Triplet Sum Close to Target (medium)

We'll cover the following ^

- Problem Statement
- Try it yourself
- Solution
 - Code
 - Time complexity
 - Space complexity

Problem Statement

Given an array of unsorted numbers and a target number, find a **triplet in the array whose sum is as close to the target number as possible**, return the sum of the triplet. If there are more than one such triplet, return the sum of the triplet with the smallest sum.

Example 1:

```
Input: [-2, 0, 1, 2], target=2
Output: 1
Explanation: The triplet [-2, 1, 2] has the closest sum to the target.
```

Example 2:

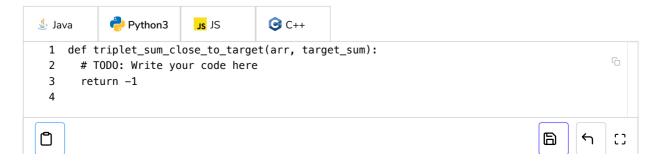
```
Input: [-3, -1, 1, 2], target=1
Output: 0
Explanation: The triplet [-3, 1, 2] has the closest sum to the target.
```

Example 3:

```
Input: [1, 0, 1, 1], target=100
Output: 3
Explanation: The triplet [1, 1, 1] has the closest sum to the target.
```

Try it yourself

Try solving this question here:







This problem follows the **Two Pointers** pattern and is quite similar to Triplet Sum to Zero (https://www.educative.io/collection/page/5668639101419520/5671464854355968/567954997300 4288/).

We can follow a similar approach to iterate through the array, taking one number at a time. At every step, we will save the difference between the triplet and the target number, so that in the end, we can return the triplet with the closest sum.

Code

Here is what our algorithm will look like:

```
🦺 Python3
                                     JS JS
                         © C++
👙 Java
 1 import math
 2
 3
 4 def triplet_sum_close_to_target(arr, target_sum):
 5
      arr.sort()
 6
      smallest_difference = math.inf
 7
      for i in range(len(arr)-2):
 8
        left = i + 1
        right = len(arr) - 1
 9
10
        while (left < right):</pre>
          target_diff = target_sum - arr[i] - arr[left] - arr[right]
11
          if target_diff == 0: # we've found a triplet with an exact sum
12
13
            return target_sum - target_diff # return sum of all the numbers
14
          # the second part of the following 'if' is to handle the smallest sum when we have m
15
16
          if abs(target_diff) < abs(smallest_difference) or (abs(target_diff) == abs(smallest_</pre>
17
            smallest_difference = target_diff # save the closest and the biggest difference
18
19
          if target diff > 0:
            left += 1 # we need a triplet with a bigger sum
20
21
          else:
22
            right -= 1 # we need a triplet with a smaller sum
23
24
      return target_sum - smallest_difference
25
26
27 def main():
28
      print(triplet sum close to target([-2, 0, 1, 2], 2))
D
```

Time complexity

Sorting the array will take O(N * log N). Overall searchTriplet() will take $O(N * log N + N^2)$, which is asymptotically equivalent to $O(N^2)$.

Space complexity

The space complexity of the above algorithm will be O(N) which is required for sorting.



ॐ



(!) Report an Issue

? Ask a Question

(https://discuss.educative.io/tag/triplet-sum-close-to-target-medium_pattern-two-pointers_grokking-the-coding-interview-patterns-for-coding-questions)