Leetcode1：两数之和：

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

你可以假设每种输入只会对应一个答案。但是，数组中同一个元素在答案里不能重复出现。

你可以按任意顺序返回答案。

示例 1：

```
输入：nums = [2,7,11,15], target = 9
输出：[0,1]
解释：因为 nums[0] + nums[1] == 9 ，返回 [0, 1] 。
```

示例 2：

```
输入：nums = [3,2,4], target = 6
输出：[1,2]
```

示例 3：

```
输入：nums = [3,3], target = 6
输出：[0,1]
```

算法思路：哈希表 ----- 使用 java 中的哈希表 HashMap < Integer, Integer> map

循环一遍 nums 数组，在每步循环中需要做两件事：

1：判断 target - nums [i] 是否在哈希表中

2：将 nums [i] 插入到哈希表中

时间复杂度：由于只扫描一遍，且哈希表的插入和查询操作的复杂度都为 O (1)，所以总时间复杂度为 O (n)

```java
class Solution {
    public int[] twoSum(int[] nums, int target) {
        HashMap<Integer, Integer> map = new HashMap<>();
        for(int i = 0; i < nums.length; i++){
            int other = target - nums[i];
            if(map.containsKey(other)){
                return new int[]{map.get(other), i};
```

```
            }
            map.put(nums[i], i);
        }
        return new int[]{-1, -1};
    }
}
```