Two Sum

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].
```

Solution 1 in C++

```
class Solution {
public:
    vector<int> twoSum(vector<int>& nums, int target) {
        vector<int> r(2);
        int i,j,cnt;
        cnt=nums.size();
        for(i=0; i<(cnt-1); i++)
        {
            r[0]=i;
            for(j=i+1; j<cnt; j++)
            {
                r[1]=j;
                if ((nums[i]+nums[j])==target) return r;
            }
        }
        return r;
    }
};
```

Solution 2 in Java : Brute Force - O(n^2)

```
public int[] twoSum(int[] nums, int target) {
    for (int i = 0; i < nums.length; i++) {
        for (int j = i + 1; j < nums.length; j++) {
            if (nums[j] == target - nums[i]) {
                return new int[] { i, j };
            }
        }
    }
    throw new IllegalArgumentException("No two sum solution");
}
```

Solution 3 in Java : Two-pass Hash Table - O(n)

```
public int[] twoSum(int[] nums, int target) {
    Map<Integer, Integer> map = new HashMap<>();
    for (int i = 0; i < nums.length; i++) {
        map.put(nums[i], i);
    }
    for (int i = 0; i < nums.length; i++) {
        int complement = target - nums[i];
        if (map.containsKey(complement) && map.get(complement) != i) {
            return new int[] { i, map.get(complement) };
        }
    }
    throw new IllegalArgumentException("No two sum solution");
}
```

Solution 3 in Java : One-pass Hash Table - O(n)

```
public int[] twoSum(int[] nums, int target) {
    Map<Integer, Integer> map = new HashMap<>();
    for (int i = 0; i < nums.length; i++) {
        int complement = target - nums[i];
        if (map.containsKey(complement)) {
            return new int[] { map.get(complement), i };
        }
        map.put(nums[i], i);
    }
    throw new IllegalArgumentException("No two sum solution");
}
```