Problem 368: Largest Divisible Subset

Problem Information

Difficulty: Medium
Acceptance Rate: 49.19%
Paid Only: No
Tags: Array, Math, Dynamic Programming, Sorting

Problem Description

Given a set of **distinct** positive integers `nums`, return the largest subset `answer` such that every pair `(answer[i], answer[j])` of elements in this subset satisfies:

* `answer[i] % answer[j] == 0`, or * `answer[j] % answer[i] == 0`

If there are multiple solutions, return any of them.

Example 1:

Input: nums = [1,2,3] **Output:** [1,2] **Explanation:** [1,3] is also accepted.

Example 2:

Input: nums = [1,2,4,8] **Output:** [1,2,4,8]

Constraints:

* `1 <= nums.length <= 1000` * `1 <= nums[i] <= 2 * 109` * All the integers in `nums` are **unique**.

Code Snippets

C++:

```
class Solution {
public:
vector<int> largestDivisibleSubset(vector<int>& nums) {

}
};
```

Java:

```
class Solution {
public List<Integer> largestDivisibleSubset(int[] nums) {

}
}
```

Python3:

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