Intuition

Since we need to form a permutation we can start from no elements and then each time we take a element we can see what further element can be taken to form permuatation with it.

Approach

Since we are exploring all possible permutation we will apply Dynamic Programming and BitMasking (for keeping visited track)

We will maintain a previous element so that at each point when we take a new element which was previously not taken we can satisfy the condition nums[i] % nums[i + 1] == 0 || nums[i + 1] % nums[i] == 0

To iterate over those elements which are previously not taken we will maintain mask.

Only taking value that satisfies the constraint:
For all indexes 0 <= i < n - 1, either nums[i] % nums[i+1] == 0 or nums[i+1] % nums[i] == 0

Using DP to store results of states based on the current bitmask and index of prev that was taken.

Complexity

- Time complexity:
The overall time complexity of the code is O(N * 2^N * N)

- Space complexity:
Since we are using the dp table space complexity goes up to O(14 + 1)(2 ^ 14)