16. 3Sum Closest ★

Question | Editorial Solution | My Submissions (/problems/3sum-closest/submissions/)

Total Accepted: **91597** Total Submissions: **305063** Difficulty: **Medium**

Given an array *S* of *n* integers, find three integers in *S* such that the sum is closest to a given number, target.

Return the sum of the three integers. You may assume that each input would have exactly one solution.

```
For example, given array S = {-1 2 1 -4}, and target = 1.

The sum that is closest to the target is 2. (-1 + 2 + 1 = 2).
```

Subscribe (/subscribe/) to see which companies asked this question

Show Tags

Show Similar Problems

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

Discuss (https://leetcode.com/discuss/questions/oj/3sum-closest)

Pick One (/problems/random-one-question/)

C++

```
class Solution {
public:
    int threeSumClosest(vector<int>& nums, int target) {
        int closest = 0, minError = INT_MAX;
        if (nums.size() < 3) {
            return 0;
        }
        sort(nums.begin(), nums.end());
        for (auto i = nums.begin(), n = nums.end(); i < n - 2; ++i) {
            auto j = i + 1;
            if (i > nums.begin() && *i == *(i - 1)) continue;
            auto k = n - 1;
            while (j < k) {
                int sum = *i + *j + *k;
                if (sum < target) {
                    int error = abs(target - sum);
                    if (minError > error) {
                        minError = error;
                        closest = sum;
                    }
                    ++j;
                    while ((j < k) && (*j == *(j - 1))) ++j;
                }
                else if (sum > target) {
                    int error = abs(target - sum);
```

```
26                      if (minError > error) {
27                          minError = error;
28                          closest = sum;
29                      }
30                      --k;
31                      while ((j < k) && (*k == *(k + 1))) --k;
32                  }
33                  else {
34                      return target;
35                  }
36              }
37          }
38          return closest;
39      }
```

Notes

Custom Testcase

Run Code | Submit Solution