

Beginner Level (1-30)

1. Print "Hello, World!" in JavaScript.
2. Reverse a string.
3. Check if a string is a palindrome.
4. Find the largest number in an array.
5. Sort an array in ascending order.
6. Print all even numbers from 1 to 100.
7. Count the number of vowels in a string.
8. Check if a given number is prime.
9. Calculate the factorial of a number.
10. Create a function to return the nth Fibonacci number.
11. Sum all numbers in an array.
12. Find the maximum difference between any two elements in an array.
13. Remove duplicates from an array.
14. Find the intersection of two arrays.
15. Check if a string contains only digits.
16. Count the number of words in a sentence.
17. Check if two strings are anagrams.
18. Swap two numbers without using a temporary variable.
19. Check if a number is even or odd.
20. Create a function that returns the square of a number.
21. Convert a given string to title case.
22. Merge two sorted arrays.
23. Remove falsy values from an array.
24. Find the second largest number in an array.
25. Check if an array is sorted.
26. Find the missing number in a given array.
27. Check if a string ends with a given substring.
28. Write a function to repeat a string n times.
29. Implement a basic queue using an array.
30. Create a function that capitalizes the first letter of each word in a string.

Intermediate Level (31-70)

31. Implement a simple linked list.
32. Create a function to flatten a nested array.
33. Generate a random hex color code.
34. Implement a debounce function.
35. Write a function to shuffle an array.
36. Create a function to get unique values from an array.
37. Implement a binary search algorithm.
38. Write a function to find the GCD of two numbers.

39. Create a function to convert RGB to hex.
40. Write a function to count the number of occurrences of each character in a string.
41. Create a function to format a number as currency.
42. Implement a basic stack using an array.
43. Create a function to merge two objects.
44. Implement a basic calculator using functions.
45. Write a function to find the longest common prefix of an array of strings.
46. Create a function to find the longest palindrome in a string.
47. Write a function to group an array of objects by a property.
48. Implement a basic LRU cache.
49. Create a function to check if an array is a subset of another array.
50. Write a function to rotate an array `k` times.
51. Create a function that deep clones an object.
52. Write a function to convert a string to a URL slug.
53. Implement a function to find the longest increasing subsequence.
54. Write a function to validate an email address.
55. Create a function that simulates the `Array.prototype.map()` method.
56. Implement a function to get the `n`th Fibonacci number iteratively.
57. Create a function to find the median of an array.
58. Implement a function that converts Roman numerals to integers.
59. Create a function to check if a given string is a valid identifier.
60. Write a function to count the number of prime numbers up to `n`.
61. Implement a simple version of the `Promise.all()` function.
62. Create a function that returns the transpose of a matrix.
63. Implement the merge sort algorithm.
64. Create a function that returns a throttled version of a function.
65. Write a function that counts the number of inversions in an array.
66. Create a function to generate all permutations of a string.
67. Implement the quicksort algorithm.
68. Write a function to find the majority element in an array.
69. Create a function to validate if an array forms a valid sequence.
70. Write a function to implement a binary search tree (BST) with basic operations.

Advanced Level (71-100)

71. Implement a function for deep comparison between two objects.
72. Create a function that memoizes another function.
73. Write a function to serialize and deserialize a binary tree.
74. Implement a function to find all subsets of a given array.
75. Write a function to find the shortest path in a graph using Dijkstra's algorithm.
76. Implement a function for a rate limiter.
77. Create a function that finds the intersection point of two linked lists.
78. Write a function to solve the N-Queens problem.
79. Implement a function to find the longest path in a DAG (Directed Acyclic Graph).

80. Create a function to detect a cycle in a linked list.
81. Write a function to solve the knapsack problem.
82. Implement a function for breadth-first search (BFS) on a graph.
83. Write a function for depth-first search (DFS) on a graph.
84. Implement a function to find all paths from the source to the destination in a graph.
85. Write a function for merge intervals.
86. Create a function that simulates `Function.prototype.bind()`.
87. Implement a function to convert an array of promises into a single chained promise.
88. Create a function to generate combinations of a string.
89. Implement a function to find the longest substring without repeating characters.
90. Write a function to solve the word ladder problem.
91. Implement a function to find the minimum window substring.
92. Create a function for topological sorting of a DAG.
93. Write a function to implement the A* search algorithm.
94. Implement a function to find the critical connections in a network.
95. Create a function to find the articulation points in a graph.
96. Write a function for backtracking to solve Sudoku.
97. Implement a function to check if a binary tree is balanced.
98. Write a function for KMP pattern matching algorithm.
99. Create a function to count the number of islands in a matrix.
100. Implement a function for Tarjan's algorithm to find strongly connected components.

