JavaScript Practice Questions

1. Variables (50 questions)

What will be logged?

```
console.log(a);
var a = 10;
```

1.

What will this print?

```
let a = 1;
{ let a = 2; console.log(a); }
console.log(a);
```

2.
3. Can you reassign a `const` variable? Try it.

4. What happens when you redeclare `var`? Write a code snippet.

5. What error occurs when you access `let` before declaration?

Predict output:

```
var a = 1;
function test() {
  console.log(a);
  var a = 2;
  console.log(a);
}
test();
```

6.

7. Write code to demonstrate block scope of `let`.

Predict output:
```
for(var i=0; i<3; i++) {
  setTimeout(() => console.log(i), 100);
}
```

8.
9. Now fix the previous code so it logs 0,1,2 instead of 3,3,3.

What will this output?
```
const obj = {};
obj.name = "JS";
console.log(obj.name);
```

10.
11. Declare a variable inside an `if` block using `var` and log it outside the block.
12. What will happen if you do `let a = 10; let a = 20;` in the same scope?
13. Create a variable `a` without declaring it. What scope does it have?
14. Explain and demonstrate the difference between `globalThis` and `window`.
15. Use `const` to declare an array and push a new element to it. Will it work?
16. Write a program to swap two variables without a temporary variable using `let`.
17. Declare a `const` object and try to reassign it. What happens?

Predict output:
```
console.log(typeof undeclaredVariable);
```

18.
19. Write code to demonstrate how `var` variables are hoisted.

Predict output:
```
console.log(foo);
let foo = "bar";
```

20.
21. Explain and show the difference between `var` and `let` in loops with closures.

22. What happens if you declare a variable with the same name in two different scopes?

23. Write code to demonstrate temporal dead zone error.

Predict output:
```
var a = 1;
function test() {
  a = 2;
  console.log(a);
  var a;
}
test();
```

24.
25. Create a global variable inside a function without `var/let/const`.

Predict output:
```
for (let i = 0; i < 3; i++) {
  setTimeout(() => console.log(i), 100);
}
```

26.
27. Declare a variable `a` with `var` and `let` in different blocks and show scope.

28. What happens if you try to redeclare a `const` variable?

29. Write a function that declares a variable with the same name as a global variable and explain scope.

Predict output:
```
{
  let a = 10;
  {
    console.log(a);
  }
}
```

30.
31. How do `let` and `const` affect the global object? Demonstrate.

32. Write code to demonstrate how variables declared with `var` behave inside `if` blocks.

33. Explain what happens if you declare a variable after its use with `var`.

34. Write code that shows the difference between declaring a variable inside and outside a function.

Predict output:
```
var a = 10;
function test() {
  console.log(a);
}
test();
```

35.
36. What will happen if you declare a variable with `const` but do not initialize it?

37. Write code that throws an error because of redeclaration of a variable.

38. Demonstrate the use of global variables inside functions with an example.

39. Explain and show the difference in redeclaring variables with `var` and `let`.

Predict output:
```
let a = 5;
{
  var a = 10;
}
```

40.
41. Write code to swap two variables using destructuring and `let`.

Predict output:
```
console.log(typeof a);
let a = 3;
```

42.

43. Write code showing that `const` variables cannot be reassigned but their contents can be mutated if they are objects.

Predict output:
```
console.log(a);
var a = 10;
console.log(a);
```

44.
45. Write code to declare a variable inside a loop using `let` and log it after the loop.

46. Explain and show the difference between function scope and block scope with variables.

Predict output:
```
var a = 10;
function f() {
  if (true) {
    var a = 20;
  }
  console.log(a);
}
f();
```

47.
48. Write code demonstrating hoisting with function declarations and function expressions.

49. What happens if you assign a value to an undeclared variable?

50. Write code showing how variable shadowing works with nested functions.

---

2. Data Types (50 questions)

1. Write code to check the type of a variable holding a string, number, boolean, null, undefined, symbol, and object.

Predict output:

```
typeof null;
```

2.
3. Write a program that shows the difference between `null` and `undefined`.

4. Create a symbol and compare it to another symbol with the same description.

Predict output:

```
typeof NaN;
```

5.
6. Write code to demonstrate `BigInt` usage and how it differs from `Number`.

Predict output:

```
console.log(true + false);
```

7.
8. Write code to convert string `"123"` to number and explain.

Predict output:

```
let a;
console.log(typeof a);
```

9.
10. Write code to demonstrate type coercion with `==` and strict equality `===`.

Predict output:

```
console.log([] == false);
```

11.
12. Write code to check if a value is an array.

Predict output:

```
console.log(typeof function() {});
```

13.

14. Write code showing that functions are objects in JavaScript.

15. Write a program to demonstrate that `typeof` an array returns `object`.

Predict output:
```
console.log(typeof Symbol("id"));
```

16.
17. Write code showing that `typeof` on a class returns `function`.

Predict output:
```
console.log(typeof Infinity);
```

18.
19. Write a program that differentiates primitive and reference data types.

Predict output:
```
console.log(typeof []);
```

20.
21. Write code to convert boolean to string and vice versa.

Predict output:

```
console.log(typeof new Date());
```

22.
23. Write a program that creates a number from a string with spaces and explains the result.

Predict output:
```
console.log(typeof undefined);
```

24.
25. Write code to demonstrate use of `typeof` with `null`.

Predict output:
```
console.log(null == undefined);
```

26.

27. Write code that throws an error when trying to modify a symbol.

Predict output:
```
console.log(typeof /regex/);
```

28.
29. Write code to demonstrate `typeof` on a Map object.

Predict output:
```
console.log(typeof 123n);
```

30.
31. Write code that shows conversion from number to string using template literals.

Predict output:
```
console.log(typeof NaN);
```

32.
33. Write code that shows what happens when you add a string to a number.

Predict output:
```
console.log(typeof JSON);
```

34.
35. Write code demonstrating conversion of string to boolean.

Predict output:
```
console.log(typeof null);
```

36.
37. Write code showing that objects are mutable even when declared with `const`.

Predict output:

```
console.log(typeof Symbol.iterator);
```

38.

39. Write code that demonstrates converting a symbol to string throws an error unless explicitly converted.

Predict output:
```
console.log(typeof Array);
```

40.
41. Write code showing that arrays are objects and how to check it.

Predict output:

```
console.log(typeof undefinedVariable);
```

42.
43. Write code demonstrating the difference between `null` and `""` (empty string).

Predict output:
```
console.log(typeof 0);
```

44.
45. Write code demonstrating how to check for NaN correctly.

Predict output:
```
console.log(typeof Symbol("desc"));
```

46.
47. Write code to show how you can create a unique property key using Symbol.

Predict output:
```
console.log(typeof new WeakMap());
```

48.
49. Write code demonstrating difference between `typeof` and `instanceof`.

Predict output:
```
console.log(typeof (() => {}));
```

50.

3. Data Conversion (50 questions)

1. Write code to convert string `"123"` to number using `Number()`, `parseInt()`, and `+` operator.

Predict output:
`Number(" 123 ");`

2.
3. Write code that converts number to string using `.toString()` and template literals.

Predict output:
`parseInt("10.5abc");`

4.
5. Write code showing difference between `parseInt` and `Number` with decimal strings.

Predict output:
`Boolean("");`

6.
7. Write code to convert boolean to number and vice versa.

Predict output:
`!!"false";`

8.
9. Write code that converts an array to a string using `.join()`.

Predict output:
`Number(null);`

10.
11. Write code to safely convert a value to a number and check if it is `NaN`.

Predict output:

```
Number(undefined);
```

12.
13. Write code showing how to convert a string to a date object.

Predict output:

```
parseFloat("12.34abc");
```

14.
15. Write code to convert number to boolean.

Predict output:

```
Boolean("0");
```

16.
17. Write code to convert string `"true"` to boolean.

Predict output:

```
Number(true);
```

18.
19. Write code to convert date to string and back to date.

