**Assignment 1**

**Question 1: Answer The Following Questions**

1. **What do you understand by hoisting in JavaScript?**

hoisting allows for variable and function declarations to be used before they are declared in the code, but the behavior differs between var, let, const, and function declarations.

|  |  |  |
| --- | --- | --- |
| **Type of Declaration** | **Behavior** | **Example** |
| **var** | Declarations are hoisted to the top of the function or global scope but initialized with undefined. | console.log(a);  // undefined  var a = 5; |
| **let** and **const** | Declarations are hoisted to the top of the block scope but are not initialized. They are in a "temporal dead zone" until the declaration is encountered. | console.log(b);  // ReferenceError  let b = 5; |
| **function** | Function declarations are hoisted with their entire definition. | console.log(foo()); // 'bar'  function foo() {    return "bar";  } |

1. **Why is super used in JavaScript?**

super is used in class inheritance to call functions on an object's parent class. It serves two primary purposes:

|  |  |  |
| --- | --- | --- |
| **Purpose** | **Description** | **Example** |
| 1. **Calling Parent Constructor** | Ensures the parent class's constructor is called to initialize inherited properties. | class Dog extends Animal {    constructor(name, breed) {      super(name);  // Calls the parent class's constructor      this.breed = breed;    }  } |
| 1. **Calling Parent Methods** | Allows the subclass to invoke methods from the parent class, providing the ability to extend or override functionality while still accessing the original method. | speak() {      super.speak();  // Calls the parent class's speak method      console.log(`${this.name} barks.`);  } |

1. **What is let and const? And how it differs from var?**

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **var** | **let** | **const** |
| **Scope** | Function-scoped | Block-scoped | Block-scoped |
| **Hoisting** | Hoisted and initialized with undefined | Hoisted but not initialized (Temporal Dead Zone) | |
| **Reassignment** | Allowed | | Not allowed |
| **Initialization** | Can be declared without initialization | | Must be initialized at the time of declaration |
| **Redeclaration** | Allowed within the same scope | Not allowed within the same scope | |
| **Use Case** | Older code, function-wide variables | Modern code, block-scoped variables | Constants, configuration objects |
| **Example** | var a = 1; | let b = 2; | const c = 3; |

1. **Discuss the Rest parameter in ES6 What is Arrow function?**

**What are all its uses? How it differs from normal function?**

**Rest Parameter in ES6:**

* **Definition:** Allows a function to accept an indefinite number of arguments as an array.
* Example:

function sum(...numbers) {

    return numbers.reduce((acc, curr) => acc + curr, 0);

}

console.log(sum(1, 2, 3, 4)); // Output: 10

**Arrow Function in ES6:**

* **Definition:** Provides a concise syntax for writing function expressions.
* **Example:**

const add = (a, b) => a + b;

console.log(add(2, 3)); // Output: 5

**Uses of Arrow Functions:**

1. **Concise Syntax:** Arrow functions are shorter and easier to read.
2. **Implicit Return:** When there is only one expression in the function body, you can omit the curly braces and return keyword.
3. **Lexical this:** Arrow functions do not have their own this context; they inherit this from the parent scope. This makes them particularly useful in cases where you want to retain the context of this in callbacks.

**Differences from Normal Functions:**

1. **Syntax:** Arrow functions are more concise.
2. **this Context:** Arrow functions inherit this from the parent scope, whereas normal functions have their own this.
3. **arguments Object:** Normal functions have an arguments object; arrow functions do not.
4. **Constructor:** Normal functions can be used as constructors; arrow functions cannot.
5. **What is the difference between the readonly and disabled attributes for the <textarea> element**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **readonly** | **disabled** |
| **Definition** | Makes the <textarea> content non-editable but interactive. | Makes the <textarea> content non-editable and non-interactive. |
| **Interactivity** | Users can scroll, highlight, and copy text. | Users cannot scroll, highlight, or copy text. |
| **Form Submission** | Value is included in form submissions. | Value is not included in form submissions. |
| **Visual Appearance** | Typically appears similar to an editable <textarea>. | Typically appears greyed out and visually distinct. |
| **Use Case** | When you want to display text that can be copied or referenced but not edited. | When you want to completely prevent interaction with the text area. |
| **Example** | <textarea readonly>This text is readonly.</textarea> | <textarea disabled>This text is disabled.</textarea> |

