Copilot

Sure! Here's how you can implement a binary search in JavaScript within an HTML file. Binary search is an efficient algorithm for finding an item from a sorted list of items. It works by repeatedly dividing in half the portion of the list that could contain the item, until you've narrowed down the possible locations to just one.

HTML and JavaScript Example

```
<!DOCTYPE html>
<html>
<head>
    <title>Binary Search Example</title>
</head>
<body>
    <script>
        function binarySearch(arr, x) {
            let start = 0, end = arr.length - 1;

            while (start <= end) {
                let mid = Math.floor((start + end) / 2);

                // Check if x is present at mid
                if (arr[mid] === x) return mid;

                // If x greater, ignore left half
                if (arr[mid] < x) start = mid + 1;
                // If x is smaller, ignore right half
                else end = mid - 1;
            }

            // If we reach here, then the element was not present
            return -1;
        }

        // Example usage:
        const sortedArray = [2, 3, 4, 10, 40];
        const target = 10;
        const result = binarySearch(sortedArray, target);

        if (result !== -1) {
            console.log("Element found at index: " + result);
        } else {
            console.log("Element not found in array.");
        }
    </script>
</body>
</html>
```

Explanation:

1. **Function Definition**: The `binarySearch` function takes a sorted array `arr` and the target value `x` as inputs.
2. **Initialization**: We initialize `start` to 0 and `end` to the last index of the array.
3. **Loop**: We use a `while` loop that continues as long as `start` is less than or equal to `end`.
4. **Middle Calculation**: Calculate the middle index `mid`.
5. **Comparison**:
    - If the middle element is equal to `x`, return the middle index.
    - If the middle element is less than `x`, adjust the `start` index to `mid + 1`.
    - If the middle element is greater than `x`, adjust the `end` index to `mid - 1`.
6. **Return**: If the element is not found, return `-1`.

This example demonstrates a simple and efficient way to perform a binary search in JavaScript. If you have any questions or need further assistance, feel free to ask!