In questions mein theory ke sath code practice karne se aapki coding skills har level pe improve hongi. Kya aap in questions ka detailed solution bhi chahte hain?

40

You said:

aur 100 questions do jis me biginner intermediate aur avanced ho

ChatGPT said:

ChatGPT

Yeh rahe aur 100 JavaScript coding questions har level ke liye:

Beginner Level (101-130)

101. Write a function to find the sum of all even numbers in an array.
102. Create a function to check if a number is a perfect square.
103. Find the longest word in a string.
104. Write a function to convert a decimal number to binary.
105. Check if a given year is a leap year.
106. Create a function that reverses an array without using the built-in `reverse()` method.
107. Find the GCD (Greatest Common Divisor) of two numbers using recursion.

108. Write a function to find the sum of digits of a number.
109. Create a function that removes specific elements from an array.
110. Check if two arrays are equal (order matters).
111. Find the first non-repeating character in a string.
112. Create a function to reverse the order of words in a sentence.
113. Implement a function to calculate the power of a number (x^y).
114. Write a function to count the frequency of elements in an array.
115. Convert a string to camelCase.
116. Find the maximum sum of a subarray using the Kadane's algorithm.
117. Write a function to check if a string starts with a specific substring.
118. Create a function that finds all prime numbers up to n using the Sieve of Eratosthenes.
119. Write a function to find the first duplicate number in an array.
120. Implement a basic version of the `filter()` function.
121. Write a function that flattens an array of arrays up to a given depth.
122. Check if a string is a valid palindrome ignoring special characters.
123. Create a function to split an array into chunks of n size.
124. Implement a function to count the number of trailing zeros in a factorial.
125. Create a function to replace all spaces in a string with `-`.
126. Write a function to check if a number is an Armstrong number.
127. Find the sum of an arithmetic progression given a , d , and n .
128. Create a function to remove all spaces from a string.
129. Write a function that returns the product of an array's elements.
130. Implement a function that checks if an array has duplicate elements.

Intermediate Level (131-170)

131. Create a function to rotate a matrix 90 degrees clockwise.
132. Implement a function to check if two strings are permutations of each other.
133. Write a function to convert a number to Roman numerals.
134. Create a function that returns all pairs of numbers in an array that sum to a target value.
135. Implement the bubble sort algorithm.
136. Write a function to find the n th ugly number.
137. Create a function that returns the sum of all left leaves in a binary tree.
138. Write a function to remove duplicates from a linked list.
139. Implement a simple hash table with basic operations.
140. Create a function to check if a binary tree is symmetric.
141. Write a function to find the longest increasing path in a matrix.
142. Create a function to count the number of unique paths in a grid.
143. Implement a function to find the longest consecutive sequence in an array.
144. Write a function to find the number of ways to climb stairs with n steps.
145. Create a function to generate the Pascal's Triangle up to a given number of rows.

146. Implement the insertion sort algorithm.
147. Write a function to validate parentheses in a string.
148. Create a function that finds the common elements in three sorted arrays.
149. Write a function to find all missing numbers in an array.
150. Create a function to convert a 12-hour time format to a 24-hour format.
151. Implement a simple version of `Array.prototype.reduce()`.
152. Write a function to check if a binary tree is balanced.
153. Create a function to find the diameter of a binary tree.
154. Implement a basic event emitter class.
155. Create a function to check if a given string can be rearranged into a palindrome.
156. Write a function to find all valid IP addresses from a given string of digits.
157. Implement the shell sort algorithm.
158. Create a function that simulates a sleep function.
159. Write a function to implement a simple version of `Promise.race()`.
160. Create a function to generate all possible permutations of an array.
161. Implement the counting sort algorithm.
162. Write a function to find the nth super ugly number.
163. Create a function to serialize a nested object to a query string.
164. Write a function to find the minimum spanning tree using Prim's algorithm.
165. Create a function to check if a string is a valid palindrome by removing at most one character.
166. Implement a function to find the largest rectangle in a histogram.
167. Write a function for the Dutch National Flag problem.
168. Create a function to find the length of the shortest path in an unweighted graph.
169. Write a function to solve the longest word ladder sequence.
170. Implement the selection sort algorithm.

