**Data Structures Algorithms**

**EXERCISE 2: E Commerce Platform Search Functionality**

**Source Code**

import java.util.Arrays;

import java.util.Comparator;

// Step 2: Product class

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

}

}

public class Main {

// Step 3: Linear search

public static Product linearSearch(Product[] products, int id) {

for (Product product : products) {

if (product.productId == id) {

return product;

}

}

return null;

}

// Step 3: Binary search

public static Product binarySearch(Product[] sortedProducts, int id) {

int left = 0;

int right = sortedProducts.length - 1;

while (left <= right) {

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

if (sortedProducts[mid].productId == id) {

return sortedProducts[mid];

} else if (sortedProducts[mid].productId < id) {

left = mid + 1;

} else {

right = mid - 1;

}

}

return null;

}

public static void main(String[] args) {

// Sample data

Product[] products = {

new Product(101, "Ear Phones", "Electronics"),

new Product(203, "Jeans", "Apparel"),

new Product(150, "Note Book", "Education"),

new Product(305, "Mobile Phone", "Electronics"),

new Product(120, "Heels", "Footwear")

};

// Step 1: Asymptotic Notation (display explanation)

System.out.println("Big O Notation helps analyze algorithm efficiency.");

System.out.println("Linear Search: Best O(1), Avg/Worst O(n)");

System.out.println("Binary Search: Best O(1), Avg/Worst O(log n) — needs sorted array\n");

// Step 3: Linear Search

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

Product result1 = linearSearch(products, 150);

System.out.println(result1 != null ? result1 : "Product not found");

// Step 3: Prepare sorted array for binary search

Product[] sortedProducts = Arrays.copyOf(products, products.length);

Arrays.sort(sortedProducts, Comparator.comparingInt(p -> p.productId));

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

Product result2 = binarySearch(sortedProducts, 150);

System.out.println(result2 != null ? result2 : "Product not found");

// Step 4: Analysis

System.out.println("\nAnalysis:");

System.out.println("Linear search scans each item — simple but slower for large datasets.");

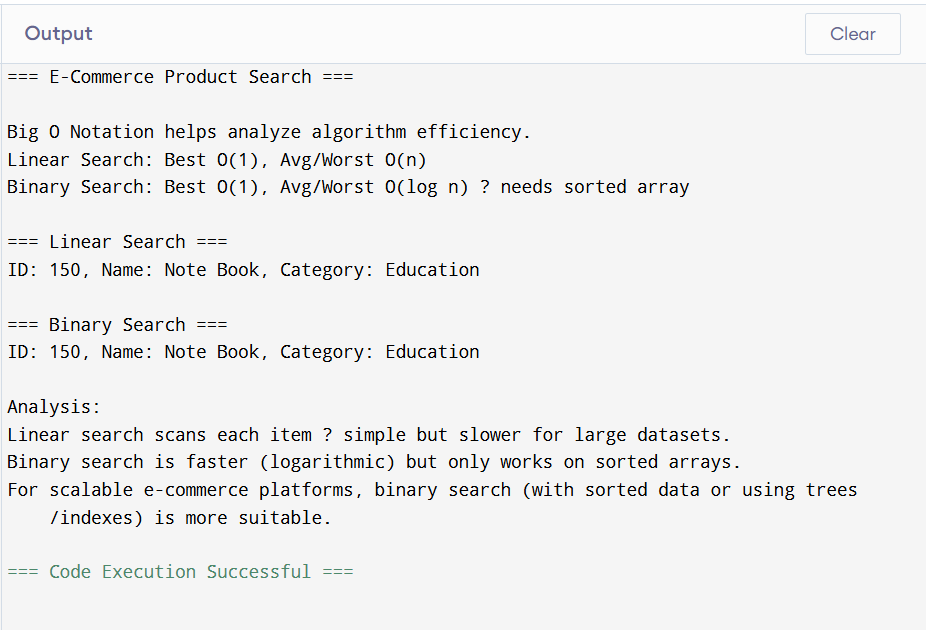
System.out.println("Binary search is faster (logarithmic) but only works on sorted arrays.");

System.out.println("For scalable e-commerce platforms, binary search (with sorted data or using trees/indexes) is more suitable.");

}

}

**Output**

****