and Monolithic architecture
  - d. Load balancing
  - e. Message queue
  - f. Design Whatsapp, Tinder, Uber system