496. Next Greater Element I

Easy 2284 2758 Add to List Share

You are given two integer arrays `nums1` and `nums2` both of **unique** elements, where `nums1` is a subset of `nums2`.

Find all the next greater numbers for `nums1`'s elements in the corresponding places of `nums2`.

The Next Greater Number of a number `x` in `nums1` is the first greater number to its right in `nums2`. If it does not exist, return `-1` for this number.

Example 1:

```
Input: nums1 = [4,1,2], nums2 = [1,3,4,2]
Output: [-1,3,-1]
Explanation:
For number 4 in the first array, you cannot find the next greater number for it in the second array, so output -1.
For number 1 in the first array, the next greater number for it in the second array is 3.
For number 2 in the first array, there is no next greater number for it in the second array, so output -1.
```

Example 2:

```
Input: nums1 = [2,4], nums2 = [1,2,3,4]
Output: [3,-1]
Explanation:
For number 2 in the first array, the next greater number for it in the second array is 3.
For number 4 in the first array, there is no next greater number for it in the second array, so output -1.
```

Constraints:

- `1 <= nums1.length <= nums2.length <= 1000`
- `0 <= nums1[i], nums2[i] <= 10^4`
- All integers in `nums1` and `nums2` are **unique**.
- All the integers of `nums1` also appear in `nums2`.

Follow up: Could you find an `O(nums1.length + nums2.length)` solution?

1. Get next greater value for nums2 and store into map.

2. Get answer for nums1 from map directly.

```cpp
class Solution {
public:
    vector<int> nextGreaterElement(vector<int>& nums1, vector<int>& nums2) {
        stack<int> stack;
        unordered_map<int, int> map;

        // maintain monotonic stack
        // store next greater element for nums2 into map
        for (auto num : nums2) {
            while (!stack.empty() && stack.top() < num) {
                map.insert(pair<int, int>(stack.top(), num));
                stack.pop();
            }
            stack.push(num);
        }

        while (!stack.empty()) {
            map.insert(pair<int, int>(stack.top(), -1));
            stack.pop();
        }

        // iterate nums1 and find the result
        vector<int> res;
        for (auto num : nums1) {
            res.push_back(map.at(num));
        }

        return res;
    }
};
```