

Name: ziad medhat ahmed mahmoud ID:22010104

e-commerce project

1)class Main

```
Start Page X Main.java X
Source History
1 package main.java;
2
3 import java.util.Scanner;
4 import javax.swing.JOptionPane;
5
6 public class Main {
7     public static void main(String[] args) {
8         Scanner in = new Scanner(System.in);
9         ElectronicProduct E = new ElectronicProduct("samsung", 1, 1, "smart phone", 599.9f);
10        ClothingProduct C = new ClothingProduct("medium", "cotton", 2, "T-Shirt", 19.99f);
11        BookProduct B = new BookProduct("o'reilly", "X publications", 3, "OOP", 39.99f);
12
13        // Taking input for customer information
14        int id = Integer.parseInt(JOptionPane.showInputDialog("Enter your ID"));
15        String name = JOptionPane.showInputDialog("Enter your name");
16        JOptionPane.showMessageDialog(null, "Hello " + name);
17        String add = JOptionPane.showInputDialog("Enter your Address");
18
19        // Creating a Customer object
20        Customer Y = new Customer(id, name, add);
21
22        // Taking input for number of products to add to cart
23        int nproduct = Integer.parseInt(JOptionPane.showInputDialog("How many products you want to add to your cart?"));
24        Cart X = new Cart();
25        X.setCustomerid(Y.getCustomerid());
26        X.setNproduct(nproduct);
27
28        // Loop to add products to the cart
29        for (int i = 0; i < nproduct; i++) {
30            int Array = Integer.parseInt(JOptionPane.showInputDialog("Which product would you like to add?\n1.smartphone\n2.T-Shirt\n3.oop "));
31            switch (Array) {
32                case 1:
33                    X.addProduct(E);
34                    break;
35                case 2:
36                    X.addProduct(C);
37                    break;
38                case 3:
39                    X.addProduct(B);
40                    break;
41                default:
42                    JOptionPane.showMessageDialog(null, "invalid choice!");
43            }
44        }
45
46        // Displaying total price of products in the cart
47        JOptionPane.showMessageDialog(null, "your total is: " + X.calculatePrice() + "$.");
48
49        // Asking user if they want to place the order
50        int n = Integer.parseInt(JOptionPane.showInputDialog(" Do you want to place the order?1.yes 2.No"));
51        X.placeOrder(n);
52
53        // Displaying order information or cart status based on user input
54        if (n == 1) {
55            Order O = new Order(X);
56            O.printOrderInfo();
57        } else if (n == 2) {
58            JOptionPane.showMessageDialog(null, "cart is now empty");
59        } else {
60            JOptionPane.showMessageDialog(null, "invalid input");
61        }
62    }
63 }
```

2)class Order

```
source History | 
64
65 class Order {
66     private int customerid;
67     private int orderid;
68     private Product[] arr;
69     private float totalprice;
70     private int nproduct;
71     static int number;
72
73     // Constructor for Order class
74     public Order(Cart c) {
75         number++;
76         this.customerid = c.getCustomerid();
77         this.orderid = number;
78         this.arr = c.getArr();
79         this.totalprice = c.calculatePrice();
80     }
81
82     // Method to print order information
83     public void printOrderInfo() {
84         System.out.println("Here's your order's summary: ");
85         System.out.println("Order id: " + orderid);
86         System.out.println("Customer id: " + customerid);
87         System.out.println("Ordered products: ");
88         for (int i = 0; i < arr.length; i++) {
89             if (arr[i] != null) {
90                 System.out.println("Name: " + arr[i].getName());
91                 System.out.println("Price: " + arr[i].getPrice());
92             }
93         }
94         System.out.println("Total price: " + totalprice);
95     }
96 }
```

