**How hard was this problem?**

If you had attempted solving this problem directly, there is a good chance that you would have struggled badly. Why? As the complexity of an interview problem increases, cognitive load increases as well. At some point, it becomes overwhelming and you stop operating at maximum efficiency. Note that this threshold may be different for different people.

Real world problems, unlike interview problems are much larger in scope and require days of effort to solve. And it is close to impossible to approach them like a 1 hr interview problem where you jump into coding. You need to divide them into multiple sub-problems, solve each part individually before assembling them back together. This is exactly what we have done here!

Consciously dividing a problem however simple into smaller parts till it becomes brain-dead and then solving them is extremely effective during interviews. Common practice among junior developers is to skip this step and jump directly into coding. How many times have you started writing down a for loop without even thinking what the problem demands? You might sometimes solve problems a tad faster, but you will definitely lack consistency and miss your estimates far too often.

Also when you approach a problem very structurally, the interviewer will clearly see that you can break down a problem of any size into smaller parts and then come up with a solution.