Q. Two Sum  
  
Given an array of integers nums and an integer target, return *indices of the two numbers such that they add up to target*.

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

You can return the answer in any order.

**Example 1:**

**Input:** nums = [2,7,11,15], target = 9

**Output:** [0,1]

**Explanation:** Because nums[0] + nums[1] == 9, we return [0, 1].

**Example 2:**

**Input:** nums = [3,2,4], target = 6

**Output:** [1,2]

**Example 3:**

**Input:** nums = [3,3], target = 6

**Output:** [0,1]

**Constraints:**

* 2 <= nums.length <= 104
* -109 <= nums[i] <= 109
* -109 <= target <= 109
* **Only one valid answer exists.**

A.   
  
class Solution {

public:

    vector<int> twoSum(vector<int>& nums, int target) {

// // bruteforce approach

        // vector<int> ans(2);

        // int k=nums.size();

        // for(int i=0;i<k;i++)

        // {

        //     int num = target - nums[i];

        //     for(int j=i+1;j<k;j++)

        //     {

        //        if(nums[j]==num)

        //        {

        //            ans[0]=i;

        //            ans[1]=j;

        //            break;

        //        }

        //     }

        // }

        // return ans;

        // // Optimized approach HashTable

        unordered\_map<int, int> numMap;

        int k=nums.size();

        for(int i=0;i<k;i++)

        {

            int r = target - nums[i];

            if(numMap.count(r))

            {

                return {numMap[r],i};

            }

            numMap[nums[i]] = i;

        }

       return {};

    }

};