Predict output:

```
Number("0xF");
```

20.
21. Write code to demonstrate implicit type coercion in addition.

Predict output:

```
"5" + 3;
```

22.

23. Write code showing difference between explicit and implicit conversion.

Predict output:

```
"5" - 3;
```

24.
25. Write code to demonstrate automatic conversion in boolean context.

Predict output:

```
[] + [];
```

26.
27. Write code to convert number to hexadecimal string.

Predict output:

```
Number("123abc");
```

28.
29. Write code that converts a string with spaces to a number.

Predict output:

```
parseInt("   10");
```

30.
31. Write code demonstrating `.toFixed()` for rounding numbers.

Predict output:

```
Boolean(null);
```

32.
33. Write code to convert an object to a string using `JSON.stringify()`.

Predict output:

```
Number("Infinity");
```

34.
35. Write code to convert string `"false"` to boolean false correctly.

Predict output:
```
Number(" ");
```

36.
37. Write code showing how to parse numbers from user input safely.

Predict output:
```
"123" * 2;
```

38.
39. Write code showing how to convert boolean to string.

Predict output:
```
Number(true);
```

40.
41. Write code to convert string `"NaN"` to a number and check result.

Predict output:

```
Boolean(" ");
```

42.
43. Write code to demonstrate type conversion inside template literals.

Predict output:

```
"true" == true;
```

44.
45. Write code showing how to convert a function to string.

Predict output:
```
Number(false);
```

46.

47. Write code to convert number to exponential form string.

Predict output:

```
Number("0b1010");
```

48.
49. Write code to parse date strings of different formats.

Predict output:

```
parseInt("0x10");
```

50.

---

4. String (50 questions)

1. Write code to find the length of a string.

Predict output:

```
"hello".charAt(1);
```

2.
3. Write code to convert a string to uppercase and lowercase.

Predict output:

```
"hello world".indexOf("world");
```

4.
5. Write code to check if a string starts with a substring.

Predict output:

```
"hello world".includes("wor");
```

6.
7. Write code to extract a substring using `.slice()`.

Predict output:

```
"abc".repeat(3);
```

8.
9. Write code to trim spaces from a string.

Predict output:

```
"123".padStart(5, "0");
```

10.
11. Write code to split a string by spaces into an array.

Predict output:

```
"hello".replace("l", "L");
```

12.
13. Write code to check equality of strings ignoring case.

Predict output:

```
"abc".concat("def");
```

14.
15. Write code to reverse a string.

Predict output:

```
"  a  ".trim();
```

16.
17. Write code to convert a string to an array of characters.

Predict output:

```
"hello".substring(1, 4);
```

18.
19. Write code to check if a string ends with a certain substring.

Predict output:

```
"abcde".substr(1, 3);
```

20.
21. Write code to find the first non-repeated character in a string.

Predict output:

```
"hello world".split(" ");
```

22.
23. Write code to replace all instances of a substring using regex.

Predict output:

```
"12345".slice(-3);
```

24.
25. Write code to check if a string contains only digits.

Predict output:

```
"JavaScript".indexOf("script");
```

26.
27. Write code to pad a string to a certain length with characters.

Predict output:

```
"a,b,c".split(",");
```

28.
29. Write code to capitalize the first letter of a string.

Predict output:

```
"hello".startsWith("h");
```

30.
31. Write code to remove vowels from a string.

Predict output:

```
"hello".match(/l/g);
```

32.
33. Write code to count the occurrences of a character in a string.

Predict output:

```
"hello world".lastIndexOf("o");
```

34.
35. Write code to convert numbers to strings.

Predict output:

```
"foo".localeCompare("bar");
```

36.
37. Write code to convert camelCase string to snake_case.

Predict output:

```
"a,b,c".join("-");
```

38.
39. Write code to split a string by multiple delimiters.

Predict output:

```
"hello world".replace(/ /g, "_");
```

40.
41. Write code to check if a string is a palindrome.

Predict output:

```
"a".charCodeAt(0);
```

42.
43. Write code to repeat a string N times without using `.repeat()`.

Predict output:

```
"Hello World".toLowerCase();
```

44.
45. Write code to find the longest word in a string.

Predict output:

```
"Hello World".split(" ")[1];
```

46.
47. Write code to convert a string to title case.

Predict output:

```
"abcde".substring(2);
```

48.
49. Write code to remove duplicate characters from a string.

Predict output:

```
"abcde".charAt(10);
```

50.

---

5. Number and Math (50 questions)

1. Write code to round a number to the nearest integer.

Predict output:

```
Math.floor(4.9);
```

2.
3. Write code to generate a random number between 1 and 10.

Predict output:

```
Math.max(1, 5, 3);
```

4.
5. Write code to calculate the square root of a number.

Predict output:

```
0.1 + 0.2 === 0.3;
```

6.
7. Write code to calculate absolute value of a number.

Predict output:

```
Math.ceil(4.1);
```

8.
9. Write code to get the minimum of an array using `Math.min`.

Predict output:

```
Math.random() < 1;
```

10.
11. Write code to truncate decimal digits of a number.

Predict output:

```
Math.pow(2, 3);
```

12.

13. Write code to check if a number is integer.

Predict output:

```
Number.isNaN("NaN");
```

14.
15. Write code to parse a float from a string.

Predict output:

```
Math.sign(-5);
```

16.
17. Write code to calculate the factorial of a number using recursion.

Predict output:

```
Math.log(1);
```

18.
19. Write code to calculate the hypotenuse of a right triangle.

Predict output:

```
Math.trunc(4.9);
```

20.
21. Write code to generate a random integer between two numbers (inclusive).

Predict output:

```
Number.isInteger(4.0);
```

22.
23. Write code to calculate the cube root of a number.

Predict output:

```
Math.E;
```

24.
25. Write code to round a number to N decimal places.

Predict output:

```
Math.min();
```

26.
27. Write code to calculate the greatest common divisor (GCD) of two numbers.

Predict output:

```
Math.random() === Math.random();
```

28.
29. Write code to convert degrees to radians.

Predict output:
```
Math.abs(-0);
```

30.
31. Write code to calculate the distance between two points (x1, y1) and (x2, y2).

Predict output:

```
Math.round(4.5);
```

32.
33. Write code to generate a random floating-point number in a range.

Predict output:

```
Number.MAX_SAFE_INTEGER;
```

34.
35. Write code to check if a number is safe integer.

Predict output:

```
Math.log10(100);
```

36.
37. Write code to calculate the area of a circle given radius.

Predict output:

```
Math.cos(0);
```

38.
39. Write code to calculate permutations of n items taken r at a time.

Predict output:

```
Math.sin(Math.PI / 2);
```

40.
41. Write code to calculate the difference between two dates in days.

Predict output:

```
Math.random() > 0;
```

42.
43. Write code to convert a number to binary string.

Predict output:

```
Number.parseInt("101", 2);
```

44.
45. Write code to calculate compound interest given principal, rate, time.

Predict output:

```
Number.isFinite(Infinity);
```

46.

47. Write code to round a number up to the nearest 10.

Predict output:

```
Math.exp(1);
```

48.
49. Write code to generate a random hex color code.

Predict output:

```
 Math.log2(8);
```

50.

---

6. Date and Time (50 questions)

1. Write code to get the current date and time.

Predict output:
```
new Date(0);
```

2.
3. Write code to get the year, month, and day from a `Date` object.

Predict output:

```
new Date("2020-02-29").getDate();
```

4.
5. Write code to set the hours and minutes of a `Date` object.

Predict output:

```
new Date("invalid date").toString();
```

6.
7. Write code to convert a `Date` to ISO string.

Predict output:

```
Date.now() > 0;
```

8.
9. Write code to calculate difference in days between two dates.

Predict output:

```
new Date("2021-12-31").getMonth();
```

10.
11. Write code to add 7 days to the current date.

Predict output:

```
new Date().getDay();
```

12.
13. Write code to format date as `"DD-MM-YYYY"`.

