**Asymptotic Notation**

Asymptotic notation is a way to describe the performance of an algorithm mathematically, it shows how the time and space increases with the input. It helps you analyze and compare algorithms regardless of computational speed or programming language.

**Big O (Worst Case)**

The maximum time your algorithm will take. Ensures upper bound.

**Big Omega (Best case)**

The minimum time your algorithm will take. Gives lower bound.

**Big Theta (Average case)**

The exact bound (both upper and lower) for typical performance.

**My Solution**

The first class is the Product class, it defines a product each product has

* productId: Unique identifier (e.g., 101).
* productName: Name of the product (e.g., "Laptop").
* category: Product category (e.g., "Electronics").

It includes getters to access these attributes.

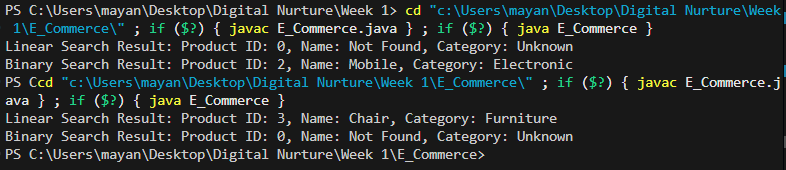
The toString() method is overridden so the object prints.

The second class is the Search class which adds linear search and binary search functionality so users can search any product and get their details.

it contains two methods – one for linear search and one for binary search.

The last class is the E\_Commerce class which is the driver class of the problem

**Sample Output**

****

**Linear vs Binary Search**

| **Algorithm** | **Best Case** | **Average Case** | **Worst Case** | **Time Complexity Class** |
| --- | --- | --- | --- | --- |
| **Linear Search** | **O(1)** (match at start) | **O(n)** (random position) | **O(n)** (not found) | **Linear** |
| **Binary Search** | **O(1)** (match at middle) | **O(log n)** | **O(log n)** | **Logarithmic** |

1. Linear Search:

* Pros:
  + Simple to implement.
  + Works on unsorted data.
  + No pre-processing or sorting required.
* Cons:
  + Slow for large datasets—checks each element one by one.
  + Inefficient when many searches are required.

2. Binary Search:

* Pros:
  + Very fast—cuts search space in half each time.
  + Ideal for read-heavy operations on large datasets.
* Cons:
  + Requires sorted data.
  + May need additional preprocessing (sorting), which is O(n log n) but only done once.

**Binary Search is More Suitable**

* E-commerce platforms often have thousands to millions of products.
* Search needs to be fast and scalable to provide good user experience.
* Products can be sorted once (e.g., alphabetically by name or by ID) and then binary search can be used repeatedly for efficient lookups.