2 Sum

1. Two Sum

Easy 51K 1.6K

Companies

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].
```

Constraints → Distinct, Negative Element,

Variants 1 → Check
2 → Return indices
} Same

arr[] = {2, 7, 11, 15} target = 9

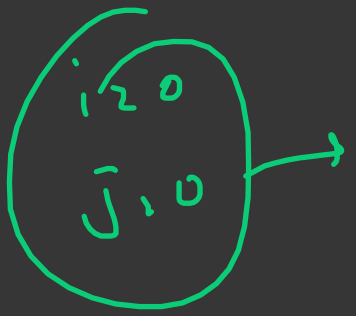
7 + 2 = 9 → {0, 1}

true

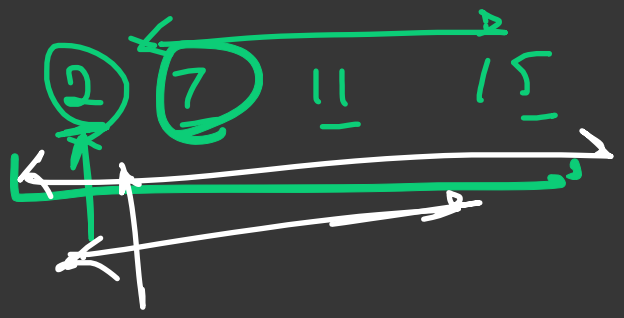
Brute force.

Distinct → Integer

```
for( i=0; i<n; i++)
    for( j=0; j<n; j++)
        if(i==j) continue;
        for if( arr[i]+arr[j] == sum)
            return ans. ((i, j))
```



Better Approach



for(i

a[i] + a[j] = target

2 + a[j] = 9

a[j] = target - a[i]

a[j] = 9 - 2 → 7

Count = 1

map<int, int>

0 → 2

1 → 7

2 → 11

3 → 15

for(i=0, i<n; i++)

i

hashmap

return {1, i},

Optimal Approach:

Two pointers algo:

arr[] : { 2, 6, 5, 8, 11 }

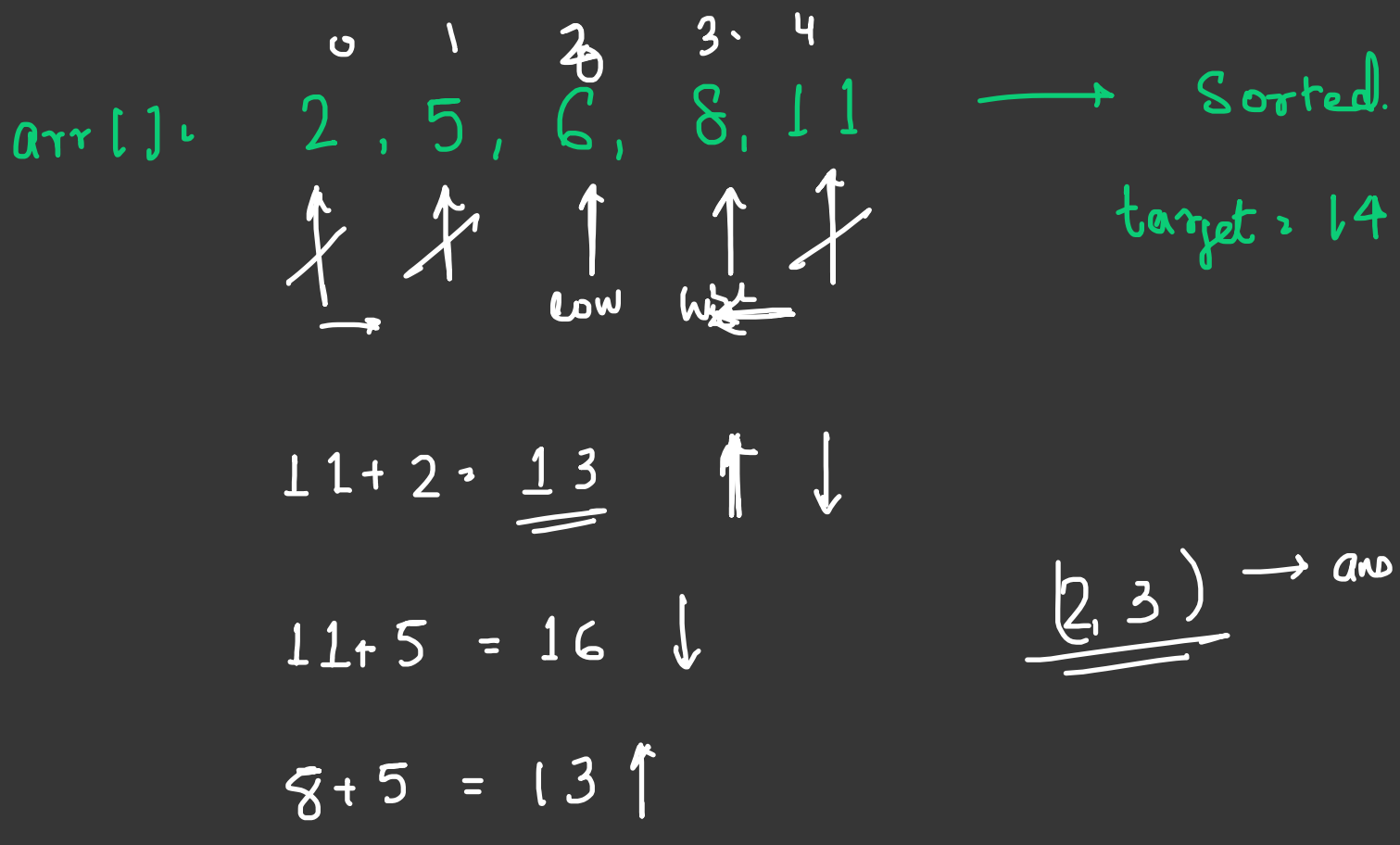
target = 14

→ $a_i + a_j$ = target

→ Greedy Algorithm →

↳ So we sort the array

2, 5, 6, 8, 11 target = 14


0 1 2 3 4
arr[]: 2, 5, 6, 8, 11
Sorted.
target : 14
low
high
11+2 = 13
11+5 = 16
8+5 = 13
8+6 = 14
(2, 3) → and

Pseudocode

Only to check for the two sum because we sort the array so the index is shuffle.

low = 0, high = n-1,

while (low < high)

if (arr[low] + arr[high] == target) return true

else if (arr[low] + arr[high] > target) high--;

else low++;

return false

T.C. :- O(N)

S.C. : O(1)