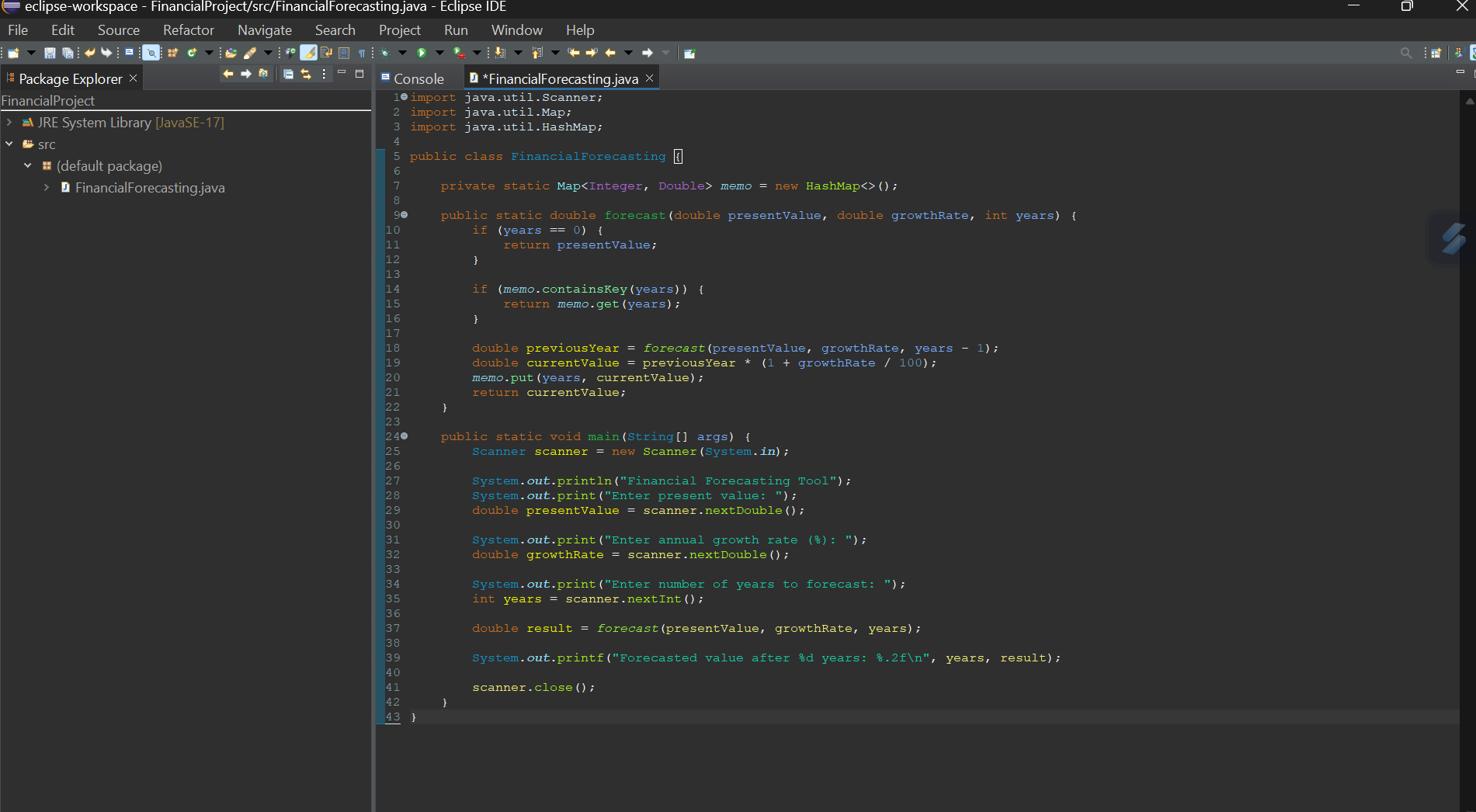
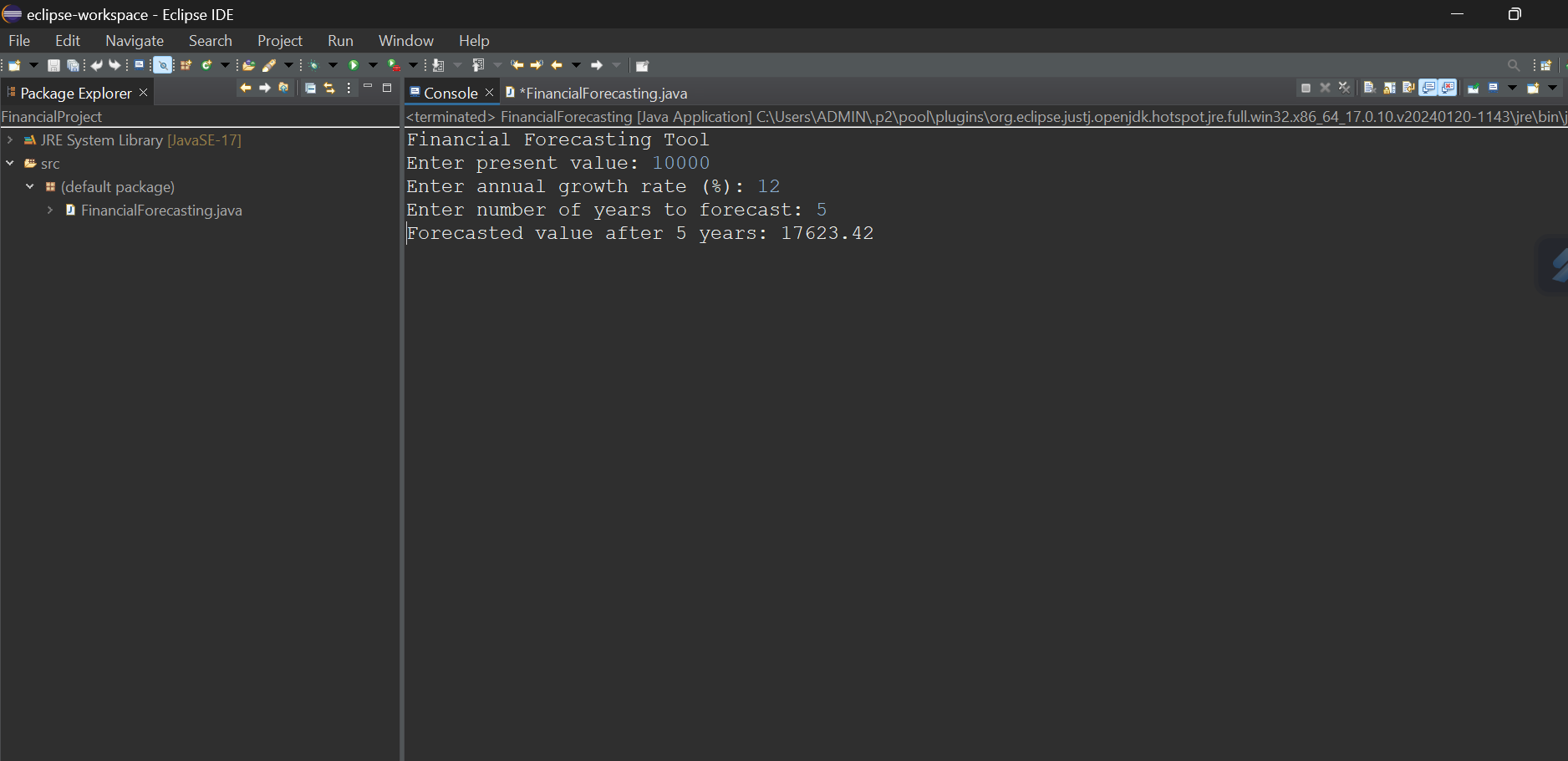
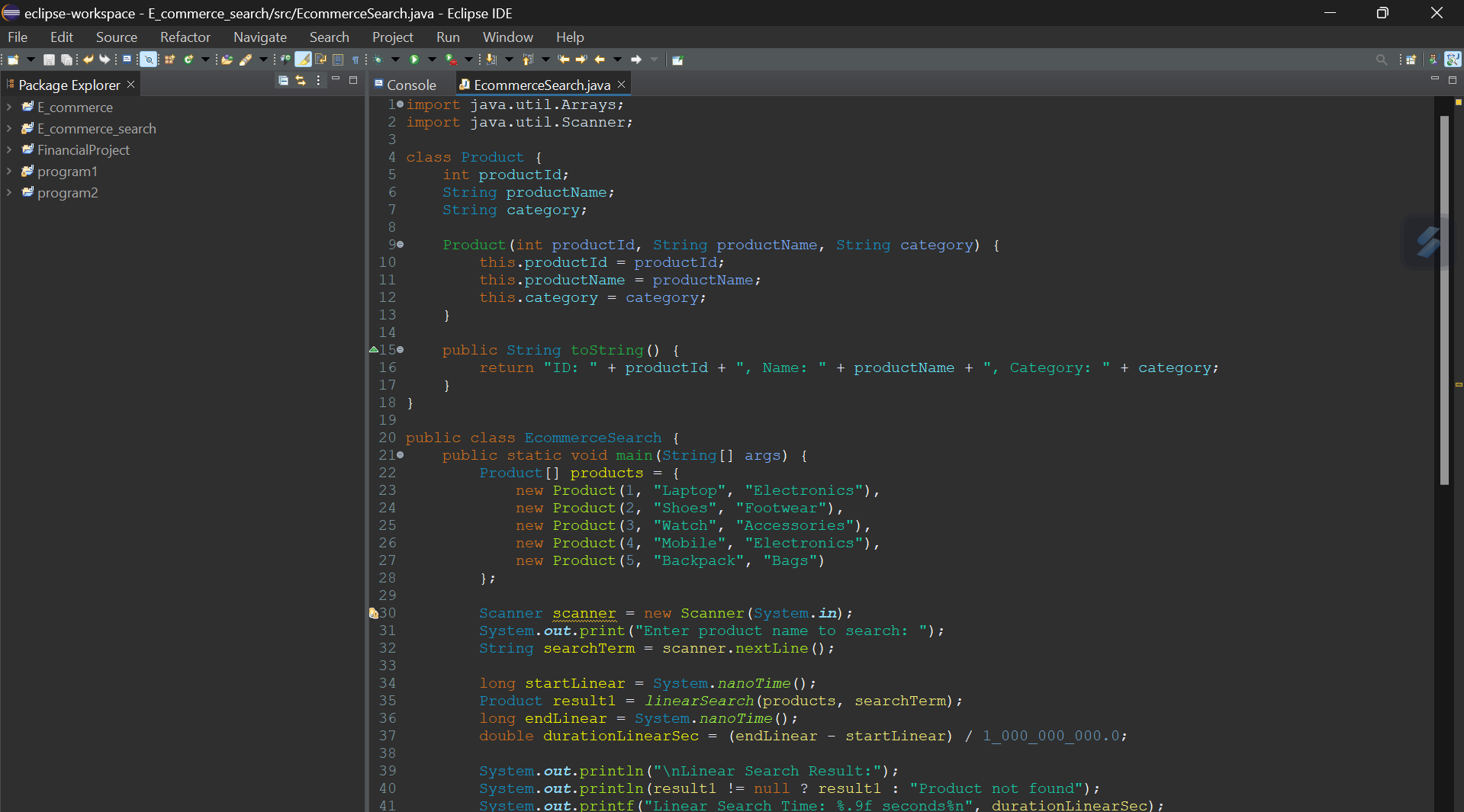
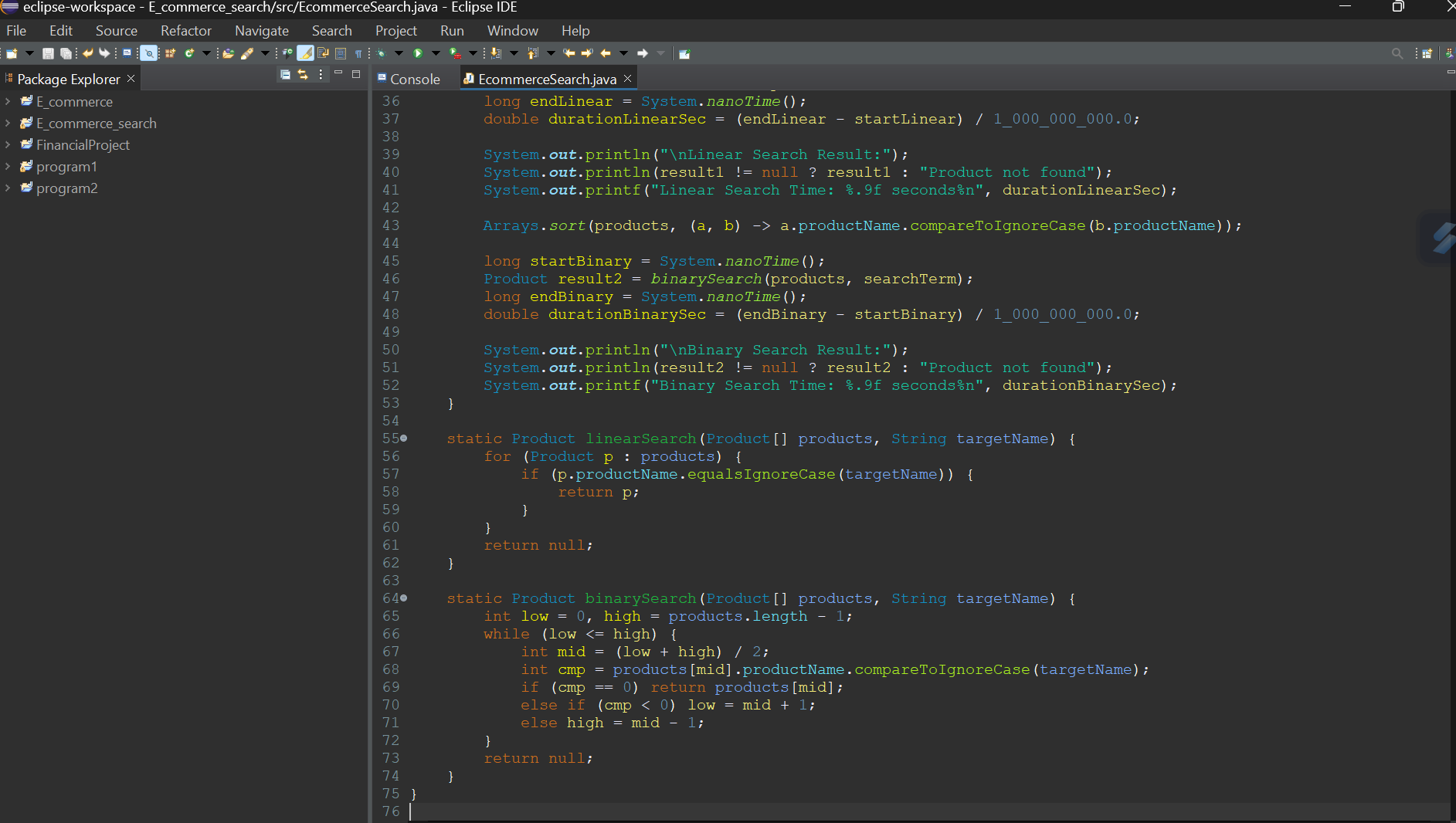
*DATA STRUCTURES AND ALGORITHMS*

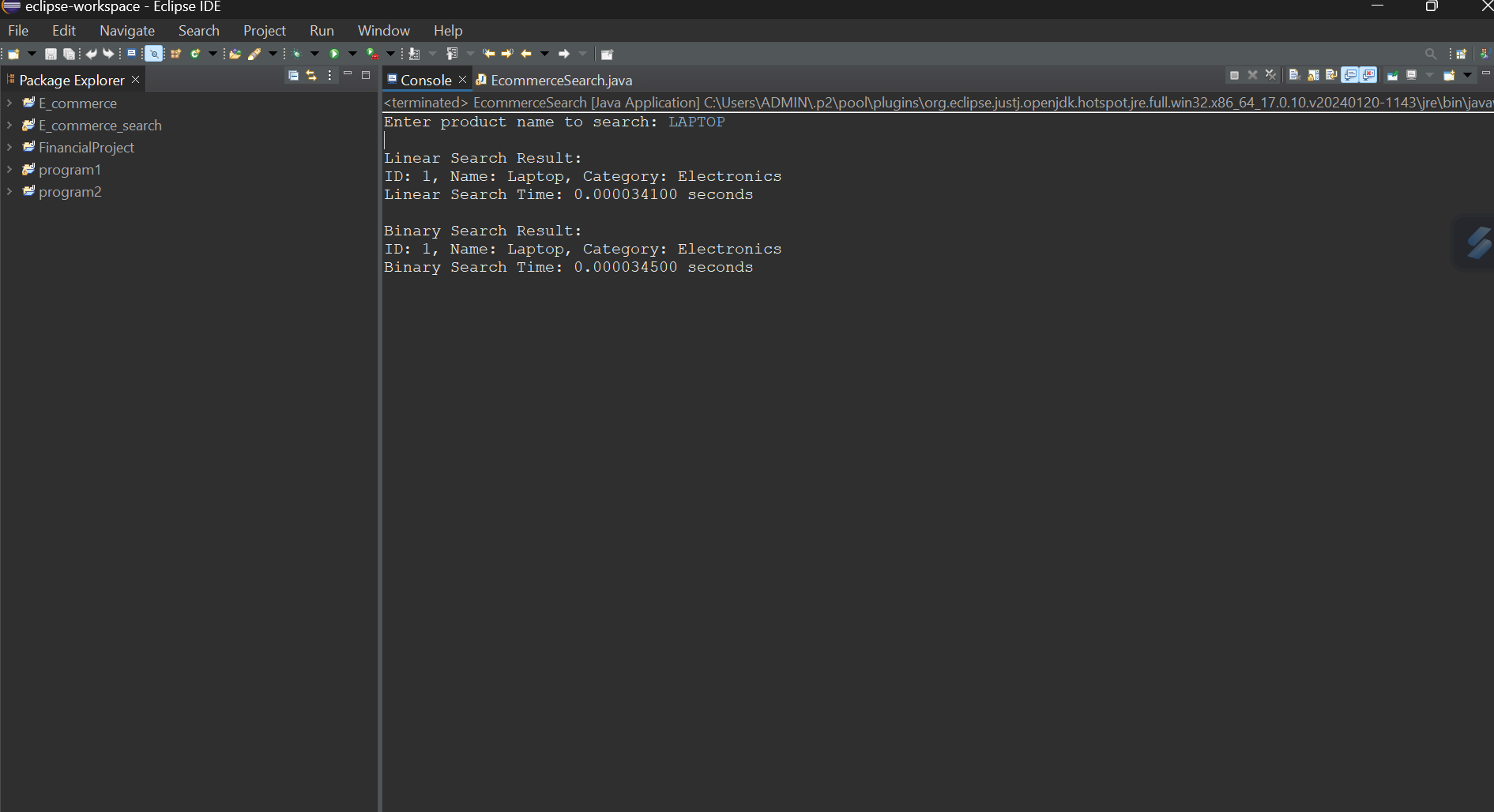
***Exercise 01: Financial Forecasting***

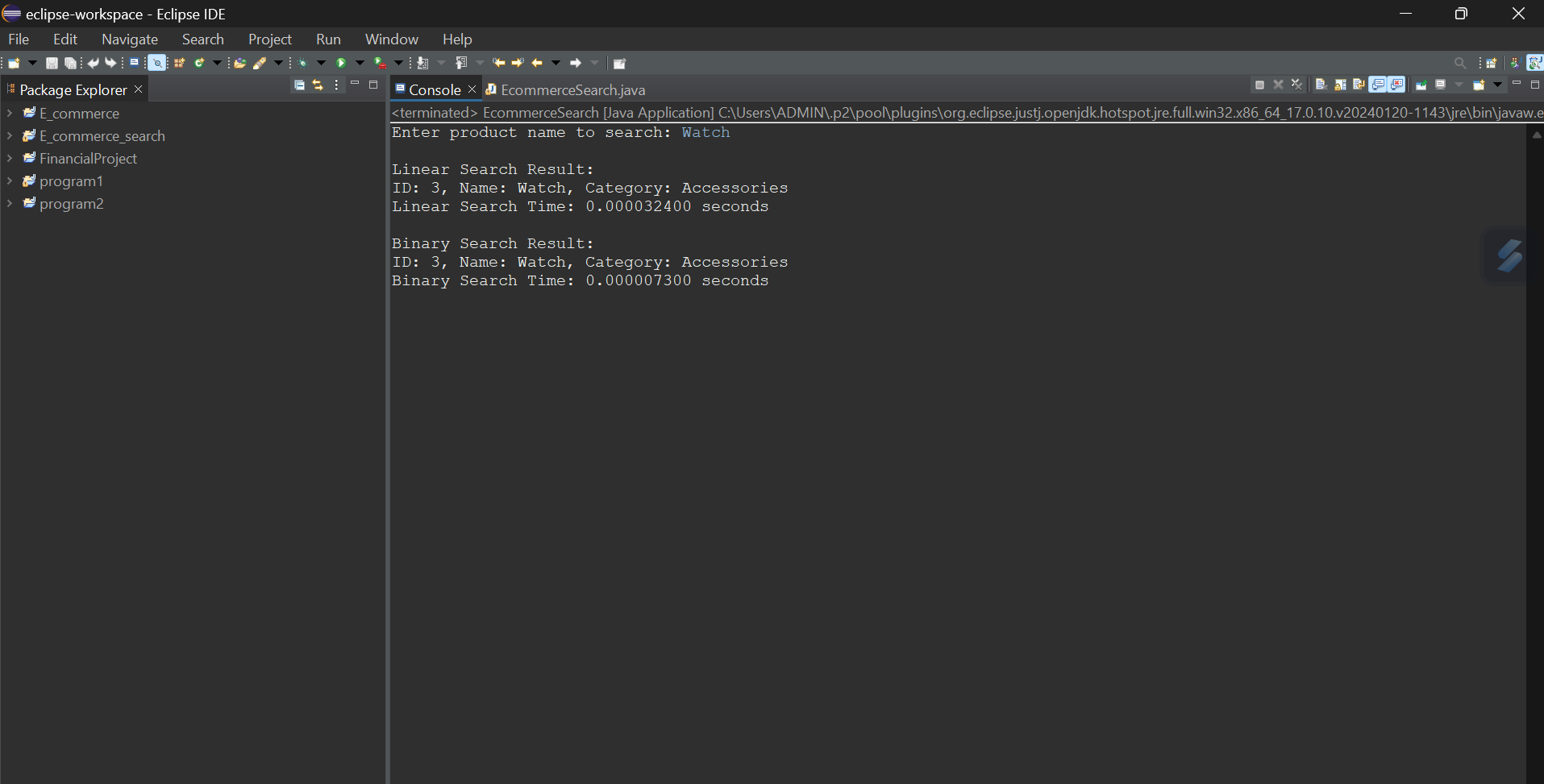
**

***I developed a recursive financial forecasting tool in Java that calculates future investment value based on a given growth rate and time period. The core logic uses recursion to repeatedly apply the compound interest formula, and I optimized the solution using memoization to avoid redundant computations, significantly improving performance. The time complexity of the unoptimized recursive approach is exponential (O(2^n)), but with memoization, it reduces to linear time (O(n)). This recursive method simplifies the logic and ensures efficient handling of forecasting over large time spans***.

***Exercise 02: E-commerce Platform Search Function***

******

******

******

***I implemented a search functionality for an e-commerce platform in Java using both linear and binary search algorithms. I created a Product class with attributes like productId, productName, and category to simulate realistic data. Products are stored in an array, and the search feature allows users to input a product name. Linear search scans the array sequentially, while binary search operates on a sorted array using a divide-and-conquer strategy. To demonstrate efficiency, I measured the runtime for both searches in seconds using System.nanoTime(). The linear search has a time complexity of O(n), making it slower on large datasets, while binary search operates in O(log n), showing significantly faster performance. This comparison clearly shows how algorithm choice impacts real-world performance and helps in choosing the most efficient method for large-scale platforms.***