Predict output:

```
new Date().toLocaleDateString("en-GB");
```

14.
15. Write code to get the UNIX timestamp from a date.

Predict output:

```
new Date(2025, 0, 1).getFullYear();
```

16.
17. Write code to parse a date string `"2023-07-23T10:00:00Z"`.

Predict output:

```
new Date().toDateString();
```

18.
19. Write code to get the number of milliseconds in a day.

Predict output:

```
new Date(86400000).getDate();
```

20.
21. Write code to compare two dates.

Predict output:

```
new Date(2020, 1, 29).toString();
```

22.
23. Write code to convert a date to UTC string.

Predict output:

```
new Date().getTimezoneOffset();
```

24.
25. Write code to get the weekday name of a date.

Predict output:

```
new Date(0).toISOString();
```

26.
27. Write code to create a date from components: year, month, day, hour, min, sec.

Predict output:

```
new Date("2022-02-30").toString();
```

28.

29. Write code to get the number of days in a given month and year.

Predict output:

```
new Date(2021, 11, 31).getDate();
```

30.
31. Write code to measure time taken by a function execution using `Date`.

Predict output:

```
new Date().getMilliseconds();
```

32.
33. Write code to convert a date to a readable string in a different locale.

Predict output:

```
new Date(2021, 1, 28).getMonth();
```

34.
35. Write code to create a `Date` object representing one week from now.

Predict output:

```
new Date().toTimeString();
```

36.
37. Write code to find the age in years given a birth date.

Predict output:

```
new Date("1970-01-01T00:00:00Z").getTime();
```

38.
39. Write code to check if a year is a leap year using date methods.

Predict output:

```
new Date("2020-02-29").getFullYear();
```

40.
41. Write code to parse date strings with different formats.

Predict output:

```
new Date("07/23/2025").toDateString();
```

42.
43. Write code to convert UNIX timestamp to a human-readable date.

Predict output:

```
new Date().toUTCString();
```

44.
45. Write code to get current time in milliseconds since epoch.

Predict output:

```
new Date(2025, 6, 23).getDay();
```

46.
47. Write code to add minutes to a `Date` object.

Predict output:

```
new Date().toISOString().includes("T");
```

48.
49. Write code to subtract two dates and get the difference in hours.

Predict output:

```
new Date("2025-07-23T00:00:00Z").getUTCDate();
```

7. Arrays (50 questions)

1. Write code to create an array with 5 elements.

Predict output:

```
[1,2,3].length;
```

2.
3. Write code to add an element to the end of an array.

Predict output:

```
[1, 2, 3].push(4);
```

4.
5. Write code to remove the last element of an array.

Predict output:

```
[1, 2, 3].pop();
```

6.
7. Write code to add an element to the beginning of an array.

Predict output:

```
[1, 2, 3].unshift(0);
```

8.
9. Write code to remove the first element of an array.

Predict output:

```
[1, 2, 3].shift();
```

10.

11. Write code to find the index of an element in an array.

Predict output:

```
[1, 2, 3].indexOf(2);
```

12.
13. Write code to check if an array includes a certain value.

Predict output:

```
[1, 2, 3].includes(4);
```

14.
15. Write code to create a copy of an array.

Predict output:

```
[...[1, 2, 3]];
```

16.
17. Write code to concatenate two arrays.

Predict output:

```
[1, 2].concat([3, 4]);
```

18.
19. Write code to flatten a nested array by one level.

Predict output:

```
[1, [2, 3]].flat();
```

20.
21. Write code to remove duplicates from an array.

Predict output:

```
[...new Set([1, 2, 2, 3])];
```

22.
23. Write code to reverse an array.

Predict output:

```
[1, 2, 3].reverse();
```

24.
25. Write code to sort an array of numbers ascending.

Predict output:

```
[3, 1, 2].sort();
```

26.
27. Write code to slice a portion of an array.

Predict output:

```
[1, 2, 3, 4].slice(1, 3);
```

28.
29. Write code to splice elements from an array.

Predict output:

```
let arr = [1, 2, 3];
arr.splice(1, 1);
```

30.
31. Write code to map over an array and square each number.

Predict output:

```
[1, 2, 3].map(x => x * x);
```

32.
33. Write code to filter an array for even numbers.

Predict output:

```
[1, 2, 3, 4].filter(x => x % 2 === 0);
```

34.
35. Write code to reduce an array to the sum of its elements.

Predict output:

```
[1, 2, 3, 4].reduce((a, b) => a + b, 0);
```

36.
37. Write code to find the maximum number in an array.

Predict output:

```
Math.max(...[1, 2, 3]);
```

38.
39. Write code to join array elements into a string separated by commas.

Predict output:

```
[1, 2, 3].join("-");
```

40.
41. Write code to check if an array is empty.

Predict output:

```
[].length === 0;
```

42.
43. Write code to find the last index of an element in an array.

Predict output:

```
[1, 2, 3, 2].lastIndexOf(2);
```

44.
45. Write code to create an array from arguments object inside a function.

Predict output:

```
Array.from("hello");
```

46.
47. Write code to fill an array with a value.

Predict output:

```
[1, 2, 3].fill(0);
```

48.
49. Write code to find if any element in an array satisfies a condition.

Predict output:

```
[1, 2, 3].some(x => x > 2);
```

---

8. Object (50 questions)

1. Create an object with three properties: name, age, city.

2. Access a property using dot notation and bracket notation.

3. Add a new property to an existing object.

4. Delete a property from an object.

5. Check if a property exists in an object.
6. Loop through all keys of an object using `for...in`.
7. Get an array of all keys using `Object.keys()`.
8. Get an array of all values using `Object.values()`.
9. Get an array of entries (key-value pairs) using `Object.entries()`.
10. Merge two objects using `Object.assign()`.
11. Use spread operator to clone an object.
12. Write code to freeze an object and show that it can't be changed.
13. Write code to seal an object and explain difference with freeze.
14. Create an object with a method that uses `this`.
15. Write code to demonstrate how `this` works inside an object method.
16. Use computed property names to create dynamic keys in an object.
17. Create a nested object and access nested properties.
18. Use destructuring to extract properties from an object.
19. Rename properties while destructuring.
20. Provide default values while destructuring an object.
21. Write code to demonstrate shallow copy of an object.
22. Explain and demonstrate deep copy of an object using JSON methods.
23. Write code to convert an object to JSON string using `JSON.stringify()`.
24. Parse a JSON string into an object using `JSON.parse()`.
25. Use `hasOwnProperty()` to check if a property belongs directly to an object.

26. Explain prototype chain and show how to access the prototype of an object.
27. Add a method to an object’s prototype.
28. Create an object using `Object.create()`.
29. Explain difference between own and inherited properties.
30. Write code to demonstrate enumerability of object properties.
31. Use `Object.defineProperty()` to create a property with specific descriptors.
32. Create a non-enumerable property and test with `for...in`.
33. Explain and show how getters and setters work in objects.
34. Create a read-only property using `Object.defineProperty()`.
35. Use `Object.freeze()` on nested objects and explain effect.
36. Write code to list all enumerable property names, including inherited ones.
37. Create an object with symbol keys and show how to access them.
38. Use `Object.getOwnPropertyNames()` and explain difference from `Object.keys()`.
39. Create an object with a method that returns the object’s keys.
40. Explain how to prevent extensions to an object.
41. Demonstrate prototype pollution risks with examples.
42. Write code to merge objects with conflicting keys, explaining overwrites.
43. Use `Object.entries()` and `Object.fromEntries()` together.
44. Write code to clone an object without prototype using `Object.create(null)`.
45. Show how to add multiple properties at once using `Object.defineProperties()`.
46. Create an object with dynamic getter property.

47. Use `in` operator to check property presence.
48. Write code to check if an object is empty (no own properties).
49. Explain how `this` differs inside arrow functions in object methods.
50. Write code to create an immutable object using proxies.

