**EXERCISE 2**

**E-commerce Platform Search Function**

**1. Understanding Asymptotic Notation**

**Big O Notation**

Big O notation is a mathematical representation used to describe the performance or complexity of an algorithm. It specifically focuses on the upper bound of the time complexity, giving an idea of how the algorithm's runtime grows relative to the input size. This helps in analyzing and comparing the efficiency of algorithms, especially for large datasets.

* **O(1)**: Constant time – The algorithm's runtime is not dependent on the input size.
* **O(n)**: Linear time – The runtime increases linearly with the input size.
* **O(log n)**: Logarithmic time – The runtime grows logarithmically with the input size.

By understanding Big O notation, we can gauge how well an algorithm scales and choose the most efficient one for our needs.

**Best, Average, and Worst-Case Scenarios for Search Operations**

* **Linear Search**:
  + **Best Case**: O(1) – The target item is found at the first position.
  + **Average Case**: O(n) – On average, the search will need to check half of the elements.
  + **Worst Case**: O(n) – The target item is found at the end or not at all, requiring a check of all elements.
* **Binary Search**:
  + **Best Case**: O(1) – The target item is found in the middle of the array.
  + **Average Case**: O(log n) – The search repeatedly halves the search space, requiring logarithmic time.
  + **Worst Case**: O(log n) – The search narrows down to a single element or concludes the item is not present.

**2. Setup**

Create a class for the items you want to search, including attributes such as productId, productName, and category.

**3. Implementation**

**Linear Search**

Linear search scans each element in the array until it finds the target item or reaches the end of the array. This algorithm does not require the array to be sorted.

**Binary Search**

Binary search is used on a sorted array. It repeatedly divides the search interval in half, comparing the target value to the middle element of the array.

**4. Analysis**

**Time Complexity Comparison**

* **Linear Search**:
  + **Time Complexity**: O(n)
  + **Description**: Scans each element in the array sequentially, making it less efficient for large datasets.
* **Binary Search**:
  + **Time Complexity**: O(log n)
  + **Description**: Efficiently narrows down the search space by half each time, making it much faster for large datasets if the array is sorted.

**Suitability**

For an e-commerce platform:

* **Binary Search** is generally more suitable if the dataset is large and sorted, due to its logarithmic time complexity. This makes it ideal for applications where fast search performance is critical.
* **Linear Search** is simpler and more flexible but less efficient for large datasets. It is suitable for smaller or unsorted datasets.

If the products can be sorted and need frequent searching, binary search offers superior performance. If sorting is not feasible or the dataset is small, linear search may be adequate.