100 JavaScript Practice Questions: Objects & Strings

Objects (Questions 1-50)

Basic Object Operations

1. Create an object representing a person with name, age, and city properties.
2. Access the 'name' property of an object using dot notation.
3. Access a property of an object using bracket notation with a variable.
4. Add a new property 'email' to an existing person object.
5. Delete the 'age' property from a person object.
6. Check if a property 'phone' exists in an object.
7. Create an object with a property name that contains spaces.
8. Modify the value of an existing property in an object.
9. Create an empty object and add 5 properties to it dynamically.
10. Use Object.keys() to get all property names from an object.

Object Methods and Properties

11. Create an object with a method that returns the person's full name.
12. Create an object method that calculates and returns the person's age in dog years.
13. Use 'this' keyword inside an object method to access other properties.
14. Create an object with both properties and methods, then call the methods.
15. Use Object.values() to get all property values from an object.
16. Use Object.entries() to get key-value pairs from an object.
17. Create a method that lists all properties of the object.
18. Create an object method that checks if the person is an adult (age >= 18).
19. Use Object.assign() to copy properties from one object to another.
20. Create a method that updates multiple properties at once.

Advanced Object Concepts

21. Create nested objects (object within object) representing a company structure.
22. Access a deeply nested property (e.g., company.department.employee.name).
23. Create an array of objects representing a list of students with grades.
24. Find a specific object in an array of objects based on a property value.
25. Sort an array of objects by a specific property (e.g., sort students by grade).

26. Filter an array of objects based on multiple criteria.
27. Use map() to transform an array of objects into an array of specific values.
28. Group objects in an array by a common property value.
29. Merge two objects with overlapping properties (handle conflicts).
30. Create a deep copy of an object (including nested objects).

Object Destructuring and Advanced Features

31. Use object destructuring to extract multiple properties into variables.
32. Use destructuring with default values for missing properties.
33. Use rest operator (...) in object destructuring.
34. Rename variables while destructuring object properties.
35. Use destructuring in function parameters to accept an object.
36. Create an object using computed property names.
37. Use shorthand property syntax when property name matches variable name.
38. Create an object with getter and setter methods.
39. Use Object.freeze() to make an object immutable.
40. Use Object.seal() to prevent adding/deleting properties but allow modifications.

Object Challenges and Real-world Applications

41. Create a shopping cart object with methods to add, remove, and calculate total.
42. Build a simple bank account object with deposit, withdraw, and balance methods.
43. Create a library system object that manages books (add, remove, search).
44. Build a student gradebook object with methods to add grades and calculate averages.
45. Create a inventory management object for a store.
46. Build a simple contact book object with search and update functionality.
47. Create a todo list object with methods to add, complete, and filter tasks.
48. Build a simple game player object with stats and level-up functionality.
49. Create a weather data object that processes and formats temperature data.
50. Build a social media post object with like, comment, and share functionality.

Strings (Questions 51-100)

Basic String Operations

51. Create a string variable and find its length.
52. Convert a string to uppercase and lowercase.

53. Extract the first 5 characters from a string using substring().
54. Extract characters from position 3 to 8 using slice().
55. Find the position of the word "JavaScript" in a sentence.
56. Check if a string contains the word "programming".
57. Replace all occurrences of "hello" with "hi" in a string.
58. Split a sentence into an array of words.
59. Join an array of words back into a sentence with spaces.
60. Remove whitespace from the beginning and end of a string.

String Searching and Manipulation

61. Count how many times the letter "a" appears in a string.
62. Find the first occurrence of a substring in a string.
63. Find the last occurrence of a character in a string.
64. Check if a string starts with a specific prefix.
65. Check if a string ends with a specific suffix.
66. Extract the file extension from a filename string.
67. Capitalize the first letter of each word in a sentence.
68. Reverse a string without using built-in reverse methods.
69. Check if a string is a palindrome (reads same forwards and backwards).
70. Remove all vowels from a string.

String Formatting and Validation

71. Create a formatted string using template literals with variables.
72. Format a number as currency string (e.g., "$1,234.56").
73. Validate if a string is a valid email address format.
74. Check if a string contains only letters (no numbers or symbols).
75. Check if a string is a valid phone number format.
76. Create a function that pads a string to a specific length with spaces.
77. Format a string to title case (First Letter Of Each Word Capitalized).
78. Remove all extra spaces from a string (multiple spaces become single space).
79. Check if a string represents a valid URL.
80. Validate if a string is a strong password (multiple criteria).

Advanced String Processing

81. Extract all email addresses from a text string using regular expressions.
82. Find all words that start with a capital letter in a text.
83. Replace multiple different substrings in a single operation.
84. Create a function that generates a random string of specified length.
85. Implement a simple encryption/decryption for strings (Caesar cipher).
86. Count the frequency of each character in a string.
87. Find the longest word in a sentence.
88. Remove duplicate characters from a string while preserving order.
89. Check if two strings are anagrams of each other.
90. Convert a string with hyphens to camelCase (e.g., "my-string" to "myString").

String Challenges and Algorithms

91. Implement a function to compress a string (e.g., "aaabbcc" becomes "a3b2c2").
92. Find the longest common prefix among an array of strings.
93. Implement string matching algorithm (check if pattern exists in text).
94. Create a function that wraps text to fit within a specified line width.
95. Build a simple text search function that highlights matching terms.
96. Create a function that generates abbreviations from a phrase.
97. Implement a function to find all permutations of a string.
98. Build a simple spell checker that suggests corrections for misspelled words.
99. Create a function that formats code strings with proper indentation.
100. Implement a advanced string similarity comparison function.

Practice Tips:

For Objects:

1. **Start with Basics**: Master property access before moving to methods
2. **Practice with Real Data**: Use objects to model real-world entities
3. **Understand 'this'**: Learn how 'this' works in different contexts
4. **Master Destructuring**: It's a powerful feature for cleaner code
5. **Think in Terms of Data Structures**: Objects are key-value stores

For Strings:

1. **Know Your Methods**: Master built-in string methods like slice, substring, replace
2. **Regular Expressions**: Learn basic regex for advanced string operations
3. **Template Literals**: Use them for string interpolation and formatting
4. **Unicode Awareness**: Understand how JavaScript handles different character sets
5. **Performance**: Some string operations are expensive, especially in loops

Example Solutions Pattern:

javascript

```javascript
// Object Example: Create a person with a greeting method
const person = {
    name: "John Doe",
    age: 30,
    city: "New York",
    greet() {
        return `Hello, I'm ${this.name} from ${this.city}`;
    },
    isAdult() {
        return this.age >= 18;
    }
};

console.log(person.greet()); // "Hello, I'm John Doe from New York"
console.log(person.isAdult()); // true

// String Example: Check if string is palindrome
function isPalindrome(str) {
    const cleaned = str.toLowerCase().replace(/[^a-z0-9]/g, '');
    return cleaned === cleaned.split('').reverse().join('');
}

console.log(isPalindrome("A man a plan a canal Panama")); // true
```

Combined Object-String Challenges:

- Create an object that processes and validates string data
- Build a text analyzer object that counts words, characters, and sentences
- Create a user profile object that validates and formats string inputs
- Build a simple template engine that processes strings with object data

Master these concepts by practicing regularly and combining objects and strings in real-world scenarios!