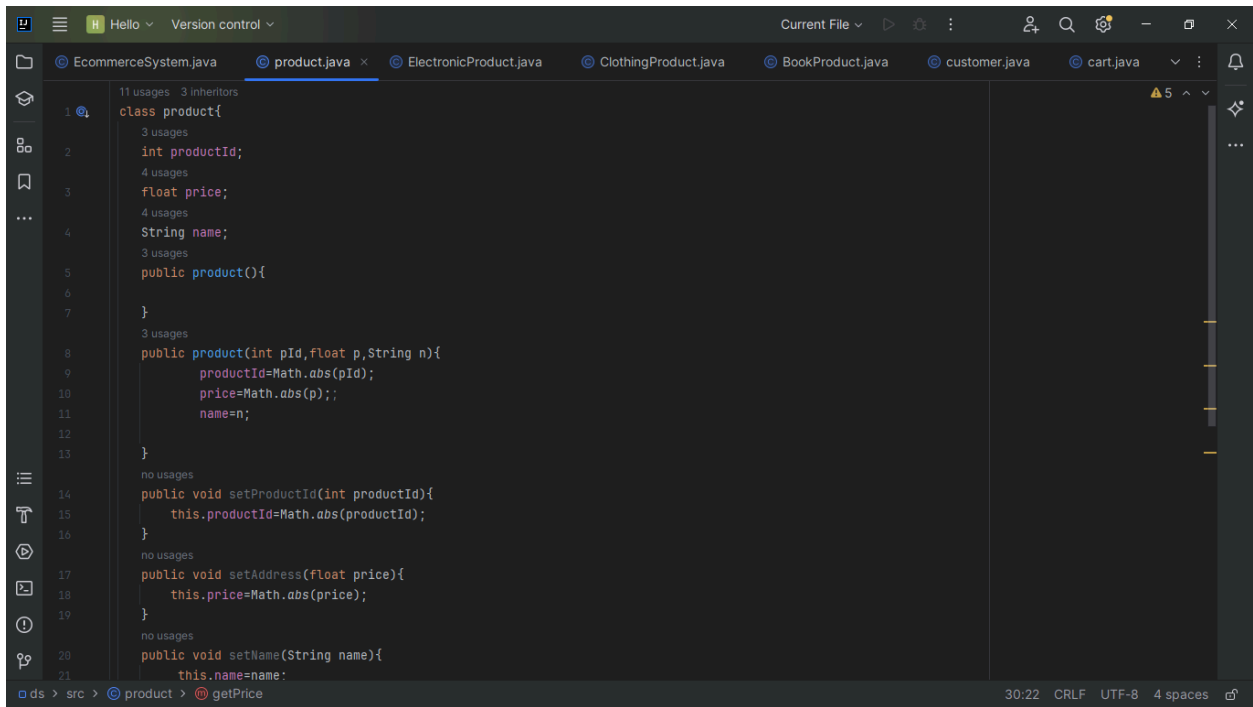


EcommerceSystem class:

```
1  import java.util.Scanner;
2  public class EcommerceSystem {
3      public static void main(String[] args) {
4          Scanner input = new Scanner(System.in);
5          ElectronicProduct e1 = new ElectronicProduct( pId: 1, p: 599.99f, n: "smartphone", b: "samsung", w: 1);
6          ClothingProduct cp1 = new ClothingProduct( pId: 2, p: 19.99f, n: "T-shirt", s: "Medium", f: "Cotton");
7          BookProduct b1 = new BookProduct( pId: 3, p: 39.99f, n: "00P", a: "O'Reilly", pu: "X Publication");
8          System.out.println("Welcome to E-commerce System!");
9          System.out.println("please enter your ID: ");
10         int customerId = input.nextInt();
11         System.out.println("please enter your name: ");
12         String customerName = input.next();
13         System.out.println("please enter your address ");
14         String customerAddress = input.next();
15         customer customer = new customer(customerId, customerName, customerAddress);
16         System.out.println("How many products you want to add to your cart?");
17         int nProduct = input.nextInt();
18         cart cart = new cart(customer.getCustomerId(), nProduct);
19         for (int i = 0; i < nProduct; i++) {
20             System.out.println("Which product would you like to add? 1- Smartphone 2- T-shirt 3- 00P ");
21             int type = input.nextInt();
22             switch (type) {
23                 case 1:
24                     cart.addProduct(e1);
25                     break;
26                 case 2:
27                     cart.addProduct(cp1);
28                     break;
29                 case 3:
30                     cart.addProduct(b1);
```

```
19         for (int i = 0; i < nProduct; i++) {
20             System.out.println("Which product would you like to add? 1- Smartphone 2- T-shirt 3- 00P ");
21             int type = input.nextInt();
22             switch (type) {
23                 case 1:
24                     cart.addProduct(e1);
25                     break;
26                 case 2:
27                     cart.addProduct(cp1);
28                     break;
29                 case 3:
30                     cart.addProduct(b1);
31                     break;
32                 default:
33                     System.out.println("invalid product type");
34                     break;
35             }
36         }
37
38         cart.placeOrder();
39     }
40 }
41
42
43
44
45
```

