

DSA QUESTIONS

Array:

Easy:

1. Two Sum - [Link](#)
2. Remove Duplicates from Sorted Array - [Link](#)
3. Best Time to Buy and Sell Stock - [Link](#)
4. Plus, One - [Link](#)
5. Missing Number - [Link](#)
6. Maximum Subarray - [Link](#)
7. Move Zeroes - [Link](#)
8. Contains Duplicate - [Link](#)
9. Intersection of Two Arrays II - [Link](#)
10. Rotate Array - [Link](#)
11. Third Maximum Number - [Link](#)
12. Valid Palindrome - [Link](#)
13. Merge Sorted Array - [Link](#)
14. Maximum Product Subarray - [Link](#)
15. Minimum Size Subarray Sum - [Link](#)

Medium:

16. Product of Array Except Self - [Link](#)
17. Container With Most Water - [Link](#)
18. Search in Rotated Sorted Array - [Link](#)
19. Combination Sum - [Link](#)
20. Next Permutation - [Link](#)
21. Find First and Last Position of Element in Sorted Array - [Link](#)

- 22. 3Sum - [Link](#)
- 23. Spiral Matrix - [Link](#)
- 24. Merge Intervals - [Link](#)
- 25. Jump Game - [Link](#)
- 26. Set Matrix Zeroes - [Link](#)
- 27. Group Anagrams - [Link](#)
- 28. Word Search - [Link](#)
- 29. First Missing Positive - [Link](#)
- 30. Find Peak Element - [Link](#)

Hard:

- 31. Trapping Rain Water - [Link](#)
- 32. Best Time to Buy and Sell Stock III - [Link](#)
- 33. First Missing Positive - [Link](#)
- 34. Median of Two Sorted Arrays - [Link](#)
- 35. Jump Game II - [Link](#)
- 36. Longest Consecutive Sequence - [Link](#)
- 37. Minimum Window Substring - [Link](#)
- 38. Gas Station - [Link](#)
- 39. Meeting Rooms II - [Link](#)
- 40. Best Time to Buy and Sell Stock IV - [Link](#)

Searching:

Easy:

- 41. First Bad Version - [Link](#)
- 42. Search Insert Position - [Link](#)
- 43. Guess Number Higher or Lower - [Link](#)
- 44. Peak Index in a Mountain Array - [Link](#)
- 45. Search a 2D Matrix II - [Link](#)
- 46. Find Minimum in Rotated Sorted Array - [Link](#)

- 47. Find K Closest Elements - [Link](#)
- 48. First Missing Positive - [Link](#)
- 49. Search in Rotated Sorted Array - [Link](#)
- 50. Search a 2D Matrix - [Link](#)

Medium:

- 51. Divide Two Integers - [Link](#)
- 52. Median of Two Sorted Arrays - [Link](#)
- 53. Find Peak Element - [Link](#)
- 54. Find Kth Smallest Element in a Sorted Matrix - [Link](#)
- 55. Search in Rotated Sorted Array II - [Link](#)
- 56. Find First and Last Position of Element in Sorted Array - [Link](#)
- 57. First Missing Positive - [Link](#)
- 58. Search a 2D Matrix II - [Link](#)
- 59. Search Insert Position - [Link](#)
- 60. Search in Rotated Sorted Array - [Link](#)

Hard:

- 61. Search in Rotated Sorted Array II - [Link](#)
- 62. Median of Two Sorted Arrays - [Link](#)
- 63. Search a 2D Matrix II - [Link](#)
- 64. Find Minimum in Rotated Sorted Array - [Link](#)
- 65. Search in Rotated Sorted Array - [Link](#)
- 66. First Missing Positive - [Link](#)
- 67. Search Insert Position - [Link](#)
- 68. Find Peak Element - [Link](#)
- 69. First Bad Version - [Link](#)
- 70. Find First and Last Position of Element in Sorted Array - [Link](#)

Recursion:

Easy:

- 71. Climbing Stairs - [Link](#)
- 72. Fibonacci Number - [Link](#)
- 73. Reverse String - [Link](#)
- 74. Pow(x, n) - [Link](#)
- 75. Merge Two Sorted Lists - [Link](#)
- 76. Maximum Depth of Binary Tree - [Link](#)
- 77. Symmetric Tree - [Link](#)
- 78. Path Sum - [Link](#)
- 79. Subsets - [Link](#)
- 80. Generate Parentheses - [Link](#)

Medium:

