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

**Question**: You are working on the search functionality of an e-commerce platform. The search needs to be optimized for fast performance.

**Code:**

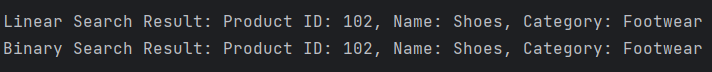
**Product.java**

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;  
 }  
  
 @Override  
 public String toString() {  
 return "Product ID: " + productId + ", Name: " + productName + ", Category: " + category;  
 }  
}

**Main.java**

import java.util.\*;  
  
public class Main {  
 public static Product linearSearch(Product[] products, int id) {  
 for (Product product : products) {  
 if (product.productId == id) {  
 return product;  
 }  
 }  
 return null;  
 }  
  
 public static Product binarySearch(Product[] products, int id) {  
 int l = 0, h = products.length - 1;  
 while (l <= h) {  
 int m = (l + h) / 2;  
  
 if (products[m].productId == id) return products[m];  
  
 else if (products[m].productId < id) l = m + 1;  
  
 else h = m - 1;  
  
 }  
 return null;  
 }  
  
 public static void main(String[] args) {  
 Product[] productList = {  
 new Product(101, "Laptop", "Electronics"),  
 new Product(102, "Shoes", "Footwear"),  
 new Product(103, "Watch", "Accessories"),  
 new Product(104, "Phone", "Electronics"),  
 new Product(105, "Bag", "Fashion")  
 };  
  
 Arrays.*sort*(productList, Comparator.*comparingInt*(p -> p.productId));  
  
 int searchId = 102;  
  
 Product result1 = *linearSearch*(productList, searchId);  
 System.*out*.println("Linear Search Result: " + (result1 != null ? result1 : "Not Found"));  
  
 Product result2 = *binarySearch*(productList, searchId);  
 System.*out*.println("Binary Search Result: " + (result2 != null ? result2 : "Not Found"));  
 }  
}

**Output:**



**Exercise 7: Financial Forecasting**

**Question:** You are developing a financial forecasting tool that predicts future values based on past data.

**Code:**

import java.util.\*;  
  
public class Main {  
 public static double forecastValue(int years, double currentValue, double growthRate) {  
 if (years == 0) {  
 return currentValue;  
 }  
 return *forecastValue*(years - 1, currentValue, growthRate) \* (1 + growthRate);  
 }  
  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
 System.*out*.println("Enter Initial Value: ");  
 double initialValue = sc.nextDouble();  
 System.*out*.println("Enter Growth Rate: ");  
 double growthRate = sc.nextDouble();  
 System.*out*.println("Enter Forecast Years: ");  
 int forecastYears = sc.nextInt();  
  
 double futureValue = *forecastValue*(forecastYears, initialValue, growthRate);  
 System.*out*.printf("Predicted value after %d years: ₹%.2f\n", forecastYears, futureValue);  
 }  
}

**Output:**

