Two - pointer approach.

1 2 3 4 5

Reach Target

tar = 4.

Sample Input 0

```
6
-1 1 2 3 4 5
4
```

Sorted.

-1 1 2 3 4 5

0 1 2 3 4 5

Sample Output 0

```
0 5
1 3
```

-1 + 5 = 4 tar.

i < j

i j

tar == 4.

-1 1 2 3 4 5

0 1 2 3 4 5

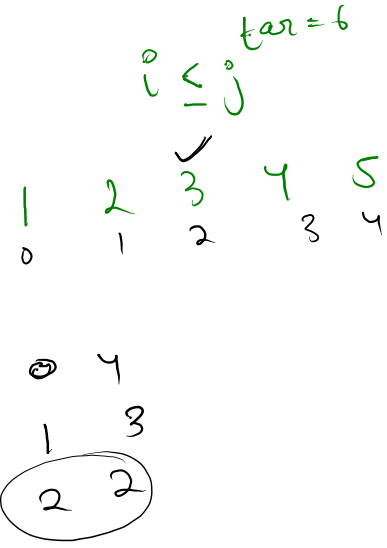
tar = 4

index → easy sort

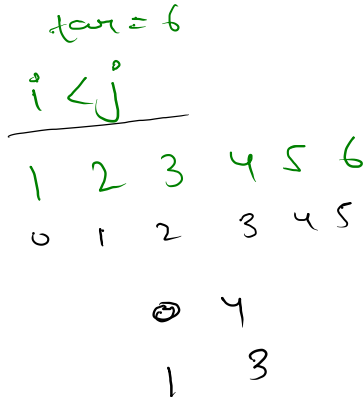
0 5

1 3

s < tar




```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int [] A = new int[n];
    for(int i = 0; i < n; i++){
        A[i] = scn.nextInt();
    }

    int tar = scn.nextInt();

    //logic
    //Arrays is already sorted
    int i = 0;
    int j = n-1;

    while(i <=j){
        int s = A[i]+A[j];

        if(s == tar){
            System.out.println(i + " " + j);
            i++;
            j--;
        }
        else if(s > tar){
            j--;
        }
        else{
            // s < tar
            i++;
        }

    }
```



Target Sum

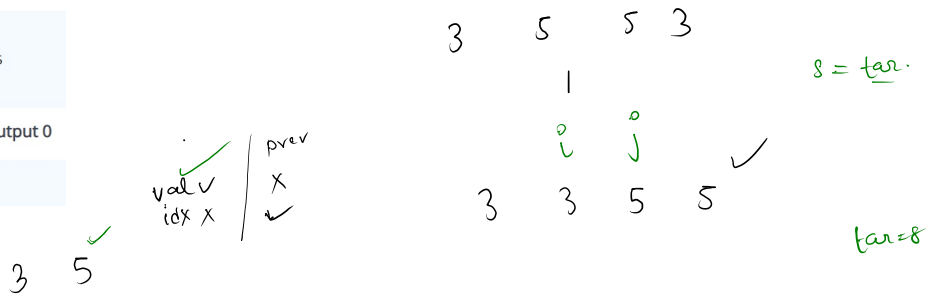
tar = 8.

Sample Input 0

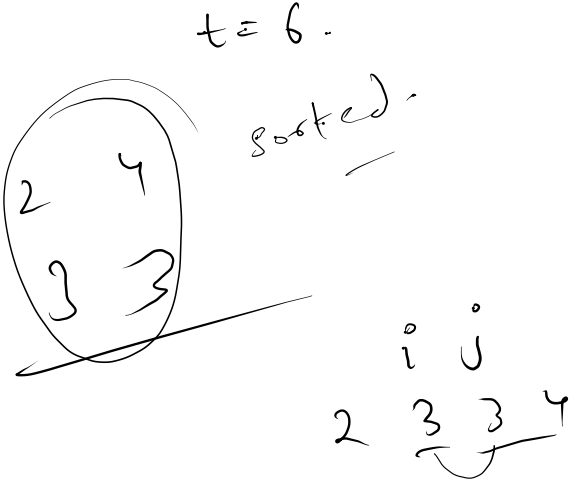
```
4
3 3 5 5
8
```

Sample Output 0

```
3 5
```



```java
//if sort
Arrays.sort(A);

int i = 0;
int j = n-1;

while(i < j){
    int sum = A[i] + A[j];

    if(sum == tar){

        //manage duplicates
        while(i+1 < j && A[i] == A[i+1]){
            i++;
        }

        while(j-1 > i && A[j] == A[j-1]){
            j--;
        }

        System.out.println(A[i] + " " + A[j]);
        i++;
        j--;

    }
    else if(sum > tar){
        j--;
    }
    else{
        //sum < tar
        i++;
    }
}
```



