Practice coding problems on platforms like LeetCode, HackerRank, or CodeWars.

Here’s a revised list of **100 easy and intermediate-level problem-solving questions** suitable for JavaScript developers. These are grouped by topics to help you organize your learning:

**1. Arrays**

1. Two Sum
2. Remove Duplicates from Sorted Array
3. Move Zeroes
4. Rotate Array
5. Maximum Subarray
6. Contains Duplicate
7. Intersection of Two Arrays
8. Plus One
9. Missing Number
10. Majority Element
11. Best Time to Buy and Sell Stock
12. Valid Mountain Array
13. Find All Numbers Disappeared in an Array
14. Height Checker
15. Squares of a Sorted Array
16. Minimize Maximum Pair Sum in Array
17. Third Maximum Number
18. Relative Sort Array
19. Check If N and Its Double Exist
20. Valid Sudoku

**2. Strings**

1. Reverse String
2. Reverse Words in a String III
3. Valid Palindrome
4. First Unique Character in a String
5. Implement strStr()
6. Longest Common Prefix
7. Count and Say
8. Valid Parentheses
9. Is Subsequence
10. Add Binary
11. Check if the Sentence Is Pangram
12. Replace All Digits with Characters
13. Determine if String Halves Are Alike
14. Merge Strings Alternately
15. Robot Return to Origin
16. Defanging an IP Address
17. Truncate Sentence
18. Sort Characters By Frequency
19. Count Binary Substrings
20. Longest Substring Without Repeating Characters

**3. Hashing**

1. Contains Duplicate II
2. Valid Anagram
3. Intersection of Two Arrays II
4. First Unique Character in a String
5. Two Sum II (Sorted Array)
6. Find the Difference
7. Single Number
8. Find All Anagrams in a String
9. Group Anagrams
10. Happy Number

**4. Math**

1. Fizz Buzz
2. Add Digits
3. Power of Two
4. Sqrt(x)
5. Guess Number Higher or Lower
6. Missing Number
7. Roman to Integer
8. Excel Sheet Column Number
9. Factorial Trailing Zeroes
10. Check If It Is a Straight Line

**5. Linked Lists**

1. Reverse Linked List
2. Merge Two Sorted Lists
3. Remove Duplicates from Sorted List
4. Delete Node in a Linked List
5. Intersection of Two Linked Lists
6. Linked List Cycle
7. Palindrome Linked List
8. Remove Linked List Elements
9. Odd Even Linked List
10. Middle of the Linked List

**6. Trees**

1. Maximum Depth of Binary Tree
2. Symmetric Tree
3. Binary Tree Inorder Traversal
4. Binary Tree Preorder Traversal
5. Path Sum
6. Diameter of Binary Tree
7. Invert Binary Tree
8. Minimum Depth of Binary Tree
9. Convert Sorted Array to Binary Search Tree
10. Same Tree

**7. Recursion**

1. Pascal's Triangle
2. Fibonacci Number
3. Power of Three
4. Climbing Stairs
5. Reverse String (Recursively)
6. Sum of Digits of a Number
7. Generate Parentheses (Intermediate)
8. Merge Two Binary Trees
9. Factorial
10. Search in a Binary Search Tree

**8. Searching & Sorting**

1. Binary Search
2. Find First and Last Position of Element in Sorted Array
3. Search Insert Position
4. Guess Number Higher or Lower
5. Merge Sorted Array
6. Sort Colors
7. Intersection of Two Arrays
8. Kth Largest Element in a Stream
9. Squares of a Sorted Array
10. Relative Ranks

**Guidance**

* **Easy Problems**: Great for beginners focusing on loops, conditionals, and simple algorithms.
* **Intermediate Problems**: Involve concepts like sliding windows, recursion, dynamic programming (basic), or simple tree traversal.

Let me know if you'd like explanations, solutions, or tips for specific problems!

