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

1. **Understanding Asymptotic Notation:**
   1. Explain Big O notation and how it helps in analysing algorithms.

* Big O Notation is a powerful tool used in computer science to describe the time or space complexity of algorithms. It is a way of expressing the upper bound of an algorithm’s time or space complexity.  
  Few Big O Notations are:  
  - O(1) [Constant time]  
  - O(log n) [Logarithmic time]  
  - O(n) [Linear time]  
  - O(n log n) [Linearithmic time]  
  - O(n2) [Quadratic time]  
  - O(2n) [Exponential time]  
  - O(n!) [Factorial time]
* Big O helps in analysing algorithms by comparing their algorithmic efficiency independently of hardware or programming language. For instance, even without running, we can say that Binary Search [O (log n)] is better than Linear Search [O(n)] for larger data. It also helps in guiding optimization efforts by pinpointing what to improve.
  1. Describe the best, average, and worst-case scenarios for search operations.
* For any general searching algorithm, the best, average and worst case scenarios are:  
  - Best Case: Minimum number of steps required.  
  - Average Case: Expected number of steps for a random output.  
  - Worst Case: Maximum number of steps required.

**2 and 3. Setup and Implementation:**

* Shown in code.

**4. Analysis:**

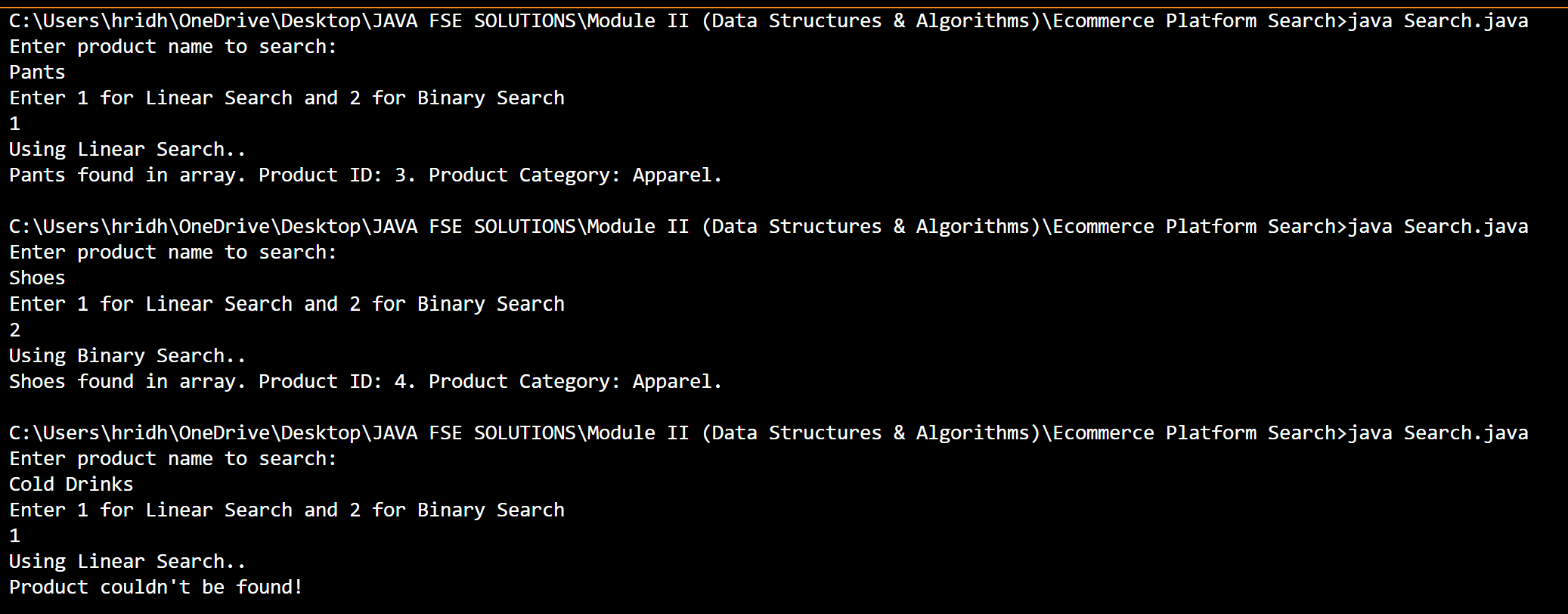
a. Comparison of time complexity of linear and binary search algorithms.

* For Linear Search:  
  Best Case: O(1)  
  Average Case: O(n)  
  Worst Case: O(n)
* For Binary Search:  
  Best Case: O(1)  
  Average Case: O(log n)  
  Worst Case: O(log n)

Although linear search works well with unsorted, small datasets, it struggles with large datasets as binary search is much faster for larger, sorted datasets as it keeps on halving the size of the array to find the target.

b. Discussing which algorithm is more suitable for this platform.  
 - Binary Search is more suitable as we sort the data before searching.

**Output:**

****