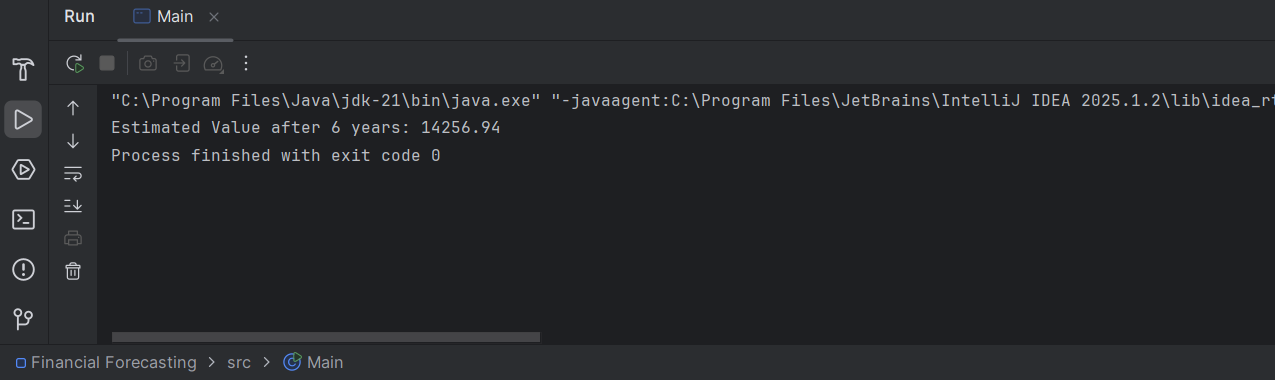
**EXERCISE 7: FINANCIAL FORECASTING**

**OUTPUT**



**1. Understanding Recursive Algorithms**

**WHAT IS RECURSION AND HOW DOES IT HELP?**

Recursion is a problem-solving method where a function calls itself repeatedly to break down a task into smaller, manageable parts. This technique is especially helpful in situations where a problem can be defined in terms of similar sub-problems. For example, computing future financial values over time becomes cleaner with recursion since each year’s value depends on the previous year’s result.

Using recursion can make the code shorter and more intuitive, especially when the pattern of calculation is repetitive and fits a divide-and-conquer approach.

**2. SETUP**

To solve the forecasting problem, a method is created that uses recursion to calculate the estimated future value. It takes the current value, annual growth rate, and number of years as input.

**3. IMPLEMENTATION**

A recursive function is implemented that multiplies the current value by (1 + rate) each year, reducing the remaining years by one until the base case is reached. This method simulates the yearly compound growth in a simplified and clear structure using recursion.

**4. Analysis**

**Time Complexity:**

The recursive solution has a time complexity of **O(n)**, where n represents the number of years. Since one recursive call is made for each year, the total number of calls is directly proportional to the number of years.

**Optimization Techniques:**

Although recursion is elegant, it can become inefficient due to the overhead of multiple function calls. This can be improved in two ways:

* **Switch to iteration**: Using a loop (like for or while) avoids the overhead of recursive stack calls, making the program more memory-efficient and faster.
* **Use memoization (for more complex variants)**: In problems where certain results are computed multiple times (overlapping subproblems), caching the results in a lookup table (memoization) helps avoid redundant calculations and significantly improves performance.

For simple linear recursion like in financial forecasting, **iteration is typically the best optimization**, since it’s more direct and avoids recursion depth issues.