**WEEK1 – Data Structures and Alogorithms HandsOn**

**Execrcise2: Objective: To implement linear and binary search for product search optimization in an e-commerce system and analyze their performance.**

# STEPS :

### Understanding Asymptotic Notation

Big O Notation Overview:

- Big O (O): Worst-case runtime.
- Big Ω (Omega): Best-case runtime.
- Big Θ (Theta): Average-case runtime.

Why it matters in search:

| Algorithm | Best Case | Average Case | Worst Case |
|---|---|---|---|
| Linear Search | O(1) | O(n) | O(n) |
| Binary Search | O(1) | O(log n) | O(log n) |

**Product.java :**

Main.java | Product.java | LinearSearch.java | BinarySearch.java

```java
public 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;
    }
}
```

**Linear Search Implementation:**

**LinearSearch.java :**

Main.java × Product.java LinearSearch.java × BinarySearch.java

```java
public class LinearSearch {  no usages
    public static Product searchById(Product[] products, int targetId) {
        for (Product p : products) {
            if (p.productId == targetId) {
                return p;
            }
        }
        return null;
    }
}
```

**BinarySearch Implementation:**

**BinarySearch.java :**

```java
import java.util.Arrays;
import java.util.Comparator;

public class BinarySearch {  no usages
    public static Product searchById(Product[] products, int targetId) {  no usages
        // Ensure the array is sorted by productId
        Arrays.sort(products, Comparator.comparingInt( Product p -> p.productId));

        int low = 0, high = products.length - 1;
        while (low <= high) {
            int mid = (low + high) / 2;
            if (products[mid].productId == targetId) {
                return products[mid];
            } else if (products[mid].productId < targetId) {
                low = mid + 1;
            } else {
                high = mid - 1;
            }
        }
        return null;
    }
}
```

## Main.java :

Main.java × Product.java LinearSearch.java BinarySearch.java

```java
public class Main {
    public static void main(String[] args) {
        Product[] products = {
                new Product( productId: 101,  productName: "Laptop",  category: "Electronics"),
                new Product( productId: 202,  productName: "Shoes",  category: "Footwear"),
                new Product( productId: 150,  productName: "Book",  category: "Stationery")
        };

        // Linear Search
        Product found1 = LinearSearch.searchById(products,  targetId: 202);
        System.out.println("Linear Search Found: " + (found1 != null ? found1.productName : "Not Found"));

        // Binary Search
        Product found2 = BinarySearch.searchById(products,  targetId: 150);
        System.out.println("Binary Search Found: " + (found2 != null ? found2.productName : "Not Found"));
    }
}
```

## Output :

Run Main ×

```
"C:\Program Files\Java\jdk-23\bin\java.exe" "-javaagent:C:\Prog
Linear Search Found: Shoes
Binary Search Found: Book

Process finished with exit code 0
```