**Week-1**

**Data Structure and Algorithm**

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

**Code:**

**Product.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;

}

public String toString() {

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

}

}

**SearchDemo.java**

import java.util.Arrays;

import java.util.Comparator;

public class SearchDemo {

// Linear Search: Go through each product one by one

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

for (Product product : products) {

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

return product;

}

}

return null;

}

// Binary Search: Works only on sorted arrays

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

int left = 0;

int right = products.length - 1;

while (left <= right) {

int mid = (left + right) / 2;

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

if (compare == 0) {

return products[mid];

} else if (compare < 0) {

left = mid + 1;

} else {

right = mid - 1;

}

}

return null;

}

public static void main(String[] args) {

Product[] products = {

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

new Product(102, "Shirt", "Clothing"),

new Product(103, "Mobile", "Electronics"),

new Product(104, "Sofa", "Furniture"),

new Product(105, "Shoes", "Footwear")

};

// Sort array for binary search by product name

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

// Searching using Linear Search

Product foundLinear = linearSearch(products, "Mobile");

if (foundLinear != null) {

System.out.println("Linear Search: Found - " + foundLinear);

} else {

System.out.println("Linear Search: Product not found.");

}

// Searching using Binary Search

Product foundBinary = binarySearch(products, "Mobile");

if (foundBinary != null) {

System.out.println("Binary Search: Found - " + foundBinary);

} else {

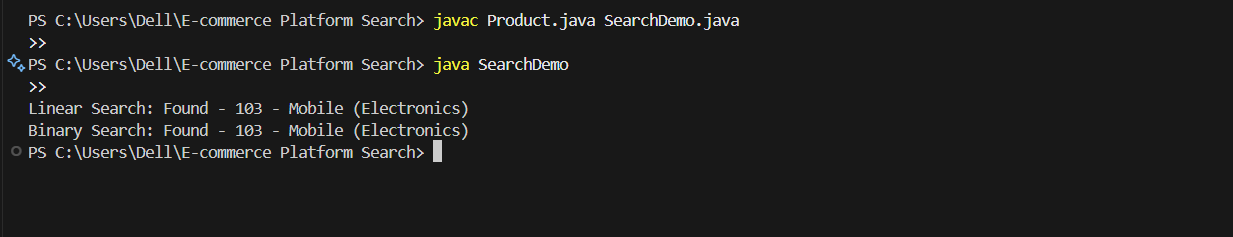
System.out.println("Binary Search: Product not found.");

}

}

}

**Output:**



**Exercise 7: Financial Forecasting**

**Code:**

**FinancialForecast.java**

public class FinancialForecast {

public static double predictFutureValue(double currentValue, double growthRate, int years) {

// Base case: no more years to grow

if (years == 0) {

return currentValue;

}

// Recursive case: apply growth for one year and call again

return predictFutureValue(currentValue \* (1 + growthRate), growthRate, years - 1);

}

public static void main(String[] args) {

double initialValue = 10000;

double annualGrowthRate = 0.07;

int years = 5;

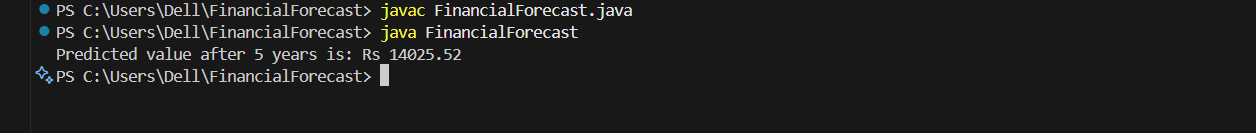
double futureValue = predictFutureValue(initialValue, annualGrowthRate, years);

System.out.printf("Predicted value after %d years is: Rs %.2f\n", years, futureValue);

}

}

**Output:**

****