Two Sum

https://leetcode.com/problems/two-sum/submissions/1508244215/

Two Sum

Easy

Given an array of integers `nums` and an integer `target`, return the indices `i` and `j` such that `nums[i] + nums[j] == target` and `i != j`.

You may assume that *every* input has exactly one pair of indices `i` and `j` that satisfy the condition.

Return the answer with the smaller index first.

Example 1:

```
Input:
nums = [3,4,5,6], target = 7

Output: [0,1]
```

Copy

Explanation: `nums[0] + nums[1] == 7`, so we return `[0, 1]`.

Example 2:

```
Input: nums = [4,5,6], target = 10

Output: [0,2]
```

Copy

Example 3:

```
Input: nums = [5,5], target = 10
```

```
Output: [0,1]
```

Copy

Constraints:

`2 <= nums.length <= 1000-10,000,000 <= nums[i] <= 10,000,000-10,000,000 <= target <= 10,000,000`

- `2 <= nums.length <= 1000`
- `10,000,000 <= nums[i] <= 10,000,000`
- `10,000,000 <= target <= 10,000,000`

```
class Solution:
    def twoSum(self, nums: List[int], target: int) -> List[int]
        if nums == []:
            return []

        stack = []

        for i in range(len(nums)):
            complement = target - nums[i]
            if complement in stack:
                return [nums.index(complement), i]
            else:
                stack.append(nums[i])
```

If i ever attempt this again, maybe use a hash map for an efficient look up