Advanced Level (171-200)

171. Create a function for matrix multiplication.
172. Write a function to solve the traveling salesman problem using dynamic programming.
173. Implement a function to find all strongly connected components using Kosaraju's algorithm.
174. Write a function to implement the min heap data structure.
175. Create a function to find the longest palindrome substring using Manacher's algorithm.
176. Implement a function for Bellman-Ford algorithm.
177. Write a function to solve the maximum flow problem using Ford-Fulkerson algorithm.
178. Create a function to solve the coin change problem using dynamic programming.
179. Implement a function to find the articulation points in an undirected graph.
180. Write a function to solve the edit distance problem using dynamic programming.
181. Implement a Trie data structure with basic insert, search, and delete operations.
182. Create a function for finding the longest repeating substring.

183. Write a function to find the largest number that can be formed from an array of numbers.
184. Implement a function for the Rabin-Karp string matching algorithm.
185. Write a function for the 0/1 knapsack problem using recursion and memoization.
186. Create a function to find all paths in a maze.
187. Implement a function for Kahn's algorithm for topological sorting.
188. Write a function to detect a negative cycle in a graph using Bellman-Ford.
189. Create a function for interval scheduling maximization.
190. Implement a function for finding maximum profit with at most two transactions.
191. Write a function for the longest repeating subsequence in a string.
192. Create a function for finding the convex hull of a set of points using Graham's scan.
193. Implement a function for balanced parentheses generation.
194. Write a function for matrix chain multiplication problem.
195. Create a function for solving the stable marriage problem.
196. Implement a function to find the minimum cut in a graph using the Stoer-Wagner algorithm.
197. Write a function for finding bridges in a graph.
198. Create a function for solving the subset sum problem using dynamic programming.
199. Implement a function for bitwise operations (AND, OR, XOR).
200. Write a function to build and solve a sudoku board using backtracking.

Yeh sawaal aapko JavaScript coding mein har level par achchi practice denge. Kya in questions ka koi specific solution ya topic chahiye?

4o

You said:

aur 100 questions do jis me bigneer intermediate aur avanced ho

ChatGPT said:

ChatGPT

Memory updated

Yeh rahe aur 100 JavaScript coding questions, har level ke liye categorized:

Beginner Level (201-230)

201. Write a function to find the factorial of a number iteratively.
202. Create a function to calculate the nth triangular number.
203. Check if a number is a power of two.
204. Write a function to merge two arrays without duplicates.
205. Create a function to find the index of an element in an array without using `indexOf()`.
206. Write a function to check if a given string is a valid URL.
207. Create a function to capitalize every alternate letter in a string.

- 208. Check if an object is empty.
- 209. Write a function to find the sum of all odd numbers in an array.
- 210. Create a function that adds two numbers without using the `+` or `-` operators.
- 211. Implement a function to generate the Fibonacci sequence up to `n` terms.
- 212. Write a function that formats a date object into `YYYY-MM-DD`.
- 213. Check if a string is a valid hexadecimal color.
- 214. Create a function that finds the longest even word in a sentence.
- 215. Write a function to return the last `n` elements of an array.
- 216. Create a function to replace all vowels in a string with a specified character.
- 217. Write a function that calculates the average of an array.
- 218. Check if a given string is a rotation of another string.
- 219. Implement a function that checks if a given string contains balanced curly braces `{}`.
- 220. Create a function to remove a property from an object.
- 221. Write a function that sorts an array of strings by their length.
- 222. Create a function that returns the sum of squares of an array.
- 223. Write a function that repeats a given string until it reaches a specified length.
- 224. Create a function to extract unique characters from a string.
- 225. Implement a function that counts how many times a substring appears in a string.
- 226. Write a function to find the first missing positive integer in an array.
- 227. Create a function to shuffle the characters in a string.
- 228. Write a function that reverses the digits of a number.
- 229. Create a function to find the average word length in a sentence.
- 230. Check if an array is a palindrome.

