Algorithms Data Structures

# Exercise 02 – E-commerce Platform Search Function

### Understanding Asymptotic Notations

Big O notation represents the upper bound of the running time of an algorithm. It determines how the runtime or space requirement of an algorithm scales within input size n. It helps in comparing algorithms independently of hardware and understanding scalability and performance of bottlenecks.

If f(n) describes the running time of an algorithm, f(n) is O(g(n)) if there exist a positive constant C and n0 such that, 0 ≤ f(n) ≤ cg(n) for all n ≥ n0

It returns the highest possible output value (big-O) for a given input. The execution time serves as an upper bound on the algorithm's time complexity.

BigO

Best, Average and Worst case in search

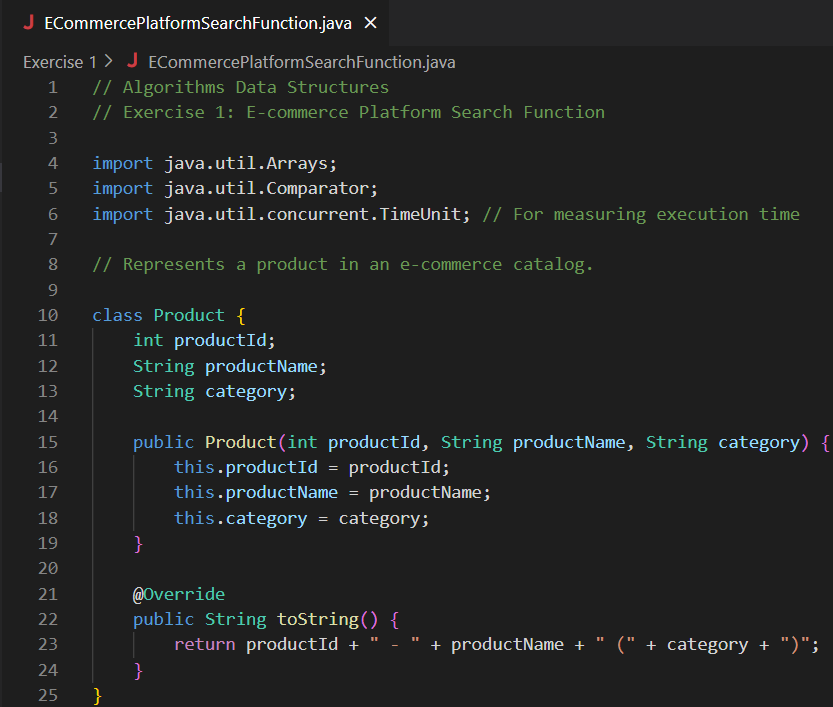
|  |  |  |  |
| --- | --- | --- | --- |
| Algorithm | Best Case | Average Case | Worst Case |
| Linear Search | O(1) | O(n) | O(n) |
| Binary Search | O(1) | O(log n) | O(log n) |

Best Case: First match.

Average Case: Element is somewhere in the middle.

Worst Case: Not found / at the end (Linear) or max log splits (Binary).

### Setup– Product Class

Represents a product in an e-commerce catalog

### Implementation – Linear Search and Binary Search

Linear and Binary Search algorithms

A computer screen shot of a program code

AI-generated content may be incorrect.

Main.java

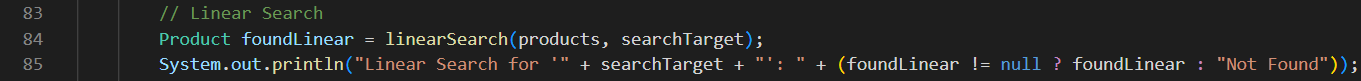
A screen shot of a computer program

AI-generated content may be incorrect.

