JavaScript Next Greater Element

Challenge

The next greater element of some element `x` in an array is the first greater element that is to the right of `x` in the same array.

You are given two distinct 0-indexed integer arrays `nums1` and `nums2`, where `nums1` is a subset of `nums2`.

For each `0 <= i < nums1.length`, find the index `j` such that `nums1[i] == nums2[j]` and determine the next greater element of `nums2[j]` in `nums2`. If there is no next greater element, then the answer for this query is `-1`.

Return an array `ans` of length `nums1.length` such that `ans[i]` is the next greater element as described above.

1st Example

```
Input: nums1 = [4,1,2], nums2 = [1,3,4,2]
Output: [-1,3,-1]
Explanation: The next greater element for each value of
             nums1 is as follows:
             - 4 is underlined in nums2 = [1,3,4,2].
               There is no next greater element, so the
               answer is -1.
             - 1 is underlined in nums2 = [1,3,4,2]. The next
               greater element is 3.
             - 2 is underlined in nums2 = [1,3,4,2]. There is
               no next greater element, so the answer is -1.
```

2nd Example

```
Input: nums1 = [2,4], nums2 = [1,2,3,4]
Output: [3,-1]
Explanation: The next greater element for each value of
             nums1 is as follows:
             - 2 is underlined in nums2 = [1,2,3,4]. The next
               greater element is 3.
             - 4 is underlined in nums2 = [1,2,3,4]. There is
               no next greater element, so the answer is -1.
```

Constraints

- `1 <= nums1.length <= nums2.length <= 1000`
- `0 <= nums1[i], nums2[i] <= 10⁴`
- All integers in `nums1` and `nums2` are unique.
- All the integers of `nums1` also appear in `nums2`.

Solution

```
const nextGreaterElement = (nums1, nums2) => {
    const map   = {},
          stack = [];
```

Solution continues on next page...

```
    nums2.forEach(n => {
        while (stack.length > 0 &&
               stack[stack.length - 1] < n) {
                   map[stack.pop()] = n;
        }

        stack.push(n);
    });

    return nums1.map(n => map[n] || -1);
};
```

Explanation

I've coded a function called `nextGreaterElement` that takes in two arrays `nums1` and `nums2`. The purpose of this function is to find the next greater element for each element in `nums1` from the corresponding position in `nums2` and return a new array with those values.

Inside the function, an empty object called `map` and an empty array called `stack` are initialized.

A `forEach` loop is used to iterate over each element `n` in the `nums2` array. Within the loop, it checks if the `stack` array is not empty and if the last element in the `stack` array is smaller than `n`.

If the condition is true, it means that the next greater element for the last element in the `stack` array has been found. In this case, the function pops the last element from the `stack` array and assigns `n` as the value for the popped element in the `map` object.

After that, the function pushes `n` into the `stack` array.

The loop continues until all elements in the `nums2` array have been processed. By the end of the loop, the `map` object contains the next greater element for each element in `nums2`.

Finally, a new array is returned by using the `map` method on the `nums1` array. Within the `map` function, it checks if the current element `n` exists as a key in the `map` object. If it does, the corresponding value from the `map` object is returned. Otherwise, `-1` is returned.

In summary, this function finds the next greater element for each element in `nums1` from the corresponding position in `nums2`. It achieves this by using a stack to keep track of elements in `nums2` and an object to store the next greater elements. The resulting values are returned as a new array.