**Superset-Id: 6364597  
Name: SRINJOY PAUL**

**WEEK- 1 DATA STRUCTURES AND ALGORITHM   
HANDS ON**

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

**Explanation of Big O Notation:**

Big O notation is a mathematical way of expressing how an algorithm's execution time or space requirement grows as the input size increases. It provides a high-level view of algorithm efficiency and helps engineers evaluate scalability and performance in various conditions**.**

**Best Case Scenario (Least Effort Required)-  
Search:**

In the most efficient scenario, the target item is found immediately—typically as the first element checked. This represents the fastest possible outcome, requiring minimal computational effort.

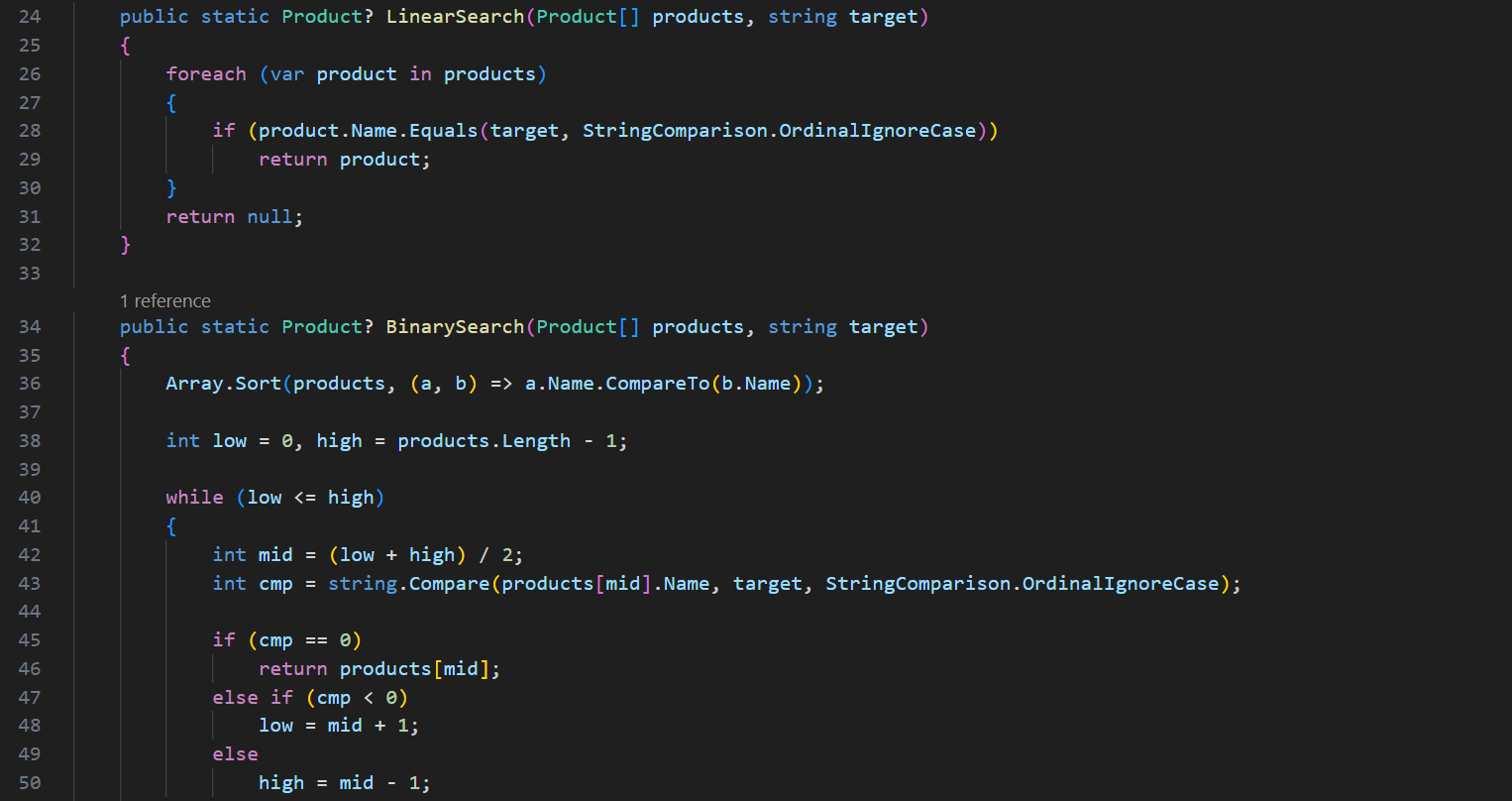
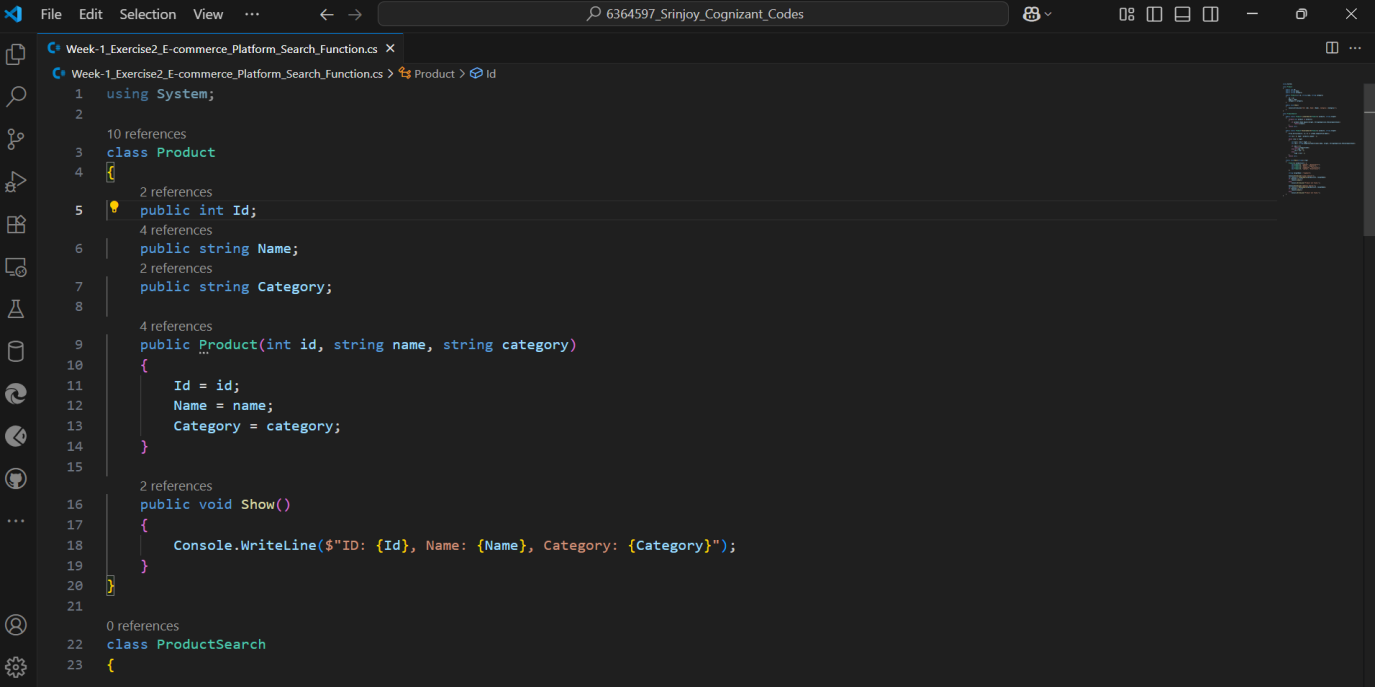
**Average Case Scenario (Moderate Effort Expected)-**

