

220. You are tasked with designing an efficient coding to generate all subsets of a given set S containing n elements. Each subset should be outputted in lexicographical order. Return a list of lists where each inner list is a subset of the given set. Additionally, find out how your coding handles duplicate elements in S . $A = [1, 2, 3]$ The subsets of $[1, 2, 3]$ are: $[], [1], [2], [3], [1, 2], [1, 3], [2, 3], [1, 2, 3]$

PROGRAM:-

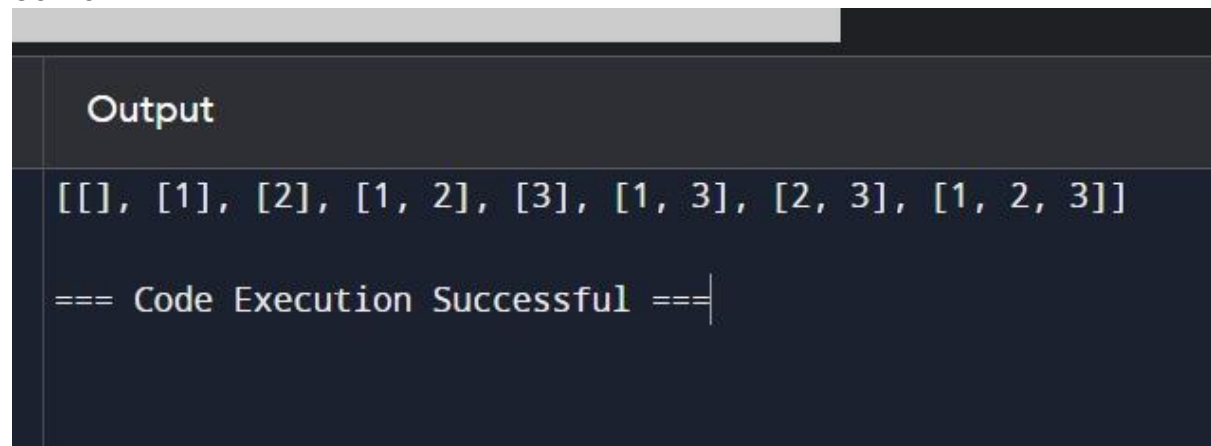
```
def subsets_lexicographical(nums):
    nums.sort() # Sort the input set to ensure subsets are generated in lexicographical order
    result = [[]] # Initialize with the empty subset

    for num in nums:
        new_subsets = []
        for subset in result:
            new_subset = subset + [num]
            new_subsets.append(new_subset)
        result.extend(new_subsets)

    return result

# Example usage:
A = [1, 2, 3]
print(subsets_lexicographical(A)) # Output: [[], [1], [2], [1, 2], [3], [1, 3], [2, 3], [1, 2, 3]]
```

OUTPUT:-

A screenshot of a code execution environment with a dark background. At the top, there is a tab labeled "Output". Below the tab, the output of the code is displayed in a monospaced font: `[[[], [1], [2], [1, 2], [3], [1, 3], [2, 3], [1, 2, 3]]`. Below the output, a status message reads `=== Code Execution Successful ===` with a vertical cursor at the end.

TIME COMPLEXITY:- $O(2^n)$