---

9. Object Constructor (50 questions)

1. Write a constructor function to create objects with name and age properties.
2. Create an object using the constructor function with `new`.
3. Add a method to the constructor’s prototype.
4. Explain difference between methods added inside constructor vs prototype.
5. Write code to check if an object is instance of a constructor.
6. Write a constructor that accepts parameters and assigns properties.
7. Override a prototype method for a specific object instance.
8. Write code to create multiple objects using a constructor function.
9. Demonstrate constructor function hoisting.
10. Write code to add static properties or methods to a constructor function.
11. Explain what happens if you call a constructor function without `new`.
12. Write code to create inheritance using constructor functions.
13. Show how to call parent constructor inside child constructor.
14. Use `Object.create()` to set prototype of an object created by constructor.
15. Write code to override prototype method in child constructor.

16. Create a constructor function that has private variables using closures.

17. Write code to list all properties (own and prototype) of an instance.

18. Explain difference between class constructors and function constructors.

19. Write code to convert a constructor function to ES6 class syntax.

20. Write code to demonstrate constructor function default parameters.

21. Create a constructor function with methods that use `this`.

22. Show prototype chain for an object created via constructor.

23. Write code to check enumerable properties of an object created via constructor.

24. Use `instanceof` to verify inheritance between constructors.

25. Write code to extend native constructors using function constructors.

26. Explain what happens if you assign a new object to prototype inside constructor.

27. Write code to add multiple methods to prototype using `Object.assign()`.

28. Write a constructor that creates objects with private properties accessible via getters.

29. Write code demonstrating constructor property on instances.

30. Write a constructor function that throws an error if called without `new`.

31. Write code to create circular references using constructors.

32. Explain performance difference between prototype and instance methods.

33. Write code to delete prototype methods and observe effect on instances.

34. Show how to add properties to the prototype after instance creation.

35. Write code demonstrating prototypal inheritance using constructor functions.

36. Explain prototype pollution in the context of constructor functions.

37. Write a constructor function that validates parameters.

38. Create a constructor function with a static method.

39. Explain how `new.target` can be used in constructor functions.

40. Write code to check if a property is own or prototype on a constructed object.

41. Write a constructor that mixes instance and prototype properties.

42. Write code to simulate private methods inside constructor functions.

43. Explain what happens if prototype is reassigned after instances are created.

44. Create a constructor function that returns a different object explicitly.

45. Write code to demonstrate chaining constructors using `.call()`.

46. Show how `this` behaves inside constructor vs prototype methods.

47. Write a constructor that logs every creation of an instance.

48. Write code to freeze prototype and observe effects.

49. Create a constructor function that supports method chaining.

50. Explain memory usage differences between methods on prototype vs instance.

---

10. Object Destructuring (50 questions)

1. Destructure an object into variables.
2. Destructure and assign default values to variables.
3. Rename variables while destructuring.
4. Destructure nested objects.
5. Destructure only some properties of an object.
6. Use rest operator to collect remaining properties during destructuring.

7. Destructure function parameters that are objects.
8. Write code to swap variables using object destructuring.
9. Destructure an array inside an object.
10. Destructure function return values that are objects.
11. Use destructuring with default values in function parameters.
12. Write code to destructure with computed property names.
13. Destructure object properties and assign to new variable names with defaults.
14. Write code to destructure nested arrays inside objects.
15. Use destructuring in a `for...of` loop over an array of objects.
16. Write code to destructure an object and collect all keys except one.
17. Write code to destructure and skip certain properties.
18. Use destructuring with symbols as keys.
19. Write code to destructure from function parameters with default object.
20. Explain error when destructuring undefined or null.
21. Write code to safely destructure nested objects using optional chaining.
22. Destructure properties from an object passed as an argument to an arrow function.
23. Destructure arrays and objects simultaneously.
24. Write code to destructure with rest operator and rename rest properties.
25. Destructure properties with special characters in keys.
26. Write code to destructure deeply nested objects with defaults.
27. Use destructuring to extract properties with fallback functions.
28. Destructure an object and create a new object with selected properties.

29. Write code to destructure array-like objects.

30. Use destructuring to pull values from Map entries.

31. Write code to destructure and ignore certain keys while copying others.

32. Explain how destructuring differs from accessing properties directly.

33. Write code to destructure with computed properties in object literals.

34. Destructure object properties inside class constructor parameters.

35. Write code to destructure and assign to variables declared outside destructuring block.

36. Destructure and rename properties from an imported module object.

37. Use destructuring to create aliases for nested properties.

38. Write code to destructure and swap two object properties.

39. Use destructuring with `try...catch` block to extract error message.

40. Destructure objects in `Promise.then()` callback parameters.

41. Write code to destructure and default nested object properties to empty objects.

42. Destructure object with methods and call a method.

43. Write code to destructure object with getters and setters.

44. Destructure and rename properties in a function returning an object.

45. Use destructuring to pull config options from an options object.

46. Write code to destructure and collect properties starting with a specific prefix.

47. Use destructuring in `async/await` with resolved object values.

48. Destructure objects with prototype inheritance and observe effects.

49. Write code to destructure and log properties in one line.

50. Destructure properties and create a shallow clone of the object.

11. Function (50 questions)

1. Write a function that returns the sum of two numbers.
2. Write a function that accepts any number of arguments and returns their sum.
3. Write a function that returns the factorial of a number using recursion.
4. Write a function expression and invoke it immediately (IIFE).
5. Write a named function and assign it to a variable.
6. Write a function that returns another function.
7. Write a function that uses default parameters.
8. Write a function that demonstrates the difference between `arguments` and rest parameters.
9. Write a function that swaps two variables using array destructuring.
10. Write a function that calculates the nth Fibonacci number using recursion.
11. Write a function that accepts a callback and invokes it after 2 seconds.
12. Write a function that returns true if a number is prime.
13. Write a function that uses closure to keep a private counter.
14. Write a function that memoizes results of a heavy calculation.
15. Write a function that uses the spread operator to accept multiple arguments.
16. Write a function that returns the maximum number from its arguments.
17. Write a function that accepts an object and prints its keys and values.
18. Write a function that reverses a string.

19. Write a function that converts a string to title case.
20. Write a function that removes duplicates from an array.
21. Write a function that flattens an array recursively.
22. Write a function that returns the type of a variable passed to it.
23. Write a function that swaps the case of letters in a string.
24. Write a function that checks if a string is a palindrome.
25. Write a function that calculates the sum of digits of a number.
26. Write a function that generates a random integer between two bounds.
27. Write a function that merges two arrays without duplicates.
28. Write a function that calculates the area of different shapes based on parameters.
29. Write a function that validates an email using regex.
30. Write a function that deep clones an object.
31. Write a function that returns the current timestamp.
32. Write a function that logs arguments passed to it.
33. Write a function that converts an object into a query string.
34. Write a function that debounces another function.
35. Write a function that throttles another function.
36. Write a function that uses recursion to sum nested arrays.
37. Write a function that counts occurrences of characters in a string.
38. Write a function that capitalizes the first letter of each word in a string.
39. Write a function that checks if two arrays have the same elements.
40. Write a function that extracts unique values from multiple arrays.

41. Write a function that calculates the GCD of two numbers.
42. Write a function that finds the longest word in a sentence.
43. Write a function that returns the nth element from the end of an array.
44. Write a function that pads a string to a given length.
45. Write a function that generates a UUID.
46. Write a function that sorts an array of objects by a property.
47. Write a function that converts a number to Roman numerals.
48. Write a function that counts the vowels in a string.
49. Write a function that removes falsy values from an array.
50. Write a function that formats a number as currency.

---

12. Function with Arrays and Objects (50 questions)

1. Write a function that takes an array and returns the sum of its elements.
2. Write a function that filters an array of objects by a property value.
3. Write a function that maps an array of objects to an array of one property.
4. Write a function that merges two arrays of objects based on an ID property.
5. Write a function that finds the object with the max value of a certain property.
6. Write a function that groups an array of objects by a key.
7. Write a function that flattens an array of objects’ nested arrays into a single array.
8. Write a function that counts objects in an array by a property.
9. Write a function that updates a property of an object in an array given the ID.

