https://course.acciojob.com/idle?question=b159eb89-5c05-4e0a-a749-49c234e5ed98

- MEDIUM
- Max Score: 40 Points
-
-
-
-
-
-

Max Number of K-Sum Pairs

You are given an integer array `nums` and an integer `k`.

In one operation, you can pick two numbers from the array whose sum equals `k` and remove them from the array.

Return *the maximum number* of operations you can perform on the array.

Input Format

Input consists of two lines.

First line contains an integer `n` which is the size of nums array.

Next line contains `n` spaced integers which are the elements of `nums`.

The last line contains `k`.

Output Format

Return *the maximum number* of operations you can perform on the array.

Example 1

Input

```
4
1 2 3 4
5
```

Output

```
2
```

Explanation

Starting with nums = [1,2,3,4]:

- Remove numbers 1 and 4, then nums = [2,3]
- Remove numbers 2 and 3, then nums = [] There are no more pairs that sum up to 5, hence a total of 2 operations.

Example 2

Input

```
5
3 1 3 4 3
6
```

Output

```
1
```

Explanation

Starting with nums = [3,1,3,4,3]:

- Remove the first two 3's, then nums = [1,4,3] There are no more pairs that sum up to 6, hence a total of 1 operation.

Constraints

1 <= nums.length <= 10^5

1 <= nums[i] <= 10^9

0 <= k <= 10^9

Topic Tags

- Hashing
- 2

My code

```
// n java
import java.util.*;

class Solution {
    public int maxOperations(int[] nums, int k) {
        //Write code here
                int n=nums.length;
                int ans=0;
                HashMap<Integer,Integer>hm=new HashMap<>();
                for(int i=0;i<n;i++)
                        {
                        if(hm.containsKey(nums[i]))
                        {
                                ans++;
                                hm.put(nums[i],hm.getOrDefault(nums[i],0)-1);
                                if(hm.get(nums[i])==0)
                                hm.remove(nums[i]);
                        }
```

```
                                else
                        {

hm.put(k-nums[i],hm.getOrDefault(k-nums[i],0)+1);
                        }
                        }
                return ans;
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n;
        n = sc.nextInt();
        int arr[] = new int[n];
        for (int i = 0; i < n; i++)
            arr[i] = sc.nextInt();
        int k;
        k = sc.nextInt();
        Solution Obj = new Solution();
        int result = Obj.maxOperations(arr, k);
        System.out.println(result);
        sc.close();
    }
}
```