Problem 3566: Partition Array into Two Equal Product Subsets

Problem Information

Difficulty: Medium
Acceptance Rate: 35.00%
Paid Only: No
Tags: Array, Bit Manipulation, Recursion, Enumeration

Problem Description

You are given an integer array `nums` containing **distinct** positive integers and an integer `target`.

Determine if you can partition `nums` into two **non-empty** **disjoint** **subsets** , with each element belonging to **exactly one** subset, such that the product of the elements in each subset is equal to `target`.

Return `true` if such a partition exists and `false` otherwise.

A **subset** of an array is a selection of elements of the array.

Example 1:

Input: nums = [3,1,6,8,4], target = 24

Output: true

Explanation: The subsets `[3, 8]` and `[1, 6, 4]` each have a product of 24. Hence, the output is true.

Example 2:

Input: nums = [2,5,3,7], target = 15

Output: false

Explanation: There is no way to partition `nums` into two non-empty disjoint subsets such that both subsets have a product of 15. Hence, the output is false.

Constraints:

* `3 <= nums.length <= 12` * `1 <= target <= 1015` * `1 <= nums[i] <= 100` * All elements of `nums` are **distinct**.

Code Snippets

C++:

```
class Solution {
public:
bool checkEqualPartitions(vector<int>& nums, long long target) {

}
};
```

Java:

```
class Solution {
public boolean checkEqualPartitions(int[] nums, long target) {

}
}
```

Python3:

```
class Solution:
def checkEqualPartitions(self, nums: List[int], target: int) -> bool:
```