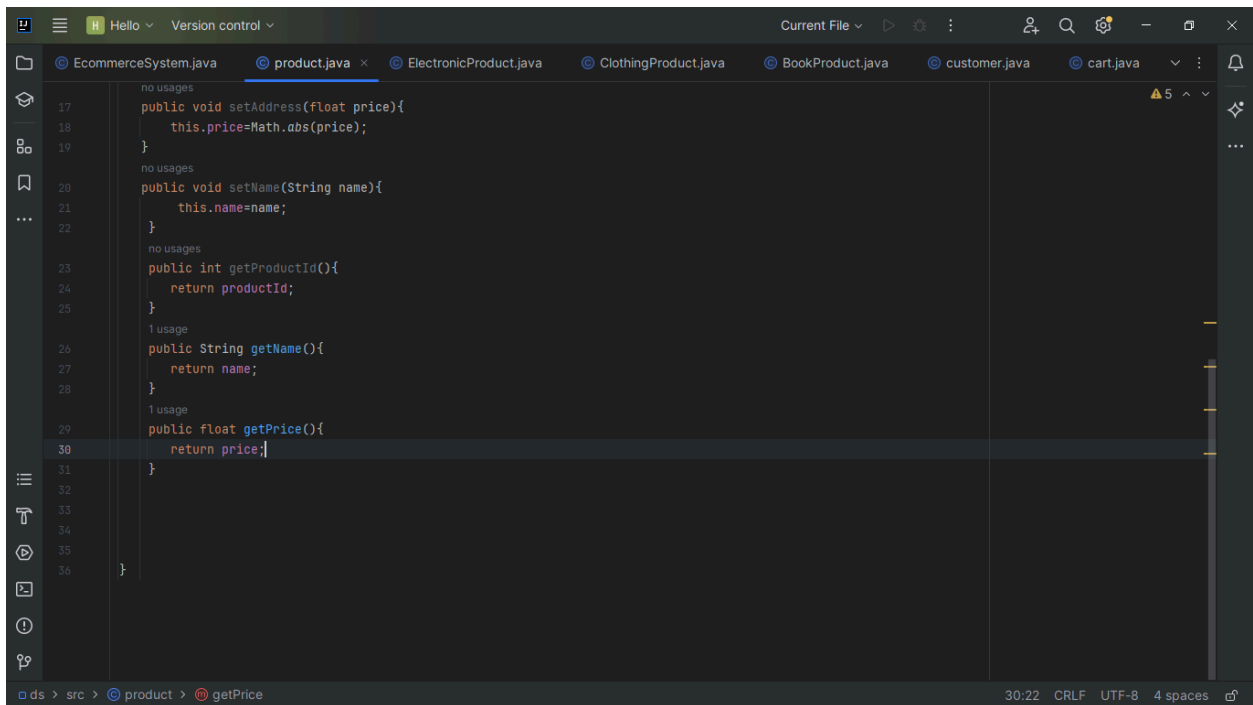
Product class:



This screenshot shows the initial state of the `product.java` file in an IDE. The code defines a `product` class with the following methods:

- `product()`: A no-argument constructor.
- `product(int pId, float p, String n)`: A constructor that initializes `productId`, `price`, and `name` using `Math.abs`.
- `setProductId(int productId)`: A setter for `productId` that uses `Math.abs`.
- `setAddress(float price)`: A setter for `price` that uses `Math.abs`.
- `setName(String name)`: A setter for `name`.

The IDE interface includes a sidebar with navigation icons, a top toolbar with search and version control options, and a status bar at the bottom indicating the file path `ds > src > product` and encoding `UTF-8`.

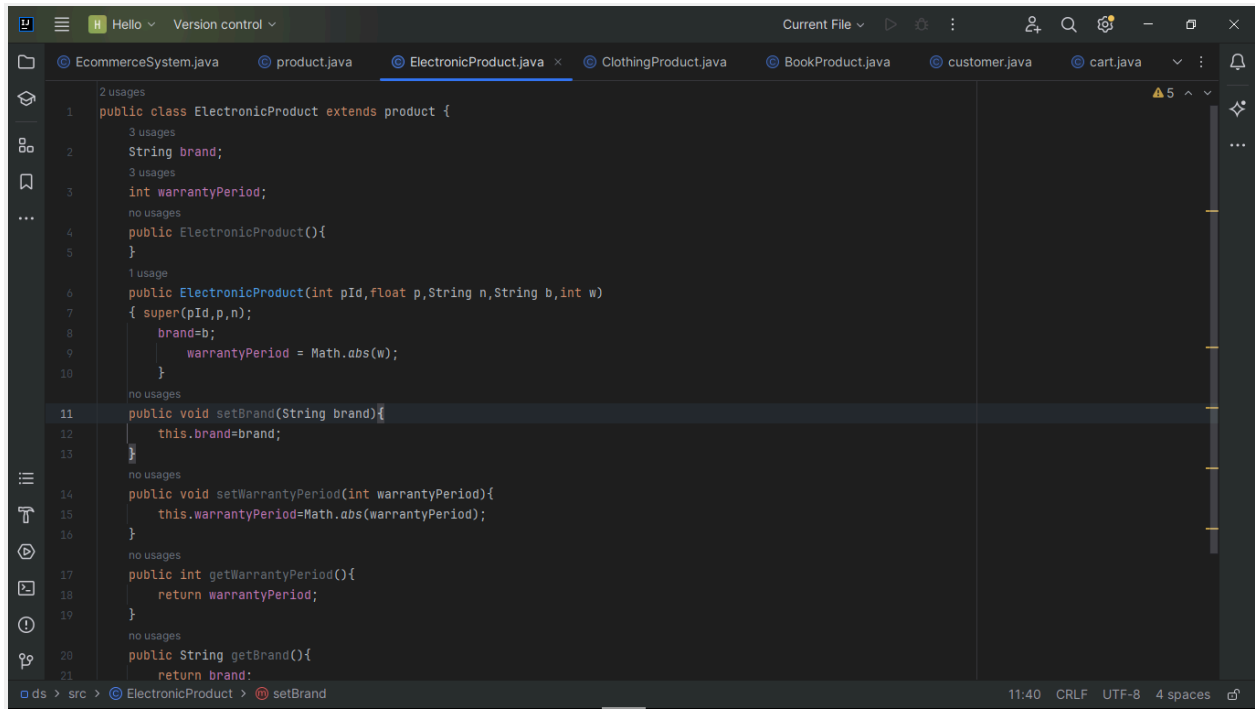


This screenshot shows the completed `product.java` file. In addition to the methods from the first screenshot, it now includes:

