3 Sum

$$arr[i] + arr[j] + arr[k] == 0$$

Idea

$$arr[i] + arr[j] == -1 * arr[k]$$

element1 element2 target

Note :- we are going to run the previous template of code, K no. of times

dry run

n = 6

arr = [-2, 0, 2, 4, -2, -8]

sort the array

0	1	2	3	4	5
-8	-2	-2	0	2	4
	k	i			j

(duplicacy)

o/p

-2 4 -2
0 2 -2
0 2 -2

i = k+1
j = n-1

k = 0, target = 8, sum = ~~2 2 4~~ 6

k = 1, target = 2, sum = ~~2~~ 2

k = 2, target = 2, sum = ~~4~~ 2

k = 3, target = 0, sum = 6

k = 4, target = -2, sum =

k = 5, target = -4, sum =

```
int i = 0;
int j = n - 1;
while ( i < j ) {
    int sum = arr[i] + arr[j];
    if ( sum == target ) {
        System.out.println(i + " " + j);
        i++;
        j--;
    } else if ( sum < target ) {
        i++;
    } else {
        j--;
    }
}
```

psudo code

1) Sort the array

2) traverse K pointer from 0 to n

2.1) declare target = -1 * arr[k]

2.2) declare i = k+1 , j = n-1 ;

2.3) loop until i < j

2.3.1) find sum = arr[i] + arr[j]

2.3.2) check if sum == target

print triplet

i++, j--;

handle duplicacy for i & j

2.3.3) check if sum < target

i++;

2.3.4) check if sum > target

j--;

2.4) handle duplicacy for K pointer

Code

```java
public static void findTriplet(int[] arr, int n) {
    Arrays.sort(arr);
    for (int k = 0; k < n; k++) {
        int target = -1 * arr[k];
        int i = k + 1;
        int j = n - 1;
        while ( i < j ) {
            int sum = arr[i] + arr[j];
            if ( sum == target ) {
                System.out.println( arr[k] + " " + arr[i] + " "  + arr[j]);
                i++;
                j--;
                while ( i < j && arr[i] == arr[i - 1] ) {
                    i++;
                }
                while ( i < j && arr[j] == arr[j + 1] ) {
                    j--;
                }
            } else if ( sum < target ) {
                i++;
            } else {
                j--;
            }
        }
        while ( k + 1 < n && arr[k] == arr[k + 1] ) {
            k++;
        }
    }
}
```

$T.C = O(n^2 + n \log(n))$

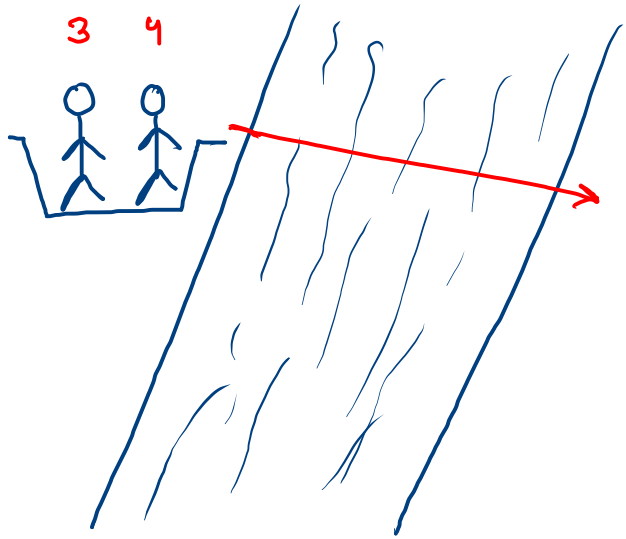
Count boat

- weight of people arr[i]
- each boat can carry 'limit' weight
- each boat can carry atmost 2 people

t

∞ boats

limit = 6



ans = no. of boats

arr[i] + arr[j] <= limit

target

arr[i] <= limit

arr[j] <= limit

} always true

Ex:- arr = [3, 2, 1, 2]

sorting arr = [1, 2, 2, 3], limit = 3

j i

arr[i] + arr[j] <= limit

sum = ~~4~~ ~~3~~ 4

boats = ~~0~~ ~~1~~ ~~2~~ 3

psudo code

1) sort
2) i = 0, j = n-1, boat = 0
3) loop until i <= j
   3.1) sum = arr[i] + arr[j]
   3.2) if sum == target
        i++, j--, boats++;
   3.3) if sum > target
        j--, boats++;
   3.4) if sum < target
        i++, j--, boats++;
4) return boats.

Code

```java
public static int countBoats(int[] arr, int n, int limit) {
    Arrays.sort(arr);
    int i = 0;
    int j = n - 1;
    int boats = 0;
    while ( i <= j ) {
        int sum = arr[i] + arr[j];
        if ( sum <= limit ) {
            i++;
            j--;
        } else {
            j--;
        }
        boats++;
    }
    return boats;
}
```

T.C = O(n + nlogn)

arr = [1, 1, 3, 4, 4, 5, 6] limit = 6

j i

sum = ~~7~~ ~~6~~ ~~5~~ ~~7~~ 6

count = 5