Javascript

1. JavaScript Basics

- **Data Types:** Primitive (string, number, boolean, null, undefined, symbol, bigint), Non-Primitive (Object, Array, Function).
- **Variable Declaration:** `var`, `let`, `const`.
  - `var`: Function-scoped, hoisted.
  - `let` & `const`: Block-scoped, not hoisted.
- **Equality Operators:**
  - `==` (loose equality): Performs type coercion.
  - `===` (strict equality): Does not perform type coercion.

Interview Questions

1. What is the difference between `var`, `let`, and `const`?
2. Explain `undefined` vs `null`.
3. What are JavaScript primitive and non-primitive data types?

---

2. Functions

- **Types of Functions:**
  - Function Declaration: `function funcName() {}`.
  - Function Expression: `const funcName = function() {}`.
  - Arrow Function: `const funcName = () => {}`.
- **Scope:**
  - Global Scope.
  - Function Scope.
  - Block Scope.
- **Closures:** Functions that "remember" the environment in which they were created.

Interview Questions

1. What are closures? Provide an example.

2. Explain the difference between function declaration and function expression.

What is the output of the following code?

javascript
Copy code

```
for (var i = 0; i < 3; i++) {
  setTimeout(() => console.log(i), 1000);
}
```

3.

---

3. Objects and Prototypes

- **Object Creation:**
  - Using Object Literals: `const obj = { key: value }`.
  - Using `Object.create()`.
  - Using Constructor Functions or Classes.
- **Prototype:** Mechanism by which JavaScript objects inherit properties and methods.

Interview Questions

1. How does JavaScript inheritance work?
2. Explain the difference between `Object.create()` and constructor functions.
3. What is the prototype chain?

---

4. Asynchronous JavaScript

- **Concepts:**
  - Callbacks.
  - Promises.
  - `async/await`.
- **Event Loop:** Handles asynchronous operations using a queue and the stack.

Interview Questions

1. What is the difference between `setTimeout` and `setInterval`?
2. Explain the event loop in JavaScript.
3. How do `Promise.all` and `Promise.race` differ?

Question 1: Reverse a String

```
function reverseString(str) {
    let reversed = '';
    for (let i = str.length - 1; i >= 0; i--) {
        reversed += str[i];
    }
    return reversed;
}
console.log(reverseString('hello')); // Output: 'olleh'
```

Question 2: Check if a String is a Palindrome

```
function isPalindrome(str) {
    let reversed = '';
    for (let i = str.length - 1; i >= 0; i--) {
        reversed += str[i];
    }
    return str === reversed;
}
console.log(isPalindrome('madam')); // Output: true
console.log(isPalindrome('hello')); // Output: false
```

Question 3: Remove Duplicates from an Array

```
function removeDuplicates(arr) {
    let uniqueArr = [];
    for (let i = 0; i < arr.length; i++) {
        let isDuplicate = false;
        for (let j = 0; j < uniqueArr.length; j++) {
            if (arr[i] === uniqueArr[j]) {
                isDuplicate = true;
                break;
            }
        }
        if (!isDuplicate) {
            uniqueArr.push(arr[i]);
        }
    }
    return uniqueArr;
}
console.log(removeDuplicates([1, 2, 2, 3])); // Output: [1, 2, 3]
```

Explanation

1. **Reverse a String:**
   - Loops through the string from the last character to the first.
   - Appends each character to a new string (`reversed`).
2. **Check if a String is a Palindrome:**
   - Reverses the string manually using a loop (same logic as above).
   - Compares the original and reversed strings for equality.
3. **Remove Duplicates from an Array:**
   - Iterates through the array.
   - Checks if the current element exists in the `uniqueArr` using a nested loop.
   - Adds the element to `uniqueArr` only if it doesn't already exist.