Intermediate Level (231-270)

- 231. Implement the merge intervals problem.
- 232. Write a function to remove all adjacent duplicates in a string.
- 233. Create a function that takes an array and returns the two numbers that sum to the closest to zero.
- 234. Implement the heap sort algorithm.
- 235. Write a function to generate a random password with specified length and characters.
- 236. Create a function to find the longest contiguous subarray with equal numbers of 0s and 1s.
- 237. Implement a function that checks if a number is a perfect number.
- 238. Write a function to create a deep freeze for objects.
- 239. Create a function that converts a number to a binary search tree path.
- 240. Implement the Tower of Hanoi solution.
- 241. Write a function that converts a CSV string into an array of objects.
- 242. Create a function that flattens an object with nested properties.
- 243. Write a function to find the longest substring with unique characters.
- 244. Create a function that finds the maximum number of overlapping intervals.

- 245. Implement a function to calculate the Hamming distance between two strings.
- 246. Write a function that simulates a `setInterval()` implementation using `setTimeout()`.
- 247. Create a function to perform matrix rotation by 180 degrees.
- 248. Implement a function to find the longest subsequence of characters that appears in both strings.
- 249. Write a function to evaluate a simple mathematical expression in string format.
- 250. Create a function to check if a given string is a valid IP address.
- 251. Implement a function that finds the second smallest element in an array.
- 252. Create a function to get the frequency of numbers in an array as an object.
- 253. Write a function that creates a sliding window sum array.
- 254. Implement a function that finds all the leaders in an array (elements larger than all to their right).
- 255. Write a function that finds the first non-repeating element in an array.
- 256. Create a function to compress a string using run-length encoding.
- 257. Implement a function that sorts an array of strings ignoring case sensitivity.
- 258. Write a function to convert an array to an object using a callback function for keys.
- 259. Create a function to find all combinations of array elements that sum to a target value.
- 260. Implement a function to calculate the Jaccard similarity between two sets.
- 261. Write a function to validate a phone number format.
- 262. Create a function to implement the nearest smaller element problem.
- 263. Write a function to calculate compound interest given principal, rate, and time.
- 264. Implement a function to find the minimum distance between two words in a sentence.
- 265. Create a function that converts a string to a zigzag pattern.
- 266. Implement a binary search on a rotated sorted array.
- 267. Write a function to check if an array of integers is monotonic.
- 268. Create a function to parse a JSON-like string to an object.
- 269. Implement a function that returns the union and intersection of two arrays.
- 270. Write a function to generate a power set of an array.

Advanced Level (271-300)

- 271. Create a function that finds the shortest path between two nodes in an unweighted graph.
- 272. Implement a function to solve the balanced parentheses problem using dynamic programming.
- 273. Write a function to find the minimum spanning tree using Kruskal's algorithm.
- 274. Create a function for checking if a given binary tree is a binary search tree.
- 275. Implement a function for the Huffman coding algorithm.
- 276. Write a function to implement Tarjan's strongly connected components algorithm.
- 277. Create a function to find the maximum path sum in a binary tree.
- 278. Implement the Gale-Shapley algorithm for stable matching.
- 279. Write a function for solving the maximum subarray sum with at most two subarrays.

280. Create a function to generate all palindromic substrings of a given string.
281. Implement a function for finding the k-th largest element in a stream.
282. Write a function to solve the word search puzzle in a 2D grid.
283. Create a function to check if a given tree is a complete binary tree.
284. Implement a function for building a min heap and extract-min operation.
285. Write a function to solve the longest path problem using Bellman-Ford.
286. Create a function that uses dynamic programming to solve the rod-cutting problem.
287. Implement a function to find the maximum size rectangle in a binary matrix.
288. Write a function for cycle detection in a directed graph.
289. Create a function to find the maximum product subarray.
290. Implement a function for finding the largest rectangle in a binary matrix.
291. Write a function to solve the egg drop problem using dynamic programming.
292. Create a function to implement Floyd-Warshall algorithm for shortest paths.
293. Implement a function for solving the longest common subsequence problem.
294. Write a function to find the articulation points in an undirected graph.
295. Create a function to convert a sparse matrix to a dictionary representation.
296. Implement a function for solving the minimum cost to connect all points.
297. Write a function to find the most frequent element in an array.
298. Create a function to solve the number of ways to make change problem.
299. Implement a function to find the critical path in a directed acyclic graph.
300. Write a function to build an autocomplete system using a Trie data structure.

In questions ko solve karne se aapki JavaScript coding skills sabhi levels par aur bhi zyada improve hongi. Agar kisi specific question ka solution chahiye ho ya koi aur help chahiye ho, bataiye!

