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

**Explain Big O notation and how it helps in analyzing algorithms.Describe the best, average, and worst-case scenarios for search operations.**

### 1 . What is Big O Notation?

Big O notation is used to describe the **efficiency** of an algorithm as the input size increases. It shows how the **time or space** needed by an algorithm grows, especially in the **worst-case scenario**.

### Best, Average & Worst Case in Search

| **Search Type** | **Best Case** | **Average Case** | **Worst Case** |
| --- | --- | --- | --- |
| **Linear Search** | O(1) | O(n) | O(n) |
| **Binary Search** | O(1) | O(log n) | O(log n) |

**Linear Search** checks each item one by one.

**Binary Search** is faster but works only with **sorted data**.

### ****Comparison of Linear Search and Binary Search****

**Linear Search** is a simple search algorithm that checks each element in a list one by one until it finds the target value or reaches the end of the list. It does not require the data to be sorted and works well for small datasets. However, as the dataset grows, linear search becomes less efficient because it may need to scan all elements.

**Binary Search**, on the other hand, is a much faster algorithm but requires the data to be sorted. It works by repeatedly dividing the search space in half, checking the middle element, and deciding whether to continue the search on the left or right half. Because of this divide-and-conquer approach, binary search significantly reduces the number of comparisons needed.

### ****Time Complexity:****

**Linear Search**:

Best Case: O(1) (target is at the beginning)

Average and Worst Case: O(n) (target is in the middle or not present)

**Binary Search**:

Best Case: O(1) (target is exactly in the middle)

Average and Worst Case: O(log n) (search space halves each time)