**[Greatest Common Divisor Traversal](https://leetcode.com/problems/greatest-common-divisor-traversal/)**

You are given a **0-indexed** integer array nums, and you are allowed to **traverse** between its indices. You can traverse between index i and index j, i != j, if and only if gcd(nums[i], nums[j]) > 1, where gcd is the **greatest common divisor**.

Your task is to determine if for **every pair** of indices i and j in nums, where i < j, there exists a **sequence of traversals** that can take us from i to j.

Return true*if it is possible to traverse between all such pairs of indices, or*false*otherwise.*

**Example 1:**

**Input:** nums = [2,3,6]

**Output:** true

**Explanation:** In this example, there are 3 possible pairs of indices: (0, 1), (0, 2), and (1, 2).

To go from index 0 to index 1, we can use the sequence of traversals 0 -> 2 -> 1, where we move from index 0 to index 2 because gcd(nums[0], nums[2]) = gcd(2, 6) = 2 > 1, and then move from index 2 to index 1 because gcd(nums[2], nums[1]) = gcd(6, 3) = 3 > 1.

To go from index 0 to index 2, we can just go directly because gcd(nums[0], nums[2]) = gcd(2, 6) = 2 > 1. Likewise, to go from index 1 to index 2, we can just go directly because gcd(nums[1], nums[2]) = gcd(3, 6) = 3 > 1.

**Example 2:**

**Input:** nums = [3,9,5]

**Output:** false

**Explanation:** No sequence of traversals can take us from index 0 to index 2 in this example. So, we return false.

**Example 3:**

**Input:** nums = [4,3,12,8]

**Output:** true

**Explanation:** There are 6 possible pairs of indices to traverse between: (0, 1), (0, 2), (0, 3), (1, 2), (1, 3), and (2, 3). A valid sequence of traversals exists for each pair, so we return true.

**Constraints:**

* 1 <= nums.length <= 105
* 1 <= nums[i] <= 105

class Solution {

public:

    unordered\_map<int, vector<int>> prime2index;

    unordered\_map<int, vector<int>> index2prime;

    void dfs(int index, vector<int>& visitedIndex, unordered\_map<int,bool>& visitedPrime){

        if(visitedIndex[index] == true) return;

        visitedIndex[index] = true;

        for(auto &prime : index2prime[index]){

            if(visitedPrime[prime] == true)

                continue;

            visitedPrime[prime] = true;

            for(auto &index1 : prime2index[prime]){

                if(visitedIndex[index1] == true) continue;

                dfs(index1, visitedIndex, visitedPrime);

            }

        }

    }

    bool canTraverseAllPairs(vector<int>& nums) {

        for (int i=0; i<nums.size(); i++) {

            int temp = nums[i];

            for (int j = 2; j\*j <= nums[i]; j++) {

                if (temp % j == 0) {

                    prime2index[j].push\_back(i);

                    index2prime[i].push\_back(j);

                    while (temp % j == 0)

                      temp /= j;

                }

            }

            if (temp > 1) {

                prime2index[temp].push\_back(i);

                index2prime[i].push\_back(temp);

            }

        }

        vector<int> visitedIndex(nums.size(),0);

        unordered\_map<int,bool> visitedPrime;

        dfs(0, visitedIndex, visitedPrime);

        for(int i=0; i<visitedIndex.size(); i++)

            if(visitedIndex[i] == false)

                return false;

        return true;

    }

};

Link : <https://leetcode.com/problems/greatest-common-divisor-traversal/?envType=daily-question&envId=2024-02-25>