YT video explaining how to use my coding template!
https://youtu.be/m71kZqOj5VU?t=380

FEEL FREE TO SAVE THIS GOOGLE SHEET / MODIFY / CREATE YOUR OWN HOWEVER YOU WANT! :) THIS IS JUST WHAT I USE. Keep track of ur own tips and tricks for when u start interviewing so that you always have a list of things to go back to.

>N!, N!, 2^n, EXPONENTIAL, NLOGN, O(N), O(LOGN), O(1)

>N!	DP? Permutation/Subset with some sort of additional calculation?
N!	Permutation
2^n	Subset
Exponential	Recursion, sliding window (variation), nested loops, nested operations
nlogn	Sorting, heaps, sliding window (variation)
O(n) time O(n^2 space), O(n) space, O(1) space	● Scan through array linearly ● Sliding Window ● Array/Hash maps ● Does storing the last index of something help? ● Peaks and Valleys? ● Does using another array as DP help solve the questions? ● Does using queue to help find something in alphabetical order help? ● Array: front, back, middle pointer? Or checks. ● Can you use a stack/array if it is about decoding or finding balanced parenthesis or subproblems
logn	Binary tree / binary search?
o(1)	Never be asked

Questions:
- What is my input
- What is my output
- What do i do if my input is empty, = 1
- Any other edge conditions?

- Are there restrictions on my input?
    - Is my input less than the MAX INTEGER REPRESENTATION (if using Java / C++ b/c of overflow, dynamic languages don't matter)
    - Positive only?
    - Integers or float?
    - Can there be a mix of string / ints?
- Should I optimize further (ask this later on before you start coding)
    - Up to you?
        - Ok, i will optimize time, then space then
- Analyze ex.
- At 10 mins get code down even if it is just a function definition
- Can you simplify the problem
- Can I take a moment to think?
- What are the problems I have?

Test cases (new section):
- Add on some additional test cases to capture the edge cases from your question if it comes up and if ex. Test cases provided already do not consider some of the edge cases u thought of.

Brute force / Intuitive Human solution:
- What this means is come up with an intuition as a human being then try to simplify the steps, so that you can find parallel points.
- Ex. If you need to do a 2 sum question, you would think:
    - Ok as a human I would look towards first checking all possible conditions where two numbers equal to my target
    - Refining that you can further say, specifically the way I do that is I have a target in mind, I subtract the base number I am trying to find pairs for and looking if that number exists
    - Which might help guide you to the answer to use some sort of data-structure to help you store a corresponding search value and a value that it pairs to.
- Other examples might be that maybe you have a problem that requires u to check back previous indexes?
    - You might notice ok, i need to check back the previous indexes
    - So checking back linearly as a human is slow, so I need something that is fast. Maybe a hashmap? Or a binary search? etc.

Pseudocode Code:
This section is just to bullet point your answer out.
Ex. If for two sum might be:

- Create a hash map
- Loop through the array

    - For every iteration of the loop, store in hashmap the [target - current value] = current value
- Loop through array again checking O(1) time against current hashmap

Time/Space Complexity of pseudocode above / before you start coding:
- Helps show the interviewer you actually know what you are doing before you start programming and can give proper analysis.
- If you cannot analyze at this point during the interview is fine, just let the interviewer know that you will give the time/space analysis at the end.

Real Code:
- Did you return the answer

Debugging:

FINAL SPACE/TIME COMPLEXITY: