Two Sum

My Submissions (/problems/two-sum/submissions/)

Question | Solution

Total Accepted: **133334** Total Submissions: **743451** Difficulty: **Medium**

Given an array of integers, find two numbers such that they add up to a specific target number.

The function twoSum should return indices of the two numbers such that they add up to the target, where index1 must be less than index2. Please note that your returned answers (both index1 and index2) are not zero-based.

You may assume that each input would have exactly one solution.

```
Input: numbers={2, 7, 11, 15}, target=9
Output: index1=1, index2=2
```

Show Tags

Show Similar Problems

Have you met this question in a real interview? Yes No

Discuss (/discuss/questions/oj/two-sum)

C

```c
#include<stdio.h>
#include<stdlib.h>
#define HASH_SIZE 10000
struct node{
    int value;
    int key;
    struct node* next;
};
struct node* data[HASH_SIZE];

int contains(int key){
    int hash = abs(key)%HASH_SIZE;
    struct node* p = data[hash];
    while(p!=NULL){
        if(p->key==key)return 1;
        p=p->next;
    }
    return 0;
}
void put(int key,int value){
```

Send Feedback (mailto:admin@leetcode.com?subject=Feedback)

```
21      int hash = abs(key)%HASH_SIZE;
22      struct node* p = data[hash];
23      struct node* s = (struct node*)malloc(sizeof(struct node));
24      s->key=key;
25      s->value=value;
26      s->next=NULL;
27      if(p==NULL){
28          data[hash] = s;
29          return;
30      }
31      while(p->next!=NULL){
32          p=p->next;
33      }
34      p->next=s;
35  }
36  int get(int key){
37      int hash = abs(key)%HASH_SIZE;
38      struct node* p = data[hash];
39      while(p!=NULL){
40          if(p->key==key)return p->value;
41          p=p->next;
42      }
43      return 0;
44  }
45  int abs(int value){
46      return value>0?value:-value;
47  }
48  int* twoSum(int* nums, int numsSize, int target) {
49       int* res = (int*)malloc(sizeof(int)*2);
50       int value;
51       for(value=0;value<HASH_SIZE;value++){
52           data[value]=NULL;
53       }
54       for(int i=0;i<numsSize;i++){
55           if(contains(target-nums[i])){
56               value = get(target-nums[i]);
57               res[0]=value;
58               res[1]=i+1;
59               return res;
60           }
61           put(nums[i],i+1);
62       }
63      return res;
64  }
```

Custom Testcase ☐

Run Code | Submit Solution