**Big O Notation**

Big O Notation is used to describe the efficiency of an algorithm in terms of its time or space complexity as the input size grows.

Big O gives a worst-case scenario of how the algorithm behaves as input size n becomes very large, ignoring constants and less significant terms.

**Best, Average, and Worst Case (for Search):**

| **Case** | **Linear Search** | **Binary Search** |
| --- | --- | --- |
| **Best** | O(1) | O(1) |
| **Average** | O(n/2) ~ O(n) | O(log n) |
| **Worst** | O(n) | O(log n) |

**ANALYSIS**  
  
**Time Complexity Comparison:**

| **Algorithm** | **Time Complexity** |
| --- | --- |
| **Linear Search** | O(n) |
| **Binary Search** | O(log n) |

**Conclusion**

**Binary Search** is much faster than Linear Search for large, **sorted** datasets, making it the better choice for an e-commerce platform where search speed is critical.