**CustomerOrderSorter**

**Main.java**

public class Main {

    public static void main(String[] args) {

        Order[] orders = {

                new Order("O1", "Alice", 1200.0),

                new Order("O2", "Bob", 450.5),

                new Order("O3", "Charlie", 1999.99),

                new Order("O4", "Diana", 700.0)

        };

        System.out.println("Original Orders:");

        for (Order o : orders)

            System.out.println(o);

        Sorter.bubbleSort(orders);

        System.out.println("\nBubble Sorted Orders:");

        for (Order o : orders)

            System.out.println(o);

        Order[] orders2 = {

                new Order("O1", "Alice", 1200.0),

                new Order("O2", "Bob", 450.5),

                new Order("O3", "Charlie", 1999.99),

                new Order("O4", "Diana", 700.0)

        };

        Sorter.quickSort(orders2, 0, orders2.length - 1);

        System.out.println("\nQuick Sorted Orders:");

        for (Order o : orders2)

            System.out.println(o);

    }

}

**Order.java**

public class Order {

    private String orderId;

    private String customerName;

    private double totalPrice;

    public Order(String orderId, String customerName, double totalPrice) {

        this.orderId = orderId;

        this.customerName = customerName;

        this.totalPrice = totalPrice;

    }

    public String getOrderId() {

        return orderId;

    }

    public String getCustomerName() {

        return customerName;

    }

    public double getTotalPrice() {

        return totalPrice;

    }

    @Override

    public String toString() {

        return "Order{" +

                "ID='" + orderId + '\'' +

                ", Customer='" + customerName + '\'' +

                ", Total=" + totalPrice +

                '}';

    }

}

**Sorter.java**

public class Sorter {

    public static void bubbleSort(Order[] orders) {

        int n = orders.length;

        for (int i = 0; i < n - 1; i++) {

            for (int j = 0; j < n - i - 1; j++) {

                if (orders[j].getTotalPrice() > orders[j + 1].getTotalPrice()) {

                    Order temp = orders[j];

                    orders[j] = orders[j + 1];

                    orders[j + 1] = temp;

                }

            }

        }

    }

    public static void quickSort(Order[] orders, int low, int high) {

        if (low < high) {

            int pi = partition(orders, low, high);

            quickSort(orders, low, pi - 1);

            quickSort(orders, pi + 1, high);

        }

    }

    private static int partition(Order[] orders, int low, int high) {

        double pivot = orders[high].getTotalPrice();

        int i = low - 1;

        for (int j = low; j < high; j++) {

            if (orders[j].getTotalPrice() < pivot) {

                i++;

                Order temp = orders[i];

                orders[i] = orders[j];

                orders[j] = temp;

            }

        }

        Order temp = orders[i + 1];

        orders[i + 1] = orders[high];

        orders[high] = temp;

        return i + 1;

    }

}