40

You said:

aur 100 questions do jis me bigneer intermediate aur avanced ho

ChatGPT said:

ChatGPT

Yeh rahe aur 100 JavaScript coding questions har level ke liye:

Beginner Level (301-330)

301. Write a function that counts the vowels in a string.
302. Create a function that checks if a given number is prime.
303. Implement a function to find the largest number in an array without using `Math.max()`.
304. Write a function to find the median of an array.
305. Create a function that checks if a given array is sorted in ascending order.
306. Write a function to convert a string to snake_case.

- 307. Create a function that reverses only the vowels in a string.
- 308. Implement a function that checks if two numbers have the same set of digits.
- 309. Write a function that returns the unique elements in an array.
- 310. Create a function to convert a 24-hour time to 12-hour format.
- 311. Implement a function that calculates the product of all positive numbers in an array.
- 312. Write a function to count how many times a specific letter appears in a string.
- 313. Create a function to remove all duplicate characters from a string.
- 314. Write a function to find the second largest number in an array.
- 315. Create a function that returns the nth Fibonacci number iteratively.
- 316. Write a function to format a number with commas as thousands separators.
- 317. Create a function that checks if a given string is an anagram of another.
- 318. Write a function to count the number of words in a sentence.
- 319. Create a function to find the maximum product of two integers in an array.
- 320. Write a function that removes falsy values from an array.
- 321. Create a function that finds the longest common prefix among an array of strings.
- 322. Write a function to reverse a string using recursion.
- 323. Implement a function to merge two sorted arrays into one.
- 324. Create a function to find the longest increasing subsequence in an array.
- 325. Write a function that rotates an array by `k` positions to the right.
- 326. Create a function that converts a string into title case.
- 327. Write a function to count the number of bits set to 1 in an integer.
- 328. Create a function to find the index of the first repeating character in a string.
- 329. Write a function that splits a string into an array of words.
- 330. Create a function to remove all HTML tags from a string.

Intermediate Level (331-370)

- 331. Implement the quicksort algorithm.
- 332. Write a function to solve the "maximum subarray sum" problem.
- 333. Create a function to validate if a given Sudoku solution is correct.
- 334. Write a function to find all subarrays in an array.
- 335. Implement a function that converts a number from base-10 to base-2.
- 336. Write a function that finds the intersection of two arrays.
- 337. Create a function that simulates a LRU (Least Recently Used) cache.
- 338. Implement a function to perform binary search on a sorted array.
- 339. Write a function that calculates the power of a number using divide-and-conquer.
- 340. Create a function that counts the number of islands in a 2D grid.
- 341. Write a function to find the longest palindrome substring in a string.
- 342. Create a function to check if a given string is valid using a set of character rules.
- 343. Implement a function that returns the maximum difference between two elements in an array.
- 344. Write a function to find the kth smallest element in an array.
- 345. Create a function that calculates the square root of a number without using `Math.sqrt()`.

- 346. Implement a function to check if a given binary tree is a BST (Binary Search Tree).
- 347. Write a function that finds the longest chain of consecutive integers in an array.
- 348. Create a function to parse an arithmetic expression into a tree.
- 349. Write a function to evaluate a postfix expression.
- 350. Implement a function that checks if a number is a palindrome without converting it to a string.
- 351. Create a function to generate all valid parentheses of length n .
- 352. Write a function that flattens an array with any level of nested arrays.
- 353. Create a function that checks if a string can be segmented into valid dictionary words.
- 354. Implement a function that finds all the permutations of a given string.
- 355. Write a function to solve the knapsack problem using recursion.
- 356. Create a function to implement a basic calculator for +, -, *, and / operations.
- 357. Write a function to remove duplicates from an unsorted linked list.
- 358. Create a function that finds the majority element in an array.
- 359. Implement a function that calculates the minimum number of edits (insertions, deletions, substitutions) to convert one string to another.
- 360. Write a function to count the number of distinct substrings in a string.
- 361. Create a function to solve the problem of finding the minimum window substring.
- 362. Implement a function that finds the common ancestor of two nodes in a binary tree.
- 363. Write a function that calculates the maximum water trapped between bars in a histogram.
- 364. Create a function to convert an infix expression to postfix notation.
- 365. Write a function that detects a cycle in a linked list.
- 366. Implement a function to find the minimum spanning tree of a graph using Prim's algorithm.
- 367. Write a function to determine if two strings are one edit distance apart.
- 368. Create a function that merges overlapping intervals.
- 369. Implement a function that finds the longest palindromic subsequence in a string.
- 370. Write a function to check if a matrix is symmetric.