- 81. Permutations - [Link](#)
- 82. Combinations - [Link](#)
- 83. Palindrome Partitioning - [Link](#)
- 84. Expression Add Operators - [Link](#)
- 85. Word Search - [Link](#)
- 86. Unique Paths - [Link](#)
- 87. Letter Combinations of a Phone Number - [Link](#)
- 88. Generate Parentheses - [Link](#)
- 89. Pow(x, n) - [Link](#)
- 90. Reverse Linked List - [Link](#)

Hard:

- 91. Regular Expression Matching - [Link](#)
- 92. Palindrome Partitioning II - [Link](#)
- 93. Expression Add Operators - [Link](#)
- 94. Word Search II - [Link](#)
- 95. Wildcard Matching - [Link](#)
- 96. Unique Paths III - [Link](#)
- 97. Decode Ways - [Link](#)
- 98. Regular Expression Matching - [Link](#)
- 99. Palindrome Partitioning II - [Link](#)
- 100. Expression Add Operators - [Link](#)

String:

Easy:

- Reverse String - [Link](#)
- Valid Anagram - [Link](#)
- Valid Palindrome - [Link](#)
- String to Integer (atoi) - [Link](#)
- Implement strStr() - [Link](#)
- Count and Say - [Link](#)
- First Unique Character in a String - [Link](#)
- Valid Parentheses - [Link](#)
- Longest Substring Without Repeating Characters - [Link](#)
- Longest Common Prefix - [Link](#)

Medium:

- Group Anagrams - [Link](#)
- Longest Palindromic Substring - [Link](#)
- ZigZag Conversion - [Link](#)
- Regular Expression Matching - [Link](#)

- Longest Valid Parentheses - [Link](#)
- Implement strStr() - [Link](#)
- String to Integer (atoi) - [Link](#)
- Palindrome Partitioning - [Link](#)
- Wildcard Matching - [Link](#)
- Valid Parentheses - [Link](#)

Hard:

- Regular Expression Matching - [Link](#)
- Longest Substring with At Least K Repeating Characters - [Link](#)
- Distinct Subsequences - [Link](#)
- Longest Palindromic Substring - [Link](#)
- Encode and Decode Strings - [Link](#)
- Palindrome Partitioning II - [Link](#)
- Text Justification - [Link](#)
- Longest Valid Parentheses - [Link](#)
- Longest Common Prefix - [Link](#)
- Minimum Window Substring - [Link](#)

Stack:

Easy:

- Min Stack - [Link](#)
- Valid Parentheses - [Link](#)
- Remove Outermost Parentheses - [Link](#)
- Implement Queue using Stacks - [Link](#)
- Implement Stack using Queues - [Link](#)
- Valid Parentheses - [Link](#)
- Remove Outermost Parentheses - [Link](#)
- Min Stack - [Link](#)

- Remove K Digits - [Link](#)
- Baseball Game - [Link](#)

Medium:

- Evaluate Reverse Polish Notation - [Link](#)
- Basic Calculator II - [Link](#)
- Daily Temperatures - [Link](#)
- Trapping Rain Water - [Link](#)
- Remove Duplicate Letters - [Link](#)
- Valid Parentheses - [Link](#)
- Longest Valid Parentheses - [Link](#)
- Simplify Path - [Link](#)
- Basic Calculator - [Link](#)
- Largest Rectangle in Histogram - [Link](#)

Queue:

Easy:

- Implement Stack using Queues - [Link](#)
- Design Circular Queue - [Link](#)
- Design Circular Deque - [Link](#)
- Implement Queue using Stacks - [Link](#)
- First Unique Character in a String - [Link](#)
- Design HashSet - [Link](#)
- Design HashMap - [Link](#)
- Implement Trie (Prefix Tree) - [Link](#)
- Implement Trie (Prefix Tree) - [Link](#)

- Sliding Window Maximum - [Link](#)

Medium:

- Shortest Subarray with Sum at Least K - [Link](#)
- Find K Pairs with Smallest Sums - [Link](#)
- Design Twitter - [Link](#)
- Number of Recent Calls - [Link](#)
- Implement Stack using Queues - [Link](#)
- Implement Queue using Stacks - [Link](#)
- First Unique Number - [Link](#)
- Sliding Window Median - [Link](#)
- Task Scheduler - [Link](#)
- Find the Most Competitive Subsequence - [Link](#)

Linked List:

Easy:

