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

**Understanding the Problem:**In e-commerce, users expect fast product searches. Using efficient algorithms reduces response time and improves user satisfaction. Depending on whether the data is sorted, different search techniques are used: linear search for unsorted arrays and binary search for sorted arrays.

**Setup and Implementation:**  
We define a Product class with:

* productId (int)
* productName (String)
* category (String)

Linear search is applied directly. For binary search, the product list is first sorted by name.

**Java Code:**

j

import java.util.\*;

class Product {

int productId;

String productName;

String category;

Product(int id, String name, String cat) {

this.productId = id;

this.productName = name;

this.category = cat;

}

}

public class EcommerceSearch {

static int linearSearch(Product[] arr, String name) {

for (int i = 0; i < arr.length; i++) {

if (arr[i].productName.equalsIgnoreCase(name)) return i;

}

return -1;

}

static int binarySearch(Product[] arr, String name) {

int l = 0, r = arr.length - 1;

while (l <= r) {

int m = (l + r) / 2;

int cmp = arr[m].productName.compareToIgnoreCase(name);

if (cmp == 0) return m;

if (cmp < 0) l = m + 1;

else r = m - 1;

}

return -1;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("How many products do you want to enter? ");

int n = sc.nextInt();

Product[] products = new Product[n];

for (int i = 0; i < n; i++) {

System.out.println("Enter details for product " + (i + 1));

System.out.print("Product ID: ");

int id = sc.nextInt();

System.out.print("Product Name: ");

String name = sc.next();

System.out.print("Category: ");

String cat = sc.next();

products[i] = new Product(id, name, cat);

}

System.out.print("Enter product name to search: ");

String key = sc.next();

int lIndex = linearSearch(products, key);

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

int bIndex = binarySearch(products, key);

System.out.println("Linear Search Result Index: " + lIndex);

System.out.println("Binary Search Result Index: " + bIndex);

}

}

**Time Complexity Analysis:**

* Linear Search: O(n)
* Binary Search: O(log n) (requires sorted array)

**Optimization Discussion:**  
If the product list is dynamic and unsorted, linear search is the fallback. However, for a sorted product list, binary search is significantly faster. To optimize further, use a HashMap<String, Product> with the product name as the key for O(1) average lookup time.