**DSA**

**Exercise 2: E-commerce Platform Search Function:**

ASYMPTOTIC NOTATION

Big O Notation:

-Big O notation describes the upper bound of the time or space complexity of an algorithm in the worst-case scenario as the input size grows.

|  |  |
| --- | --- |
| Complexity | Description |
| O(1) | Constant time |
| O(log n) | Logarithmic time |
| O(n) | Linear time |
| O(n log n) | Linearithmic time |
| O(n²) | Quadratic time |

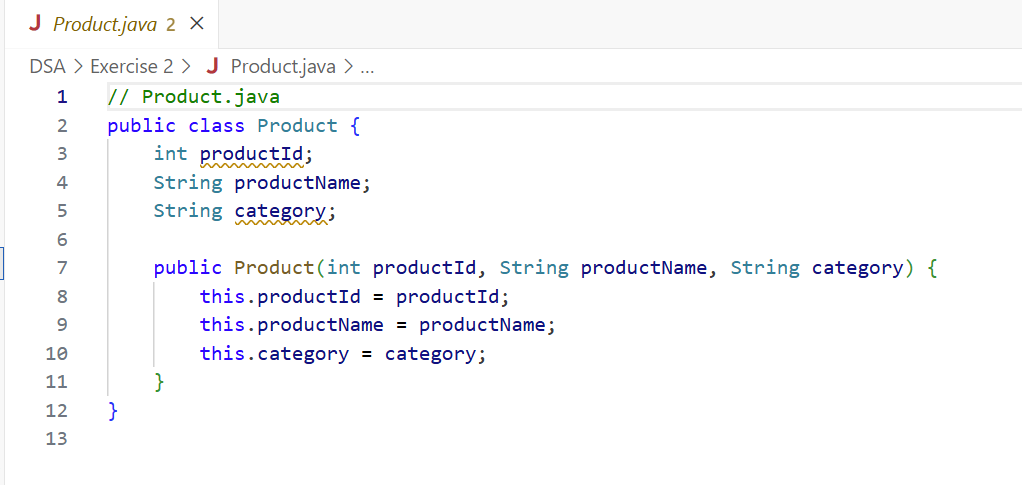
Linear Search: Binary Search (on sorted data):

Best: O(1) → if the element is the first Best: O(1)

Average: O(n/2) ≈ O(n) Average: O(log n)

Worst: O(n) Worst: O(log n)

