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

**Scenario:**

You are working on the search functionality of an e-commerce platform. The search needs to be optimized for fast performance.

**Steps:**

1. **Understand Asymptotic Notation:**
   * Explain Big O notation and how it helps in analyzing algorithms.
   * Describe the best, average, and worst-case scenarios for search operations.
2. **Setup:**
   * Create a class **Product** with attributes for searching, such as **productId, productName**, and **category**.
3. **Implementation:**
   * Implement linear search and binary search algorithms.
   * Store products in an array for linear search and a sorted array for binary search.
4. **Analysis:**
   * Compare the time complexity of linear and binary search algorithms.
   * Discuss which algorithm is more suitable for your platform and why.

import java.util.Arrays;

import java.util.Comparator;

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;

}

@Override

public String toString() {

return "Product ID: " + productId + ", Name: " + productName + ", Category: " + category;

}

}

public class ECommerceSearch {

// Linear Search - Searches each product one by one

public static Product linearSearch(Product[] products, String targetName) {

for (Product product : products) {

if (product.productName.equalsIgnoreCase(targetName)) {

return product;

}

}

return null;

}

// Binary Search - Works only on sorted array

public static Product binarySearch(Product[] products, String targetName) {

int left = 0, right = products.length - 1;

while (left <= right) {

int mid = left + (right - left) / 2;

int comparison = products[mid].productName.compareToIgnoreCase(targetName);

if (comparison == 0)

return products[mid];

else if (comparison < 0)

left = mid + 1;

else

right = mid - 1;

}

return null;

}

public static void main(String[] args) {

// Sample Product List

Product[] products = {

new Product(1, "Laptop", "Electronics"),

new Product(2, "Phone", "Electronics"),

new Product(3, "Shoes", "Footwear"),

new Product(4, "Watch", "Accessories"),

new Product(5, "T-Shirt", "Clothing")

};

// Linear Search Example

System.out.println("=== Linear Search ===");

Product linearResult = linearSearch(products, "Watch");

if (linearResult != null)

System.out.println("Found: " + linearResult);

else

System.out.println("Product not found.");

// Sort products by name before Binary Search

Arrays.sort(products, Comparator.comparing(p -> p.productName.toLowerCase()));

// Binary Search Example

System.out.println("\n=== Binary Search ===");

Product binaryResult = binarySearch(products, "Watch");

if (binaryResult != null)

System.out.println("Found: " + binaryResult);

else

System.out.println("Product not found.");

}

}

**Output**

=== Linear Search ===

Found: Product ID: 4, Name: Watch, Category: Accessories

=== Binary Search ===

Found: Product ID: 4, Name: Watch, Category: Accessories