10. Write a function that removes an object from an array by property value.
11. Write a function that clones an array of objects deeply.
12. Write a function that sorts an array of objects by date property.
13. Write a function that sums all numeric properties in an array of objects.
14. Write a function that converts an array of key-value pairs into an object.
15. Write a function that creates an array of unique property values from objects.
16. Write a function that filters an array of objects with nested arrays based on nested criteria.
17. Write a function that maps an array of objects to formatted strings.
18. Write a function that converts an object into an array of entries and sorts them.
19. Write a function that extracts values of nested properties from objects.
20. Write a function that checks if all objects in an array satisfy a condition.
21. Write a function that finds the first object in an array matching a condition.
22. Write a function that reduces an array of objects to a single object merging properties.
23. Write a function that counts occurrences of nested property values in objects array.
24. Write a function that merges default properties into objects in an array.
25. Write a function that filters out duplicate objects from an array by a property.
26. Write a function that converts an array of objects to CSV format string.
27. Write a function that groups objects by multiple properties.
28. Write a function that checks if an object exists in an array by deep equality.
29. Write a function that updates multiple properties in objects inside an array.
30. Write a function that maps an array of objects asynchronously with promises.

31. Write a function that finds the index of the object with the minimum value of a property.
32. Write a function that extracts keys of objects that have a specific value.
33. Write a function that filters objects by multiple criteria combined with AND/OR logic.
34. Write a function that flattens arrays inside objects recursively.
35. Write a function that merges two arrays of objects removing duplicates by a key.
36. Write a function that creates a lookup map from an array of objects by a key.
37. Write a function that filters out objects with missing or null properties.
38. Write a function that calculates average of a numeric property in an array of objects.
39. Write a function that converts an array of objects into a tree structure.
40. Write a function that deeply compares two arrays of objects for equality.
41. Write a function that extracts nested object values and sums them.
42. Write a function that removes a nested object property from all objects in an array.
43. Write a function that toggles a boolean property on objects in an array by ID.
44. Write a function that maps an array of objects into another format asynchronously.
45. Write a function that finds duplicates in an array of objects by a key.
46. Write a function that sorts array of objects by multiple properties.
47. Write a function that counts total occurrences of a nested array property.
48. Write a function that filters objects based on dynamic property names.
49. Write a function that finds the max value of a nested property across objects.
50. Write a function that transforms an array of objects into an object keyed by a property.

---

13. Scope (50 questions)

1. Write code to demonstrate global vs local scope.

2. Write code to show how `var` behaves differently from `let` in block scope.

3. Write code to demonstrate function scope using `var`.

4. Write code to show block scope with `let` and `const`.

Predict output:

js
CopyEdit
```
{
  var a = 1;
  let b = 2;
}
console.log(a, b);
```

5.
6. Write code to demonstrate variable shadowing in nested functions.

7. Write code to show how redeclaration works for `var` but not for `let`.

8. Write code to demonstrate temporal dead zone (TDZ) with `let`.

9. Write code to explain difference between lexical scope and dynamic scope.

10. Write code where an inner function accesses outer function variables (closure).

Predict output:
```
let x = 10;
function test() {
  console.log(x);
  let x = 5;
}
test();
```

11.
12. Write code to demonstrate implicit global variables (without `var/let/const`).

13. Write code to show how `const` variables behave with objects.

14. Write code to demonstrate hoisting of `var` variables.

15. Write code to demonstrate no hoisting for `let` and `const`.

16. Write code to explain scope inside `for` loops with `var` vs `let`.

17. Write code to demonstrate scope of function parameters.

18. Write code where two functions share global variables.

19. Write code to demonstrate nested block scope.

20. Write code where closure preserves variable value after function execution.

Predict output:

```
var a = 1;
function foo() {
  console.log(a);
  var a = 2;
}
foo();
```

21.
22. Write code to show scope of variables in IIFE.

23. Write code where nested functions modify variables from parent scope.

24. Write code to demonstrate module scope (using ES modules).

25. Write code to explain scope differences inside `eval()`.

26. Write code to demonstrate function expressions and their scope.

27. Write code where reassigning outer scope variable affects inner function.

28. Write code where closure creates private variables.

29. Write code to demonstrate variable lifetime (creation to garbage collection).

30. Write code to explain how `window` object relates to global scope in browsers.

31. Write code to differentiate between script scope and module scope in ES6.

32. Write code to show scope chain resolution in nested functions.

33. Write code where variable name conflict occurs between global and function scope.

34. Write code demonstrating difference between `let` in loop and closure inside setTimeout.

35. Write code to demonstrate scope leakage without `"use strict"`.

36. Write code to explain `var` declaration hoisting with initialization.

37. Write code to explain shadowing in block scopes.

38. Write code where variable in parent scope is inaccessible due to shadowing.

39. Write code to show nested `try...catch` scope for variables.

40. Write code to demonstrate block-scoped function in strict mode.

41. Write code where arrow function uses variable from outer lexical scope.

42. Write code to explain difference between global object in Node.js and browser.

43. Write code to demonstrate scope difference in strict vs non-strict mode.

44. Write code to check variable accessibility after block execution.

45. Write code where variable hoisting changes expected output.

46. Write code to demonstrate scope of variables declared inside `switch` case.

47. Write code to explain scope of loop counters declared with `var` vs `let`.

48. Write code to show how `const` behaves in nested scopes.

49. Write code to illustrate scope isolation using IIFE.

50. Write code where closure remembers variable even after outer function returns.

14. This and Arrow Functions (50 questions)

1. Write code to show how `this` refers to global object in non-strict mode.
2. Write code to show how `this` is `undefined` in strict mode inside a function.
3. Write code to demonstrate `this` inside an object method.
4. Write code to show how arrow functions inherit `this` from surrounding scope.

Predict output:

```
let obj = { name: "JS", arrow: () => console.log(this.name) };
obj.arrow();
```

5.
6. Write code to demonstrate `this` inside nested objects.
7. Write code to fix `this` using `.bind()` in callbacks.
8. Write code to show difference between `call`, `apply`, and `bind`.
9. Write code to demonstrate `this` in event listeners.
10. Write code to show `this` inside constructor functions.
11. Write code to show `this` behavior in arrow functions used as constructors (error).
12. Write code to demonstrate `this` inside class methods.
13. Write code to show `this` inside static methods of classes.
14. Write code where arrow function used in setTimeout preserves `this`.
15. Write code to compare `this` in arrow function vs regular function in `setTimeout`.
16. Write code to demonstrate `this` in object method assigned to a variable.

17. Write code to show `this` inside IIFE.

18. Write code where arrow function inside object method accesses outer `this`.

19. Write code to demonstrate `this` in a function passed as callback.

20. Write code to show how `.call()` changes `this` value.

21. Write code where `this` changes based on function invocation context.

22. Write code to show `this` behavior in class field arrow functions.

23. Write code where `this` is lost in event handler without binding.

24. Write code to demonstrate lexical `this` in arrow functions inside classes.

25. Write code to fix `this` using self-assignment (`const self = this`) pattern.

Predict output:

js
CopyEdit
```
const obj = {
  a: 10,
  f: function() {
    const inner = () => console.log(this.a);
    inner();
  }
};
obj.f();
```

26.
27. Write code to compare `this` inside object literal vs function declaration.

