**Data Structures and Algorithms**

**E-commerce platform search function**

import java.util.\*;

import java.util.stream.Collectors;

 class Product {

    private String name;

    private String category;

    private double price;

    public Product(String name, String category, double price)

    {

        this.name = name;

        this.category = category;

        this.price = price;

    }

    public String getName(){

         return name;

        }

    public String getCategory()

    {

         return category;

     }

    public double getPrice() {

         return price;

         }

    @Override

    public String toString() {

        return name + " | " + category + " | Rs:" + price;

    }

}

public class EcommerceSearch {

    private List<Product> productList = new ArrayList<>();

    public EcommerceSearch() {

        productList.add(new Product("White Shoes", "shoes", 2999));

        productList.add(new Product("Casual Sneakers", "Footwear", 1999));

        productList.add(new Product("Red T-Shirt", "Clothing", 999));

    }

    public List<Product> search(String keyword, String category, Double maxPrice) {

        return productList.stream()

                .filter(p -> p.getName().toLowerCase().contains(keyword.toLowerCase()))

                .filter(p -> category == null || p.getCategory().equalsIgnoreCase(category))

                .filter(p -> maxPrice == null || p.getPrice() <= maxPrice)

                .collect(Collectors.toList());

    }

    public static void main(String[] args) {

        EcommerceSearch searchEngine = new EcommerceSearch();

        System.out.println("Search Results for blue shoes under Rs.2000 ");

        List<Product> results = searchEngine.search("black", "Clothing", 2000.0);

        for (Product p : results) {

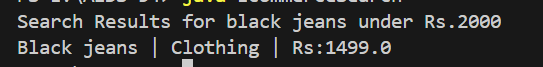
            System.out.println(p);

        }

    }

}

**Output**



**Financial Forecasting**

public class FinancialForecasting {

    public static void main(String[] args) {

        double[] months = {1, 2, 3, 4, 5}; // Time (e.g., Jan to May)

        double[] revenue = {1000, 1200, 1300, 1500, 1700}; // Revenue in $

        int n = months.length;

        double sum1 = 0, sum2 = 0, sum3 = 0, sum4 = 0;

        for (int i = 0; i < n; i++) {

            sum1 += months[i];

            sum2 += revenue[i];

            sum3 += months[i] \* revenue[i];

            sum4 += months[i] \* months[i];

        }

        double m = (n \* sum3 – sum1 \* sum2) / (n \* sum4 – sum1 \* sum1);

        double c = (sum2 - m \* sum1) / n;

        System.out.printf("Forecast Equation: Revenue = %.2f \* Month + %.2f\n", m, c);

        for (int month = 6; month <= 8; month++) {

            double forecast = m \* month + c;

            System.out.printf("Forecast for month %d: $%.2f\n", month, forecast);

        }

    }

}

**Output**

