Name : Ricardo Ferreira

1. Name the three ways to declare a variable?

Basically we can declare variables in three different ways by using var, let and const keyword.

2. Which of the three variable declarations should you avoid and why?

We should avoid var because variables created using var will be accessible throughout the function ie they will live in the function scope.

But we want our variables to only be accessible within the block, it is created in.

3. What rules should you follow when naming variables?

The first character should be a letter. You can't use a number as the first character.

You can't use one of JavaScript's reserved words as a variable name.

Name your variables based on the terms of the subject area, so that the variable name clearly describes its purpose.

Create variable names by deleting spaces that separate the words. ...

Do not begin variable names with an underscore.

Do not use variable names that consist of a single character.

4. What should you look out for when using the + operator with numbers and strings?

When used on strings, the + operator is called the concatenation operator.

Adding two numbers, will return the sum, but adding a number and a string will return a string.

5. How does the % operator work?

Returns remainder of two operands.

6. Explain the difference between == and ===.

operands of different types are converted to numbers by the equality operator ==

A strict equality operator === checks the equality without type conversion. In other words, if a and b are of different types, then a === b immediately returns false without an attempt to convert them.

7. When would you receive a NaN result?

NaN represents a computational error. It is a result of an incorrect or an undefined mathematical operation.

8. How do you increment and decrement a number?

JavaScript has an even more succinct syntax to increment a number by 1. The increment operator ( ++ ) increments its operand by 1 ; that is, it adds 1 to the existing value. There's a corresponding decrement operator ( -- ) that decrements a variable's value by 1 . That is, it subtracts 1 from the value.

9. Explain the difference between prefixing and post-fixing increment/decrement operators.

Postfix decrement operator means the expression is evaluated first using the original value of the variable and then the variable is decremented(decreased). Prefix increment operator means the variable is incremented first and then the expression is evaluated using the new value of the variable.

10. What is operator precedence and how is it handled in JS?

Operator precedence determines how operators are parsed concerning each other. Operators with higher precedence become the operands of operators with lower precedence.

11. How do you log information to the console?

log() is a function in JavaScript which is used to print any kind of variables defined before in it or to just print any message that needs to be displayed to the user. Syntax: console. log(A);

12. What does unary plus operator do to string representations of integers?

It can convert all string representations of numbers, boolean values ( true and false ), and null to numbers. Numbers will include both integers, floats, hexadecimal, scientific (exponent) notation, and Infinity

13. What are the eight data types in JavaScript?   
String.

Number.

Boolean.

Null.

Undefined.

Symbol.

BigInt.

Object.

14. Which data type is NOT primitive?

non-primitive data type 'object'.

15. What is the relationship between null and undefined?

undefined is a variable that refers to something that doesn't exist, and the variable isn't defined to be anything. null is a variable that is defined but is missing a value

16. What is the difference between single, double, and backtick quotes for strings?

Double and single quotes are “simple” quotes. There’s practically no difference between them.

Backticks are “extended functionality” quotes. They allow us to embed variables and expressions into a string by wrapping them in ${…}

17. What is the term for embedding variables/expressions in a string?

Is ${…}.

18. Which type of quote lets you embed variables/expressions in a string?

Backticks quotes.

19. How do you embed variables/expressions in a string?

You can join together two variables: const one = 'Hello, '; const two = 'how are you?' ; const joined = `${one}${two}`; console

20. How do you escape characters in a string?

We can use the backslash ( \ ) escape character to prevent JavaScript from interpreting a quote as the end of the string. The syntax of \' will always be a single quote, and the syntax of \" will always be a double quote, without any fear of breaking the string

21. What is the difference between the slice/substring/substr string methods?

The substring() function is the most common way to get a substring in JavaScript. It takes two parameters: indexStart and indexEnd. It returns the portion of the string that starts at indexStart and ends the character immediately preceding indexEnd.

The key difference between substring() and substr() is that substr() has a different 2nd parameter. The first parameter to substr() is start, and the 2nd is length.

The slice() function is less common than substring() and substr(). However, it has the best aspects of both substring() and substr(). Like substring(), the slice() function takes the start and end indices as parameters, and is not considered a legacy function. Like substr(), the slice() function supports negative indices.

The slice() function seems like the clear winner out of the 3:

Not considered a "legacy function"

Supports negative indices

Less name confusion: there's no String#splice()

22. What are the three logical operators and what do they stand for?

&& Logical AND: true if both the operands/boolean values are true, else evaluates to false

|| Logical OR: true if either of the operands/boolean values is true . evaluates to false if both are false

! Logical NOT: true if the operand is false and vice-versa.

23. What are the comparison operators?

Equal (==) Returns true if the operands are equal.

Not equal (!=) Returns true if the operands are not equal.

Strict equal (===) Returns true if the operands are equal and of the same type. See also Object.is and sameness in JS

Strict not equal (!==) Returns true if the operands are of the same type but not equal, or are of different type.

Greater than (>) Returns true if the left operand is greater than the right operand.

Greater than or equal (>=) Returns true if the left operand is greater than or equal to the right operand.

Less than (<) Returns true if the left operand is less than the right operand.

Less than or equal (<=) Returns true if the left operand is less than or equal to the right operand.

24. What are truthy and falsy values?

In JavaScript, a truthy value is a value that is considered true when encountered in a Boolean context. All values are truthy unless they are defined as falsy. That is, all values are truthy except false , 0 , -0 , 0n , "" , null , undefined , and NaN . JavaScript uses type coercion in Boolean contexts

Truthy values are values that evaluate to True in a boolean context. Falsy values are values that evaluate to False in a boolean context.

25. What are the falsy values in JavaScript?

undefined, null, NaN, 0, "" (empty string), and false of course.

26. What are conditionals?

onditional statements control behavior in JavaScript and determine whether or not pieces of code can run.

27. What is the syntax for an if/else conditional?

If(condition)

{}

else(condition)

{}

28. What is the syntax for a switch statement?

switch(expression) {

case x:

// code block

break;

case y:

// code block

break;

default:

// code block

}

29. What is the syntax for a ternary operator?

condition ? expression1 : expression2

30. What is nesting?

Nesting is when you write something inside of something else. You can have a function inside of another function: function x () { function y() { // something; } } You can have an if condition inside of another if condition: if (daylight) { if (before 12) { //It's morning; } else { // it's afternoon; } }

31. What are functions useful for?

a function allows you to define a block of code, give it a name and then execute it as many times as you want

32. How do you invoke a function?

function myFunction(a, b) {

return a \* b;

}

myFunction(10, 2);

33. What are anonymous functions?

This is called an anonymous function, because it has no name.

34. What is function scope?

JavaScript has function scope: Each function creates a new scope. Variables defined inside a function are not accessible (visible) from outside the function. Variables declared with var , let and const are quite similar when declared inside a function. They all have Function Scope: function myFunction()

35. What are return values?

the values that a function returns when it has completed.

36. What are arrow functions

Arrow functions allow us to write shorter function syntax:

let myFunction = (a, b) => a \* b;

[Try it Yourself »](https://www.w3schools.com/js/tryit.asp?filename=tryjs_arrow_function)