**E-commerce Platform Search Function**

1. **Big O Notation**

* Big O describes how the runtime or space requirements grow as the input size increases.
* Focuses on worst-case complexity (common in performance analysis).

| **Scenario** | **Linear Search** | **Binary Search** |
| --- | --- | --- |
| **Best Case** | O(1) – First element | O(1) – Middle element |
| **Average Case** | O(n/2) → O(n) | O(log n) |
| **Worst Case** | O(n) | O(log n) |

1. **Setup:**

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 productId + " - " + productName + " (" + category + ")";

}

}

1. **Implementation for Linear Search**

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

for (Product product : products) {

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

return product;

}

}

return null;

}

1. **Implementation for binary Search**

import java.util.Arrays;

import java.util.Comparator;

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

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

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

while (left <= right) {

int mid = (left + right) / 2;

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

if (cmp == 0) return products[mid];

else if (cmp < 0) right = mid - 1;

else left = mid + 1;

}

return null;

}

1. **Implementation of Main class**

public class SearchDemo {

public static void main(String[] args) {

Product[] products = {

new Product(101, "Laptop", "Electronics"),

new Product(102, "Shampoo", "Personal Care"),

new Product(103, "Chair", "Furniture"),

new Product(104, "Mobile", "Electronics"),

new Product(105, "Table", "Furniture")

};

String searchName = "Mobile";

Product result1 = linearSearch(products, searchName);

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

Product result2 = binarySearch(products, searchName);

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

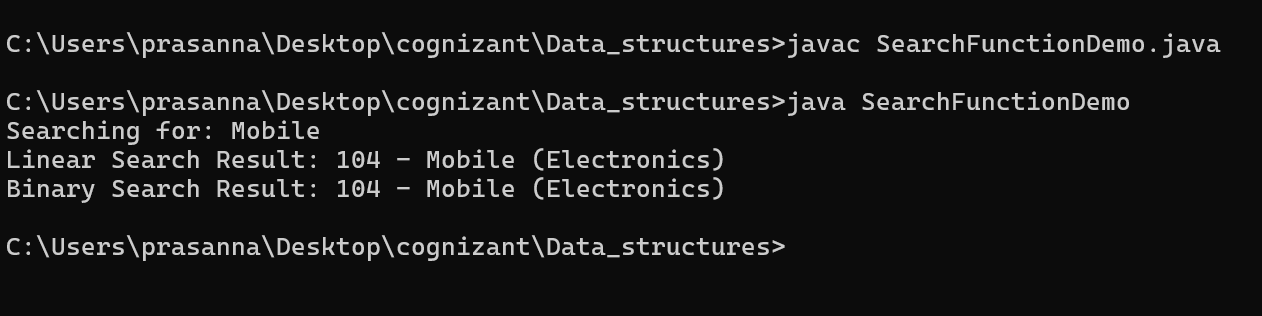
}

}

1. **Analysis:**

| **Algorithm** | **Time Complexity** | **Best For** |
| --- | --- | --- |
| **Linear Search** | O(n) | Small datasets or unsorted lists |
| **Binary Search** | O(log n) | Large, sorted datasets |

**Output**

****