- `getProductId()`: A getter for `productId` that returns `productId`.
- `getName()`: A getter for `name` that returns `name`.
- `getPrice()`: A getter for `price` that returns `price`.

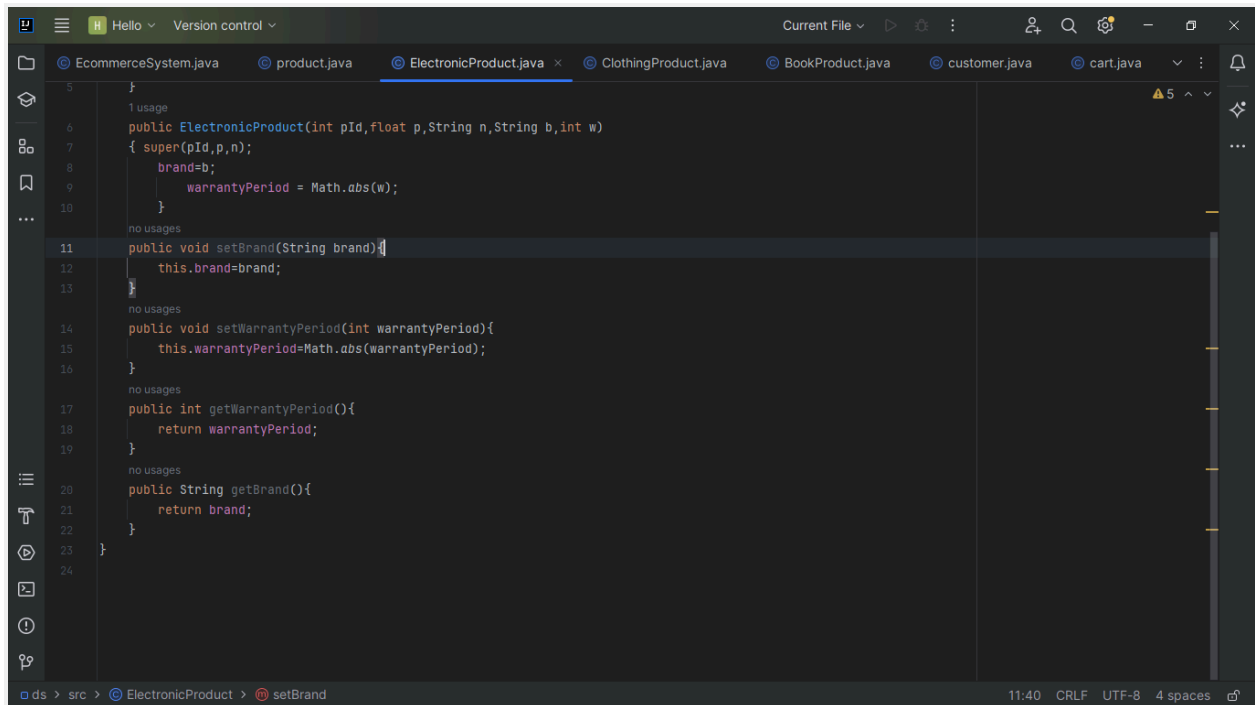
The IDE interface remains the same, with the status bar at the bottom showing the file path `ds > src > product` and encoding `UTF-8`.

ElectronicProduct class:



```
1 2 usages
2 public class ElectronicProduct extends product {
3     3 usages
4     String brand;
5     3 usages
6     int warrantyPeriod;
7     no usages
8     public ElectronicProduct(){
9     }
10    1 usage
11    public ElectronicProduct(int pId,float p,String n,String b,int w)
12    { super(pId,p,n);
13        brand=b;
14        warrantyPeriod = Math.abs(w);
15    }
16    no usages
17    public void setBrand(String brand){
18        this.brand=brand;
19    }
20    no usages
21    public void setWarrantyPeriod(int warrantyPeriod){
22        this.warrantyPeriod=Math.abs(warrantyPeriod);
23    }
24    no usages
25    public int getWarrantyPeriod(){
26        return warrantyPeriod;
27    }
28    no usages
29    public String getBrand(){
30        return brand;
31    }
32 }
```

