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

**Scenario:**

You are working on the search functionality of an e-commerce platform. The search needs to be optimized for fast performance.

**What is Big O Notation?**

**Big O Notation** describes how fast or slow an algorithm runs as the input size grows.  
It tells you the worst-case performance and helps compare which algorithm is more efficient.

**Why is it important?**

* It helps you choose the fastest algorithm when performance matters.
* It shows how your code will perform with a small vs very large number of items.

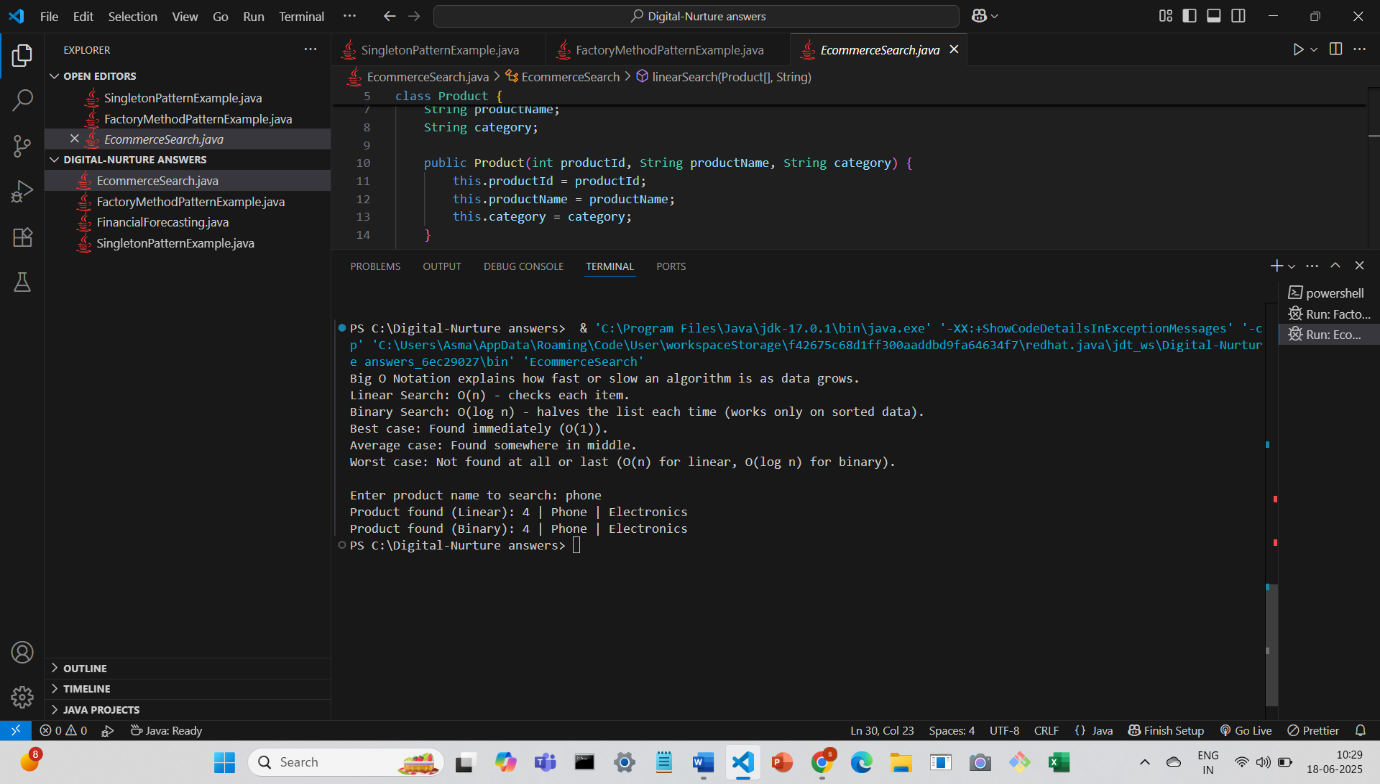
| **Big O** | **Meaning** | **Example** |
| --- | --- | --- |
| O(1) | Constant time | Direct access in array |
| O(n) | Linear time | Linear search |
| O(log n) | Logarithmic time | Binary search |
| O(n²) | Quadratic time | Nested loops (bad!) |

**Best, Average, and Worst Case (for search)**

Let’s take searching for a product in an array:

| **Case** | **Linear Search (unsorted array)** | **Binary Search (sorted array)** |
| --- | --- | --- |
| **Best** | O(1) — found at start | O(1) — found in middle |
| **Average** | O(n/2) ≈ O(n) | O(log n) |
| **Worst** | O(n) — found at end or not | O(log n) — continues halving |

Output Screenshot:



**Conclusion:**

Use Linear Search if:

* You have a small list
* Your data is unsorted

Binary Search if:

* You have a large list
* You can keep the array sorted

In e-commerce, binary search is preferred because performance matters — but you must sort data first or use tree-based or index-based structures for large-scale systems.