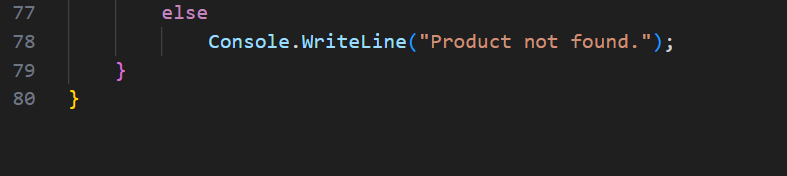
**Search:**

For most practical cases, the target item is located somewhere in the middle of the dataset. This scenario reflects a balanced workload, where the search algorithm performs a moderate number of comparisons before success.

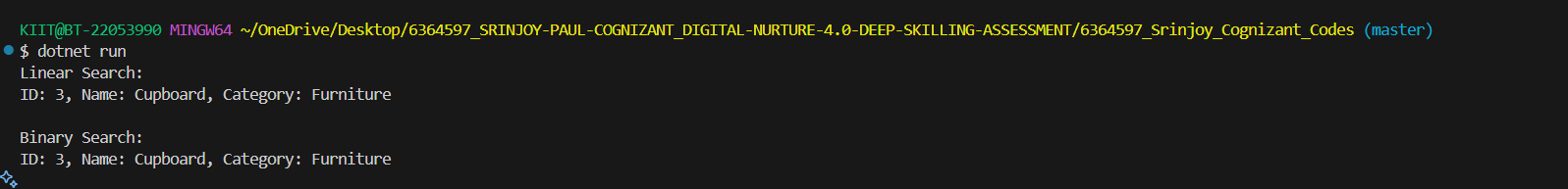
**Worst Case Scenario (Maximum Effort Needed)-  
Search:**

The least efficient situation occurs when the target item is either absent from the list or positioned as the last element to be checked. Here, the algorithm must traverse the entire dataset, resulting in the highest computational cost.