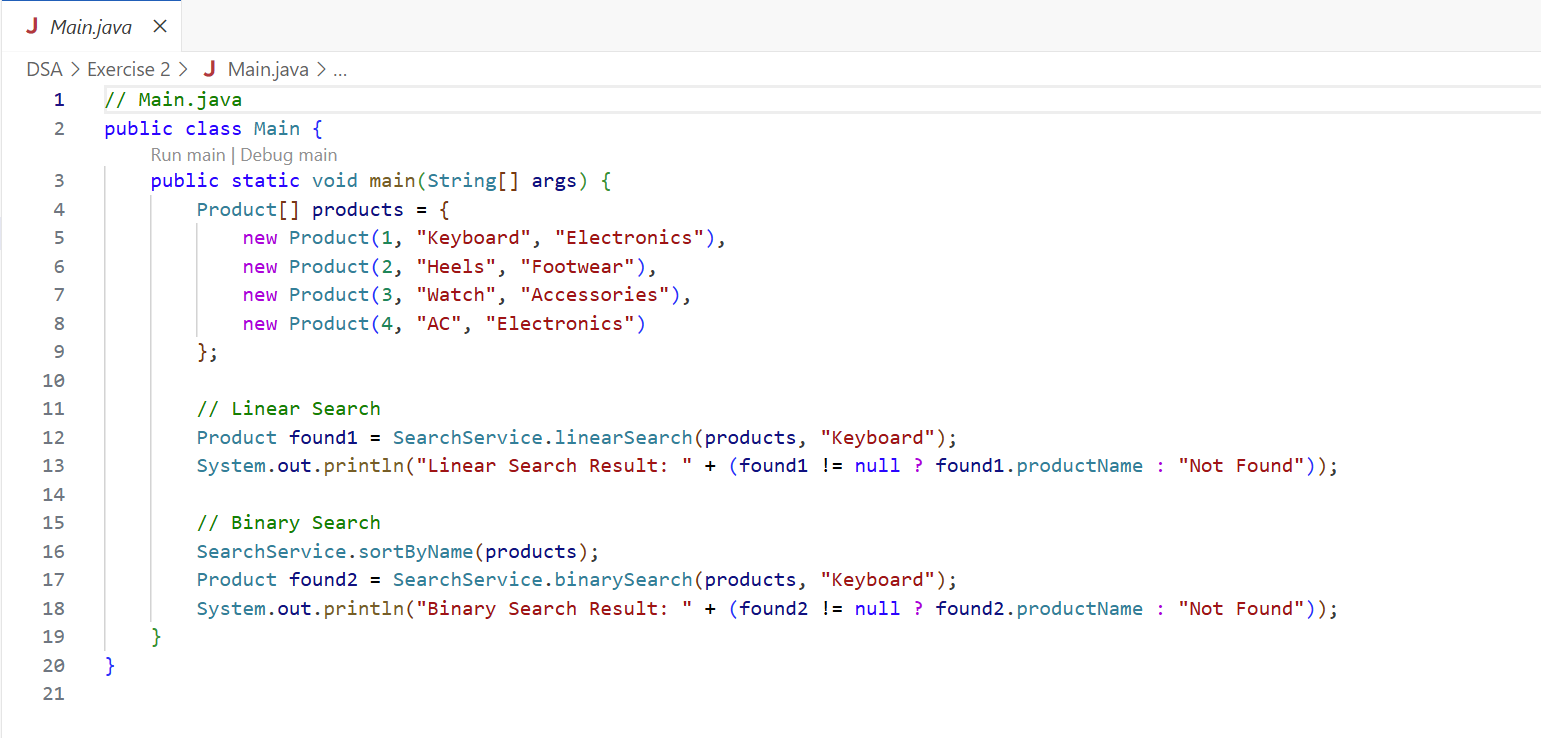
**Product.java:**



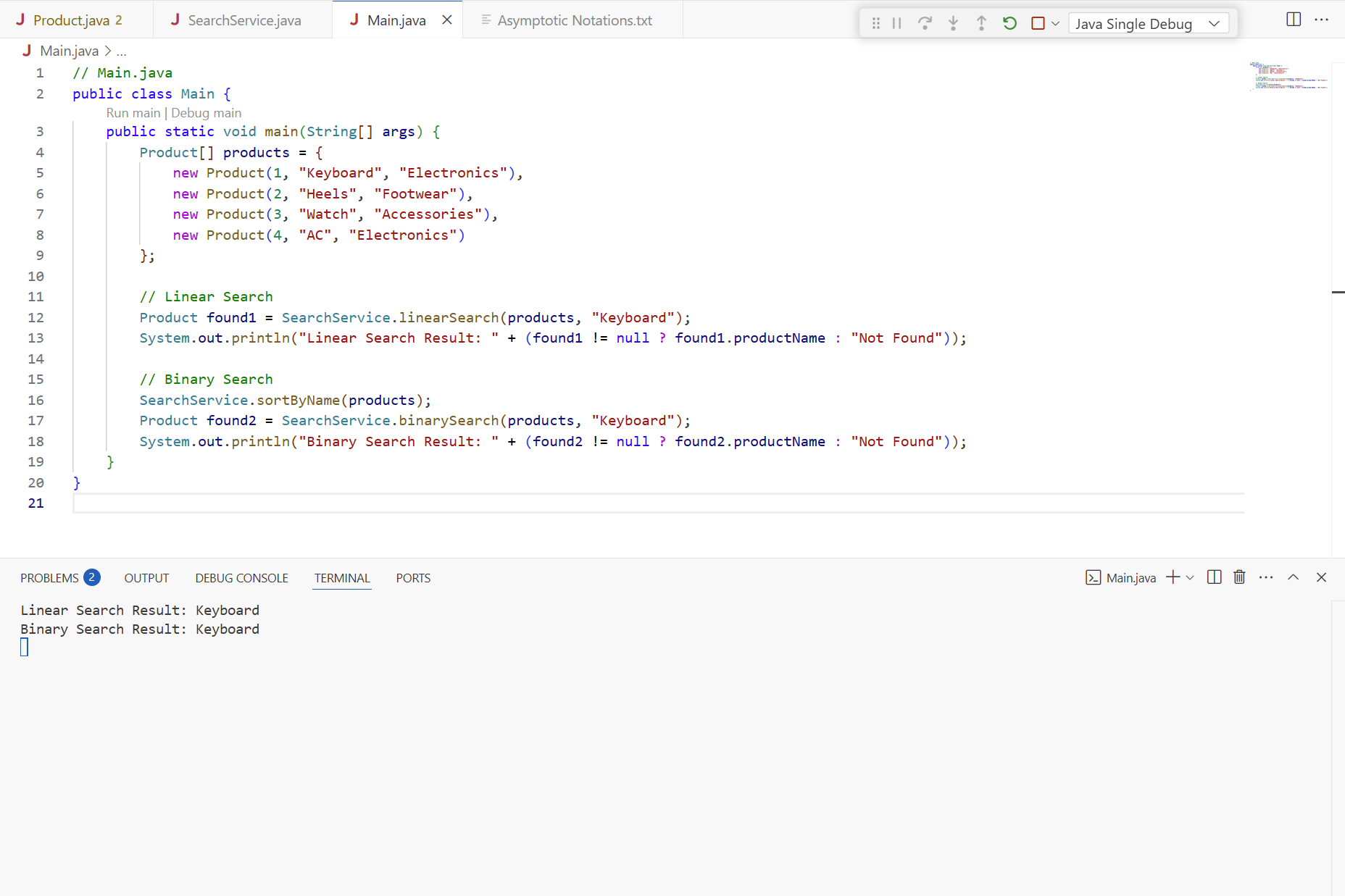
**SearchService.java:**



**Main.java:**



**OUTPUT:**

****

Analysis:

> Time Complexity:

|  |  |  |
| --- | --- | --- |
| Algorithm | Time Complexity | Suitable For |
| Linear Search | O(n) | Small or unsorted datasets |
| Binary Search | O(log n) | Large, sorted datasets |

> Which is more suitable for an E-commerce platform?

Binary Search is much faster than Linear Search for large datasets.

But it requires sorting beforehand (O(n log n)).

**Exercise 7: Financial Forecasting:**

What is Recursion?

- Recursion is a technique where a function calls itself to solve smaller instances of a problem.

Why Use Recursion?

- It breaks down complex problems into simpler subproblems.

- Often used for problems like factorial, Fibonacci, tree traversal, etc.

- In forecasting, it can naturally express repeated patterns like compound growth.

Setup:

We’ll assume the forecast follows compound growth:

Future Value 𝑛 = Future Value 𝑛-1 x (1 + Growth Rate)

We will:

