Exercise 2: E-commerce Platform Search Function

# Objective

To implement and compare linear and binary search algorithms in a simulated e-commerce platform, where product information is searched by name. This exercise demonstrates performance differences and optimization using appropriate algorithm selection.

# Java Code

import java.util.Arrays;  
import java.util.Comparator;  
import java.util.Scanner;  
  
class Product {  
 int productId;  
 String productName;  
 String category;  
  
 public Product(int productId, String productName, String category) {  
 this.productId = productId;  
 this.productName = productName;  
 this.category = category;  
 }  
  
 public String toString() {  
 return "Product ID: " + productId + ", Name: " + productName + ", Category: " + category;  
 }  
}  
  
public class ECommerceSearch {  
  
 public static Product linearSearch(Product[] products, String targetName) {  
 for (Product p : products) {  
 if (p.productName.equalsIgnoreCase(targetName)) {  
 return p;  
 }  
 }  
 return null;  
 }  
  
 public static Product binarySearch(Product[] products, String targetName) {  
 Arrays.sort(products, Comparator.comparing(p -> p.productName.toLowerCase()));  
 int left = 0, right = products.length - 1;  
  
 while (left <= right) {  
 int mid = left + (right - left) / 2;  
 int compare = products[mid].productName.compareToIgnoreCase(targetName);  
 if (compare == 0) return products[mid];  
 else if (compare < 0) left = mid + 1;  
 else right = mid - 1;  
 }  
 return null;  
 }  
  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.in);  
 Product[] products = {  
 new Product(101, "Laptop", "Electronics"),  
 new Product(102, "Shoes", "Footwear"),  
 new Product(103, "Smartphone", "Electronics"),  
 new Product(104, "T-shirt", "Clothing"),  
 new Product(105, "Book", "Stationery")  
 };  
  
 System.out.println("Enter product name to search:");  
 String targetName = sc.nextLine();  
  
 Product linearResult = linearSearch(products, targetName);  
 System.out.println("Linear Search Result: " + (linearResult != null ? linearResult : "Product not found."));  
  
 Product binaryResult = binarySearch(products, targetName);  
 System.out.println("Binary Search Result: " + (binaryResult != null ? binaryResult : "Product not found."));  
  
 System.out.println("Time Complexity:");  
 System.out.println("Linear Search: O(n)");  
 System.out.println("Binary Search: O(log n)");  
 }  
}

# Sample Input & Output

Input:  
Enter product name to search: Laptop  
  
Output:  
Linear Search Result: Product ID: 101, Name: Laptop, Category: Electronics  
Binary Search Result: Product ID: 101, Name: Laptop, Category: Electronics  
  
Time Complexity:  
Linear Search: O(n)  
Binary Search: O(log n)

# Analysis

Time Complexity:  
- Linear Search: O(n) — Scans each product one by one.  
- Binary Search: O(log n) — Requires sorted input, but faster for large datasets.  
  
Optimization:  
Use linear search for small or unsorted product arrays. For large datasets that are sorted, binary search provides better performance. In real-world platforms, indexing or hash maps can further improve search performance.