**Exercise 2: E-commerce Platform Search Function**

**Step 1: Understand Asymptotic Notation**

**Big O Notation**

Big O notation is used to describe the upper bound of the time complexity of an algorithm in the worst-case scenario. It provides a high-level understanding of the algorithm's efficiency by focusing on how the runtime or space requirements grow as the input size increases.

* **O(1)**: Constant time - The algorithm's runtime does not depend on the input size.
* **O(n)**: Linear time - The runtime grows linearly with the input size.
* **O(log n)**: Logarithmic time - The runtime grows logarithmically with the input size.
* **O(n^2)**: Quadratic time - The runtime grows quadratically with the input size.

**Best, Average, and Worst-Case Scenarios**

* **Best-case**: The minimum time an algorithm can take to complete. For example, in searching, this could be finding the desired element at the first position.
* **Average-case**: The expected time an algorithm takes to complete, averaged over all possible inputs.
* **Worst-case**: The maximum time an algorithm can take to complete. This is crucial for understanding the performance guarantees.

**Time Complexity of Linear and Binary Search**

* **Linear Search:**
  + **Best-case:** O(1) - The product is found at the first position.
  + **Average-case:** O(n) - The product is somewhere in the middle of the array.
  + **Worst-case:** O(n) - The product is at the last position or not present.
* **Binary Search:**
  + **Best-case:** O(1) - The product is found at the middle position initially.
  + **Average-case:** O(log n) - The product is found after a few comparisons.
  + **Worst-case:** O(log n) - The product is at the first or last position, or not present, requiring the maximum number of comparisons.

**Suitable Algorithm for the E-commerce Platform**

* **Binary Search:**
  + **Advantages:** Much more efficient for larger datasets due to its logarithmic time complexity. It is especially suitable for read-heavy operations where the data is mostly static.
  + **Requirements:** The dataset must be sorted, which adds a preprocessing step (O(n log n) for sorting) but is negligible compared to the gains in search efficiency for large datasets.
  + **Use Case:** Ideal for an e-commerce platform with a large number of products where quick search performance is crucial.
* **Linear Search:**
  + **Advantages:** Simple and does not require the data to be sorted.
  + **Disadvantages:** Inefficient for large datasets due to its linear time complexity.
  + **Use Case:** Suitable for small datasets or when the overhead of maintaining a sorted dataset is not justified.