Take initial value P

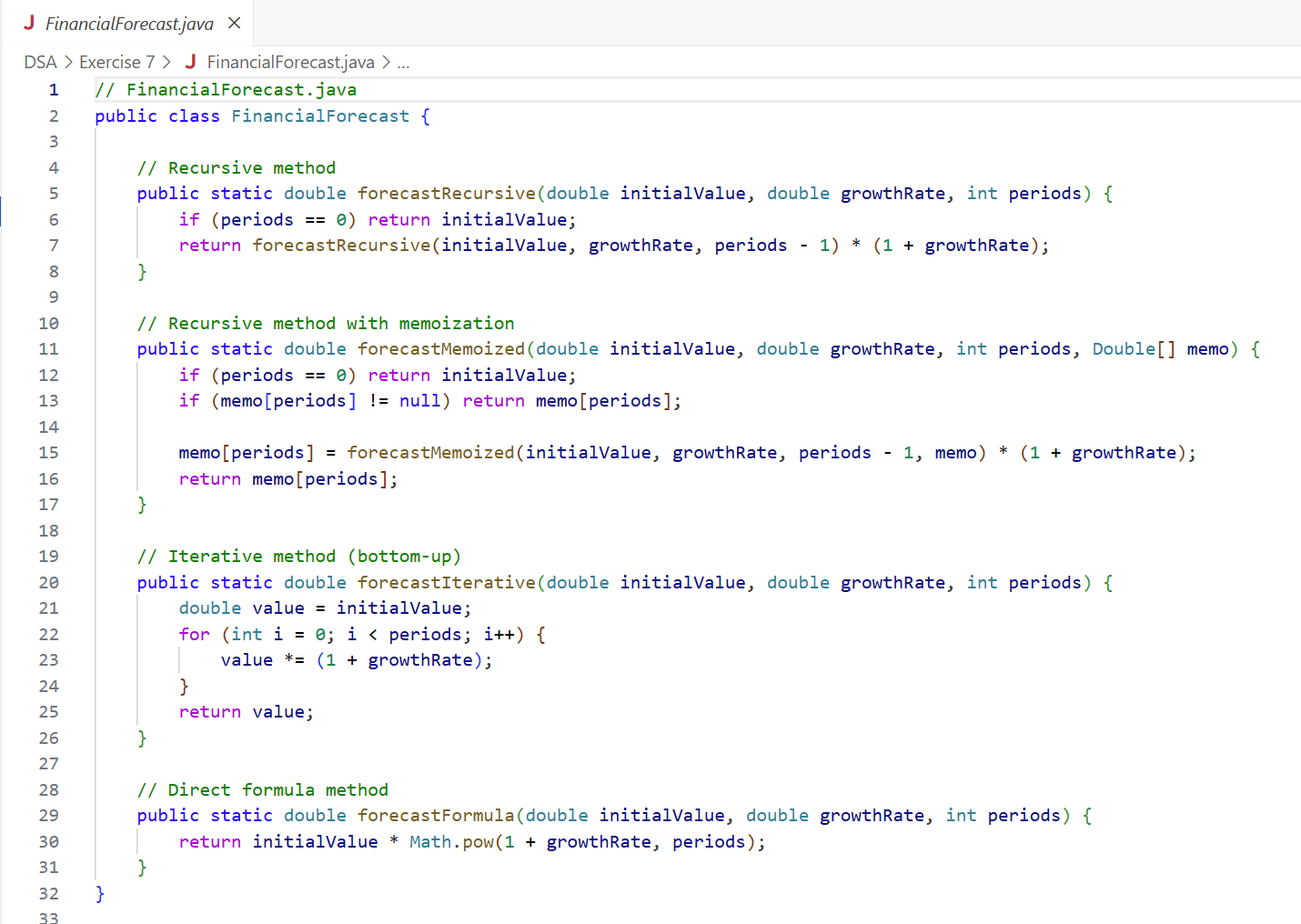
Growth rate r

Number of periods n

Compute value after n periods using recursion.