1. **How do you specify units in the CSS? What are the different ways to do it?**

|  |  |  |
| --- | --- | --- |
| **Unit Type** | **Unit** | **Description** |
| **Absolute Units** |  |  |
| **Pixels** | px | Represents one pixel on the screen. |
| **Points** | pt | Represents 1/72 of an inch, traditionally used in print media. |
| **Inches** | in | Represents a physical inch. |
| **Centimeters** | cm | Represents a physical centimeter. |
| **Millimeters** | mm | Represents a physical millimeter. |
| **Picas** | pc | 1 pica equals 12 points. |
| **Relative Units** |  |  |
| **Percentages** | % | Relative to the parent element’s size. |
| **Ems** | em | Relative to the font-size of the element itself. If the font-size is 16px, then 1em is 16px. |
| **Root ems** | rem | Relative to the font-size of the root element (<html>). If the root font-size is 16px, then 1rem is 16px. |
| **View Width** | vw | 1vw is 1% of the viewport width. |
| **View Height** | vh | 1vh is 1% of the viewport height. |
| **View Min** | vmin | 1vmin is 1% of the smaller dimension (width or height) of the viewport. |
| **View Max** | vmax | 1vmax is 1% of the larger dimension (width or height) of the viewport. |
| **Character (ch)** | ch | Relative to the width of the zero (0) character of the element’s font. |
| **Ex (ex)** | ex | Relative to the height of the lowercase x of the element’s font. |
| **Cap Height** | cap | Relative to the height of the capital letters in the font. |
| **Line Height** | lh | Relative to the line-height of the element. |

1. **What property is used for changing the font face?**

The CSS property used for changing the font face is the font-family property.

1. **How to center align a div inside another div? [2 Ways]**

* **Method 1: Using Flexbox**

.parent {

  display: flex;

  justify-content: center; /\* Centers horizontally \*/

  align-items: center;     /\* Centers vertically \*/

  height: 100vh;           /\* Full viewport height, adjust as needed \*/

}

.child {

  width: 200px;  /\* Set width if needed \*/

  height: 100px; /\* Set height if needed \*/

}

* **Method 2: Using CSS Grid**

.parent {

  display: grid;

  place-items: center; /\* Centers both horizontally and vertically \*/

  height: 100vh;       /\* Full viewport height, adjust as needed \*/

}

.child {

  width: 200px;  /\* Set width if needed \*/

  height: 100px; /\* Set height if needed \*/

}

**Question2: True Or False**

1. False
2. True
3. True
4. False
5. False
6. True
7. True
8. True

**Question 3: MCQ**

1. JavaScript is 🡪 **2- asynchronous, non-blocking, single-threaded language.**
2. is the concept of object-oriented programming used to hide the internal representation, or state, of an object from the outside 🡪 **2) Encapsulation**

**Question 4: What is The Output**

1. TypeError: member.getFullName is not a function
2. 6 – 8
3. TypeError: freddie.colorChange is not a function
4. boolean – string
5. 0 – 1 – 4 – 2 – 3

* 0 is printed immediately when the script runs.
* After 2s, 1 is printed.
* Immediately after printing 1, a setTimeout with no delay is set up, and 4 is printed.
* After 1s from the previous setTimeout, 2 is printed.
* Finally, the innermost setTimeout with no delay executes, printing 3.

1. i is not defined
2. hello world – 10
3. [59.52, 83.7, 93]
4. ['Batman', 'Bane']

**Question 5: In VS Code**