## **What’s Recursion?**

Recursion is like telling a friend to solve a puzzle by first solving a smaller version of it, then building on that. In our case, the forecasting tool uses recursion to figure out future money values by calculating each year’s value based on the previous year. It’s a clean way to break down a big problem into bite-sized steps, making the code easy to follow.

## **How Our Tool Predicts the Future**

Our tool starts with some money and a list of growth rates, like 5% or 3%. It predicts what you’ll have after a few years by applying one growth rate at a time, using a formula: new value = old value × (1 + growth rate). Recursion does this by working backward, solving for year n-1 before year n, until it reaches the starting amount.

## **How Fast Is This Thing?**

* **Speed**: Without any tricks, our recursive method would take O(n) time, where n is the number of years. It makes one call per year, each doing a quick calculation. But if we keep asking for the same years, it could waste time redoing the math.
* **Memory**: It uses O(n) space because each recursive call stacks up, holding data until we reach year 0.

## **Making It Snappier**

To avoid repeating calculations, we stash results in a cache (like a notebook for quick lookups). If we need the value for year 3 again, we just grab it from the cache, no extra work needed. This keeps the first run at O(n) but makes repeat predictions for the same years super fast—O(1). The cache uses O(n) space, but that’s a fair trade for speed, especially if we’re forecasting often. We could also ditch recursion and loop through the years instead, saving some memory (O(1) space), but our recursive version with caching is clear and works great for most financial predictions.