Pre-Placements Checklist

Data Structures:

1. Array
    a. Kadane's Algorithm
    https://www.geeksforgeeks.org/largest-sum-contiguous-subarray/
    b. N/2, N/3 greatest Number
    https://leetcode.com/problems/majority-element/
    https://leetcode.com/problems/majority-element-ii/
    https://www.geeksforgeeks.org/given-an-array-of-of-size-n-finds-all-the-elements-that-appear-more-than-nk-times/
    c. Merge overlapping intervals
    https://leetcode.com/problems/merge-intervals/
    d. Rotate matrix
    https://leetcode.com/problems/rotate-image/
    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)
    https://www.geeksforgeeks.org/kmp-algorithm-for-pattern-searching/
    https://www.geeksforgeeks.org/rabin-karp-algorithm-for-pattern-searching/
    b. Using StringBuilder class -> Add, Multiply Strings
    https://www.geeksforgeeks.org/stringbuilder-class-in-java-with-examples/
    https://www.geeksforgeeks.org/stringbuilder-append-method-in-java-with-examples/

c. String compression algorithm
https://leetcode.com/problems/string-compression/

3. LinkedList
    a. Implementation of Linkedlist
    https://www.geeksforgeeks.org/implementing-a-linked-list-in-java-using-class/
    https://leetcode.com/problems/design-linked-list/
    b. Detect cycle in a linkedlist - Floyd Algo
    https://leetcode.com/problems/linked-list-cycle/
    c. Reverse a linked list + reverse in groups
    https://leetcode.com/problems/reverse-linked-list/
    https://leetcode.com/problems/reverse-nodes-in-k-group/
4. Stack
    a. Implementation of Stack
    https://www.geeksforgeeks.org/stack-data-structure-introduction-program/
    https://www.geeksforgeeks.org/stack-class-in-java/
    b. Balance parenthesis
    https://leetcode.com/problems/valid-parentheses/
    c. Trapping rain water
    https://leetcode.com/problems/trapping-rain-water/
    d. Implement min stack
    https://leetcode.com/problems/min-stack/

5. Queue
    a. Implementation of Queue + Deque
    https://www.geeksforgeeks.org/queue-set-1introduction-and-array-implementation/
    https://www.geeksforgeeks.org/queue-interface-java/
    https://www.geeksforgeeks.org/implementation-deque-using-circular-array/
    https://www.geeksforgeeks.org/deque-interface-java-example/

    b. Sliding window maximum
    https://leetcode.com/problems/sliding-window-maximum/
    c. Implement BFS
    https://www.geeksforgeeks.org/breadth-first-search-or-bfs-for-a-graph/
    d. Implement Level order in Binary tree
    https://leetcode.com/problems/binary-tree-level-order-traversal/

6. PriorityQueue or Heap
    a. Implementation of Heap Data structure
    https://www.geeksforgeeks.org/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
    https://leetcode.com/problems/lru-cache/
    https://leetcode.com/problems/lfu-cache/

7. Set & Map

a. Internal working of HashMap
https://www.geeksforgeeks.org/internal-working-of-hashmap-java/
b. 4-sum
https://leetcode.com/problems/4sum/
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 view, left view, right view, bottom view, level order, zig-zag traversal of Binary tree
    https://www.geeksforgeeks.org/print-nodes-top-view-binary-tree/
    https://www.geeksforgeeks.org/print-left-view-binary-tree/
    https://leetcode.com/problems/binary-tree-right-side-view/
    https://www.geeksforgeeks.org/bottom-view-binary-tree/
    https://www.geeksforgeeks.org/level-order-tree-traversal/
    https://leetcode.com/problems/binary-tree-zigzag-level-order-traversal/
    c. Invert a binary tree: https://leetcode.com/problems/invert-binary-tree/
    d. Lowest common ancestor
    https://leetcode.com/problems/lowest-common-ancestor-of-a-binary-tree/

