the **top searching problems** on LeetCode frequently asked in interviews, grouped by their types of searching techniques:

**1. Binary Search Problems**

1. **Binary Search** (LeetCode #704):
   * Classic binary search to find the target in a sorted array.
2. **Search Insert Position** (LeetCode #35):
   * Find the index where a target should be inserted in a sorted array.
3. **First Bad Version** (LeetCode #278):
   * Use binary search to find the first bad version in a sequence.
4. **Find Peak Element** (LeetCode #162):
   * Find a peak element in an array where neighbors are smaller.
5. **Find Minimum in Rotated Sorted Array** (LeetCode #153):
   * Find the minimum element in a rotated sorted array.
6. **Find Minimum in Rotated Sorted Array II** (LeetCode #154):
   * Same as above but with duplicate elements.
7. **Search in Rotated Sorted Array** (LeetCode #33):
   * Search for a target in a rotated sorted array.
8. **Search in Rotated Sorted Array II** (LeetCode #81):
   * Same as above but with duplicate elements.
9. **Median of Two Sorted Arrays** (LeetCode #4):
   * Find the median of two sorted arrays in O(log(min(m, n))) time.
10. **Kth Smallest Element in a Sorted Matrix** (LeetCode #378):
    * Find the k-th smallest element in a sorted 2D matrix using binary search.

**2. Searching in a 2D Matrix**

1. **Search a 2D Matrix** (LeetCode #74):
   * Search for a target in a 2D matrix where rows and columns are sorted.
2. **Search a 2D Matrix II** (LeetCode #240):
   * Search for a target in a 2D matrix with rows and columns sorted.

**3. Sliding Window and Two-Pointer Search**

1. **Longest Substring Without Repeating Characters** (LeetCode #3):
   * Use a sliding window to find the length of the longest substring without repeating characters.
2. **Minimum Size Subarray Sum** (LeetCode #209):
   * Find the minimal length of a subarray with a sum ≥ target.
3. **Longest Subarray of 1s After Deleting One Element** (LeetCode #1493):
   * Find the longest subarray containing only 1s after deleting one element.
4. **Max Consecutive Ones III** (LeetCode #1004):
   * Find the maximum number of consecutive 1s in a binary array after flipping at most k zeros.

**4. Hashing and Search**

1. **Two Sum** (LeetCode #1):
   * Find two numbers in an array that add up to a target.
2. **Four Sum II** (LeetCode #454):
   * Count all tuples (a, b, c, d) such that a + b + c + d = 0.
3. **Intersection of Two Arrays II** (LeetCode #350):
   * Find the intersection of two arrays, allowing duplicate elements.

**5. Graph Search (DFS/BFS)**

1. **Number of Islands** (LeetCode #200):
   * Count the number of islands in a binary grid using DFS or BFS.
2. **Word Ladder** (LeetCode #127):
   * Find the shortest transformation sequence from beginWord to endWord.
3. **Word Search** (LeetCode #79):
   * Check if a word exists in a 2D grid using backtracking.

**6. Advanced Search**

1. **Allocate Books (Split Array Largest Sum)** (LeetCode #410):
   * Split an array into k subarrays such that the largest sum is minimized.
2. **Capacity to Ship Packages Within D Days** (LeetCode #1011):
   * Minimize the capacity needed to ship all packages within D days.
3. **Aggressive Cows (Maximum Distance)**:
   * Place cows in stalls to maximize the minimum distance between them.
4. **Longest Increasing Subsequence** (LeetCode #300):
   * Use binary search and dynamic programming to find the LIS.

**7. String Search**

1. **Implement strStr()** (LeetCode #28):
   * Implement substring search using KMP or other methods.
2. **Longest Palindromic Substring** (LeetCode #5):
   * Find the longest palindromic substring using expand-around-center or dynamic programming.
3. **Repeated Substring Pattern** (LeetCode #459):
   * Check if a string can be formed by repeating a substring.

**8. Bit Manipulation Search**

1. **Single Number** (LeetCode #136):
   * Find the single number in an array where every other number appears twice using XOR.
2. **Missing Number** (LeetCode #268):
   * Find the missing number in a range using XOR or summation.

Would you like detailed explanations or implementations for any of these problems?