

[View all algorithm tutorials](#)

Two sum problem

JavaScript

The two sum problem is a common interview question, and it is a variation of the [subset sum](#) problem. There is a popular dynamic programming solution for the subset sum problem, but for the two sum problem we can actually write an algorithm that runs in $O(n)$ time. The challenge is to find all the pairs of two integers in an unsorted array that sum up to a given S .

For example, if the array is $[3, 5, 2, -4, 8, 11]$ and the sum is 7 , your program should return $[[11, -4], [2, 5]]$ because $11 + -4 = 7$ and $2 + 5 = 7$.

Naive solution

A naive approach to this problem would be to loop through each number and then loop again through the array looking for a pair that sums to S . The running time for the below solution would be $O(n^2)$ because in the worst case we are looping through the array twice to find a pair.

```
// our two sum function which will return
// all pairs in the array that sum up to S
function twoSum(arr, S) {

  var sums = [];

  // check each element in array
  for (var i = 0; i < arr.length; i++) {

    // check each other element in the array
    for (var j = i + 1; j < arr.length; j++) {

      // determine if these two elements sum to S
      if (arr[i] + arr[j] === S) {
        sums.push([arr[i], arr[j]]);
      }

    }

  }

  // return all pairs of integers that sum to S
  return sums;

}

twoSum([3, 5, 2, -4, 8, 11], 7);
```

Faster solution

We can write a faster algorithm that will find pairs that sum to S in linear time. The algorithm below makes use of [hash tables](#) which have a constant lookup time. As we pass through each element in the array, we check to see if S minus the current element exists in the hash table. We only need to loop through the array once, resulting in a running time of $O(n)$ since each lookup and insertion in a hash table is $O(1)$.

Example

If the array is: [4, 5, 1, 8] and the sum is 6 the algorithm would proceed with the steps below:

- (1) The hash table is initially empty and the first element in the array is 4. We simply put 4 into the hash table.
- (2) The next element is 5. We check to see if the sum minus the current element exists in the hash table. $6 - 5 = 1$ does not exist in the hash table. So add 5 to the hash table.
- (3) The next element is 1. We check to see if the sum minus the current element exists in the hash table. $6 - 1 = 5$ does exist in the hash table so we found a pair!

Code for faster solution

```
// our two sum function which will return
// all pairs in the array that sum up to S
function twoSum(arr, S) {

  var sums = [];
  var hashTable = {};

  // check each element in array
  for (var i = 0; i < arr.length; i++) {

    // calculate S - current element
    var sumMinusElement = S - arr[i];

    // check if this number exists in hash table
    // if so then we found a pair of numbers that sum to S
    if (hashTable[sumMinusElement.toString()] !== undefined) {
      sums.push([arr[i], sumMinusElement]);
    }

    // add the current number to the hash table
    hashTable[arr[i].toString()] = arr[i];
  }

  // return all pairs of integers that sum to S
  return sums;
}

twoSum([3, 5, 2, -4, 8, 11], 7);
```

[Run Code](#)

You can edit the above code...

Sources

<http://www.careercup.com/question?id=15206700>

http://www.glassdoor.com/Interview/Given-a-sorted-array-write-a-program-to-decide-if-two-elements-sum-up-to-a-third-QTN_242990.htm

http://www.glassdoor.com/Interview/Given-a-sum-find-two-numbers-in-an-array-with-that-sum-QTN_864790.htm



mrdaniel published this on 11/9/15 | subset, bootcamp, hash, search, array, Google, Facebook, Amazon, Fullstack Academy

Comments



why do you have to convert, .toString()?



[df5890](#)

commented on 04/08/16


- +

4

-

@df5890, in JavaScript hash tables, the keys cannot be integers. JavaScript will automatically convert the number to a string, but I just did it explicitly in the code above.


<http://stackoverflow.com/questions/2002923/javascript-using-integer-as-key-in-associative-array>

[mrdaniel](#)
commented on 04/13/16
- +

2

-


what if there are duplicates in the given array?

[jayceeyili](#)
commented on 09/27/16
- +

1

-


How do i get the brackets to show up in my browser, when I run the code it shows up like this 5,2,-4,11, instead of [[5,2],[-4,11]]

[jcapra](#)
commented on 10/13/16
- +

0

-


Can you guys give code/comments for the ArrayAdditionI version of this classic problem, ie. returns true if there is a subset of any number of elements (not just two) adding up to the largest element? The solutions provided don't have comments for the most part. Thanks.

[rhodesd](#)
commented on 01/24/16
- +

-6

-

This actually doesn't even correctly solve the challenge. It returns an empty array

[adrian08](#)
commented on 04/07/17

[Log in](#) to submit a comment.

[Challenges](#) [About](#) [Sign Up/Login](#) [Blog](#) [Social](#) [Privacy](#) [Terms](#) [Contact](#) [Membership](#)