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

**Product.java**

package SearchFunction;

public class Product

{

int productId;

String productName;

String category;

public Product(int productId, String productName, String category) {

super();

this.productId = productId;

this.productName = productName;

this.category = category;

}

*@Override*

public String toString() {

return "Product [productId=" + productId + ", productName=" + productName + ", category=" + category + "]";

}

public int getProductId() {

return productId;

}

public void setProductId(int productId) {

this.productId = productId;

}

public String getProductName() {

return productName;

}

public void setProductName(String productName) {

this.productName = productName;

}

public String getCategory() {

return category;

}

public void setCategory(String category) {

this.category = category;

}

}

**Main.java**

package SearchFunction;

import java.util.Arrays;

import java.util.Comparator;

public class Main

{

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

for (Product p : products) {

if (p.productName.equalsIgnoreCase(name)) {

return p;

}

}

return null;

}

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

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

while (left <= right) {

int mid = (left + right) / 2;

int cmp = products[mid].productName.compareToIgnoreCase(name);

if (cmp == 0)

return products[mid];

else if (cmp < 0)

left = mid + 1;

else

right = mid - 1;

}

return null;

}

public static void main(String[] args) {

Product[] products = {

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

new Product(2, "Shirt", "Clothing"),

new Product(3, "Book", "Education"),

new Product(4, "Phone", "Electronics")

};

Product result1 = linearSearch(products, "Phone");

System.out.println("Linear Search Result: " + (result1 != null ? result1 : "Not Found"));

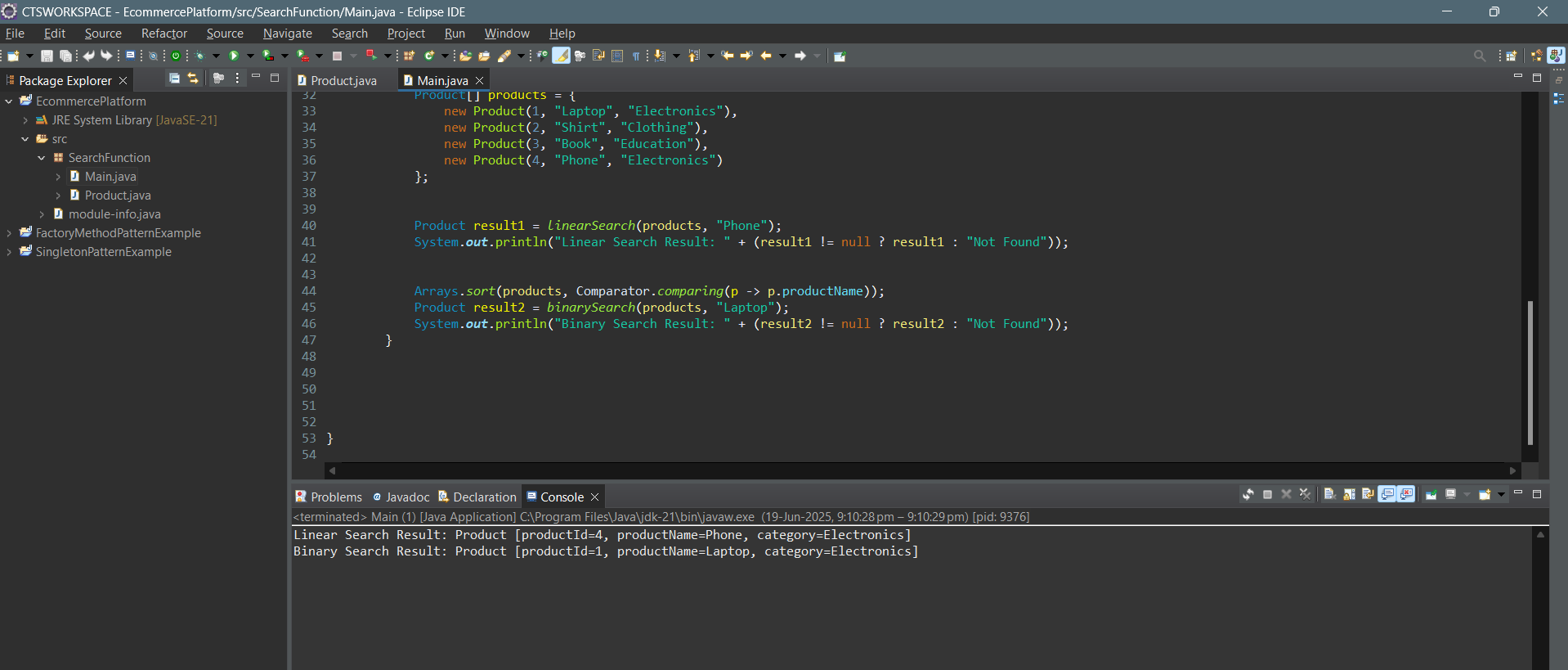
Arrays.sort(products, Comparator.comparing(p -> p.productName));

Product result2 = binarySearch(products, "Laptop");

System.out.println("Binary Search Result: " + (result2 != null ? result2 : "Not Found"));

}

}

**Output:**

**4. Analysis:**

**Compare the time complexity of linear and binary search algorithms.**

1. **Linear Search:**
   * Best Case: O(1) — when the element is found at the beginning.
   * Average Case: O(n)
   * Worst Case: O(n) — when the element is at the end or not present.
   * Requires Sorting: No
2. **Binary Search:**
   * Best Case: O(1) — when the element is found at the middle.
   * Average Case: O(log n)
   * Worst Case: O(log n)
   * Requires Sorting: Yes

**Discuss which algorithm is more suitable for your platform and why.**

Binary Search is more suitable for an e-commerce platform when the dataset (products) is sorted by name or ID. It significantly reduces search time for large data due to its logarithmic time complexity. This is ideal for platforms where fast, frequent searches are needed and the product list doesn't change often.

Linear Search can still be useful for small datasets or unsorted data but becomes inefficient as the data grows.