9. Binary Search Tree
    a. Implementation
    https://www.geeksforgeeks.org/binary-search-tree-set-1-search-and-insertion/

    b. Check if a tree is BST or not

https://www.geeksforgeeks.org/a-program-to-check-if-a-binary-tree-is-bst-or-not/

c. AVL tree and rotation
https://www.geeksforgeeks.org/avl-tree-set-1-insertion/
https://www.geeksforgeeks.org/avl-tree-set-2-deletion/

10. Graph
a. Implementation, BFS and DFS traversals
https://www.geeksforgeeks.org/graph-and-its-representations/
https://www.geeksforgeeks.org/breadth-first-search-or-bfs-for-a-graph/
https://www.geeksforgeeks.org/depth-first-search-or-dfs-for-a-graph/
b. Topological sorting
https://www.geeksforgeeks.org/topological-sorting/
c. Bellman ford Algorithm
https://www.geeksforgeeks.org/bellman-ford-algorithm-dp-23/
d. Dijkstra's Algorithm
https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-greedy-algo-7/
e. Prim's Algorithm
https://www.geeksforgeeks.org/prims-minimum-spanning-tree-mst-greedy-algo-5/
f. Kruskal's Algorithm
https://www.geeksforgeeks.org/kruskals-minimum-spanning-tree-algorithm-greedy-algo-2/
g. Unique Islands Problem: https://www.geeksforgeeks.org/find-the-number-of-distinct-islands-in-a-2d-matrix/

11. Trie
a. Implementation

https://www.geeksforgeeks.org/trie-insert-and-search/

12. Segment Trees : More important in CP
    a. Implementation https://www.hackerearth.com/practice/data-structures/advanced-data-structures/segment-trees/tutorial/

Algorithms:

1. Two pointers Algorithm
    a. 3-Sum https://leetcode.com/problems/3sum/
    b. Container with most water https://leetcode.com/problems/container-with-most-water/
    c. Sort the array containing only 0, 1 and 2 https://www.geeksforgeeks.org/sort-an-array-of-0s-1s-and-2s/
2. Math
    a. Fast Power: https://www.youtube.com/watch?v=dyrRM8dTEus
    b. Euclid GCD: https://www.geeksforgeeks.org/euclidean-algorithms-basic-and-extended/
    c. Sieve of Eratosthenes: https://www.geeksforgeeks.org/sieve-of-eratosthenes/
3. Recursion + Backtracking
    a. Sudoku solver https://leetcode.com/problems/sudoku-solver/
    b. N-Queens Problem https://leetcode.com/problems/n-queens/
    c. Permutation and Combinations (Bruteforce) https://www.geeksforgeeks.org/permutation-and-combination/
4. Bits Manipulation + Mathematics

    a. Find one non-repeating number, find two
       https://www.geeksforgeeks.org/non-repeating-element/
       https://www.geeksforgeeks.org/find-two-non-repeating-elements-in-an-array-of-repeating-elements/
    b. Count 1 bits in a number
       https://leetcode.com/problems/number-of-1-bits/
5. Divide & Conquer
    a. Merge Sort
       https://www.geeksforgeeks.org/merge-sort/
    b. Median of two sorted arrays
       https://leetcode.com/problems/median-of-two-sorted-arrays/

6. Binary Searching
    a. Find upper and lower bound using Binary search
       https://www.geeksforgeeks.org/find-first-and-last-positions-of-an-element-in-a-sorted-array/
    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
       https://www.geeksforgeeks.org/fractional-knapsack-problem/
8. Dynamic Programming
    a. 0/1 Knapsack: https://www.youtube.com/watch?v=y6kpGJBI7t0
    b. Longest increasing subsequence
       https://leetcode.com/problems/longest-increasing-subsequence/
    c. Matrix chain multiplication
       https://www.geeksforgeeks.org/matrix-chain-multiplication-dp-8/
    d. Coin change problem

https://leetcode.com/problems/coin-change/

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, Foriengn 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