001. Two Sum

001 Two Sum

- **Array**
- **Hash Table** + two pointers

Description

Given an array of integers, return **indices** of the two numbers such that they add up to a specific target.

You may assume that each input would have ***exactly*** one solution, and you may not use the *same* element twice.

Example:

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

Because nums[0] + nums[1] = 2 + 7 = 9,
return [0, 1].
```

1. Thought Line

(1) map struture **mapHash** is for storing the occurrence numbers of each number.

2. Hash Table + Two Pointers

```
class Solution {
public:
    vector<int> twoSum(vector<int>& nums, int target) {
        vector<int> result;
        map<int,int> mapHash;
        if (nums.size()<2) return result;

        for (vector<int>::size_type i=0; i<=nums.size()-1; ++i)
            ++mapHash[nums[i]];

        for (vector<int>::size_type i=0; i<=nums.size()-2; ++i){
            int currentValue = nums[i];
            int biasValue = target-nums[i];

            // cannot use the same element twice.
            --mapHash[currentValue];
            if (mapHash[biasValue]>0){
                for (vector<int>::size_type j=i+1; j<=nums.size()-1; ++j){
                    if (nums[j]==biasValue){
                        result.push_back(i);
                        result.push_back(j);
                        return result;
                    }
                }
            }
        }
        return result;
    }
};
```