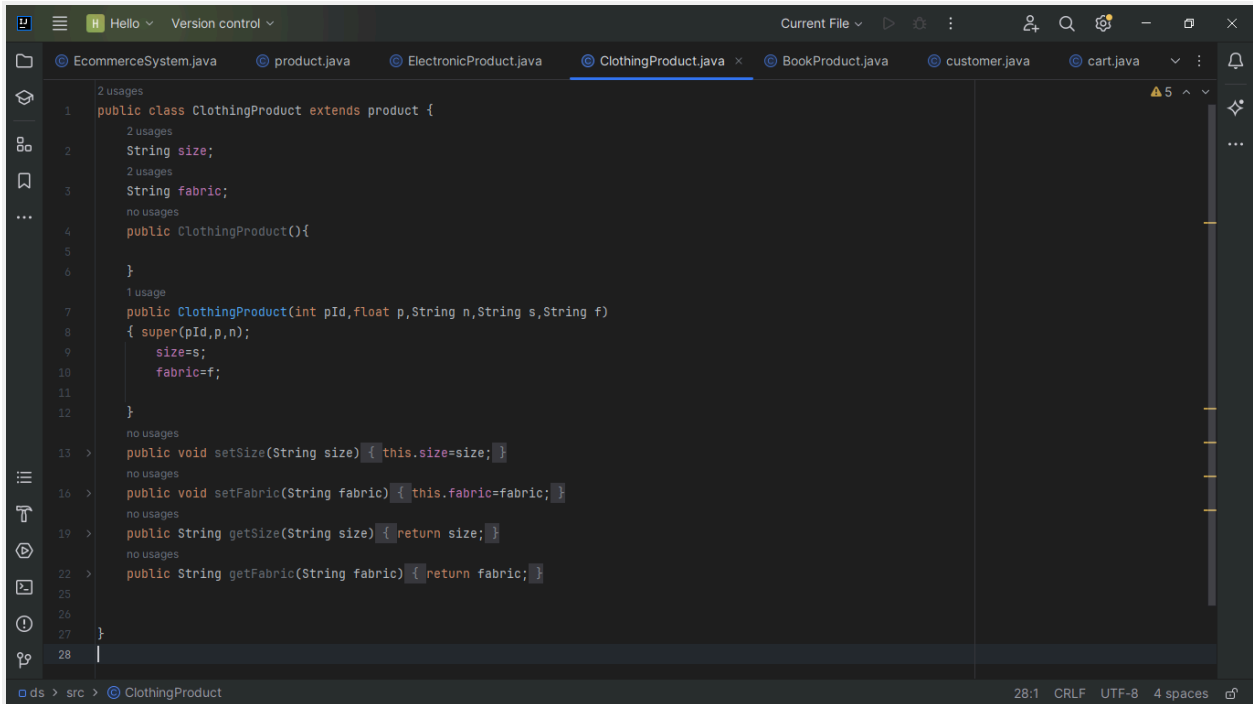
ds > src > ElectronicProduct > setBrand 11:40 CRLF UTF-8 4 spaces



```
5 }
6 1 usage
7 public ElectronicProduct(int pId,float p,String n,String b,int w)
8 { super(pId,p,n);
9     brand=b;
10    warrantyPeriod = Math.abs(w);
11 }
12 no usages
13 public void setBrand(String brand){
14     this.brand=brand;
15 }
16 no usages
17 public void setWarrantyPeriod(int warrantyPeriod){
18     this.warrantyPeriod=Math.abs(warrantyPeriod);
19 }
20 no usages
21 public int getWarrantyPeriod(){
22     return warrantyPeriod;
23 }
24 no usages
25 public String getBrand(){
26     return brand;
27 }
28 }
```

ds > src > ElectronicProduct > setBrand 11:40 CRLF UTF-8 4 spaces

ClothingProduct class:



```
1 2 usages
2 public class ClothingProduct extends product {
3     2 usages
4     String size;
5     2 usages
6     String fabric;
7     no usages
8     public ClothingProduct(){
9
10    }
11    1 usage
12    public ClothingProduct(int pId,float p,String n,String s,String f)
13    { super(pId,p,n);
14        size=s;
15        fabric=f;
16    }
17    no usages
18    public void setSize(String size) { this.size=size; }
19    no usages
20    public void setFabric(String fabric) { this.fabric=fabric; }
21    no usages
22    public String getSize(String size) { return size; }
23    no usages
24    public String getFabric(String fabric) { return fabric; }
25
26 }
27
28
```

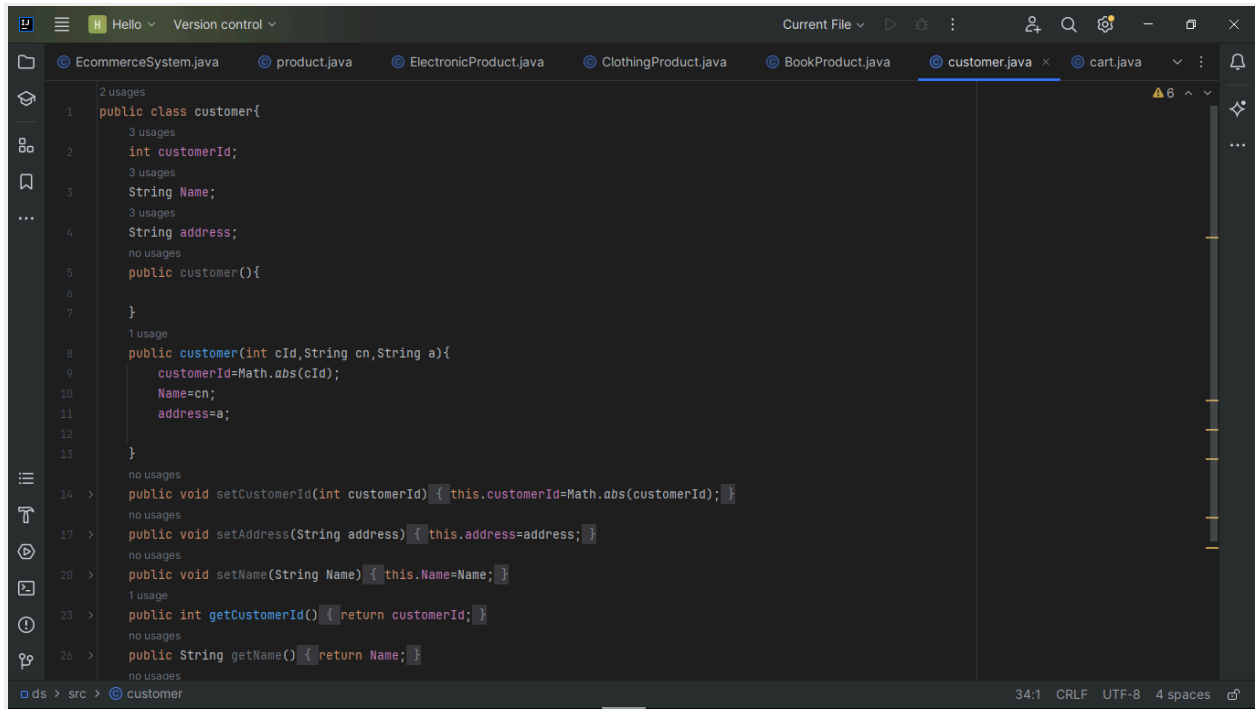
ds > src > ClothingProduct 28:1 CRLF UTF-8 4 spaces