Advanced Level (371-400)

- 371. Implement a function to solve the traveling salesman problem.
- 372. Write a function for the N-Queens problem.
- 373. Create a function that implements a basic version of a search engine ranking algorithm.
- 374. Implement the A* algorithm for pathfinding in a grid.
- 375. Write a function for the maximum flow problem using the Edmonds-Karp algorithm.
- 376. Create a function that generates all possible subsets of a set.
- 377. Implement a function to solve the minimum cut problem in a graph.
- 378. Write a function that finds the diameter of a binary tree.
- 379. Create a function to detect and return the articulation points in a graph.
- 380. Implement a function for solving the job scheduling problem with deadlines.

381. Write a function to solve the word ladder problem.
382. Create a function for building a Huffman tree from a given string.
383. Implement a function that solves the 2-sum problem using hashing.
384. Write a function that finds the shortest path in a weighted graph using Dijkstra's algorithm.
385. Create a function that builds a binary search tree from a preorder traversal.
386. Implement a function to solve the coin change problem with an unlimited supply of coins.
387. Write a function that simulates a blockchain with basic transaction validation.
388. Create a function for solving the minimum vertex cover problem in a tree.
389. Implement a function that finds the maximum path sum between two leaves in a binary tree.
390. Write a function that solves the sliding puzzle problem.
391. Create a function to detect cycles in a directed graph using depth-first search.
392. Implement a function for finding strongly connected components in a graph using Kosaraju's algorithm.
393. Write a function that finds the shortest path in a maze using BFS.
394. Create a function for solving the longest path problem in a weighted DAG.
395. Implement a function for finding the number of connected components in a graph.
396. Write a function to solve the painter's partition problem.
397. Create a function for solving the wild card pattern matching problem.
398. Implement a function to find the bridges in a graph.
399. Write a function to solve the meeting rooms problem with intervals.
400. Create a function for implementing a priority queue using a heap.

Yeh additional sawaal aapko JavaScript mein aur bhi zyada proficiency denge. Kya aap inme se kisi specific problem ka solution dekhna chaahenge?

40

You said:

aur 100 questions do jis me bigneer intermediate aur avanced ho

ChatGPT said:

ChatGPT

Here's another set of 100 JavaScript coding questions:

Beginner Level (401-430)

401. Write a function to check if an array contains any duplicates.
402. Create a function that returns the factorial of a number recursively.
403. Write a function that counts the number of consonants in a string.
404. Create a function to convert a number to Roman numerals.
405. Write a function to return the length of the longest word in a sentence.

- 406. Create a function to find the sum of all even numbers in an array.
- 407. Write a function to capitalize the first letter of each word in a string.
- 408. Create a function that returns the intersection of two arrays.
- 409. Write a function that finds the maximum difference between elements in an array.
- 410. Create a function to reverse a number.
- 411. Write a function to check if two strings are rotations of each other.
- 412. Create a function that removes a specific character from a string.
- 413. Write a function that returns the sum of the elements in an array, skipping duplicates.
- 414. Create a function to find the sum of the first n prime numbers.
- 415. Write a function to flatten an array using iteration.
- 416. Create a function to check if a string contains only unique characters.
- 417. Write a function to return the longest common suffix in an array of strings.
- 418. Create a function to implement a basic Caesar cipher.
- 419. Write a function that checks if a number is an Armstrong number.
- 420. Create a function to remove whitespace from the start and end of a string.
- 421. Write a function to check if a number is a palindrome using string conversion.
- 422. Create a function to sort an array of strings by the last letter.
- 423. Write a function that repeats a given string a specified number of times.
- 424. Create a function that finds the GCD (Greatest Common Divisor) of two numbers.
- 425. Write a function to generate all substrings of a given string.
- 426. Create a function that checks if an array contains consecutive numbers.
- 427. Write a function that calculates the sum of the digits of a number.
- 428. Create a function to find the most common element in an array.
- 429. Write a function that swaps two variables without using a third variable.
- 430. Create a function that filters out all non-alphabetic characters from a string.

