Answers to Questions (while trying my hand at writing in Markdown)

Esad Kaya 300283659

Q1) What is the output of the following script? (3 marks)

```
for (let count = 1; count <= 10; count++) {
    if (count !== 5) {
        document.write(count + " ");
    }
}
```

Output: 1 2 3 4 6 7 8 9 10

Q2) Explain the concept of variable scope. What is the difference between a local variable and a global variable? (3 marks)

Variable Scope: Variable scope refers to the region of a program where a declared variable is accessible. It determines where a variable can be used and referenced.

Local Variable vs. Global Variable:

- **Local Variable:** A local variable is declared inside a function or a block (e.g., within `if` statements, `for` loops). It is only accessible within that specific function or block where it is defined. Once the function or block execution finishes, the local variable ceases to exist.
- **Global Variable:** A global variable is declared outside of any function or block. It is accessible from anywhere in the program, including inside functions and blocks. Global variables persist throughout the entire execution of the program.

Q3) Fill in the blanks in each of the following statements (4 marks):

a. The elements of an array are related by the fact that they have the same **name**. b. A **constructor** can be used to specify the type of an array and the number of elements to be allocated. c. The three ways to return control from a called function to a caller are **return**, **reaching the end of the function**, and **throwing an exception**. d. Variables declared in a block or in a function’s parameter list are of **local** scope.

Q4) State whether each of the following is true or false. If false, explain why. (4 marks)

a. The `Math.PI` property represents the mathematical constant π. **True**

b. The `localStorage` object can store any JavaScript data type, including objects and arrays, without any conversion. **False**. `localStorage` only stores strings. Objects and arrays must be converted to JSON strings (e.g., using `JSON.stringify()`) before being stored, and parsed back (e.g., using `JSON.parse()`) when retrieved.

c. The `String` method `charAt(0)` returns the first character of a string. **True**

d. `JSON.parse()` is used to convert a JavaScript object into a JSON string for storage or transmission. **False**. `JSON.parse()` is used to convert a JSON string into a JavaScript object. `JSON.stringify()` is used to convert a JavaScript object into a JSON string.

Q5) Explain the concept of event bubbling. How can it be stopped? (3 marks)

Event Bubbling: Event bubbling is a mechanism in the DOM (Document Object Model) where an event, when triggered on an element, first executes on that element, then propagates upwards through its ancestors in the DOM tree, from the innermost element to the outermost `document` object. For example, if you click a button inside a `div`, the click event will first fire on the button, then on the `div`, then on the `body`, and finally on the `document`.

How to stop it: Event bubbling can be stopped using the `event.stopPropagation()` method within an event listener. When called, this method prevents the event from propagating further up the DOM tree to parent elements.

Q6) Explain the relationship between parent nodes, child nodes, and sibling nodes in the DOM tree. (3 marks)

In the DOM (Document Object Model) tree, elements are organized in a hierarchical structure, representing the logical structure of an HTML or XML document.

- **Parent Node:** A parent node is an element that has one or more direct descendants. Every node in the DOM tree, except the root node (e.g., `<html>`), has exactly one parent node.
- **Child Nodes:** Child nodes are direct descendants of a parent node. An element can have multiple child nodes. For example, `<body>` is a child of `<html>`, and `<p>` and `<div>` might be children of `<body>`.
- **Sibling Nodes:** Sibling nodes are nodes that share the same parent node. They are at the same level in the DOM hierarchy. For example, if a `<div>` and a `<p>` element are both direct children of the same `<body>` element, then the `div` and `p` are siblings.