28. Write code to demonstrate `this` in getter and setter methods.

29. Write code to show difference between arrow and regular function in array methods.

30. Write code to explain why arrow functions cannot be used as constructors.

31. Write code to demonstrate `this` in DOM event delegation handler.
32. Write code to show arrow function ignoring `new.target`.
33. Write code to demonstrate `this` behavior in async/await functions.
34. Write code to show arrow function inside Promise using `this`.
35. Write code to demonstrate global `this` in module vs non-module script.
36. Write code to show how `.bind()` works with partially applied arguments.
37. Write code where arrow function inside method causes unexpected `this` behavior.
38. Write code to compare `this` in strict mode vs non-strict arrow function.
39. Write code to demonstrate `this` in nested regular and arrow function mix.
40. Write code where `this` points to window in browser but undefined in Node.
41. Write code to explain why `this` in arrow functions cannot be dynamically bound.
42. Write code to demonstrate `this` inside class static arrow functions.
43. Write code to compare `this` in object methods defined via prototype vs inline.
44. Write code to demonstrate how `this` behaves with `Object.assign` methods.
45. Write code where binding `this` twice with `.bind()` does not change `this`.
46. Write code to demonstrate `this` in nested `setTimeout` and `setInterval`.
47. Write code where arrow function in constructor keeps parent `this`.
48. Write code to demonstrate `this` inside chained methods of object.
49. Write code to show how arrow function lexical `this` helps avoid `bind()` in React-like components.
50. Write code to illustrate pitfalls of using arrow functions as object methods.

15. IIFE (Immediately Invoked Function Expression) – 50 Questions

1. Write an IIFE that logs `"Hello World"`.
2. Write an IIFE that returns the sum of two numbers.
3. Create an IIFE that accepts parameters and logs them.
4. Write an IIFE that creates a private counter variable.
5. Create an IIFE that initializes a configuration object.

Predict output:

```
(function() {

  var x = 5;

})();

console.log(x);
```

6.
7. Write an arrow function IIFE that returns a string.
8. Write an IIFE that attaches a method to the `window` object.
9. Write an IIFE that immediately executes an async function.
10. Write an IIFE that prevents variable leakage to global scope.
11. Create an IIFE that logs current date and time.
12. Write an IIFE that initializes a module with private data.
13. Write an IIFE that creates a singleton object.

Predict output:

```
var result = (function(a, b) { return a + b })(5, 10);

console.log(result);
```

14.
15. Write an IIFE that returns another function and calls it immediately.

16. Write an IIFE that runs only if a condition is true.

17. Write an IIFE to polyfill a missing browser feature.

18. Write an IIFE that stores configuration settings in closure.

19. Write an IIFE that immediately sets event listeners.

20. Create an IIFE that initializes app data on page load.

21. Write an IIFE that logs `this` and explain result.

22. Write an IIFE using arrow function syntax with parameters.

23. Write an IIFE that prevents variable naming conflicts.

Predict output:

```
(function() {

  let a = 10;

  return function() { console.log(a); }

})()();
```

24.
25. Write an IIFE that modifies a global variable safely.

26. Create an IIFE that provides public methods but hides private variables (Module Pattern).

27. Write an async IIFE that fetches data from an API and logs it.

28. Create an IIFE that returns an object with getter and setter methods.

29. Write an IIFE that demonstrates lexical scoping.

30. Write an IIFE that adds custom methods to `Array.prototype`.

Predict output:
```
(() => { return typeof this })();
```

31.
32. Write an IIFE that immediately registers DOM events on elements.

33. Write an IIFE that logs environment details (browser or Node).

34. Write an IIFE that performs currency conversion and returns result.

35. Write an IIFE that accepts a callback and invokes it.

36. Write an IIFE that keeps track of number of times it’s been invoked.

37. Write an IIFE that returns current timestamp.

38. Write an IIFE that logs sum of elements in an array.

39. Write an IIFE that provides utility methods like `sum`, `max`, `min`.

40. Write an IIFE that returns different values based on environment (dev/prod).

41. Write an IIFE that demonstrates variable hoisting inside it.

42. Write an IIFE that demonstrates strict mode isolation.

43. Create an IIFE that accepts object destructuring as parameters.

44. Write an IIFE that runs only once even if called multiple times (singleton pattern).

Predict output:

```
const val = (function(x) { return x * x })(3);

console.log(val);
```

45.
46. Write an IIFE that creates a private event bus (publish/subscribe system).

47. Write an IIFE that caches results of a heavy computation.

48. Write an IIFE that creates a countdown timer.

49. Write an IIFE that dynamically adds styles to a page.
50. Write an IIFE that generates unique IDs each time it's called.

---

16. Control Statements (50 questions)

1. Write code to demonstrate `if` statement with multiple conditions.
2. Write code using `if...else if...else` to classify numbers as positive, negative, zero.
3. Write code using `switch` to map day numbers to day names.
4. Write code using `switch` with fallthrough behavior.

Predict output:

```
switch (2) {

  case 1:

  case 2: console.log("Two");

  case 3: console.log("Three");

}
```

5.
6. Write code using `for` loop to sum numbers 1 to 100.
7. Write code using `while` loop to print even numbers 1–20.
8. Write code using `do...while` loop to print numbers until user input is 0.
9. Write code using nested loops to print a multiplication table.
10. Write code using `break` to exit a loop early.
11. Write code using `continue` to skip odd numbers.

12. Write code using `return` inside a function to exit early.

Predict output:

```
for (let i = 0; i < 5; i++) {

  if (i === 3) break;

  console.log(i);

}
```

13.
14. Write code using `for...in` to iterate over object properties.
15. Write code using `for...of` to iterate over array elements.
16. Write code to demonstrate labeled statements with `break label`.
17. Write code to check leap year using `if...else`.
18. Write code to calculate factorial using `while` loop.
19. Write code to print Fibonacci series using `for` loop.
20. Write code to reverse a string using loop control statements.
21. Write code to find first prime number in a range using `break`.
22. Write code to skip multiples of 3 using `continue`.
23. Write code using nested `if` statements to check grading system.
24. Write code using `switch` to handle multiple user roles.
25. Write code to break out of nested loops using labels.
26. Write code to simulate retry logic using `do...while`.
27. Write code to sum digits of a number using `while` loop.

28. Write code to print all divisors of a number using `for` loop.
29. Write code to find largest number in array using loop.

Predict output:

```
 let i = 0;

while (i++ < 3) console.log(i);
```

30.
31. Write code to check palindrome using `for` loop.
32. Write code to simulate simple calculator using `switch`.
33. Write code using `for` loop to print pyramid pattern.
34. Write code to find greatest common divisor (GCD) using loop.
35. Write code using `if...else` to validate password strength.
36. Write code to count vowels in string using loop control statements.
37. Write code to print numbers divisible by 5 using `for...of`.
38. Write code to find smallest number in array using `while` loop.
39. Write code to print sum of squares up to `n` using `for` loop.
40. Write code to simulate traffic light using `switch`.
41. Write code to print prime numbers between 1–50 using `continue`.
42. Write code to print pattern of stars using nested loops.
43. Write code to validate user input repeatedly using `do...while`.
44. Write code to count digits of number using loop.
45. Write code to find LCM of two numbers using loops.

46. Write code using labeled `break` in nested loops.
47. Write code to find sum of even and odd separately using loop.
48. Write code to remove duplicates from array using loop control statements.
49. Write code to print reverse of number using `while` loop.
50. Write code to check Armstrong number using `for` loop.

---

17. Higher Order Loops (forEach, map, filter, reduce, etc.) – 50 Questions

1. Use `forEach` to print all elements of an array.
2. Use `forEach` to calculate the sum of numbers in an array.
3. Use `map` to create a new array with squares of all numbers.
4. Use `map` to convert an array of strings to uppercase.
5. Use `filter` to return all even numbers from an array.
6. Use `filter` to get words longer than 5 characters.
7. Use `reduce` to find the sum of elements in an array.
8. Use `reduce` to find the maximum number in an array.
9. Use `reduce` to flatten a nested array.
10. Combine `map` and `filter` to get squares of even numbers.
11. Use `forEach` to update values in an object array.
12. Use `map` to extract a specific property from an array of objects.

