WEEK:1 INTERVIEW QUESTIONS – JAVASCRIPT BASICS

Data Types in JavaScript

1. What are the different data types present in JavaScript?
   Primitive types: String, Number, BigInt, Boolean, Undefined, Symbol, and Null.
   Non-primitive type: Object.
2. Can you explain the difference between \`null\` and \`undefined\`?
   \`undefined\` indicates that a variable has been declared but has not yet been assigned a value. \`null\` is an assignment value that represents a deliberate non-value.
3. How would you determine the data type of a variable in JavaScript?
   Use the \`typeof\` operator for primitive data types, and for more complex checks (like arrays or null), use \`Array.isArray()\` or compare with \`null\` directly.

Variables in JavaScript

1. What are \`var\`, \`let\`, and \`const\` in JavaScript?
   \`var\` is function-scoped, \`let\` and \`const\` are block-scoped. \`const\` is used to declare constants.
2. What happens if you try to use a variable declared with \`let\` or \`const\` before its declaration?
   A ReferenceError will be thrown due to the Temporal Dead Zone (TDZ).
3. How does the scope of \`var\` differ from \`let\` and \`const\` when used inside a loop?
   Variables declared with \`var\` are function-scoped (or globally-scoped if declared outside of a function), while \`let\` and \`const\` are block-scoped, meaning they are only accessible within the loop block.

Hoisting

1. What is hoisting in JavaScript?
   Hoisting is JavaScript's default behavior of moving declarations to the top of their scope before code execution.
2. Are variables declared with \`let\` and \`const\` hoisted?
   Yes, variables declared with \`let\` and \`const\` are hoisted but cannot be accessed before their actual declaration line in the code, due to the TDZ.
3. Given a function that logs a variable declared later with \`var\`, what gets logged?
   The variable will be \`undefined\` due to hoisting, as the declaration (but not the initialization) is hoisted to the top.

Operators

1. What is the difference between \`==\` and \`===\` in JavaScript?
   \`==\` compares values after converting them to a common type (coercion), while \`===\` compares both value and type without conversion.

2. What does the `??` operator do?
   The nullish coalescing operator `??` returns its right-hand side operand when its left-hand side operand is `null` or `undefined`, and otherwise returns its left-hand side operand.
3. How can you use the ternary operator to assign a value based on a condition?
   `let result = condition ? value1 : value2;` assigns `value1` if `condition` is true, and `value2` otherwise.

Conditional and Looping Statements

1. What are the different types of loops available in JavaScript?
   For, while, and do-while loops.
   The `for` loop is the most commonly used loop and has three parts: initialization, condition, and increment/decrement.
   The `while` loop runs as long as the specified condition evaluates to true. The condition is evaluated before the execution of the loop's body.
   The `do-while` loop is similar to the `while` loop, but it executes the code block once before checking the condition. Thus, the loop will always execute at least once.
2. How does the `break` statement work in a nested loop?
   It terminates the current loop or switch statement and transfers program control to the statement following the terminated statement.

String and Its Methods in JavaScript

1. What is the purpose of the `trim()` method in JavaScript?
   `trim()` is used to remove whitespace from both ends of a string.
   let str = ' Hello JavaScript '; console.log(str.trim());
2. How can you convert a string to lower case?
   You can use the `toLowerCase()` method to convert a string to lower case.
   let str = 'Hello JavaScript';
   console.log(str.toLowerCase());
   Output: ‘hello javascript’
3. How do you check if a string contains a certain word in JavaScript?
   Use the `includes()` method to check if a string contains a specific word or substring.
   let str = 'Hello JavaScript';
   console.log(str.includes('JavaScript'));
   Output: true
4. What is the difference between `slice()` and `substring()` methods in JavaScript?
   Both methods return a part of the string, but `slice()` can accept negative indexes, which `substring()` cannot. `slice()` treats negative parameters as `str.length + startIndex`, whereas `substring()` swaps its two parameters if `indexStart` is greater than `indexEnd`, making it more forgiving.
   ```
   let str = 'Hello JavaScript';
   console.log(str.slice(-5));          // 'JavaScript'
   console.log(str.substring(0, 5));    // 'Hello'`
   ```

5. How would you reverse a string in JavaScript?

   Though JavaScript does not have a built-in method to reverse strings, you can reverse a string by converting it into an array, using the `reverse()` array method, and then joining it back into a string.

   ```
   let str = 'Hello';
   let reversed = str.split('').reverse().join('');
   console.log(reversed);                    //olleH
   ```

6. Write a function to check if a string is a palindrome.

   A palindrome is a word that reads the same backward as forward. You can check for a palindrome by reversing the string and comparing it to the original.

   ```
   function isPalindrome(str) {
       let reversed = str.split('').reverse().join('');
       return str === reversed;
       }
   console.log(isPalindrome('madam'));
   ```