001-两数之和

题目

给定一个整数数组 `nums` 和一个目标值 `target`，请你在该数组中找出和为目标值的那 **两个** 整数，并返回他们的数组下标。

你可以假设每种输入只会对应一个答案。但是，你不能重复利用这个数组中同样的元素。

示例:

```
给定 nums = [2, 7, 11, 15], target = 9

因为 nums[0] + nums[1] = 2 + 7 = 9
所以返回 [0, 1]
```

思路

对于每一个元素 x，我们都需要去检查 $target - x$ 是否在数组中。最简单的思路就是直接遍历整个数组来查找，但是这样时间复杂度达到 $O(n)$。

因此我们需要一种更有效的方法来检查数组中是否存在目标元素。如果存在，我们需要找出它的索引。保持数组中的每个元素与其索引相互对应的最好方法是什么？哈希表。通过以空间换取速度的方式，我们可以将查找时间从 $O(n)$ 降低到 $O(1)$。

最简单的实现使用了两次迭代。在第一次迭代中，我们将每个元素的值和它的索引添加到表中。然后，在第二次迭代中，我们将检查每个元素所对应的目标元素（$target - nums[i]$）是否存在于表中。注意，该目标元素不能是 $nums[i]$ 本身！

更巧妙的实现是在进行迭代并将元素插入到表中的同时，我们还会回过头来检查表中是否已经存在当前元素所对应的目标元素。如果它存在，那我们已经找到了对应解，并立即将其返回。

我的解答:

```python
def twoSum(self, nums: List[int], target: int) -> List[int]:
        temp = {}
        for index,num in enumerate(nums):
            if num in temp:
                return [temp[num], index]
            else:
                temp[target - num] = index
```