E-Commerce Platform

Q1). Explain Big O notation and how it helps in analyzing algorithms.

Ans 1) **Big O notation** describes how an algorithm's time (or space) requirement grows relative to input size n. It focuses on the **upper bound** — the worst-case growth rate.

**Why it's helpful**: It lets us compare algorithm performance regardless of machine or environment — important for scaling.

Best , Average , and Worst Case for Search Operations :

1. Linear Search :

Best Case : O(1)

Average Case : O(n/2) ~ O(n)

Worst Case : O(log n)

1. Binary Search :

Best Case : O(1)

Average Case : O(log n)

Worst Case : O(log n )

Q2. Compare the time complexity of linear and binary search algorithms.

Ans 2)1. Linear Search :

Time Complexity : O(n)

Requires Sorting : NO

2. Binary Search :

Time Complexity : O(log n)

Requires Sorting : Yes

Q3. Which is more suitable ?

Ans 3)

|  |  |
| --- | --- |
| Platform Size | Suitable Algo. |
| Small Inventory | Linear Search |
| Large E-commerce | Binary Search(with sorting) |

SUMMARY(Flow)

|  |  |
| --- | --- |
| Steps | Description |
| 1. | Explained **Big O Notation** and search performance |
| 2. | |  | | --- | | Created Product class with searchable fields |  |  | | --- | |  | |
| 3. | Implemented **Linear** and **Binary Search** |
| 4. | Analyzed time complexity and linked which to use based on scale |