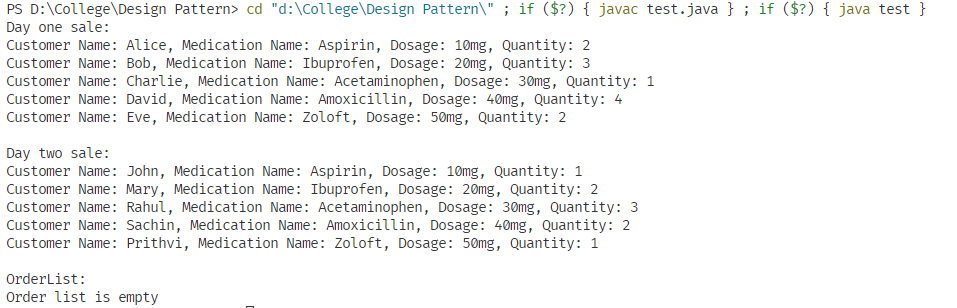
**AIM: Apply Prototype Design Pattern on Pharmacy (Medical Store) Problem**

Code

|  |
| --- |
| /\*   \* Author: Saurav Navdhare   \* Roll Number: 21BCP157   \* Purpose: To implement deep copy in prototype design pattern   \*   \* Explanation: There's a Medical Store which sells medicine,   \* and they want to keep a record of their sales. So, they have a OrderList class   \* which stores the orders. Now, they want to keep a record of their sales for   \* each day. So, they want to clone the OrderList object and store it in a   \* separate object. So, they can keep a record of their sales for each day.   \* Then they reset the main billing object and start billing for the next day.   \*   \* For example purposes, I have kept limit of OrderList as 10 orders.   \*/  class Order { // Order class;      private String customerName;      private String medicationName;      private String dosage;      private int quantity;      public Order(String customerName, String medicationName, String dosage, int quantity) { // Constructor;          this.customerName = customerName;          this.medicationName = medicationName;          this.dosage = dosage;          this.quantity = quantity;      }      public Order(Order order) { // Copy constructor;          this.customerName = order.customerName;          this.medicationName = order.medicationName;          this.dosage = order.dosage;          this.quantity = order.quantity;      }      public String getCustomerName() { // Getters and setters          return this.customerName;      }      public void setCustomerName(String value) {          this.customerName = value;      }      public String getMedicationName() {          return this.medicationName;      }      public void setMedicationName(String value) {          this.medicationName = value;      }      public String getDosage() {          return this.dosage;      }      public void setDosage(String value) {          this.dosage = value;      }      public int getQuantity() {          return this.quantity;      }      public void setQuantity(int value) {          this.quantity = value;      }      public String toString() {          return "Customer Name: " + customerName + ", Medication Name: " + medicationName + ", Dosage: " + dosage                  + ", Quantity: " + quantity;      }  }  class OrderList implements Cloneable { // OrderList class implementing Cloneable interface      private int orderCount;      private Order[] orders; // Array of Order objects;      public OrderList() { // Constructor;          this.orderCount = 0;          this.orders = new Order[10];      }      public void addOrder(Order order) { // Add order to the list          if (orderCount < 1000) {              orders[orderCount] = order;              orderCount++;          } else {              System.out.println("Order list is full");          }      }      public void addOrder(String customerName, String medicationName, String dosage, int quantity) { // Overloaded method          if (orderCount < 1000) {              orders[orderCount] = new Order(customerName, medicationName, dosage, quantity);              orderCount++;          } else {              System.out.println("Order list is full");          }      }      public Order getOrder(int index) { // Get order from the list          if (index < orderCount) {              return orders[index];          } else {              return null;          }      }      public int getOrderCount() { // Get order count          return this.orderCount;      }      public OrderList clone() { // Applying deepCopy;          OrderList clone = new OrderList();          for (int i = 0; i < orderCount; i++) {              clone.addOrder(new Order(orders[i]));          }          return clone;      }      public void resetOrderList() { // Reset order list          orderCount = 0;          orders = new Order[10];      }      public String toString() { // To string method          if (orderCount == 0)              return "Order list is empty";          String result = "";          for (int i = 0; i < orderCount; i++) {              result += orders[i].toString() + "\n";          }          return result;      }  }  public class test {      public static void main(String[] args) { // To test the code;          OrderList orderList = new OrderList();          Order order1 = new Order("Alice", "Aspirin", "10mg", 2);          orderList.addOrder(order1);          Order order2 = new Order("Bob", "Ibuprofen", "20mg", 3);          orderList.addOrder(order2);          Order order3 = new Order("Charlie", "Acetaminophen", "30mg", 1);          orderList.addOrder(order3);          Order order4 = new Order("David", "Amoxicillin", "40mg", 4);          orderList.addOrder(order4);          Order order5 = new Order("Eve", "Zoloft", "50mg", 2);          orderList.addOrder(order5);          // Day one sale over;          OrderList dayOne = orderList.clone();          orderList.resetOrderList();          Order order6 = new Order("John", "Aspirin", "10mg", 1);          orderList.addOrder(order6);          Order order7 = new Order("Mary", "Ibuprofen", "20mg", 2);          orderList.addOrder(order7);          Order order8 = new Order("Rahul", "Acetaminophen", "30mg", 3);          orderList.addOrder(order8);          Order order9 = new Order("Sachin", "Amoxicillin", "40mg", 2);          orderList.addOrder(order9);          Order order10 = new Order("Prithvi", "Zoloft", "50mg", 1);          orderList.addOrder(order10);          // Day two sale over;          OrderList dayTwo = orderList.clone();          orderList.resetOrderList();          System.out.println("Day one sale:\n" + dayOne);          System.out.println("Day two sale:\n" + dayTwo);          System.out.println("OrderList:\n" + orderList);      }  } |

**Output on next page**

**Output:**



**1**: Output of the above code