442. Find all duplicates in an Array:

Given an integer array `nums` of length `n` where all the integers of `nums` are in the range `[1, n]` If each integer appears **at most twice**, return *an array of all the integers that appears **twice***.

You must write an algorithm that runs in `O(n)` time and uses only *constant* auxiliary space, excluding the space needed to store the output

Example 1:

```
Input: nums = [4,3,2,7,8,2,3,1]
Output: [2,3]
```

Example 2:

```
Input: nums = [1,1,2]
Output: [1]
```

Example 3:

```
Input: nums = [1]
Output: []
```

Constraints:

- `n == nums.length`
- `1 <= n <= 105`
- `1 <= nums[i] <= n`
- Each element in `nums` appears **once** or **twice**.

Solution: Time = O(n) and space = O(n)

```
class Solution:
    def findDuplicates(self, nums: List[int]) -> List[int]:
        lst = []
```

```
        s = set()
        for i in nums:
            if i not in s:
                s.add(i)
            else:
                lst.append(i)
        return lst
```

Optimal solution: Time = O(n) and Space = O(1)

```
class Solution:
    def findDuplicates(self, nums: List[int]) -> List[int]:
        result = []

        for num in nums:
            index = abs(num) - 1

            if nums[index] < 0:
                result.append(abs(num))
            else:
                nums[index] = -nums[index]

        return result
```