Session 2 - September 08, 2022

Triplet Sum to Zero: (unique triplets)

arr:

-3	0	1	2	-1	1	-2

ans: [-3, 1, 2], [-2, 0, 2], [-2 1 1]

[-1, 0, 1]

ans = { }

Brute f

```
for i in range( len(arr) - 2):
    for j in range(i+1, len(arr)-1):
        for k in range( j+1, len(arr)):
            if arr[i] + arr[j] + arr[k] == 0:
                ans.add( [arr[k], arr[j], arr[i]] )
```

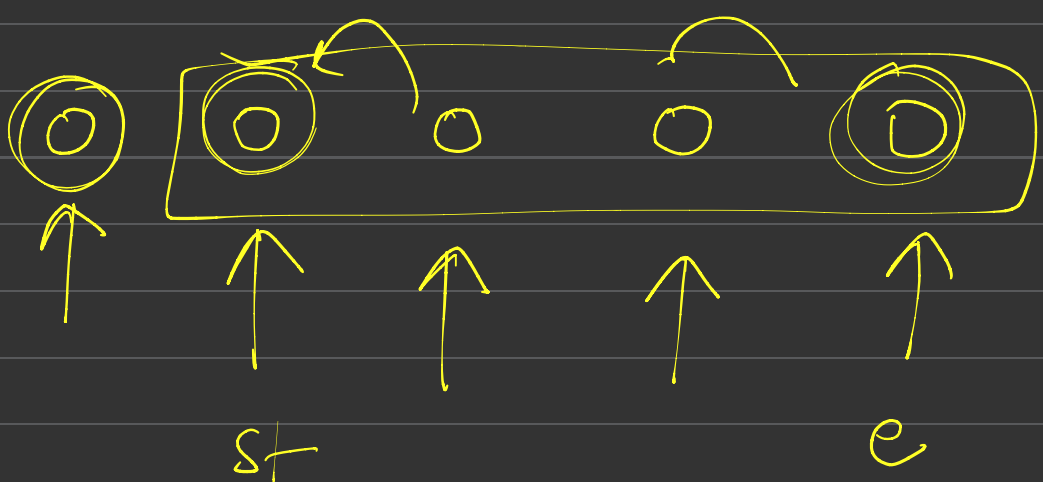
T.C:
$O(N^3)$

arr:	-3	0	1	2	-1	1	-2

↓

sort(arr)

arr:	-3	-3	-1	0	1	1	2	2



st + e = 0

```
def solve(arr):
    triplets = []
    for i in range(len(arr)):
        target = -arr[i]
        if i>0 and arr[i] == arr[i-1]:
            continue
        find_pair(arr, i+1, target, triplets)

def find_pair(arr, start, target, triplets):
    right = len(arr)-1
    while left < right:
        currsum = arr[left] + arr[right]
        if currsum == target:
            triplets.append([-target, arr[left], arr[right]])
            left += 1
            right -= 1
            while left<right and arr[left] == arr[left-1]:
                left += 1
            while left<right and arr[right] == arr[right+1]:
                right -= 1
        elif curr < target:
            left += 1
        else:
            right -= 1
```

pair → duplicate remove

-3 -2 1 0 1 2 1 2 1 2

-3 | -2 0 1 1 1 1 2 2 2

↑

Count no. of triplets whose sum < given target. -1+2+4<5 5<5 No

arr: | -1 | 4 | 2 | 1 | 3 |

<5

-1+2+3<5 (5)

4<5

0	1	2	3	4
-1	1	2	3	4

~~-1+1+4<5~~

4<5

-1 2 3

i ↑ start ↑ end

end-sta = 1

count += end-start

4-1 = (3)

-1 1 2

-1 1 3

-1 1 4

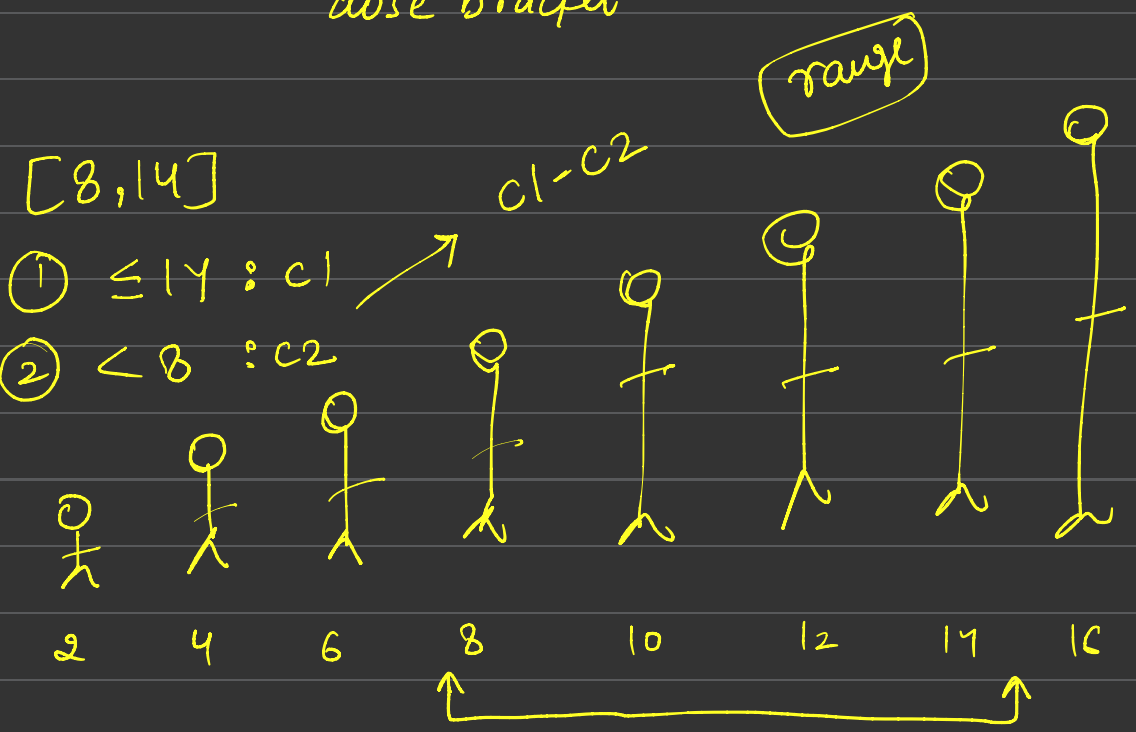
```
def countTriplets(arr, target):
    arr.sort()
    count = 0
    for i in range(len(arr)-2):
        count += find_pair(arr, arr[i], i+1, target);
    return count

def find_pair(arr, first, start, target):
    count = 0
    end = len(arr)-1
    while start < end:
        if first + arr[start] + arr[end] < target:
            count += (end-start)
            start += 1
        else:
            end -= 1
    return count
```

Problem: You are given an array nums and a range [a,b].

of triplets whose sum lies in range [a,b].

close bracket



3sum closest