JavaScript Refresher Quiz (30 Questions with Explanations)

1. What is the output of this code?

console.log(typeof null);

a) 'null'
b) 'object'
c) 'undefined'
d) Error
Correct Answer: b) 'object'
Explanation: In JavaScript, typeof null returns 'object' due to a historical bug retained for backward compatibility.

2. What will be printed?

let a = [1, 2]; let b = a; b.push(3); console.log(a);

a) [1, 2]
b) [1, 2, 3]
c) undefined
d) Error
Correct Answer: b) [1, 2, 3]
Explanation: Arrays are reference types. Assigning 'b = a' means they point to the same memory.

3. What is the output?

console.log(1 + '1' - 1);

a) '10'
b) 10
c) 11
d) NaN
Correct Answer: b) 10
Explanation: 1 + '1' = '11'; then '11' - 1 coerces '11' to number -> 11 - 1 = 10.

4. What is printed by this code?

setTimeout(() => console.log('A'), 0);

console.log('B');

a) B then A
b) A then B
c) Error
d) Undefined
Correct Answer: a) B then A
Explanation: setTimeout is asynchronous and scheduled after the current call stack, so 'B' logs first.

5. What will this log?

console.log(2 == '2');

console.log(2 === '2');

a) true false
b) false true
c) true true
d) false false
Correct Answer: a) true false
Explanation: '==' allows coercion, '===' checks both type and value.

6. What happens when you run this?

const x;

x = 10;

a) 10 is assigned
b) Error
c) undefined
d) null
Correct Answer: b) Error
Explanation: const requires a value at declaration time this throws a SyntaxError.

7. What does this return?

[1, 2, 3].map(n => { if (n > 1) return; });

a) [undefined, undefined, undefined]
b) [false, true, true]
c) [1, 2]
d) Error
Correct Answer: a) [undefined, undefined, undefined]
Explanation: No return value is returned explicitly from the callback undefined is default.

8. What is the result of this code?

let x = 10; function test() { console.log(x); let x = 5; } test();

a) 10
b) 5
c) ReferenceError
d) undefined
Correct Answer: c) ReferenceError
Explanation: 'let' is not hoisted like 'var' and lives in the temporal dead zone.

9. What is the output?

function outer() { let count = 0; return function inner() { count++; return count; } }

const fn = outer();

console.log(fn());

console.log(fn());

a) 1 and 1
b) 1 and 2
c) 0 and 1

d) Error

Correct Answer: b) 1 and 2

Explanation: Closures allow 'fn' to retain access to 'count' even after 'outer' has returned.

10. What is the result?

console.log([] + []);

a) ''

b) []

c) undefined

d) Error

Correct Answer: a) ''

Explanation: [] is coerced to '', and '' + '' results in an empty string.

11. What does typeof [] return?

a) 'array'

b) 'object'

c) 'list'

d) 'undefined'

Correct Answer: b) 'object'

Explanation: Arrays are technically objects in JavaScript, hence typeof [] is 'object'.

12. What does this log?

const obj = { name: 'Ali', greet: () => console.log(this.name) }; obj.greet();

a) 'Ali'

b) undefined

c) Error

d) 'this.name'

Correct Answer: b) undefined

Explanation: Arrow functions dont bind their own 'this'; 'this.name' refers to the global object.

13. console.log('5' + 1); console.log('5' - 1);

a) 6 and 4

b) '51' and 4

c) '6' and NaN

d) 5 and 4

Correct Answer: b) '51' and 4

Explanation: '5' + 1 = '51'; '5' - 1 = 4 due to coercion.

14. [1,2,3].filter(n => n % 2 === 0);

a) [1,3]

b) [2]

c) [1,2,3]

d) []

Correct Answer: b) [2]

Explanation: Only 2 is even it matches the filter condition.

15. Which of the following is falsy?

a) 'false'
b) '0'
c) 0
d) []
Correct Answer: c) 0
Explanation: Only the numeric 0 is falsy; the others are truthy values.

16. Which keyword defines a block-scoped constant?

a) var
b) let
c) const
d) define
Correct Answer: c) const
Explanation: 'const' creates a block-scoped, immutable binding.

17. Which is not a valid data type?

a) string
b) number
c) character
d) object
Correct Answer: c) character
Explanation: JavaScript has no 'character' type it uses strings for characters.

18. Which method modifies the original array?

a) push
b) map
c) filter
d) slice
Correct Answer: a) push
Explanation: 'push' appends items to the original array.

19. What does === check?

a) Value
b) Type
c) Value and type
d) Type coercion
Correct Answer: c) Value and type
Explanation: === checks both value and type with no coercion.

20. JSON.stringify({ a: 1 }) returns:

a) {a:1}
b) [object Object]
c) '{"a":1}'
d) null
Correct Answer: c) '{"a":1}'

Explanation: It returns a JSON-formatted string representation of the object.

21. typeof NaN is:

a) 'NaN'
b) 'number'
c) 'undefined'
d) 'object'
Correct Answer: b) 'number'
Explanation: Although NaN means 'Not a Number', its type is 'number'.

22. Array.isArray([]) returns:

a) false
b) true
c) undefined
d) error
Correct Answer: b) true
Explanation: Array.isArray checks if the value is an array.

23. 0.1 + 0.2 === 0.3?

a) true
b) false
c) NaN
d) undefined
Correct Answer: b) false
Explanation: Floating point precision makes 0.1 + 0.2 slightly off from 0.3.

24. Which adds to the end of an array?

a) unshift
b) concat
c) push
d) pop
Correct Answer: c) push
Explanation: 'push' appends elements to the end of an array.

25. !!'text' returns:

a) true
b) false
c) 'text'
d) undefined
Correct Answer: a) true
Explanation: !! forces a truthy/falsy conversion non-empty strings are truthy.

26. typeof function(){} is:

a) object
b) function
c) undefined
d) null

Correct Answer: b) function

Explanation: Functions have a specific typeof return 'function'.

27. console.log(0 == false); console.log(0 === false);

a) true true

b) false true

c) true false

d) false false

Correct Answer: c) true false

Explanation: '0 == false' because of coercion; '0 === false' is false due to type.

28. console.log('5' - - '2');

a) 7

b) NaN

c) 3

d) Error

Correct Answer: a) 7

Explanation: '5' and '2' are coerced to numbers; minus negative = plus.

29. setTimeout behavior?

a) Blocks the thread

b) Runs immediately

c) Runs after delay

d) Runs first

Correct Answer: c) Runs after delay

Explanation: setTimeout schedules the task after the current stack and delay.

30. What is the nullish coalescing operator?

a) &&

b) ||

c) ??

d) ?:

Correct Answer: c) ??

Explanation: The '??' operator returns the right-hand value if the left-hand side is null or undefined.

31. What happens here?

```
async function test() {
  try {
    await Promise.reject('Error!');
  } catch (e) {
    console.log('Caught');
  } finally {
    console.log('Finally');
  }
```

}

a) Error

b) Caught

c) Caught + Finally

d) Nothing

Correct Answer: c) Caught + Finally

Explanation: The rejection is caught, and then the finally block runs. So it logs 'Caught' and 'Finally'.