BookProduct class:

```
1 public class BookProduct extends product{
2     String author;
3     String publisher;
4     public BookProduct(){
5
6     }
7     public BookProduct(int pId,float p,String n,String a,String pu){
8         super(pId,p,n);
9         author=a;
10        publisher=pu;
11    }
12    public void setAuthor(String author){
13        this.author=author;
14    }
15    public void setPublisher(String publisher){
16        this.publisher=publisher;
17    }
18    public String getAuthor(){
19        return author;
20    }
21    public String getPublisher(){
```

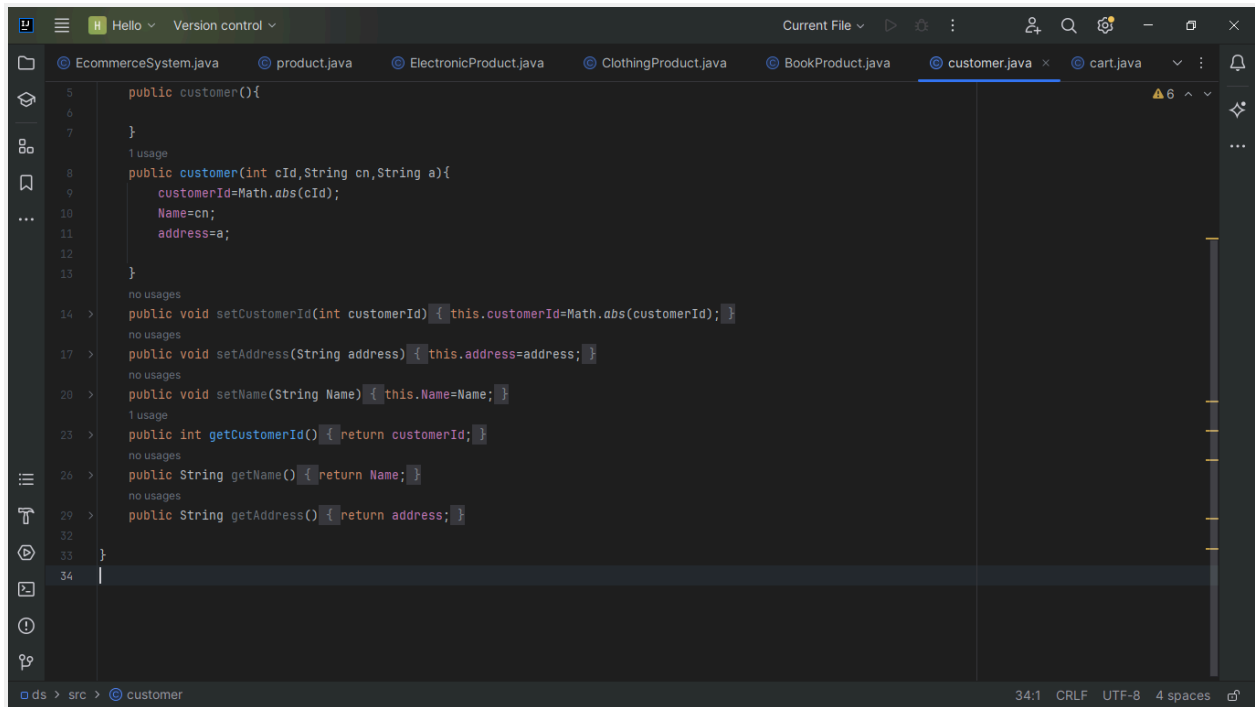
```
22        return publisher;
23    }
24
25
26 }
27
```

Customer class:



```
1 2 usages
2 public class customer{
3     3 usages
4     int customerId;
5     3 usages
6     String Name;
7     3 usages
8     String address;
9     no usages
10    public customer(){
11
12    }
13    1 usage
14    public customer(int cId,String cn,String a){
15        customerId=Math.abs(cId);
16        Name=cn;
17        address=a;
18    }
19    no usages
20    public void setCustomerId(int customerId){ this.customerId=Math.abs(customerId); }
21    no usages
22    public void setAddress(String address){ this.address=address; }
23    no usages
24    public void setName(String Name){ this.Name=Name; }
25    1 usage
26    public int getCustomerId(){ return customerId; }
27    no usages
28    public String getName(){ return Name; }
29    no usages
30    }
```