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Time Complexity | Space Complexity | Notes |
| Recursive | O(n) | O(n) | Simple but stack-heavy |
| Recursive + Memo | O(n) | O(n) | Avoids re computation |
| Iterative | O(n) | O(1) | More efficient in practice |
| Formula (Best) | O(1) | O(1) | Direct and most optimized |

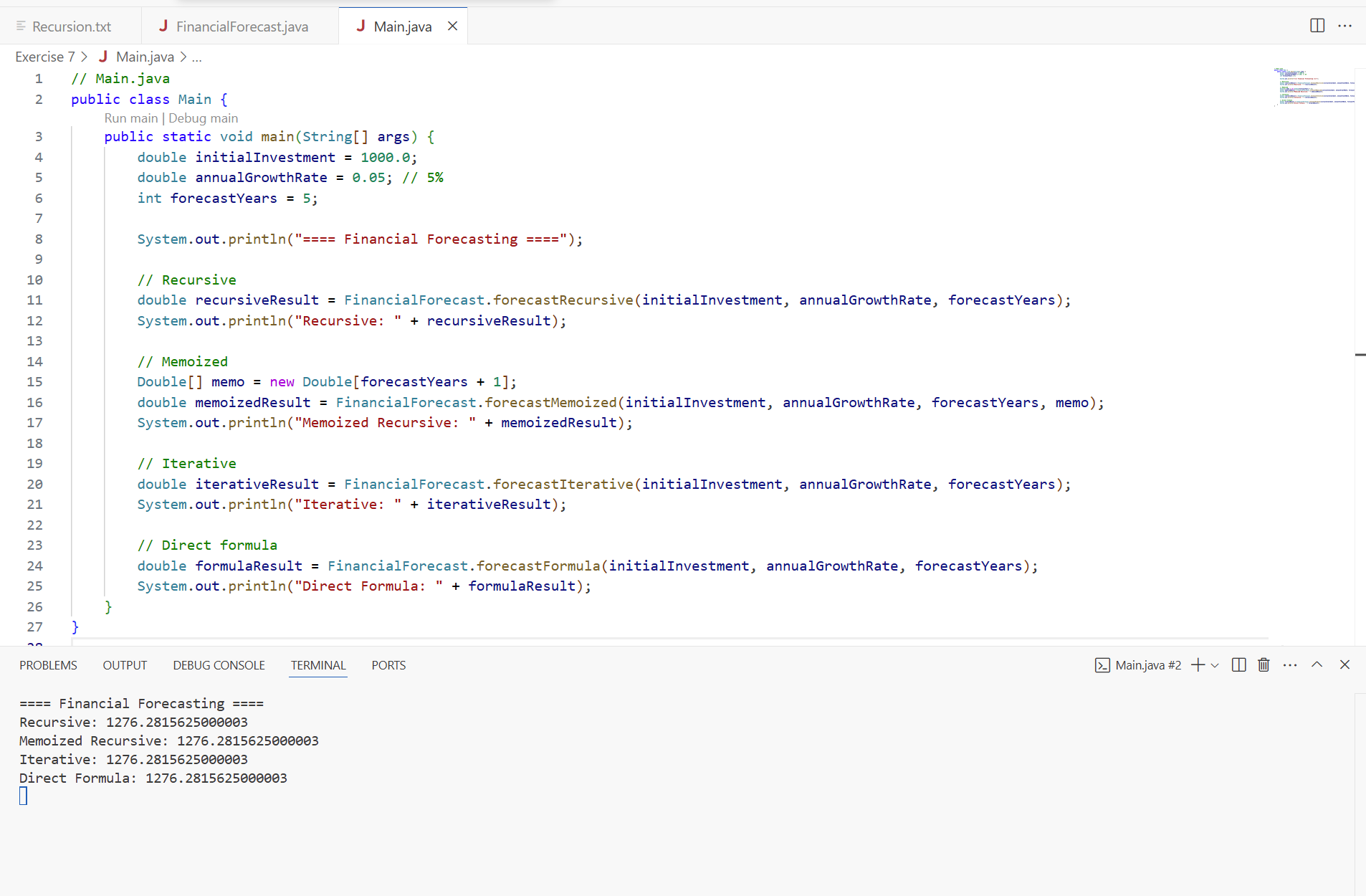
**FinancialForecast.java:**

****

**Main.java:**

****

**OUTPUT:**

****