**823. Binary Trees With Factors: -**

Medium Accepted: 124.6K Submissions: 239.6K Acceptance Rate: 52.0%

Given an array of unique integers, arr, where each integer arr[i] is strictly greater than 1.

We make a binary tree using these integers, and each number may be used for any number of times. Each non-leaf node's value should be equal to the product of the values of its children.

Return *the number of binary trees we can make*. The answer may be too large so return the answer **modulo** 109 + 7.

**Example 1:**

**Input:** arr = [2,4]

**Output:** 3

**Explanation:** We can make these trees: [2], [4], [4, 2, 2]

**Example 2:**

**Input:** arr = [2,4,5,10]

**Output:** 7

**Explanation:** We can make these trees: [2], [4], [5], [10], [4, 2, 2], [10, 2, 5], [10, 5, 2].

**Constraints:**

* 1 <= arr.length <= 1000
* 2 <= arr[i] <= 109
* All the values of arr are **unique**.

**Code: -**

class Solution {

public:

    int numFactoredBinaryTrees(vector<int>& arr) {

        int n = arr.size(), mod = 1e9 + 7;

        unordered\_map<int, long long> mp;

        sort(arr.begin(), arr.end());

        for(auto &i : arr)

            ++mp[i];

        for(int i = 1; i < n; ++i){

            long long extracount = 0;

            for(int j = 0; j < i; ++j){

                if(arr[i] % arr[j] == 0 and mp.find(arr[i] / arr[j]) != mp.end()){

                    long long first = mp[arr[j]];

                    long long second = mp[arr[i] / arr[j]];

                    extracount += first \* second;

                }

            }

            mp[arr[i]] = mp[arr[i]] + extracount;

        }

        int ans = 0;

        for(auto &p : mp){

            ans = ((ans % mod) + (p.second % mod)) % mod;

        }

        return ans;

    }

};

**T.C: - O(N2)**

**S.C: - O(N)**