**Algorithmic Thinking in Problem Solving**

**Framework**

1. List the steps one should take when tackling a coding problem during a coding interview. Be very specific. Your task is to create a framework/problem-solving strategy.
2. What do you think should be done when someone “gets stuck” when solving a coding problem? List the steps. Be very specific. Feel free to use examples to illustrate your points.
3. Imagine you are being interviewed and you are asked to use a whiteboard to problem solve. How would you use the whiteboard? Draw a diagram that shows the different sections that you would have and justify your decisions.
4. Create a checklist for assessing your problem-solving process when tackling a coding problem. For example, one of you items in your checklist could be “Ask Clarifying Questions” or “Test your code”
5. Come up with a list of DOs and DON’Ts that people can use in the context of coding interviews.
6. Use your proposed framework/strategy, whiteboard section distribution, and checklist to tackle the following problems – If you had to refine anything based on your experience tackling the problems, state what you changed and why.

Problem A: Given a collection of **distinct** integers, return all possible permutations.

**Example:**

**Input:** [1,2,3]

**Output:**

[

[1,2,3],

[1,3,2],

[2,1,3],

[2,3,1],

[3,1,2],

[3,2,1]

]

Problem B: Given an array of integers nums and a positive integer k, find whether it's possible to divide this array into k non-empty subsets whose sums are all equal.

**Example 1:**

**Input:** nums = [4, 3, 2, 3, 5, 2, 1], k = 4

**Output:** True

**Explanation:** It's possible to divide it into 4 subsets (5), (1, 4), (2,3), (2,3) with equal sums.

**Note:**

* 1 <= k <= len(nums) <= 16.
* 0 < nums[i] < 10000.