3)class Cart

```
98 class Cart {
99     private int customerid;
100     private int nproduct;
101     protected Product[] arr;
102
103     // Getter for the array of products in the cart
104     public Product[] getArr() {
105         return arr;
106     }
107
108     // Setter for the array of products in the cart
109     public void setArr(Product[] arr) {
110         this.arr = new Product[nproduct];
111     }
112
113     // Getter for customer ID
114     public int getCustomerid() {
115         return customerid;
116     }
117
118     // Setter for customer ID
119     public void setCustomerid(int customerid) {
120         this.customerid = customerid;
121     }
122
123     // Getter for number of products in the cart
124     public int getNproduct() {
125         return nproduct;
126     }
127
128     // Setter for number of products in the cart
129     public void setNproduct(int nproduct) {
130         this.nproduct = Math.abs(nproduct);
131         arr = new Product[this.nproduct];
132     }
133
134     // Method to add a product to the cart
135     public void addProduct(Product product) {
136         for (int i = 0; i < nproduct; i++) {
137             if (arr[i] == null) {
138                 arr[i] = product;
139                 return;
140             }
141         }
142         JOptionPane.showMessageDialog(null, "The cart is full, cannot add more products.");
143     }
144
145     // Method to remove a product from the cart
146     public void removeProduct(int index) {
147         if (index >= 0 && index < nproduct) {
148             arr[index] = null;
149         } else {
150             JOptionPane.showMessageDialog(null, "Invalid index, cannot remove product.");
151         }
152     }
153
154     // Method to calculate the total price of products in the cart
155     public float calculatePrice() {
156         float total = 0;
157         for (int i = 0; i < nproduct; i++) {
158             if (arr[i] != null) {
159                 total += arr[i].getPrice();
160             }
161         }
162         return total;
163     }
164 }
```

```

161         }
162         return total;
163     }
164
165     // Method to place an order based on user input
166     public void placeOrder(int choice) {
167         switch (choice) {
168             case 1:
169                 System.out.println("Order placed successfully.");
170                 break;
171             case 2:
172                 for (int i = 0; i < arr.length; i++) {
173                     arr[i] = null;
174                 }
175                 System.out.println("Cart is now empty.");
176                 break;
177             default:
178                 System.out.println("Invalid input.");
179         }
180     }
181 }

```

4) class Product

```

182
183 class Product {
184     protected int productid;
185     protected String name;
186     protected float price;
187
188     // Getters and setters for product attributes
189     public int getProductid() {
190         return productid;
191     }
192
193     public void setProductid(int productid) {
194         this.productid = Math.abs(productid);
195     }
196
197     public String getName() {
198         return name;
199     }
200
201     public void setName(String name) {
202         this.name = name;
203     }
204
205     public float getPrice() {
206         return price;
207     }
208
209     public void setPrice(float price) {
210         this.price = Math.abs(price);
211     }
212 }

```

5) class ElectronicProduct

```
213
214 class ElectronicProduct extends Product {
215     protected String brand;
216     protected int warrantyperiod;
217
218     // Constructor for ElectronicProduct class
219     public ElectronicProduct(String brand, int productId, int warrantyPeriod, String name, float price) {
220         this.brand = brand;
221         this.setProductId(productId);
222         this.setWarrantyperiod(warrantyPeriod);
223         this.setName(name);
224         this.setPrice(price);
225     }
226
227     // Getters and setters for electronic product attributes
228     public String getBrand() {
229         return brand;
230     }
231
232     public void setBrand(String brand) {
233         this.brand = brand;
234     }
235
236     public int getWarrantyperiod() {
237         return warrantyperiod;
238     }
239
240     public void setWarrantyperiod(int warrantyperiod) {
241         this.warrantyperiod = Math.abs(warrantyperiod);
242     }
243 }
244
```

6) class ClothingProduct

```
244
245 class ClothingProduct extends Product {
246     protected String size;
247     protected String fabric;
248
249     // Constructor for ClothingProduct class
250     public ClothingProduct(String size, String fabric, int productId, String name, float price) {
251         this.size = size;
252         this.fabric = fabric;
253         this.setProductId(productId);
254         this.setName(name);
255         this.setPrice(price);
256     }
257
258     // Getters and setters for clothing product attributes
259     public String getSize() {
260         return size;
261     }
262
263     public void setSize(String size) {
264         this.size = size;
265     }
266
267     public String getFabric() {
268         return fabric;
269     }
270
271     public void setFabric(String fabric) {
272         this.fabric = fabric;
273     }
274 }
```

