0 - intro to leetcode

The ultimate goal of this project is to learn & implement algorithms by solving Leetcode problems. We'll mostly focus on easy to medium problems, hard can wait for later. For most of your career, you'll rarely have to implement any hard algorithms, trust me.

The biggest benefit that comes with solving leetcode problem, however, is it allows you to practice & improve your coding skills. At some point, you'll see yourself immediately know how to tackle some (recurrent) problems, what data structure & algorithm to use.

Here are some important notes, and please do practice them all the time:

- Click on the link right away so you don't see the solution I write down below 🤣
- Read the problem carefully first!!! You must understand what the problem is, what the inputs and outputs are. This will go the same for your job later.
- Think about the problem in words first, not code. Later on, when you're used to it, you'll see your brain throwing code like popcorn - *for easy problems*. For hard ones, you'll find coming back to words is much easier.
- Think of the easy solution (bruteforce) first. For example, if it's a search problem, try search using loop.
    - come up with examples, use pen/paper to solve those examples first
- Now look at the bruteforce solution and see where you can improvement
    - oftentimes you can find a better solution this way. For example, you can find a nested 3-loop (so O(n^3) time complexity) can be restructured to use only two nested-double-loop (so 2 x O(n^2) which is O(n^2)).

For each problem:

- don't worry if you can't solve it yet. Think for 30m, if you can't solve it, read the solution.
- you'll go through these problems again and again and again (every single time you plan to interview lolz) and trust me you'll have forgotten them already so they'll be brand new again.

It's crucial to understand Big O notation to learn algorithm (there are also other notations). Please read the following tutorials on big O notation:

- https://web.mit.edu/16.070/www/lecture/big_o.pdf
- https://www.geeksforgeeks.org/analysis-algorithms-big-o-analysis/
- https://www.reddit.com/r/learnprogramming/comments/370908/a_simple_introduction_to_big_o_notation/
- asking for help from TA & your own lecture notes

Finally, I must have some typos here and there. Hopefully they don’t make the explanation too confusing.