Intermediate Level (431-470)

- 431. Implement a function to perform merge sort on an array.
- 432. Write a function to find the longest word in a string that appears more than once.
- 433. Create a function that returns the n th prime number.
- 434. Write a function to find the most frequent character in a string.
- 435. Implement a function to check if a given year is a leap year.
- 436. Write a function to generate all permutations of an array.
- 437. Create a function to implement a stack using JavaScript arrays.
- 438. Write a function to implement binary search on a rotated sorted array.
- 439. Create a function that finds the missing number in an array of integers from 1 to n .
- 440. Implement a function to perform matrix multiplication.
- 441. Write a function to check if a given number is a perfect square.
- 442. Create a function to find the first recurring character in a string.
- 443. Implement a function to perform insertion sort on an array.
- 444. Write a function that checks if a string is a valid palindrome considering only alphanumeric characters.

- 445. Create a function to calculate the nth number in the Fibonacci sequence using dynamic programming.
- 446. Write a function that counts the number of prime numbers less than a given number.
- 447. Implement a function that sorts an array using the bubble sort algorithm.
- 448. Create a function to solve the coin change problem using recursion.
- 449. Write a function to find the maximum depth of a nested array.
- 450. Create a function that calculates the edit distance between two strings.
- 451. Implement a function to find the transpose of a matrix.
- 452. Write a function that determines if a string can be rearranged to form a palindrome.
- 453. Create a function that finds the kth largest element in an array using a min-heap.
- 454. Write a function to find the longest substring without repeating characters.
- 455. Create a function that solves the "House Robber" problem.
- 456. Write a function to rotate a matrix by 90 degrees clockwise.
- 457. Implement a function that checks if a number is a power of three.
- 458. Write a function that counts the number of unique characters in a string.
- 459. Create a function to find the second highest number in an array without sorting it.
- 460. Write a function that implements the Sieve of Eratosthenes algorithm for finding prime numbers.
- 461. Create a function that finds the maximum product of any two numbers in an array.
- 462. Implement a function to perform quickselect for finding the kth smallest element in an array.
- 463. Write a function to validate a given string as a valid email address.
- 464. Create a function that merges two objects and resolves key conflicts by taking the sum of the values.
- 465. Implement a function to find all pairs of integers in an array that sum to a given number.
- 466. Write a function that counts the number of set bits in an integer.
- 467. Create a function to solve the "Buy and Sell Stock" problem for maximum profit.
- 468. Implement a function to convert an integer to a binary string.
- 469. Write a function that finds the common elements in two sorted arrays.
- 470. Create a function that calculates the median of two sorted arrays.

Advanced Level (471-500)

- 471. Implement a function to solve the maximum flow problem in a network graph.
- 472. Write a function to find all strongly connected components in a directed graph.
- 473. Create a function for implementing a data structure similar to a Trie for fast prefix searches.
- 474. Write a function that solves the Longest Increasing Path in a Matrix problem.
- 475. Implement a function that finds the minimum spanning tree using Kruskal's algorithm.
- 476. Create a function for the coin change problem using dynamic programming.
- 477. Write a function to implement the KMP algorithm for substring search.
- 478. Implement a function for solving the maximum rectangle problem in a binary matrix.

479. Write a function that solves the shortest path problem in a weighted graph using Bellman-Ford algorithm.
480. Create a function to find all possible subsets that sum to a given value.
481. Implement a function for the Word Break problem using recursion and memoization.
482. Write a function that simulates the "Game of Life" cellular automaton.
483. Create a function to check if a binary tree is balanced.
484. Write a function for finding the longest valid parentheses substring.
485. Implement a function that generates permutations with repetition for a given set.
486. Create a function that finds the number of unique paths in a grid using dynamic programming.
487. Write a function to solve the edit distance problem using dynamic programming.
488. Implement a function to find the bridges in an undirected graph.
489. Create a function to implement the Rabin-Karp algorithm for substring search.
490. Write a function that solves the Travelling Salesperson problem using brute force.
491. Implement a function to find the articulation points in a graph.
492. Create a function for finding the maximal rectangle in a 2D binary matrix.
493. Write a function that implements the Boyer-Moore voting algorithm for majority element.
494. Create a function that solves the sudoku puzzle.
495. Implement a function that returns the k-th smallest element in a matrix sorted row-wise and column-wise.
496. Write a function that solves the minimum window substring problem using the sliding window technique.
497. Create a function for implementing a priority queue using a binary heap.
498. Implement a function for finding the longest palindromic subsequence in a string.
499. Write a function to find the critical connections in a network using Tarjan's algorithm.
500. Create a function to find all unique paths in a grid with obstacles.