13. Use `filter` to remove duplicate numbers from an array.

14. Use `reduce` to count occurrences of each element in an array.

15. Use `map` to format an array of numbers as currency strings.

16. Use `filter` to remove falsy values from an array.

17. Use `reduce` to group objects by a property (e.g., category).

18. Use `forEach` to create a string from array values.

19. Use `map` to increment every number in an array by 2.

20. Use `reduce` to calculate the product of numbers in an array.

21. Use `filter` to find all prime numbers in an array.

22. Use `map` to create a boolean array indicating even/odd.

23. Use `reduce` to merge an array of objects into one object.

24. Use `forEach` to log index along with element value.

25. Use `filter` to extract objects with a certain property value.

26. Combine `map` and `reduce` to calculate average of numbers.

27. Use `reduce` to create a frequency map of characters in a string.

28. Use `map` to transform an array of strings into their lengths.

29. Use `filter` to remove numbers less than 10.

30. Use `reduce` to reverse an array.

31. Use `forEach` to mutate original array by doubling numbers.

32. Use `map` to add a new property to each object in an array.

33. Use `filter` to get students with grades above 80 from an object array.
34. Use `reduce` to find longest word in an array of strings.
35. Use `map` to convert temperatures from Celsius to Fahrenheit.
36. Use `filter` to select strings containing a specific substring.
37. Use `reduce` to partition numbers into even and odd arrays.
38. Use `forEach` to build a string with commas between values.
39. Use `map` to round all decimal numbers to nearest integer.
40. Use `filter` to exclude null or undefined values from array.
41. Use `reduce` to convert an array into an object with index keys.
42. Use `map` to generate HTML list elements from an array of strings.
43. Use `filter` to select unique values (simulate `Set`).
44. Use `reduce` to implement `map` functionality manually.
45. Use `reduce` to implement `filter` functionality manually.
46. Use `map` and `reduce` together to sum squares of numbers.
47. Use `forEach` to populate DOM elements from array data.
48. Use `filter` to select objects within a range (e.g., age 18–30).
49. Use `reduce` to find first duplicate in an array.
50. Use all four (`forEach`, `map`, `filter`, `reduce`) in a single problem: process student scores to calculate average of passed students.

---

18. Higher Order Functions (Functions as arguments/return values) – 50 Questions

1. Write a function that takes another function as argument and calls it.
2. Write a function that returns another function which logs “Hello”.
3. Write a function `multiplier(factor)` that returns a function to multiply numbers.
4. Write a function that accepts two callbacks: one for success and one for error.
5. Create a function that takes a number and returns a function to add to that number.
6. Write a function that executes a callback after a given delay (wrapper for `setTimeout`).
7. Write a function that returns different functions based on input parameter.
8. Create a function `compose` that combines two functions into one.
9. Write a `pipe` function that chains multiple functions.
10. Write a function that accepts multiple callbacks and executes them in sequence.
11. Write a function that memoizes results of another function (caching).
12. Write a function that debounces another function using closures.
13. Write a function that throttles another function using closures.
14. Write a function `once` that ensures a function is called only one time.
15. Write a function `after(n, fn)` that runs `fn` only after being called `n` times.
16. Write a higher order function to retry an operation N times before failing.
17. Create a function that transforms a synchronous function into asynchronous (using `Promise`).
18. Write a function `wrapLogger(fn)` that logs arguments and result of `fn`.

19. Write a function that takes array and callback, returning new array with transformed values.

20. Write a function `filterWith(fn)` that mimics `Array.filter` using callbacks.

21. Write a function `mapWith(fn)` that mimics `Array.map` using callbacks.

22. Write a function that takes multiple functions and returns their composition.

23. Write a function `delay(fn, ms)` that delays execution of `fn` by `ms` milliseconds.

24. Create a function that returns an object with multiple methods using closures.

25. Write a function `withCounter(fn)` that counts how many times `fn` is executed.

26. Create a higher order function that adds logging to any function.

27. Write a function `partial(fn, ...args)` for partial function application.

28. Write a function `curry(fn)` to convert function into curried form.

29. Write a function that takes a callback and invokes it with squared numbers of an array.

30. Create `oncePerSecond(fn)` that repeatedly calls `fn` every second.

31. Write a higher order function to validate inputs before calling callback.

32. Write `negate(fn)` that inverts the result of predicate function.

33. Write `composeAsync` to compose async functions that return Promises.

34. Write a higher order function that adds caching to fetch API calls.

35. Write a higher order function that ensures a callback is executed only if condition passes.

36. Write a higher order function to measure execution time of another function.

37. Write `tap(fn)` that runs `fn` with given value but returns value unchanged (for chaining).

38. Write `onceAndReturn(fn)` that runs `fn` first time and returns cached result later.

39. Write higher order function to create range validators (min, max).
40. Write a higher order function `chain(fn1, fn2, fn3)` to call multiple functions sequentially.
41. Write `wrapAsync(fn)` to convert callback-based function to promise-based.
42. Create a higher order function that logs arguments and rethrows errors.
43. Write a higher order function that auto-binds `this` for class methods.
44. Create `rateLimit(fn, limitPerSecond)` using closures and timestamps.
45. Write higher order function `delayAndRetry(fn, retries)` to retry failed operations.
46. Write `composeRight` that composes functions from right to left.
47. Write `identity` function and use it in composition chain.
48. Create `memoizeAsync(fn)` for caching async function results.
49. Write higher order function `logger(fn)` that logs before and after execution.
50. Write higher order function that applies a given function to each value in an object (like `mapValues`).

---

19. DOM (Document Object Model) – 50 Questions

1. Select an element by `id` and change its text content.
2. Select all elements with a specific class and change their background color.
3. Select all `p` tags and append `!` to their text.
4. Create a new `<div>` element and append it to the body.

5. Remove an element from the DOM by selecting it.
6. Replace an existing element with a new element dynamically.
7. Clone an existing DOM element and append it somewhere else.
8. Access and log all attributes of an element.
9. Modify the `src` attribute of an `<img>` tag dynamically.
10. Add a new class to an element without overwriting existing ones.
11. Remove a class from an element dynamically.
12. Toggle a class on an element (add if absent, remove if present).
13. Change inline styles of an element (e.g., `color`, `font-size`).
14. Get computed styles of an element and log them.
15. Change the inner HTML of an element with `innerHTML`.
16. Change only text content using `textContent`.
17. Create a list of items (`<ul>`) dynamically from an array.
18. Add multiple elements dynamically using `DocumentFragment`.
19. Access parent element of a specific node.
20. Access all child elements of a parent node.
21. Access the first and last child of an element.
22. Navigate to sibling elements (next/previous).
23. Change the value of an input element dynamically.
24. Get the value of an input field when user types.
25. Set multiple attributes on an element using `setAttribute`.

26. Remove an attribute from an element.

27. Check if an element has a specific attribute.

28. Dynamically create and insert a `<style>` tag into DOM.

29. Insert an element before another element using `insertBefore`.

30. Insert an element after another element (simulate `insertAfter`).

31. Move an existing element to a different position in DOM.

32. Change background color of body every second dynamically.

33. Select nested elements using `querySelector` and `querySelectorAll`.

34. Create a table dynamically and populate with data.

35. Remove all child nodes of a given parent element.

36. Change multiple styles at once using `cssText`.

37. Create tooltip element dynamically and attach to body.

38. Dynamically load an external script file.

39. Dynamically load an external CSS file.

40. Find the depth of nested DOM elements programmatically.

41. Highlight all `<a>` elements with a specific `href` pattern.

42. Change the `title` attribute of multiple elements at once.

43. Create a function to toggle visibility of an element.

44. Dynamically build a navigation menu from an array of links.

45. Get all form elements and log their names and values.

46. Programmatically scroll to a specific DOM element.

47. Create and append multiple elements in a loop efficiently.
48. Dynamically wrap an element with another element.
49. Remove duplicate child nodes from a parent element.
50. Build a real-time character counter for a textarea using DOM methods.

