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

**Big O Notation**

Big O notation is a mathematical representation used to describe the upper bound of an algorithm’s runtime or space complexity in terms of the size of the input data. It helps in analyzing the efficiency of algorithms by focusing on their growth rates rather than exact runtimes, providing a way to compare different algorithms regardless of hardware and software differences.

O(1): Constant time – the operation’s time is unaffected by the size of the input data.

O(n): Linear time – the operation’s time grows linearly with the input size.

O(log n): Logarithmic time – the operation’s time grows logarithmically with the input size.

O(n^2): Quadratic time – the operation’s time grows quadratically with the input size.

Best, Average, and Worst-case Scenarios for Search Operations

**Linear Search:**

Best-case: O(1) – The target element is at the first position.

Average-case: O(n/2) = O(n) – The target element is in the middle.

Worst-case: O(n) – The target element is at the last position or not present.

**Binary Search:**

Best-case: O(1) – The target element is in the middle.

Average-case: O(log n) – The target element is anywhere in the array.

Worst-case: O(log n) – The target element is at the last position to be checked or not present.

Time Complexity Comparison

Linear Search:

Best-case: O(1)

Average-case: O(n)

Worst-case: O(n)

Binary Search:

Best-case: O(1)

Average-case: O(log n)

Worst-case: O(log n)

Linear Search:

Suitable for small datasets.

Doesn’t require data to be sorted.

Binary Search:

More suitable for large datasets due to its O(log n) time complexity.

Requires data to be sorted, which can be an overhead if the dataset changes frequently.