Step 1: Understand the Problem

# Explain Big O notation and how it helps in analyzing algorithms.

Big O notation is a mathematical notation used to describe the performance or complexity of an algorithm, specifically how its runtime or memory usage grows as the input size increases.

Example-

O(n): Time grows linearly

O(log n) = time grows slowly even with large input

O(1) = constant time

# Describe the best, average, and worst-case scenarios for search operations.

Search type

### Binary Search:

Best case scenario – O(1)

Average scenario - O(n/2)

Worst Case - O(n)

### Linear Search:

Best case scenario – O(1)

Average scenario - O(log n)

Worst Case - O(log n)

Step 4: Analysis

## Compare the time complexity of linear and binary search algorithms.

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

## Discuss which algorithm is more suitable for your platform and why.

E-commerce deals with a huge dataset(thousands of products) and generally, products are stored in a sorted manner using their product Id, so the best algorithm for this will be Binary Search as it will be faster than Linear Search and will be efficient in searching products.