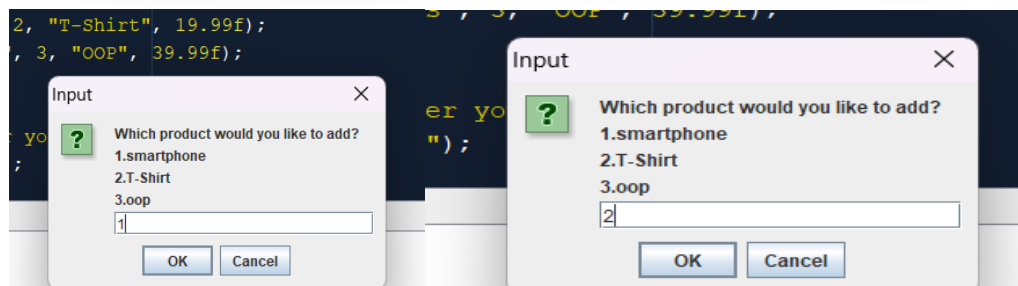
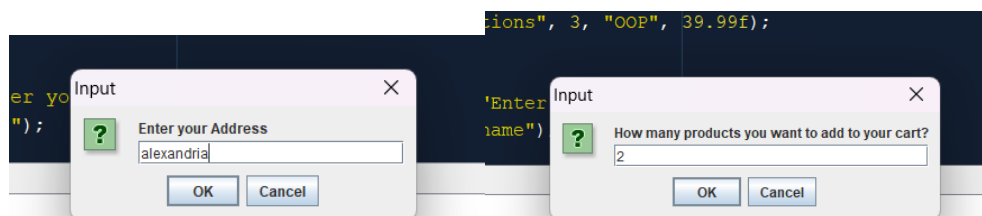
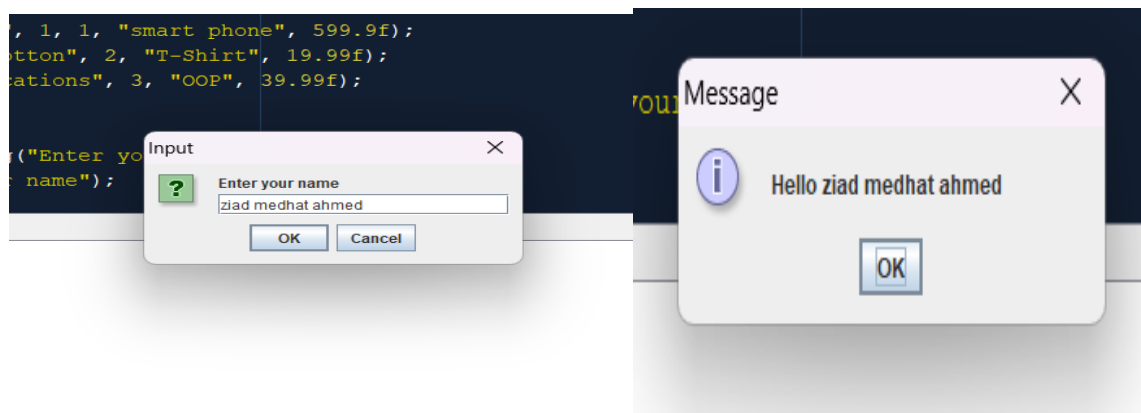
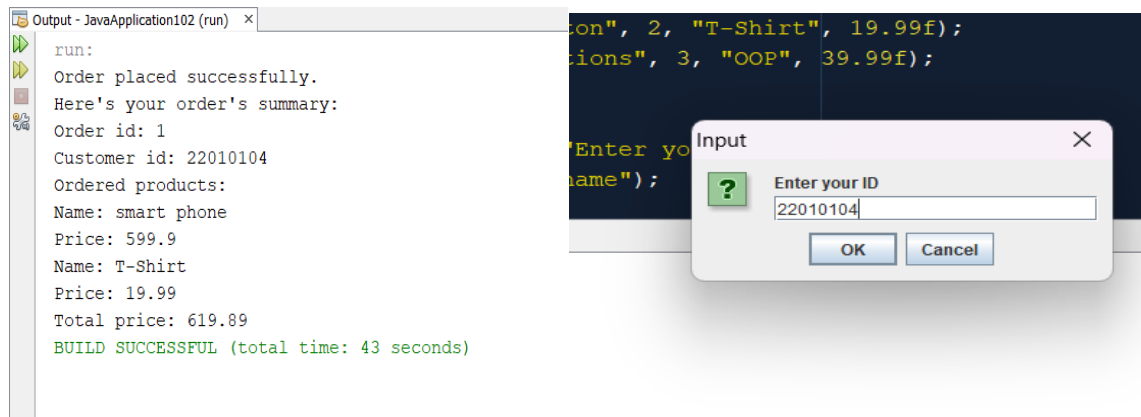
7) class BookProduct

