Name: ziad medhat ahmed mahmoud ID:22010104

e-commerce project

1)class Main

```java
package main.java;

import java.util.Scanner;
import javax.swing.JOptionPane;

public class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        ElectronicProduct E = new ElectronicProduct("samsung", 1, 1, "smart phone", 599.9f);
        ClothingProduct C = new ClothingProduct("medium", "cotton", 2, "T-Shirt", 19.99f);
        BookProduct B = new BookProduct("o'reilly", "X publications", 3, "OOP", 39.99f);

        // Taking input for customer information
        int id = Integer.parseInt(JOptionPane.showInputDialog("Enter your ID"));
        String name = JOptionPane.showInputDialog("Enter your name");
        JOptionPane.showMessageDialog(null, "Hello " + name);
        String add = JOptionPane.showInputDialog("Enter your Address");

        // Creating a Customer object
        Customer Y = new Customer(id, name, add);

        // Taking input for number of products to add to cart
        int nproduct = Integer.parseInt(JOptionPane.showInputDialog("How many products you want to add to your cart?"));
        Cart X = new Cart();
        X.setCustomerid(Y.getCustomerid());
        X.setNproduct(nproduct);

        // Loop to add products to the cart
        for (int i = 0; i < nproduct; i++) {
            int Array = Integer.parseInt(JOptionPane.showInputDialog("Which product would you like to add?\n1.smartphone\n2.T-Shirt\n3.oop "));
            switch (Array) {
                case 1:
                    X.addProduct(E);
                    break;
                case 2:
```

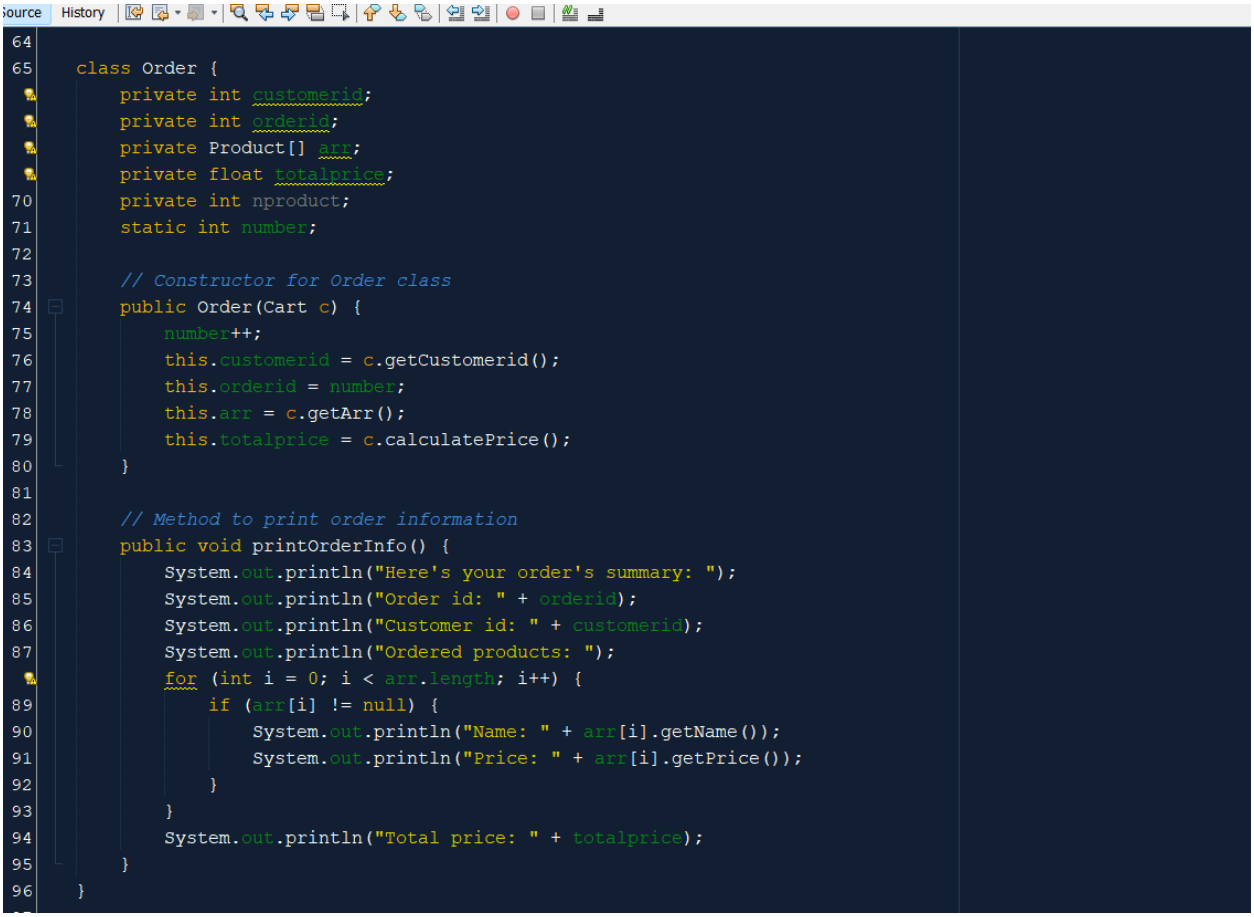
```java
        // Loop to add products to the cart
        for (int i = 0; i < nproduct; i++) {
            int Array = Integer.parseInt(JOptionPane.showInputDialog("Which product would you like to add?\n1.smartphone\n2.T-Shirt\n3.oop "));
            switch (Array) {
                case 1:
                    X.addProduct(E);
                    break;
                case 2:
                    X.addProduct(C);
                    break;
                case 3:
                    X.addProduct(B);
                    break;
                default:
                    JOptionPane.showMessageDialog(null, "invalid choice!");
            }
        }

        // Displaying total price of products in the cart
        JOptionPane.showMessageDialog(null, "your total is: " + X.calculatePrice() + "$.");

        // Asking user if they want to place the order
        int n = Integer.parseInt(JOptionPane.showInputDialog(" Do you want to place the order?1.yes  2.No"));
        X.placeOrder(n);

        // Displaying order information or cart status based on user input
        if (n == 1) {
            Order O = new Order(X);
            O.printOrderInfo();
        } else if (n == 2) {
            JOptionPane.showMessageDialog(null, "cart is now empty");
        } else {
            JOptionPane.showMessageDialog(null, "invalid input");
        }
    }
```

