**Big O notation:**

* Big O notation is a mathematical notation that describes the complexity of an algorithm, which is the amount of time or space it requires as the size of the input increases. It's used to classify algorithms according to how their run time or space requirements grow as the input size grows.

**How Big O notation helps in analyzing algorithms:**

* Big O notation helps in analyzing algorithms by providing a standardized way to measure their performance. It allows developers to compare the efficiency of different algorithms and predict how they will perform on large inputs.

**Best, average, and worst-case scenarios for search operations:**

* **Best-case scenario:** The best-case scenario for a search operation is when the target element is found at the beginning of the search space. This is usually represented as O(1), which means the algorithm takes constant time.
* **Average-case scenario:** The average-case scenario for a search operation is when the target element is found in the middle of the search space. This is usually represented as O(n/2), which means the algorithm takes linear time.
* **Worst-case scenario:** The worst-case scenario for a search operation is when the target element is not found in the search space. This is usually represented as O(n), which means the algorithm takes linear time.

**Comparing the time complexity of linear and binary search algorithms:**

* **Linear search:** The time complexity of linear search is O(n), where n is the number of products. This is because in the worst-case scenario, the algorithm has to iterate through the entire array to find the target element.
* **Binary search:** The time complexity of binary search is O(log n), where n is the number of products. This is because the algorithm divides the search space in half with each iteration, reducing the number of comparisons required to find the target element.

**Algorithm most suitable for this problem**:

* Binary search is more suitable for our e-commerce platform because it has a faster time complexity than linear search. This is particularly important for large datasets, where linear search can become slow and inefficient. Binary search, on the other hand, can handle large datasets with ease, making it a more scalable solution.