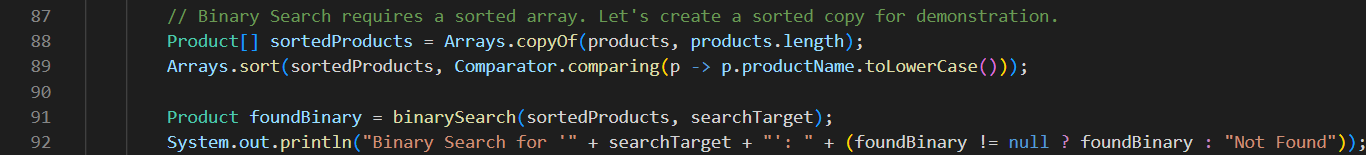
Search Algorithms



Linear Search



Binary Search



Performance Analysis with a Large Dataset

A screen shot of a computer

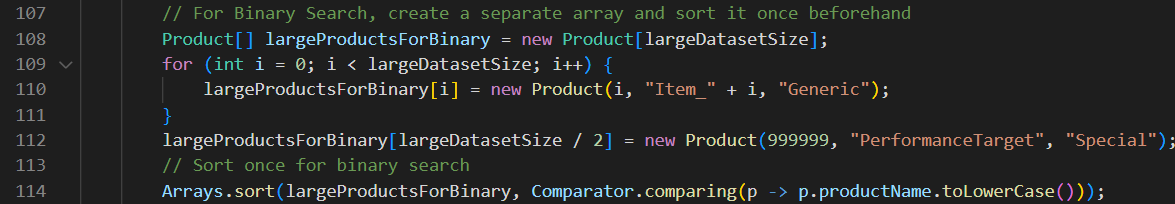
AI-generated content may be incorrect.

Create products for linear search

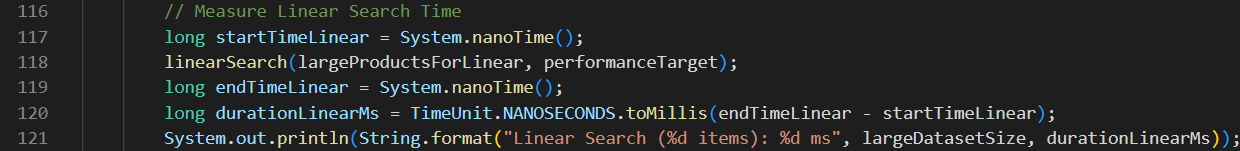
A black background with colorful text

AI-generated content may be incorrect.

For Binary Search



Measure Linear Search Time



Measure Binary Search Time (on pre-sorted array)

A black screen with colorful text

AI-generated content may be incorrect.

Outputs:

A computer screen with white text

AI-generated content may be incorrect.

### Analysis and Comparison

Time Complexity of Linear Search and Binary Search

|  |  |  |
| --- | --- | --- |
| Algorithm | Time Complexity | Space Complexity |
| Linear Search | O(n) | O(1) |
| Binary Search | O(log n) | O(1) |

Binary Search algorithm is more suitable for the E-Commerce platform:

* Performance advantage - Binary search is significantly faster (O (log n)) compared to linear search (O(n)). For 1,000,000 products: Linear search may take up to 1,000,000 comparisons. Binary search will take about log₂ (1,000,000) ≈ 20 comparisons.
* Sorted data is common: E-commerce platforms typically maintain sorted data for fast search, sorting, and filtering (by product ID, price, name, etc.). Product listings are often indexed in databases, which naturally support efficient search trees or binary search structures.
* Better User experience: Fast search = faster results = better user experience. Customers expect real-time or near-instant search results.

# Exercise 02 – Financial Forecasting

### Understanding recursive algorithms

Recursion is a programming technique where a function calls itself to solve a problem.

Each recursive call works on a smaller subproblem, bringing the solution closer with each call until it reaches a base case — a condition that stops further recursion.

### Setup and Implementation

A screen shot of a computer program

AI-generated content may be incorrect.Recursive method to calculate future value

Memoized (optimized recursive) method

A computer screen with text

AI-generated content may be incorrect.

Iterative method

A screen shot of a computer code

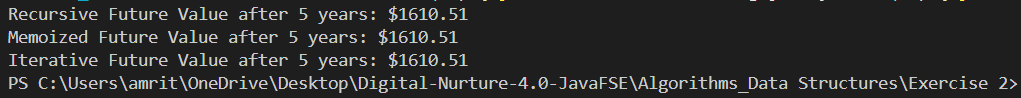
AI-generated content may be incorrect.

Main.java

A screen shot of a computer code

AI-generated content may be incorrect.

Output:



### 4. Analysis

Time complexity of recursive algorithm:

|  |  |  |
| --- | --- | --- |
| Method | Time Complexity | Space Complexity |
| Basic Recursion | O(n) | O(n) |
| Memoized Version | O(n) | O(n) |

The basic recursive version makes n recursive calls. Memoized version stores intermediate results, avoiding recomputation.

Optimizing recursive solutions to minimize computation:

* Memoization: Store results of already computed subproblems (as shown above).
* Bottom-Up Approach (DP): Use a loop instead of recursion to build the result iteratively.
* For very large values of n where recursion depth could lead to StackOverflowError.
* In performance-critical real-world financial systems, iterative solutions or precomputed formulas are preferred.