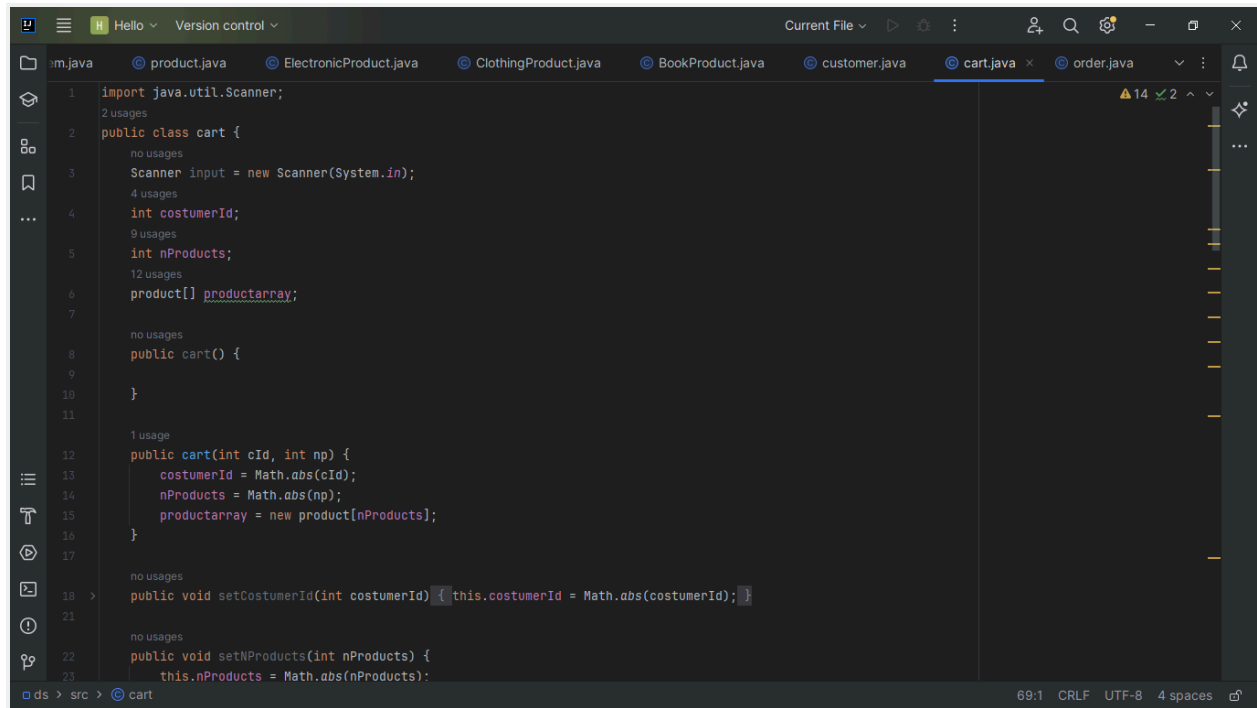
ds > src > customer 34:1 CRLF UTF-8 4 spaces



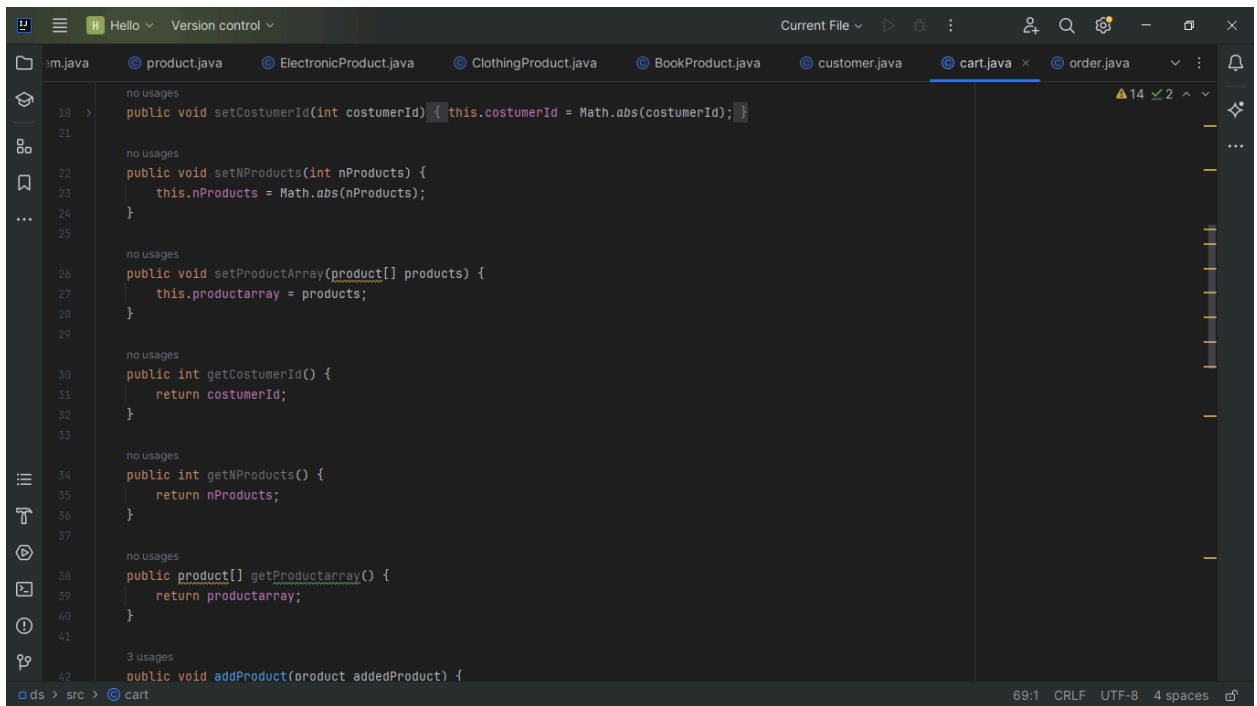
```
5 public customer(){
6
7    }
8    1 usage
9    public customer(int cId,String cn,String a){
10        customerId=Math.abs(cId);
11        Name=cn;
12        address=a;
13    }
14    no usages
15    public void setCustomerId(int customerId){ this.customerId=Math.abs(customerId); }
16    no usages
17    public void setAddress(String address){ this.address=address; }
18    no usages
19    public void setName(String Name){ this.Name=Name; }
20    1 usage
21    public int getCustomerId(){ return customerId; }
22    no usages
23    public String getName(){ return Name; }
24    no usages
25    public String getAddress(){ return address; }
26
27 }
28
29
30
31
32
33
34
```

