Problem Link:

https://leetcode.com/problems/intersection-of-two-arrays/

Problem Description:

Given two integer arrays `nums1` and `nums2`, return *an array of their intersection*. Each element in the result must be **unique** and you may return the result in **any order**.

Problem Approach:

Sets and iteration of the arrays.

Sample Test Case:

Input: nums1 = [1,2,2,1], nums2 = [2,2]

Output: [2]

Solution:

We start by initializing a set, which will store the elements of the first array, and an empty result array. We do it so we can search for elements in O(1) time complexity. After adding all the elements of the first array to the set, we start iterating the second array, and for each element, we check if it exists in the set. If it does, then we add it to the result list, as it's a common element in both of the arrays. Following that, we remove the element from the set, since we don't want duplicate values in the final result. By the end, we'd have the result array as our answer!

Example:
Let's say if we have [1,2,3,4,5] and [0,1,4,4,5]. So, the common elements are [1,4,5]. If we do not remove the 4 from the set once the 4 at index 2 of the second array is added to the result array, the next 4 will also get added to the result array, since 4 will be present in the set. The time complexity of this solution is O(n+m) and space complexity is O(n).

Code (Python):

```python
def intersection(self, nums1: List[int], nums2: List[int]) -> List[int]:
       nums1Elements = set()
       result = []
       for num in nums1:
           nums1Elements.add(num)
       for num in nums2:
           if num in nums1Elements:
               result.append(num)
               nums1Elements.remove(num)
       return result
```