**Two Sum**

* **Question Description:**

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. The answers (index1 and index2) are not zero-based.

* **Required Complexity:**

O(n)

* **Input:**

Given nums = [2, 7, 11, 15], target = 9. Assume that each input would have exactly one solution, and you may not use the same element twice.

* **Explanation:**

Because nums [0] + nums [1] = 2 + 7 = 9

return [index1, index2] = [0, 1]

* **Analyze:**

1. Select the max value from the input array which equals to 15.

max\_value = 15

1. Create a hash table with size of max\_value in the input array. The hash table is used to store whether the value appears in the input array.

(Zero indicates that the value doesn’t appear in the input array meanwhile non-zero indicates that the location of the value appears in the input array. This will improve the speed of output index.)

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]

[0, 0, 1, 0, 0, 0, 0, 2, 0, 0, 0, 3, 0, 0, 0, 4]

which stands for:

hash\_table [0] = 0, hash\_table [1] = 0, hash\_table [2] = 1, hash\_table [3] = 0,

hash\_table [4] = 0, hash\_table [5] = 0, hash\_table [6] = 0, hash\_table [7] = 2,

hash\_table [8] = 0, hash\_table [9] = 0, hash\_table [10] = 0,

hash\_table [11] = 3, hash\_table [12] = 0, hash\_table [13] = 0,

hash\_table [14] = 0, hash\_table[15] = 4

1. Scan the hash table from scratch to the middle.

If hash\_table [index1] = 0, go to the next index, index1++.

Else hash\_table [index1]! = 0, check that hash\_table [target – index1]! = 1.

If hash\_table [target – index1]! = 0, return index pair.

Else hash\_table [target – index1] = 0, go to the next index, index1 ++.

* First Round:

hash\_table [0] = 0, goto the next index, 0++ = 1.

* Second Round:

hash\_table [1] = 0, goto the next index, 1++ = 2.

* Third Round:

hash\_table [2] = 1 and target = 9

hash\_table [9 – 2] = hash\_table [7] = 2, return pair [index1, index2] = [1 - 1, 2 - 1] = [0, 1].

1. Return index pair [0, 1] as result.

* **Special Circumstances:**

Input array is empty, then return null as index pair.

* **Space Complexity:**

O(n)

* **Speed Complexity:**

O(n)