ds > src > customer 34:1 CRLF UTF-8 4 spaces

Cart class:



```
1 import java.util.Scanner;
2 public class cart {
3     Scanner input = new Scanner(System.in);
4     int costumerId;
5     int nProducts;
6     product[] productarray;
7
8     public cart() {
9     }
10
11     public cart(int cId, int np) {
12         costumerId = Math.abs(cId);
13         nProducts = Math.abs(np);
14         productarray = new product[nProducts];
15     }
16
17     public void setCostumerId(int costumerId) { this.costumerId = Math.abs(costumerId); }
18
19     public void setNProducts(int nProducts) {
20         this.nProducts = Math.abs(nProducts);
21     }
22
23 }
```



```
18 public void setCostumerId(int costumerId) { this.costumerId = Math.abs(costumerId); }
19
20 public void setNProducts(int nProducts) {
21     this.nProducts = Math.abs(nProducts);
22 }
23
24 public void setProductArray(product[] products) {
25     this.productarray = products;
26 }
27
28 public int getCostumerId() {
29     return costumerId;
30 }
31
32 public int getNProducts() {
33     return nProducts;
34 }
35
36 public product[] getProductarray() {
37     return productarray;
38 }
39
40 public void addProduct(product addedProduct) {
```

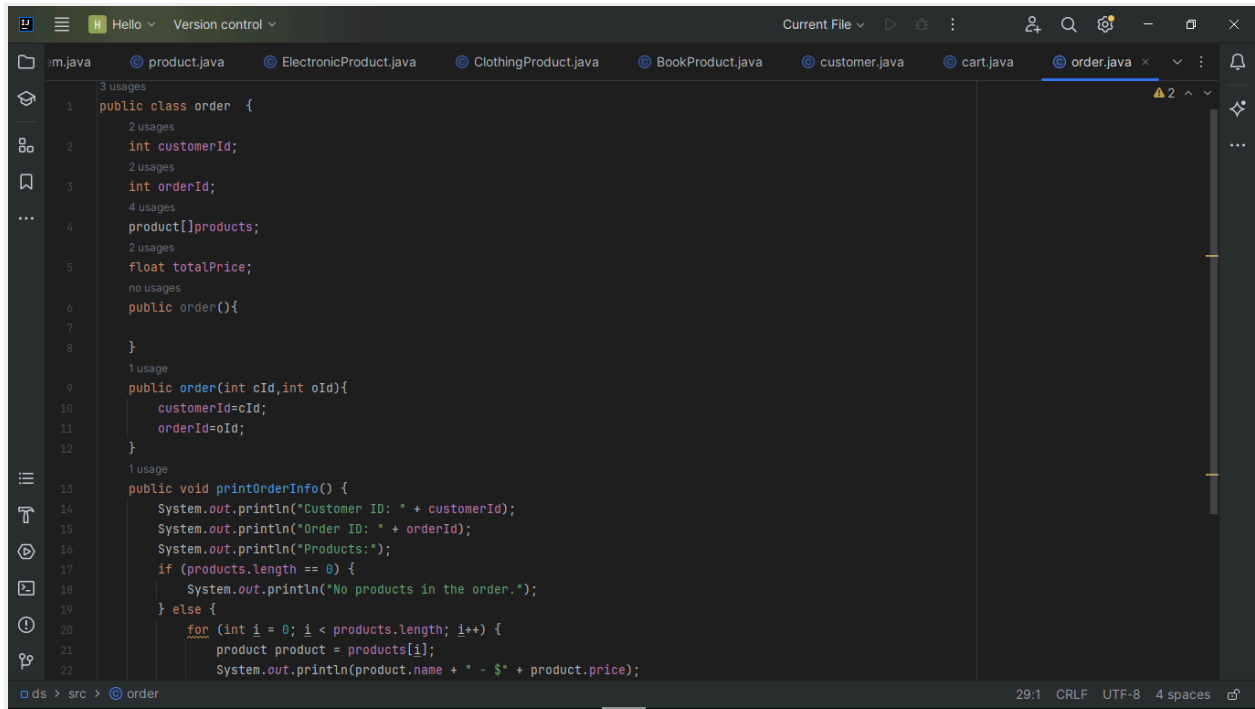
This screenshot shows the 'cart.java' file in an IDE. The 'addProduct' method (lines 42-50) iterates through the 'productarray' and adds a product if it's not null. The 'removeProduct' method (lines 51-68) iterates through the array, shifts elements to the left to remove a specified product, and prints a confirmation message. The status bar at the bottom indicates the cursor is at line 69, column 1, with CRLF line endings, UTF-8 encoding, and 4 spaces for indentation.

```
41
42 3 usages
43 public void addProduct(product addedProduct) {
44     for (int i = 0; i < productarray.length; i++) {
45         if (productarray[i] == null) {
46             productarray[i] = addedProduct;
47             return;
48         }
49     }
50 }
51
52 no usages
53 public void removeProduct(product removedProduct) {
54     boolean found = false;
55     for (int i = 0; i < nProducts; i++) {
56         if (productarray[i] == removedProduct) {
57             for (int j = i; j < nProducts - 1; j++) {
58                 productarray[j] = productarray[j + 1];
59             }
60             productarray[nProducts - 1] = null;
61             nProducts--;
62             found = true;
63             System.out.println(removedProduct.getName() + " removed from the cart.");
64             break;
65         }
66     }
67     if (!found) {
68         System.out.println("Product not found in the cart.");
69 }
70 }
```