---

20. Events – 50 Questions

1. Add a `click` event listener to a button that logs “Clicked!”.
2. Add a `mouseover` event that changes element color.
3. Add a `mouseout` event to reset the color of an element.
4. Add a `keydown` event that logs pressed key code.
5. Add a `keyup` event that shows which key was released.
6. Add a `dblclick` event to toggle visibility of an element.
7. Add an `input` event to display typed value in real time.
8. Add a `focus` event to highlight input border.
9. Add a `blur` event to reset input border style.
10. Add a `submit` event handler to validate a form.
11. Add an event handler to prevent form submission.
12. Add a `contextmenu` event to create a custom right-click menu.
13. Add a `scroll` event to log scroll position.
14. Add a `resize` event to log window size dynamically.

15. Add a `load` event to show alert when page finishes loading.

16. Add an `unload` event to confirm navigation away from page.

17. Add a `change` event to log selected dropdown option.

18. Add an event to dynamically remove itself after firing once.

19. Add multiple event listeners to the same element.

20. Use `addEventListener` with `capture` phase enabled.

21. Use `stopPropagation()` to prevent bubbling.

22. Use `preventDefault()` to stop link navigation.

23. Demonstrate event delegation using parent container.

24. Log event target vs currentTarget in a nested element event.

25. Trigger custom events using `dispatchEvent()`.

26. Create a custom event and listen for it.

27. Add a `wheel` event to detect scroll direction.

28. Add a `dragstart` event to make an element draggable.

29. Add a `dragover` event to allow drop.

30. Add a `drop` event to handle dragged data.

31. Add a `touchstart` event for mobile touch interactions.

32. Add a `touchmove` event and log coordinates.

33. Add a `touchend` event to finalize a touch action.

34. Use `once: true` in `addEventListener` to fire event only once.

35. Create a function that attaches the same handler to multiple elements.

36. Dynamically remove event listener on button click.

37. Use `passive: true` option in event listener for performance.

38. Attach event listener to window object for global shortcuts.

39. Attach event listener to dynamically created element.

40. Demonstrate bubbling vs capturing with nested divs.

41. Build a simple click counter using event listeners.

42. Build a character counter using `input` event.

43. Build a to-do list with click event to mark items completed.

44. Build a toggle switch with `change` event on checkbox.

45. Create hover effect using `mouseenter` and `mouseleave`.

46. Attach multiple handlers to same event and remove one dynamically.

47. Build custom double-tap detection for mobile devices.

48. Demonstrate difference between inline event and `addEventListener`.

49. Build a key combination detector (e.g., `Ctrl + S`).

50. Build real-time form validation using multiple events (`blur`, `input`, `submit`).

---

21. Promise – 50 Questions

1. Create a Promise that resolves with `"Hello"` after 1 second.

2. Create a Promise that rejects with an error message after 2 seconds.

3. Chain `.then()` methods to transform resolved values step by step.

4. Handle rejection with `.catch()` and log the error message.
5. Use `.finally()` to log a message regardless of resolve or reject.
6. Combine two Promises with `Promise.all()` and log both results.
7. Use `Promise.race()` to get the first resolved promise.
8. Use `Promise.any()` to get first fulfilled result ignoring rejects.
9. Use `Promise.allSettled()` to log statuses of multiple promises.
10. Create a function that returns a Promise simulating API fetch.
11. Create a Promise that resolves with sum of two numbers.
12. Simulate a delay function using Promises (`sleep`).
13. Create a Promise that rejects randomly (50/50 chance).
14. Chain three asynchronous tasks using Promises.
15. Create a Promise that resolves conditionally based on input.
16. Convert a callback-based function to return a Promise.
17. Create a Promise wrapper around `setTimeout`.
18. Use `.then()` chaining to parse JSON data from a resolved string.
19. Nest promises (not recommended) and rewrite to chain them.
20. Create multiple Promises and handle them using `Promise.all`.
21. Convert synchronous function into a Promise-returning one.
22. Create a Promise that resolves only if input is even, else reject.
23. Chain `.then()` and `.catch()` to handle both success and error.
24. Create Promise-based delay and log timestamp after each delay.

25. Wrap `fetch()` in custom Promise to handle success/failure.

26. Create `retryPromise(fn, retries)` to retry failed Promise.

27. Implement a timeout mechanism for a Promise.

28. Handle multiple Promises where one rejects using `allSettled`.

29. Convert `XMLHttpRequest` to return a Promise.

30. Write a Promise to simulate database query.

31. Write a Promise that resolves after multiple chained timers.

32. Use Promises to fetch data sequentially from two APIs.

33. Implement a cancelable Promise.

34. Create a function returning a Promise that resolves to factorial of number.

35. Implement a queue of Promises executed one by one.

36. Use Promises to preload multiple images.

37. Create a Promise that logs intermediate results during execution.

38. Use Promises to simulate parallel API calls and merge data.

39. Write a Promise-based function for file reading (simulate).

40. Use `.then()` to return a new Promise inside callback.

41. Demonstrate how unhandled Promise rejections behave.

42. Wrap `setInterval` logic in a Promise and clear after condition.

43. Use Promise chaining to perform form validation steps.

44. Implement a polling mechanism using Promises (repeat until success).

45. Create a Promise that resolves with nested object data.

46. Use Promises to load and execute scripts in order.
47. Write a Promise that batches API calls in groups of 3.
48. Convert event-based code (like `onload`) into Promise style.
49. Use `Promise.race()` to simulate request timeout handling.
50. Combine `Promise.all()` and `Promise.race()` for fallback logic.

---

22. Asynchronous Functions (async/await) – 50 Questions

1. Write an async function that returns `"Hello Async"`.
2. Write an async function using `await` on a resolved Promise.
3. Use `await` to pause execution for 1 second (simulate sleep).
4. Handle errors in async functions using `try...catch`.
5. Combine multiple `await` calls sequentially.
6. Combine multiple `await` calls concurrently with `Promise.all`.
7. Write async function that fetches data from an API and logs it.
8. Use `await` to transform data before returning.
9. Create async function that throws error and catch outside.
10. Chain async functions (async calls another async).
11. Convert `.then()` chain into `async/await` equivalent.
12. Use `await` inside loop to process array sequentially.

13. Use `Promise.all` with `await` to process array in parallel.

14. Handle `Promise.race` with async/await.

15. Use `await` inside `try...finally`.

16. Write async IIFE that logs message immediately.

17. Convert callback-based function into async using `await`.

18. Use `await` for retry mechanism inside loop.

19. Build async function that validates user input with API call.

20. Simulate async file read with `await`.

21. Build async function that preloads images and logs when done.

22. Use async/await for chained mathematical computations.

23. Combine async with DOM updates (e.g., loading spinner).

24. Create async function that performs sequential database calls (mock).

25. Write async function to check multiple URLs for status.

26. Use `await` in nested `try...catch` blocks.

27. Convert Promise-based timer to async/await style.

28. Use async function to fetch and merge data from two APIs.

29. Implement async function for exponential backoff retry logic.

30. Write async function that reads/writes localStorage data.

31. Use `await` in recursion (e.g., countdown).

32. Build async function that simulates queue processing.

33. Combine `await` with `for...of` for ordered async processing.

34. Write async generator function (`async function*`) and iterate over results.

35. Use `await` to pause animation frames in `requestAnimationFrame`.

36. Build async function that polls server every 5 seconds until success.

37. Use async/await to validate form fields one by one.

38. Combine async and higher order functions (map + await).

39. Write async function to simulate real-time chat message fetch.

40. Use async/await with `fetch()` error handling (404, network error).

41. Convert multiple `.then()` API calls to async/await for readability.

42. Use async IIFE to initialize app state on page load.

43. Combine async/await with `AbortController` for cancelable fetch.

44. Implement async pagination: fetch next page on scroll.

45. Write async function that processes large files in chunks.

46. Combine async/await with web workers (simulate).

47. Use `await` inside class method and call from constructor.

48. Use async function to lazy load components dynamically.

49. Write async function with timeout using `Promise.race`.

50. Build async pipeline where each step depends on previous result.