. Boats to Save People

You are given an array `people` where `people[i]` is the weight of the i^{th} person, and an **infinite number of boats** where each boat can carry a maximum weight of `limit`. Each boat carries at most two people at the same time, provided the sum of the weight of those people is at most `limit`.

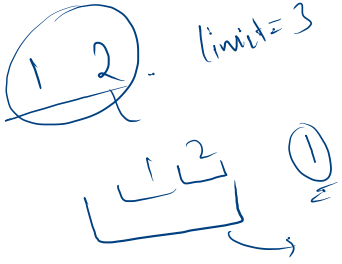
Return *the minimum number of boats to carry every given person.*

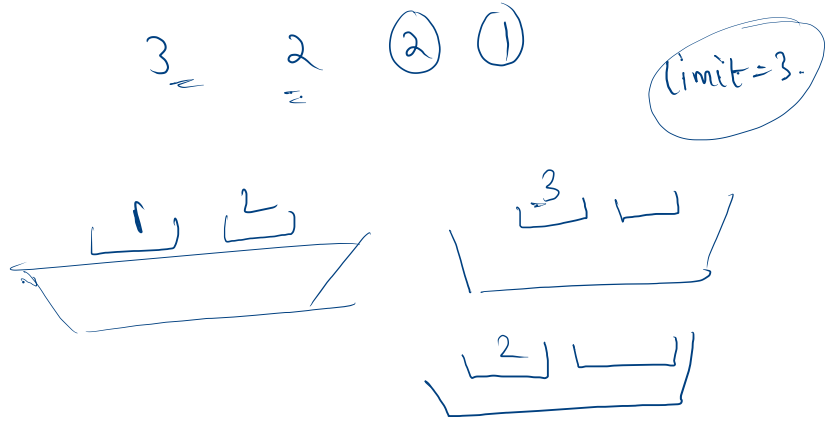
Example 1:

```
Input: people = [1,2], limit = 3
Output: 1
Explanation: 1 boat (1, 2)
```

Example 2:

```
Input: people = [3,2,2,1], limit = 3
Output: 3
Explanation: 3 boats (1, 2), (2) and (3)
```

