Problem Link:

<https://leetcode.com/problems/largest-divisible-subset/description/?envType=daily-question&envId=2025-04-06>

Solution:

class Solution {

public:

vector<int> largestDivisibleSubset(vector<int>& nums) {

if(nums.empty())

return {};

sort(nums.begin(), nums.end());

int n = nums.size();

vector<int> dp(n, 1);

vector<int> p(n, -1);

int ms = 1;

int mi = 0;

for(int i = 1; i < n; ++i)

{

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

{

if(nums[i] % nums[j] == 0)

{

if(dp[i] < dp[j] + 1)

{

dp[i] = dp[j] + 1;

p[i] = j;

}

}

}

if(dp[i] > ms)

{

ms = dp[i];

mi = i;

}

}

vector<int> v;

while(mi != -1)

{

v.push\_back(nums[mi]);

mi = p[mi];

}

reverse(v.begin(), v.end());

return v;

}

};