- Reverse Linked List - [Link](#)
- Middle of the Linked List - [Link](#)
- Merge Two Sorted Lists - [Link](#)
- Remove Nth Node From End of List - [Link](#)
- Palindrome Linked List - [Link](#)
- Intersection of Two Linked Lists - [Link](#)
- Design Linked List - [Link](#)
- Linked List Cycle - [Link](#)
- Flatten a Multilevel Doubly Linked List - [Link](#)

- Reverse Nodes in k-Group - [Link](#)

Medium:

- Add Two Numbers - [Link](#)
- Odd Even Linked List - [Link](#)
- Remove Linked List Elements - [Link](#)
- Copy List with Random Pointer - [Link](#)
- Swap Nodes in Pairs - [Link](#)
- Merge k Sorted Lists - [Link](#)
- LRU Cache - [Link](#)
- Palindrome Linked List - [Link](#)
- Design Linked List - [Link](#)
- Split Linked List in Parts - [Link](#)

Hard:

- Reverse Nodes in k-Group - [Link](#)
- LRUCache - [Link](#)
- Serialize and Deserialize Binary Tree - [Link](#)
- Merge k Sorted Lists - [Link](#)
- Copy List with Random Pointer - [Link](#)
- Merge Two Sorted Lists - [Link](#)
- LRU Cache - [Link](#)
- Palindrome Linked List - [Link](#)
- Design Linked List - [Link](#)
- Reverse Nodes in k-Group - [Link](#)

Tree:

Easy:

- Maximum Depth of Binary Tree - [Link](#)
- Validate Binary Search Tree - [Link](#)
- Symmetric Tree - [Link](#)
- Convert Sorted Array to Binary Search Tree - [Link](#)
- Minimum Depth of Binary Tree - [Link](#)
- Diameter of Binary Tree - [Link](#)
- Same Tree - [Link](#)
- Subtree of Another Tree - [Link](#)
- Invert Binary Tree - [Link](#)

201.Path Sum - [Link](#)

Medium:

- Binary Tree Inorder Traversal - [Link](#)
- Construct Binary Tree from Preorder and Inorder Traversal - [Link](#)
- Construct Binary Tree from Inorder and Postorder Traversal - [Link](#)
- Lowest Common Ancestor of a Binary Tree - [Link](#)
- Binary Tree Level Order Traversal - [Link](#)
- Validate Binary Search Tree - [Link](#)
- Binary Tree Zigzag Level Order Traversal - [Link](#)
- Path Sum II - [Link](#)
- Count Complete Tree Nodes - [Link](#)
- Flatten Binary Tree to Linked List - [Link](#)

Hard:

- Binary Tree Maximum Path Sum - [Link](#)
- Serialize and Deserialize Binary Tree - [Link](#)
- Construct Binary Tree from String - [Link](#)
- Populating Next Right Pointers in Each Node - [Link](#)
- Populating Next Right Pointers in Each Node II - [Link](#)
- House Robber III - [Link](#)
- Kth Smallest Element in a BST - [Link](#)
- Closest Binary Search Tree Value II - [Link](#)
- Convert Sorted List to Binary Search Tree - [Link](#)
- Maximum Binary Tree - [Link](#)

Graph:

Easy:

- Number of Islands - [Link](#)
- Valid Sudoku - [Link](#)
- Course Schedule - [Link](#)
- Is Graph Bipartite? - [Link](#)
- Friend Circles - [Link](#)
- Clone Graph - [Link](#)
- Pacific Atlantic Water Flow - [Link](#)
- Course Schedule II - [Link](#)
- Minimum Height Trees - [Link](#)
- Reconstruct Itinerary - [Link](#)

Medium:

- Word Ladder - [Link](#)
- Number of Connected Components in an Undirected Graph - [Link](#)
- Graph Valid Tree - [Link](#)
- Course Schedule III - [Link](#)
- Alien Dictionary - [Link](#)
- Network Delay Time - [Link](#)
- Number of Islands II - [Link](#)
- Pacific Atlantic Water Flow - [Link](#)
- Reconstruct Itinerary - [Link](#)
- Course Schedule - [Link](#)

Hard:

- Number of Islands III - [Link](#)
- Bus Routes - [Link](#)
- Critical Connections in a Network - [Link](#)
- Evaluate Division - [Link](#)
- Number of Restricted Paths From First to Last Node - [Link](#)
- Minimum Swaps to Group All 1's Together - [Link](#)
- Swim in Rising Water - [Link](#)
- Optimize Water Distribution in a Village - [Link](#)
- Regions Cut By Slashes - [Link](#)
- Most Stones Removed with Same Row or Column - [Link](#)

