823. Binary Trees With Factors

DescriptionHintsSubmissionsDiscussSolution

Given an array of unique integers, each integer 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 it's children.

How many binary trees can we make? Return the answer **modulo 10 ** 9** + **7**.

Example 1:

```
Input: A = [2, 4]
Output: 3
Explanation: We can make these trees: [2], [4], [4, 2, 2]
```

Example 2:

```
Input: A = [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].
```

Note:

1. `1 <= A.length <= 1000.`
2. `2 <= A[i] <= 10 ^ 9.`

Let `dp[i] be the number of trees with root I`
`we have the following functions:`
`dp[i] = sum(dp[j] * dp[j/i])`
`res = sum(dp[i])`

```
class Solution {
public:
    long mod = 1000000007;
    int numFactoredBinaryTrees(vector<int>& A) {
        unordered_map<int,long> dp;
        sort(A.begin(),A.end());
        for(int i=0;i<A.size();++i)
        {
            dp[A[i]]=1;
            for(int j=0;j<i;++j)
            {
                if(A[i]%A[j]==0 && dp.count(A[i]/A[j]))
                {
                    dp[A[i]] = (dp[A[i]]+dp[A[i]/A[j]]*dp[A[j]])%mod;
                }
```

```
            }
        }
        int sum=0;
        for(auto it:dp)
        {
            sum=(sum+it.second)%mod;
        }
        return sum;
    }
};
```