**Q5: Searching in a Rotated Sorted Array**: Solve the problem of searching for a specific element in a **rotated sorted array** using a modified **binary search** algorithm.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Rotated Sorted Array Search</title>

<style>

body { font-family: Arial, sans-serif; padding: 20px; }

input, button { padding: 10px; margin: 5px; }

</style>

</head>

<body>

<h1>Search in Rotated Sorted Array</h1>

<label for="arr">Enter Rotated Sorted Array:</label>

<input type="text" id="arr" placeholder="e.g., 4,5,6,7,0,1,2" />

<label for="target">Target:</label>

<input type="text" id="target" placeholder="Target number" />

<button onclick="search()">Search</button>

<h3 id="result"></h3>

<script>

function search() {

const arr = document.getElementById('arr').value.split(',').map(Number);

const target = Number(document.getElementById('target').value);

const result = rotatedBinarySearch(arr, target);

document.getElementById('result').innerText = result !== -1 ? `Target found at index: ${result}` : 'Target not found';

}

function rotatedBinarySearch(arr, target) {

let left = 0, right = arr.length - 1;

while (left <= right) {

const mid = Math.floor((left + right) / 2);

if (arr[mid] === target) return mid;

// Determine which side is sorted

if (arr[left] <= arr[mid]) {

if (arr[left] <= target && target < arr[mid]) {

right = mid - 1;

} else {

left = mid + 1;

}

} else {

if (arr[mid] < target && target <= arr[right]) {

left = mid + 1;

} else {

right = mid - 1;

}

}

}

return -1; // Not found

}

</script>

</body>

</html>