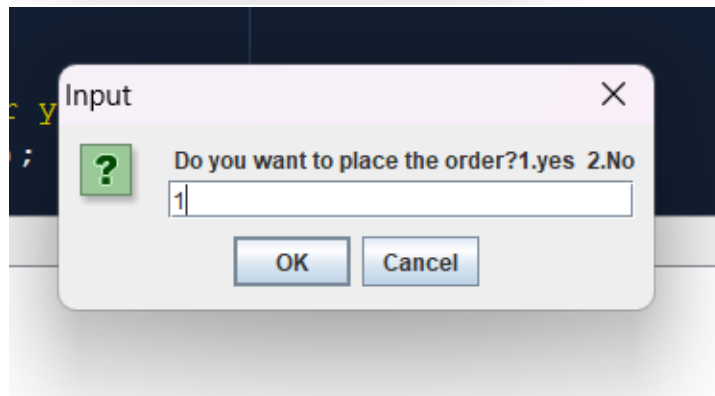
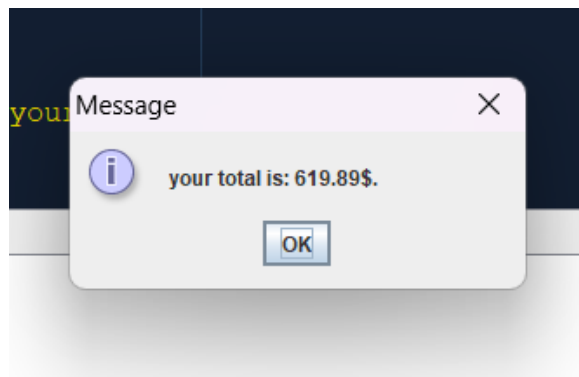
```
Source History
275
276 class BookProduct extends Product {
277     protected String author;
278     protected String publisher;
279
280     // Constructor for BookProduct class
281     public BookProduct(String author, String publisher, int productId, String name, float price) {
282         this.author = author;
283         this.publisher = publisher;
284         this.setProductId(productId);
285         this.setName(name);
286         this.setPrice(price);
287     }
288
289     // Getters and setters for book product attributes
290     public String getAuthor() {
291         return author;
292     }
293
294     public void setAuthor(String author) {
295         this.author = author;
296     }
297
298     public String getPublisher() {
299         return publisher;
300     }
301
302     public void setPublisher(String publisher) {
303         this.publisher = publisher;
304     }
305 }
```

8) class Customer

```
Source History
307 class Customer {
308     private int customerId;
309     private String name;
310     private String address;
311
312     // Constructor for Customer class
313     public Customer(int id, String name, String add) {
314         this.setCustomerId(id);
315         this.setName(name);
316         this.setAddress(add);
317     }
318
319     // Getters and setters for customer attributes
320     public int getCustomerId() {
321         return customerId;
322     }
323
324     public void setCustomerId(int customerId) {
325         this.customerId = Math.abs(customerId);
326     }
327
328     public String getName() {
329         return name;
330     }
331
332     public void setName(String name) {
333         this.name = name;
334     }
335
336     public String getAddress() {
337         return address;
338     }
339
340     public void setAddress(String address) {
341         this.address = address;
342     }
343 }
```

Output





```
Output - JavaApplication102 (run) x
run:
Order placed successfully.
Here's your order's summary:
Order id: 1
Customer id: 22010104
Ordered products:
Name: smart phone
Price: 599.9
Name: T-Shirt
Price: 19.99
Total price: 619.89
BUILD SUCCESSFUL (total time: 1 minute 38 seconds)
|
```