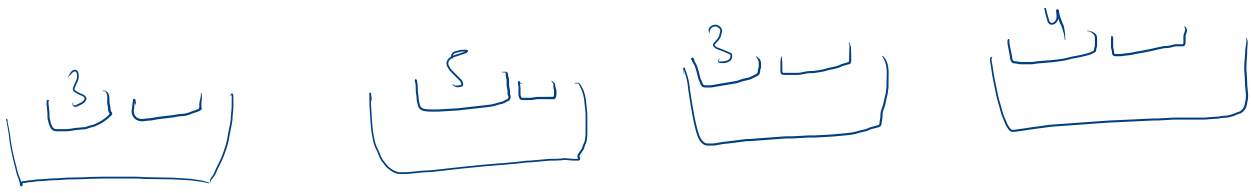




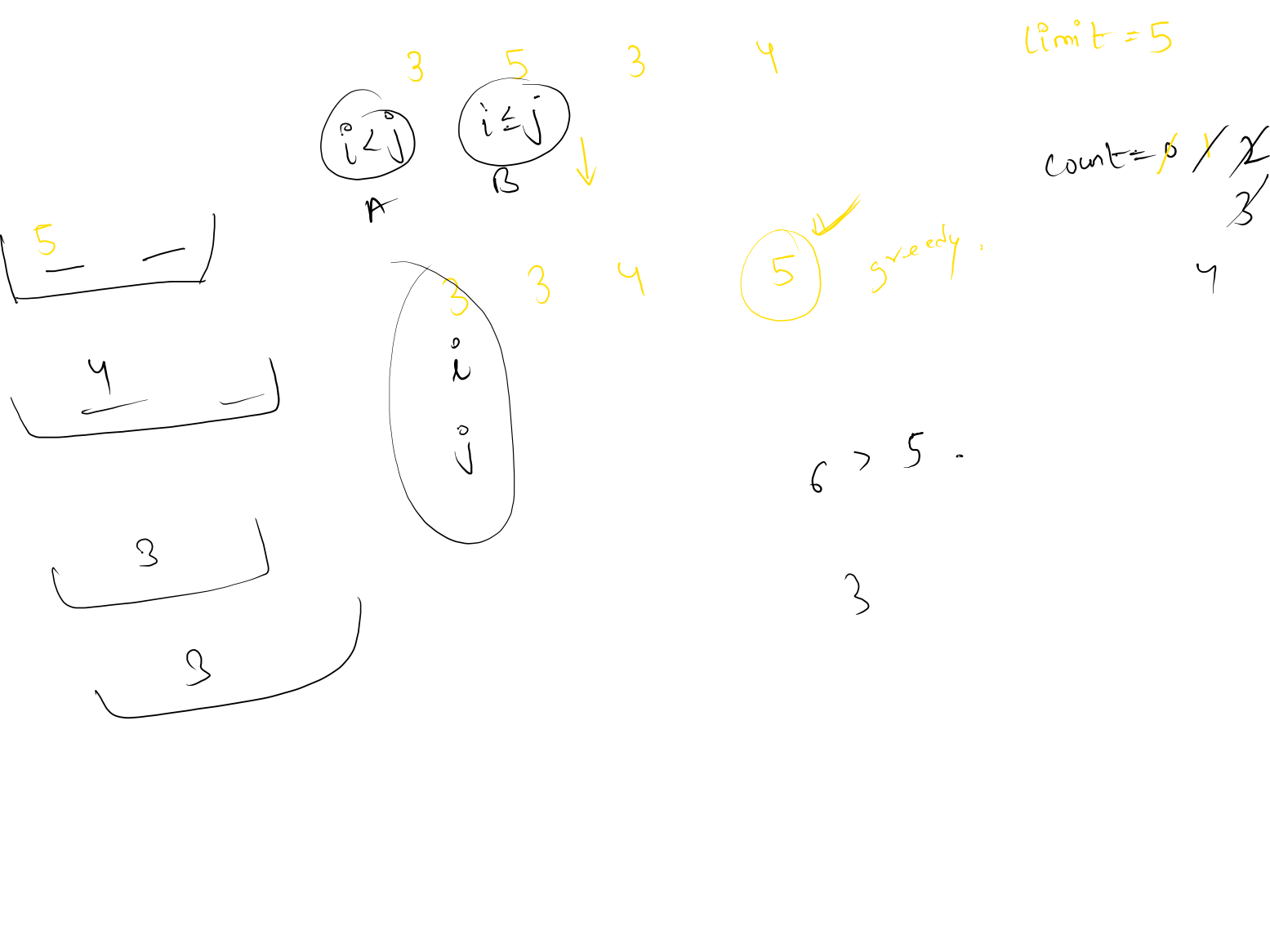
limit = 5.

Input: people = [3,5,3,4], limit = 5
Output: 4
Explanation: 4 boats (3), (3), (4), (5)

3 5 3 4



Eg:


3 5 3 4
limit = 5
i < j
A
i ≤ j
B
count = 0 1 2 3
4
5
greedy.
5
3 4
3
i
j
6 > 5.
3
3
3

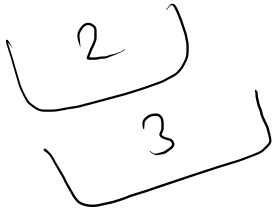
limit = 5

(2) 3 3 5

c = 1

5 + 2 > 5

5 + 3 > 5



```java
public int numRescueBoats(int[] people, int limit) {
    int count = 0;
    int n = people.length;

    Arrays.sort(people);
    int i = 0;
    int j = n-1;


    while(i <= j){
        int sum = people[i] + people[j];
        if(sum <= limit){
            // carry 2 people
            i++;
            j--;
        }
        else{
            // carry just 1 person
            j--;
        }

        count++;
    }
    return count;


}
```

Greatest Till Me

```
7
1
88
3
2
16
10
9
```

Sample Output 0

```
1
88
88
88
88
88
88
```