**3.EcommercePlatformSearchFunction**

**Product.java**

package com.ecommerce;

public class Product {

    private int productId;

    private String productName;

    private String category;

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

        this.productId = productId;

        this.productName = productName;

        this.category = category;

    }

    public int getProductId() {

        return productId;

    }

    public String getProductName() {

        return productName;

    }

    public String getCategory() {

        return category;

    }

    @Override

    public String toString() {

        return "Product{" +

               "productId=" + productId +

               ", productName='" + productName + '\'' +

               ", category='" + category + '\'' +

               '}';

    }

}

**LinearSearch.java**

package com.ecommerce;

public class LinearSearch {

    public static Product search(Product[] products, int targetProductId) {

        for (Product product : products) {

            if (product.getProductId() == targetProductId) {

                return product;

            }

        }

        return null;

    }

}

**BinarySearch.java**

package com.ecommerce;

public class BinarySearch {

    public static Product search(Product[] products, int targetProductId) {

        int low = 0;

        int high = products.length - 1;

        while (low <= high) {

            int mid = low + (high - low) / 2;

            if (products[mid].getProductId() == targetProductId) {

                return products[mid];

            }

            if (targetProductId < products[mid].getProductId()) {

                high = mid - 1;

            } else {

                low = mid + 1;

            }

        }

        return null;

    }

}

**Test.java**

package com.ecommerce;

import java.util.Arrays;

import java.util.Comparator;

public class Test {

    public static void main(String[] args) {

        Product[] products = {

            new Product(102, "Smartphone", "Electronics"),

            new Product(201, "T-Shirt", "Apparel"),

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

            new Product(301, "Book", "Books"),

            new Product(202, "Jeans", "Apparel")

        };

        System.out.println("--- Testing Linear Search ---");

        int targetIdLinear = 101;

        Product foundProductLinear = LinearSearch.search(products, targetIdLinear);

        System.out.println("Searching for Product ID: " + targetIdLinear);

        if (foundProductLinear != null) {

            System.out.println("Found: " + foundProductLinear);

        } else {

            System.out.println("Product not found.");

      }

        System.out.println("--- Testing Binary Search ---");

        System.out.println("First, sorting the array by Product ID for Binary Search...");

        Arrays.sort(products, Comparator.comparingInt(Product::getProductId));

        System.out.println("Sorted Products:");

        for(Product p : products) {

            System.out.println("  " + p);

        }

        System.out.println();

        int targetIdBinary = 101;

        Product foundProductBinary = BinarySearch.search(products, targetIdBinary);

        System.out.println("Searching for Product ID: " + targetIdBinary);

        if (foundProductBinary != null) {

            System.out.println("Found: " + foundProductBinary);

        } else {

            System.out.println("Product not found.");

        }

        System.out.println();

        int targetIdNotFound = 999;

        Product notFoundProduct = BinarySearch.search(products, targetIdNotFound);

        System.out.println("Searching for Product ID: " + targetIdNotFound);

        if (notFoundProduct != null) {

            System.out.println("Found: " + notFoundProduct);

        } else {

            System.out.println("Product with ID " + targetIdNotFound + " was not found, as expected.");

        }

    }

}

Output:

A screenshot of a computer program

AI-generated content may be incorrect.

**4.FinancialForecast**

FinancialForecast.java

package com.forecast;

public class FinancialForecast {

    public static double calculateFutureValue(double initialInvestment, double growthRate, int years) {

        if (years == 0) {

            return initialInvestment;

        }

        return calculateFutureValue(initialInvestment, growthRate, years - 1) \* (1 + growthRate);

    }

    public static void main(String[] args) {

        double initialInvestment = 10000.0;

        double annualGrowthRate = 0.08;

        int numberOfYears = 5;

        double futureValue = calculateFutureValue(initialInvestment, annualGrowthRate, numberOfYears);

        System.out.printf("Future Value after %d years: ₹%.2f%n", numberOfYears, futureValue);

    }

}

**Output:**

A screenshot of a computer program

AI-generated content may be incorrect.