Here are **more examples** of **easy and intermediate-level problems** from **Strings**, **Arrays**, and **Hashing**, specifically tailored for JavaScript developers:

**Strings**

1. **Remove Vowels from a String**
   * Remove all vowels from the input string.
2. **Check for Palindrome**
   * Check if a string reads the same forward and backward.
3. **Longest Palindromic Substring**
   * Find the longest substring that is a palindrome.
4. **Find the Index of the First Occurrence in a String**
   * Implement a function like strStr() to locate the first occurrence of a substring.
5. **Count Words in a String**
   * Count the number of words in a string separated by spaces.
6. **Find and Replace Pattern**
   * Given a list of words, return the ones matching a given pattern.
7. **Reverse Only Letters**
   * Reverse the letters in a string, keeping non-letter characters in their original positions.
8. **Buddy Strings**
   * Check if two strings are similar by swapping two letters in one of them.
9. **Rotate String**
   * Check if one string is a rotation of another.
10. **Longest Common Subsequence**
    * Find the longest common subsequence between two strings.
11. **String to Integer (atoi)**
    * Implement string-to-integer conversion.
12. **Reorganize String**
    * Rearrange a string so that no adjacent characters are the same.
13. **Repeated Substring Pattern**
    * Check if a string is a repeated substring of itself.
14. **Check if a String Contains All Binary Codes of Length K**
    * Determine if a string contains all possible binary codes of length K.
15. **Add Strings**
    * Add two non-negative numbers represented as strings.

**Arrays**

1. **Find Pivot Index**
   * Find the index where the sum of elements on the left equals the sum on the right.
2. **Largest Number at Least Twice of Others**
   * Find if the largest number in an array is at least twice as large as the others.
3. **Replace Elements with Greatest Element on Right Side**
   * Replace every element with the greatest element among the elements to its right.
4. **Find the Distance Value Between Two Arrays**
   * Calculate the distance value between two arrays based on given conditions.
5. **Array Partition I**
   * Partition an array to maximize the sum of min-pairs.
6. **Maximum Average Subarray I**
   * Find the subarray with the maximum average.
7. **Binary Search in a Rotated Array**
   * Search for an element in a rotated sorted array.
8. **Split Array into Consecutive Subsequences**
   * Check if the array can be split into consecutive subsequences.
9. **Can Place Flowers**
   * Check if flowers can be placed in a flowerbed without violating the rule.
10. **Max Consecutive Ones**
    * Find the maximum number of consecutive 1s in a binary array.
11. **Array Nesting**
    * Find the length of the longest set created by a nesting operation.
12. **Find Pivot in a Rotated Sorted Array**
    * Find the index of the smallest element in a rotated sorted array.
13. **Candy Distribution**
    * Distribute candies to children with specific rules.
14. **Toeplitz Matrix**
    * Check if a 2D array is a Toeplitz matrix.
15. **Sort Array by Parity**
    * Rearrange an array so that all even elements come before all odd elements.

**Hashing**

1. **Unique Email Addresses**
   * Count the number of unique email addresses in a list.
2. **Find the Difference**
   * Find the letter added to a string after shuffling and adding a single extra character.
3. **Single Number**
   * Find the number that appears exactly once in an array where others appear twice.
4. **Distribute Candies**
   * Determine the maximum number of candy types a person can eat.
5. **Longest Harmonious Subsequence**
   * Find the longest subsequence in which the difference between max and min is 1.
6. **Count Good Meals**
   * Count pairs of dishes whose total sum is a power of two.
7. **Smallest Subarray with All Occurrences of the Most Frequent Element**
   * Find the shortest subarray containing all occurrences of the most frequent number.
8. **Two Sum II (Sorted Input)**
   * Return indices of two numbers in a sorted array that sum up to a target.
9. **Group People Given the Group Size They Belong To**
   * Group people based on the sizes given.
10. **Subdomain Visit Count**
    * Count the number of visits to each subdomain in a given list.
11. **Jewels and Stones**
    * Count the number of jewels in a string of stones.
12. **Sum of Unique Elements**
    * Sum all unique elements in an array.
13. **Check If Two Arrays Are Equal by Reversing Subarrays**
    * Determine if two arrays can be made equal using subarray reversals.
14. **Ransom Note**
    * Determine if a ransom note can be created using letters from a magazine.
15. **Max Number of Balloons**
    * Count how many times the word "balloon" can be formed from a given string.

Let me know if you’d like explanations, solutions, or specific problem patterns to focus on!