**Big O Notation**

**Big O notation** is a mathematical way to describe the **efficiency** of an algorithm, especially in terms of:

* **Time complexity** – how fast an algorithm runs as the input size increases.
* **Space complexity** – how much memory it uses.

It focuses on the **growth rate** of an algorithm rather than the exact time or memory used.

The algorithm is analysed by the time taking to execute when the input length is increased and also the memory taken by the code.

We have many algorithms for one action we analyse all codes and compare the time taken by them and memory occupied for same input only then we can know the best algo for the action.

**Best, Average, and Worst-Case in Search**

We assume that we have an array of elements and **Linear Search Algo** is performed then,

* If the search element is found at the starting of the execution then it is the **Best case** i.e. the Search Element is at the starting of the array at 0th index.
* If the search element is in the middle of the array then it is **Average Case.**
* If the Search element is at the end of the array i.e. at the last index then it is the **Worst Case.**

**Comparing Time Complexity: Linear Search vs. Binary Search**

The time complexity for,

**Linear Search** is O(n)

**Binary Search** is O(log n)

But the memory required is more in Binary Search than in Linear Search  
  
Our Platform is an E-commerce, So we require the fastest searching algo because the Serach is main function in our platform.

Our platform deals with **frequent search on sorted data**.  
  
So I finally Conclude that Binary Search Algorithm is most suitable for our Platform