**CODE:**

****

**Output:**



**Analysis**

Method of search:

* **Linear Search**-   
  Time Complexity : O(n)  
  Use Case (Best): Ideal for small datasets or unsorted lists.
* **Binary Search -**

Time Complexity: O(log n)

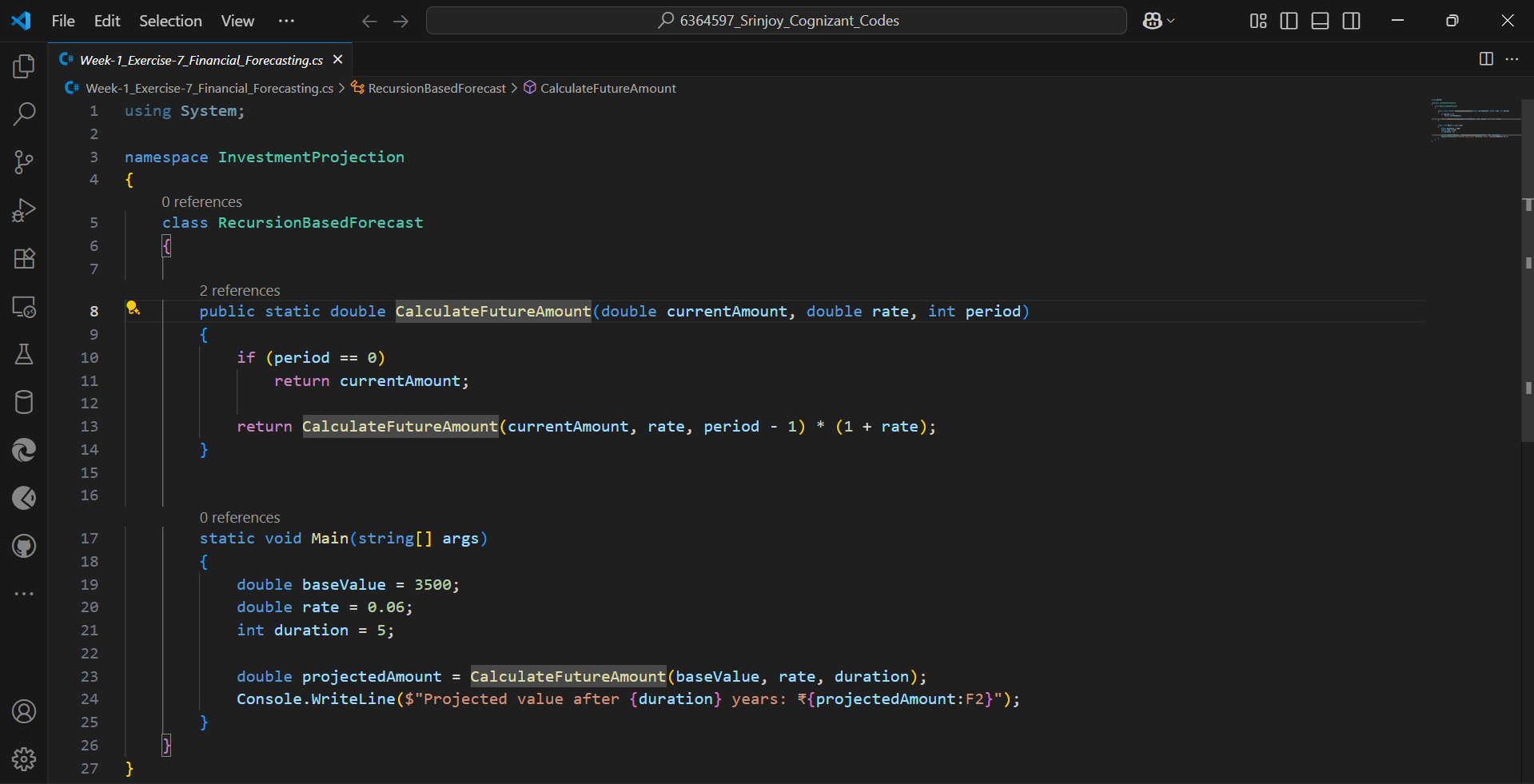
Use Case (Best): Efficient for large, sorted datasets.

Binary Search O(log n) Efficient for large, sorted datasets

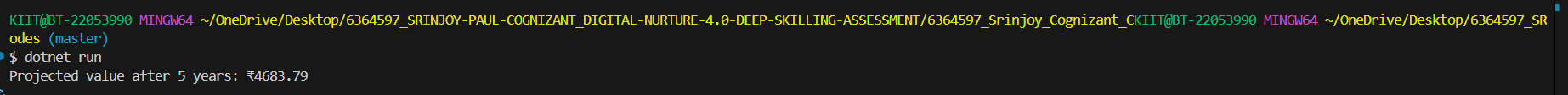
**Exercise 7: Financial Forecasting**

Recursion is a programming technique where a function solves a problem by calling itself to break the problem into smaller, more manageable sub-problems. By dividing the task into simpler versions of itself, recursion often leads to cleaner and more concise code, particularly for tasks involving repetitive operations or nested structures.

**CODE:**

****

**Output:**

****

-> The algorithm has a linear time complexity of O(n), as it makes one recursive call per year in the calculation.  
-> Optimization Strategy:

* If results for certain years are computed multiple times , it leads to unnecessary repetition.
* We can enhance performance by:

1. Applying memoization to cache previous results.
2. Or by rewriting it as an iterative - based approach (loop) .