2)class Order



```
64
65  class Order {
        private int customerid;
        private int orderid;
        private Product[] arr;
        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: ");
            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

```
class Cart {
    private int customerid;
    private int nproduct;
    protected Product[] arr;

    // Getter for the array of products in the cart
    public Product[] getArr() {
        return arr;
    }

    // Setter for the array of products in the cart
    public void setArr(Product[] arr) {
        this.arr = new Product[nproduct];
    }

    // Getter for customer ID
    public int getCustomerid() {
        return customerid;
    }

    // Setter for customer ID
    public void setCustomerid(int customerid) {
        this.customerid = customerid;
    }

    // Getter for number of products in the cart
    public int getNproduct() {
        return nproduct;
    }

    // Setter for number of products in the cart
    public void setNproduct(int nproduct) {
        this.nproduct = Math.abs(nproduct);
        arr = new Product[this.nproduct];
    }
```

Source History

```
    // Method to add a product to the cart
    public void addProduct(Product product) {
        for (int i = 0; i < nproduct; i++) {
            if (arr[i] == null) {
                arr[i] = product;
                return;
            }
        }
        JOptionPane.showMessageDialog(null, "The cart is full, cannot add more products.");
    }

    // Method to remove a product from the cart
    public void removeProduct(int index) {
        if (index >= 0 && index < nproduct) {
            arr[index] = null;
        } else {
            JOptionPane.showMessageDialog(null, "Invalid index, cannot remove product.");
        }
    }

    // Method to calculate the total price of products in the cart
    public float calculatePrice() {
        float total = 0;
        for (int i = 0; i < nproduct; i++) {
            if (arr[i] != null) {
                total += arr[i].getPrice();
            }
        }
        return total;
    }
```

```
        }
        return total;
    }

    // Method to place an order based on user input
    public void placeOrder(int choice) {
        switch (choice) {
            case 1:
                System.out.println("Order placed successfully.");
                break;
            case 2:
                for (int i = 0; i < arr.length; i++) {
                    arr[i] = null;
                }
                System.out.println("Cart is now empty.");
                break;
            default:
                System.out.println("Invalid input.");
        }
    }
}
```

4) class Product

```

class Product {
    protected int productid;
    protected String name;
    protected float price;

    // Getters and setters for product attributes
    public int getProductid() {
        return productid;
    }

    public void setProductid(int productid) {
        this.productid = Math.abs(productid);
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public float getPrice() {
        return price;
    }

    public void setPrice(float price) {
        this.price = Math.abs(price);
    }
}
```

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;
             this.setProductid(productId);
             this.setWarrantyperiod(warrantyPeriod);
             this.setName(name);
             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;
             this.setProductid(productId);
             this.setName(name);
             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

```
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;
             this.setProductid(productId);
             this.setName(name);
             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

```
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) {
             this.setCustomerid(id);
             this.setName(name);
             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;
```