Absolutely! Here's a curated list of **100 medium-level DSA theory questions** focused on **Arrays, Strings, and Linked Lists** — no basics, just solid intermediate conceptual depth.

🔢 Arrays – Medium Theory Questions (1–33)

1. Explain how you would rotate an array k times efficiently.
2. How do you find the majority element in an array?
3. Describe Kadane's algorithm and its use.
4. What is the Dutch National Flag algorithm? Where is it used?
5. How does a two-pointer approach help in solving array problems?
6. Explain how binary search can be applied in a rotated sorted array.
7. Describe the sliding window technique for subarray problems.
8. How do prefix sums help with range queries?
9. What is a difference array? How is it useful?
10. How do you find the longest consecutive subsequence?
11. What is an equilibrium index? How can it be found?
12. How do you count inversions in an array? Why is it useful?
13. What is the time complexity of merging two sorted arrays in-place?
14. How does hashing help in array problems?
15. How do you detect duplicates in an array without modifying it?
16. Describe an efficient approach to merge overlapping intervals.
17. How do you rearrange elements in alternating positive and negative order?
18. What is the maximum product subarray problem?
19. What are sparse arrays? How are they implemented?
20. Explain the approach to find the smallest missing positive integer.
21. How do you detect subarrays with a given sum?
22. How do you find all pairs with a given sum using hashing?
23. What's the approach to sort an array of 0s, 1s, and 2s?
24. How to apply binary search on an infinite array?

25. Explain how to find duplicate numbers using Floyd’s cycle detection.
26. What are monotonic arrays? How do they help in optimization?
27. How can stacks help in solving histogram problems (like largest rectangle)?
28. Explain segment trees in context of arrays (conceptually).
29. What is the optimal way to rotate an array using reversal?
30. How do suffix sums differ from prefix sums and when to use them?
31. What is a circular array and how to handle index wrapping?
32. How does QuickSelect algorithm help in finding the k-th smallest element?
33. How do you find the next greater/smaller element using stacks?

□ Strings – Medium Theory Questions (34–66)

34. How does Rabin-Karp algorithm optimize substring search?
35. Describe the KMP algorithm and its applications.
36. What is Z-Algorithm in string matching?
37. How do you find the longest palindrome substring efficiently?
38. Explain how trie structures work for prefix-based searching.
39. What is string hashing and how does it prevent collisions?
40. Explain the concept of rolling hash with example.
41. How does Manacher’s algorithm find palindromes in linear time?
42. How to perform wildcard pattern matching with * and ??
43. What is the time complexity of concatenating strings in Java vs Python?
44. Explain the difference between lexicographic order and ASCII order.
45. How to check if two strings are rotations of each other?
46. What is the use of a suffix array in string problems?
47. Describe the concept of LPS (Longest Prefix Suffix) array.
48. How do you find the longest common prefix among a group of strings?
49. How to group anagrams efficiently?
50. What is Unicode normalization? Why is it important?

51. How does immutability of strings impact algorithm design?
52. What is the edit distance (Levenshtein distance)?
53. How do you find the minimum window substring that contains all characters?
54. How does regular expression matching work internally?
55. How is a string builder different from string concatenation?
56. How do you compress a string using run-length encoding?
57. How to validate numeric strings using FSM (finite state machines)?
58. Explain how tries can be used for autocomplete systems.
59. How to count the number of distinct substrings of a string?
60. What is the difference between subsequence and substring?
61. How do you tokenize a string efficiently?
62. What is the concept of string interning and how does it save memory?
63. Explain the sliding window maximum substring problem.
64. What are efficient ways to detect repeating characters in a substring?
65. How does Boyer-Moore pattern search differ from KMP?
66. What are grapheme clusters in Unicode strings?

📎 Linked Lists – Medium Theory Questions (67–100)

67. How does Floyd's Cycle Detection algorithm work?
68. How do you find the start node of a loop in a linked list?
69. What is the difference between singly, doubly, and circular linked lists?
70. How do you reverse a linked list recursively?
71. How do you find the intersection point of two linked lists?
72. How is memory allocation different in arrays and linked lists?
73. What are sentinel (dummy) nodes and how do they simplify logic?
74. What's the time complexity of insertion at the head and tail in various linked lists?
75. How to delete a node in O(1) time without head pointer?
76. How can you detect and remove a loop using hashing?

77. How is LRU cache implemented using doubly linked lists?
78. What is the use of slow and fast pointers in linked list problems?
79. How do you merge two sorted linked lists?
80. How do you add two numbers represented by linked lists?
81. How to check if a linked list is a palindrome using O(1) space?
82. How to sort a linked list using merge sort?
83. What are the pros and cons of linked lists over arrays?
84. What are circular linked lists and where are they used?
85. How to rotate a linked list k places?
86. How to flatten a multilevel linked list?
87. How to detect a loop in a linked list with double pointers?
88. How to find the N-th node from the end in a single pass?
89. What are use cases of dummy nodes in edge case handling?
90. What is structural sharing in persistent linked lists?
91. How is a linked list implemented in functional programming languages?
92. How do stacks/queues differ in linked list vs array implementations?
93. How to reverse alternate k nodes in a linked list?
94. What are real-life applications of linked lists in operating systems?
95. What is memory fragmentation and how do linked lists help?
96. What are XOR linked lists and how do they save memory?
97. How to clone a linked list with random pointers?
98. What’s the difference between deep copy and shallow copy in linked structures?
99. How do you implement a circular queue using linked lists?
100. How is garbage collection handled in languages with dynamic memory and linked structures?