Wet Shark and Two Subsequences

by trophies

One day, Wet Shark was given an array $X = \{x_1, x_2, \ldots, x_m\}$. As always, he started playing with its subsequences.

When you came to know about this habit, you presented him a task of finding all pairs of subsequences, (A, B), which satisfies all of the following constraints. We will represent a pair of subsequence as $A = \{x_{a_1}, x_{a_2}, \ldots, x_{a_n}\}$ and $B = \{x_{b_1}, x_{b_2}, \ldots, x_{b_n}\}$

- A and B must be of same length, i.e., $|A| = |B|$.
- $\sum_{i=1}^{n}(x_{a_i} + x_{b_i}) = r$
- $\sum_{i=1}^{n}(x_{a_i} - x_{b_i}) = s$

Please help Wet Shark determine how many possible subsequences A and B can exist. Because the number of choices may be big, output your answer modulo $10^9 + 7 = 1000000007$.

Note:

- Two segments are different if there's exists at least one index i such that element x_i is present in exactly one of them.
- Both subsequences can overlap each other.
- Subsequences do not necessarily have to be distinct

Input Format

The first line consists of 3 space-separated integers m, r, s, where m denotes the length of the original array, X, and r and s are as defined above. The next line contains m space-separated integers, $x_1, x_2, \ldots, x_m$, representing the elements of X.

Constraints

- $1 \leq m \leq 100$
- $0 \leq r,\ s \leq 2000$
- $1 \leq x_i \leq 2000$

Output Format

Output total number of pairs of subsequences, (A, B), satisfying the above conditions. As the number can be large, output it's modulo $10^9 + 7 = 1000000007$

Sample Input 0

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

Sample Output 0

```
3
```

Explanation 0

For array $X = \{x_1, x_2, x_3, x_4\} = \{1, 1, 1, 4\}$ there are three pairs of subsequences:

1. $A = \{x_4\} = \{4\}; B = \{x_1\} = \{1\}$
2. $A = \{x_4\} = \{4\}; B = \{x_2\} = \{1\}$
3. $A = \{x_4\} = \{4\}; B = \{x_3\} = \{1\}$

Submissions: 812
Max Score: 80
Difficulty: Medium

Rate This Challenge:

More

Current Buffer (saved locally, editable) Java 8

```java
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    static long mod = 1000000007;


    public static void main(String args[] ) throws Exception {

        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        String[] num = br.readLine().split("\\s");

        int m = Integer.parseInt(num[0]);
        int r = Integer.parseInt(num[1]);
        int s = Integer.parseInt(num[2]);

        num = br.readLine().split("\\s");

        int[] arr = new int[m];

        for(int i = 0 ; i < m ; i++){
            arr[i] = Integer.parseInt(num[i]);
        }

        Arrays.sort(arr);

        if(r == 0 && s == 0){
            System.out.println(0);
            return;
        }

        System.out.println(dp(arr,arr.length - 1, r+s, r-s, new HashMap<String,Long>()));

    }

```

```
40
41      public static long dp(int[] arr, int index, int sumA, int sumB,Map<String,Long> memo){
42
43          if(sumA < 0 || sumB < 0){
44              return 0;
45          }
46          else if(sumA == 0 && sumB == 0){
47              return 1;
48          }
49
50          if(memo.containsKey(sumA + "-" + sumB + "-"+ index)){
51              return memo.get(sumA + "-" + sumB + "-" + index);
52          }
53
54          long output = 0;
55
56          if(index < 0){
57              return output % mod;
58          }
59
60          output = ( output +  (dp(arr,index-1,sumA-2*arr[index],sumB-2*arr[index],memo) % mod)) % mod;
61          output = ( output +  (dp(arr,index-1,sumA-2*arr[index],sumB,memo) % mod)) % mod;
62          output = ( output +  (dp(arr,index-1,sumA,sumB-2*arr[index],memo) % mod)) % mod;
63          output = ( output +  (dp(arr,index-1,sumA,sumB,memo) % mod)) % mod;
64
65          memo.put(sumA + "-" + sumB + "-" + index, output);
66          return output;
67
68      }
69  }
70
```

Line: 1 Col: 1

Upload Code as File Test against custom input Run Code Submit Code