**Digital Nurture 4.0 Java FSE**

**Week 1 Data Structure and Algorithm**

**(Mandatory)**

**Exercise 02: E-commerce Platform Search Function**

**Code:**

import java.util.\*;  
  
class SearchProduct {  
 int productId;  
 String productName;  
 String category;  
  
 public SearchProduct(int id, String name, String category) {  
 this.productId = id;  
 this.productName = name;  
 this.category = category;  
 }  
  
 @Override  
 public String toString() {  
 return productId + " | " + productName + " | " + category;  
 }  
}  
  
public class Ecommerce {  
  
 static SearchProduct linearSearch(SearchProduct[] products, String name) {  
 for (SearchProduct p : products) {  
 if (p.productName.equalsIgnoreCase(name)) return p;  
 }  
 return null;  
 }  
  
 static SearchProduct binarySearch(SearchProduct[] products, String name) {  
 int low = 0, high = products.length - 1;  
 while (low <= high) {  
 int mid = (low + high) / 2;  
 int cmp = products[mid].productName.compareToIgnoreCase(name);  
 if (cmp == 0) return products[mid];  
 else if (cmp < 0) low = mid + 1;  
 else high = mid - 1;  
 }  
 return null;  
 }  
  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
  
 SearchProduct[] items = {  
 new SearchProduct(11, "Notebook", "Stationery"),  
 new SearchProduct(7, "Earbuds", "Electronics"),  
 new SearchProduct(19, "Charger", "Accessories"),  
 new SearchProduct(3, "Desk Lamp", "Home"),  
 new SearchProduct(25, "Backpack", "Travel")  
 };  
  
 Arrays.*sort*(items, Comparator.*comparing*(p -> p.productName.toLowerCase()));  
  
 System.*out*.print("Enter product name to search: ");  
 String query = sc.nextLine();  
  
 System.*out*.println("\nLinear Search Result:");  
 SearchProduct result1 = *linearSearch*(items, query);  
 System.*out*.println(result1 != null ? result1 : "Product not found.");  
  
 System.*out*.println("\nBinary Search Result:");  
 SearchProduct result2 = *binarySearch*(items, query);  
 System.*out*.println(result2 != null ? result2 : "Product not found.");  
  
 sc.close();  
 }  
}

**Output:**

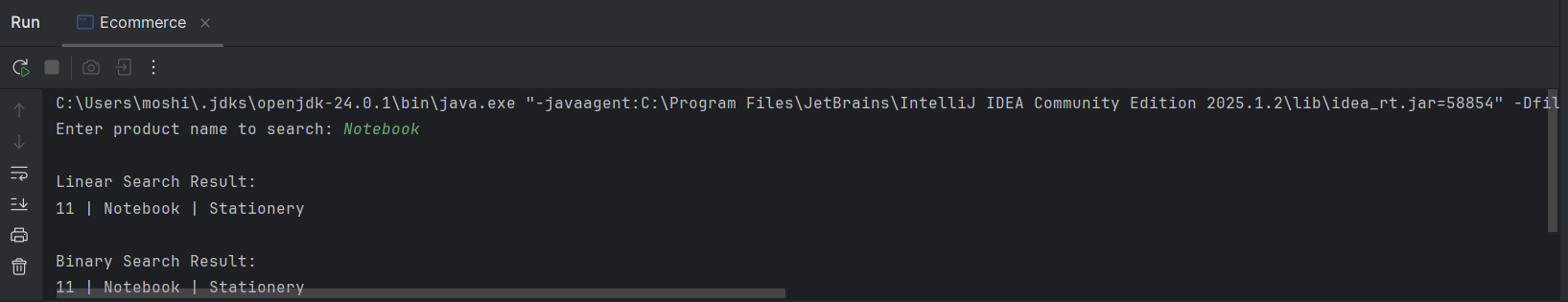
Enter product name to search: Notebook

Linear Search Result:

11 | Notebook | Stationery

Binary Search Result:

11 | Notebook | Stationery



**Exercise 07: Financial Forecasting**

**Code:**

import java.util.Scanner;

public class FinancialForecasting {

static double forecastRecursive(double value, double growth, int years) {

if (years == 0) return value;

return forecastRecursive(value \* (1 + growth), growth, years - 1);

}

static double forecastIterative(double value, double growth, int years) {

for (int i = 0; i < years; i++) {

value \*= (1 + growth);

}

return value;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter starting value (₹): ");

double start = sc.nextDouble();

System.out.print("Enter annual growth rate (in %): ");

double rate = sc.nextDouble() / 100;

System.out.print("Enter number of years: ");

int years = sc.nextInt();

double resultRecursive = forecastRecursive(start, rate, years);

double resultIterative = forecastIterative(start, rate, years);

System.out.printf("\nPredicted (Recursive): ₹%.2f\n", resultRecursive);

System.out.printf("Predicted (Iterative): ₹%.2f\n", resultIterative);

sc.close();

}

}

**Output:**

Enter starting value (₹): 20000

Enter annual growth rate (in %): 5

Enter number of years: 6

Predicted (Recursive): ₹26801.91

Predicted (Iterative): ₹26801.91