This screenshot shows the 'placeOrder' method in 'cart.java' (lines 78-97). It uses a 'Scanner' to get user input, calculates the total price, and creates an 'Order' object if the user chooses to place the order. The status bar at the bottom shows the cursor at line 37, column 1, with CRLF line endings, UTF-8 encoding, and 4 spaces for indentation.

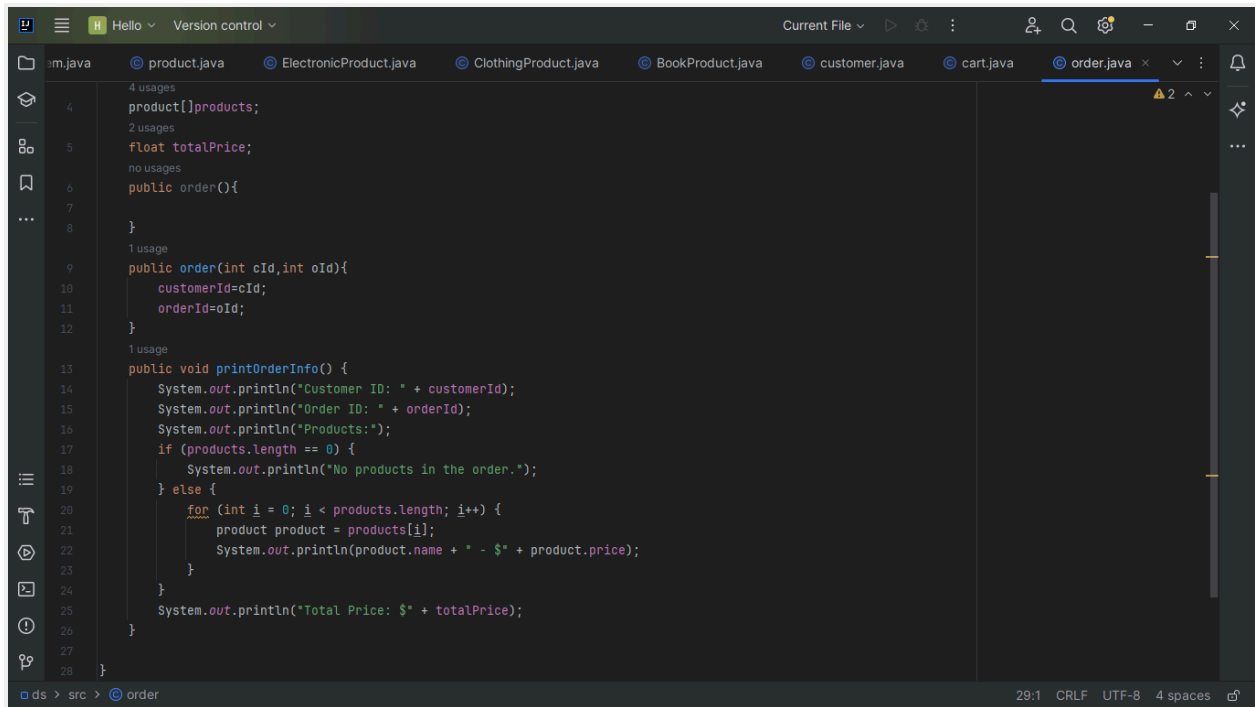
```
74 }
75     return totalPrice;
76 }
77
78 1 usage
79 public order placeOrder() {
80     Scanner input = new Scanner(System.in);
81     System.out.println("Your total price is: $" + calculateTotalPrice());
82     System.out.println("Would you like to place the order? (1=yes, 2=no)");
83     int choice = input.nextInt();
84     if (choice == 1) {
85         System.out.println("Order placed successfully!");
86         order order = new order(costumerId, old: 1);
87         order.products = productarray;
88         order.totalPrice = calculateTotalPrice();
89         order.printOrderInfo();
90         return order;
91     } else if (choice == 2) {
92         System.out.println("Order canceled.");
93     } else {
94         System.out.println("Invalid choice.");
95     }
96     return null;
97 }
```


Order class:



This screenshot shows the 'order.java' file in an IDE. The code defines a 'public class order' with attributes 'customerId', 'orderId', and 'products', and methods 'order()', 'order(int cId, int oId)', and 'printOrderInfo()'. The 'products' attribute is annotated with '3 usages'. The 'printOrderInfo()' method contains logic to print order details, including a loop over the 'products' array.

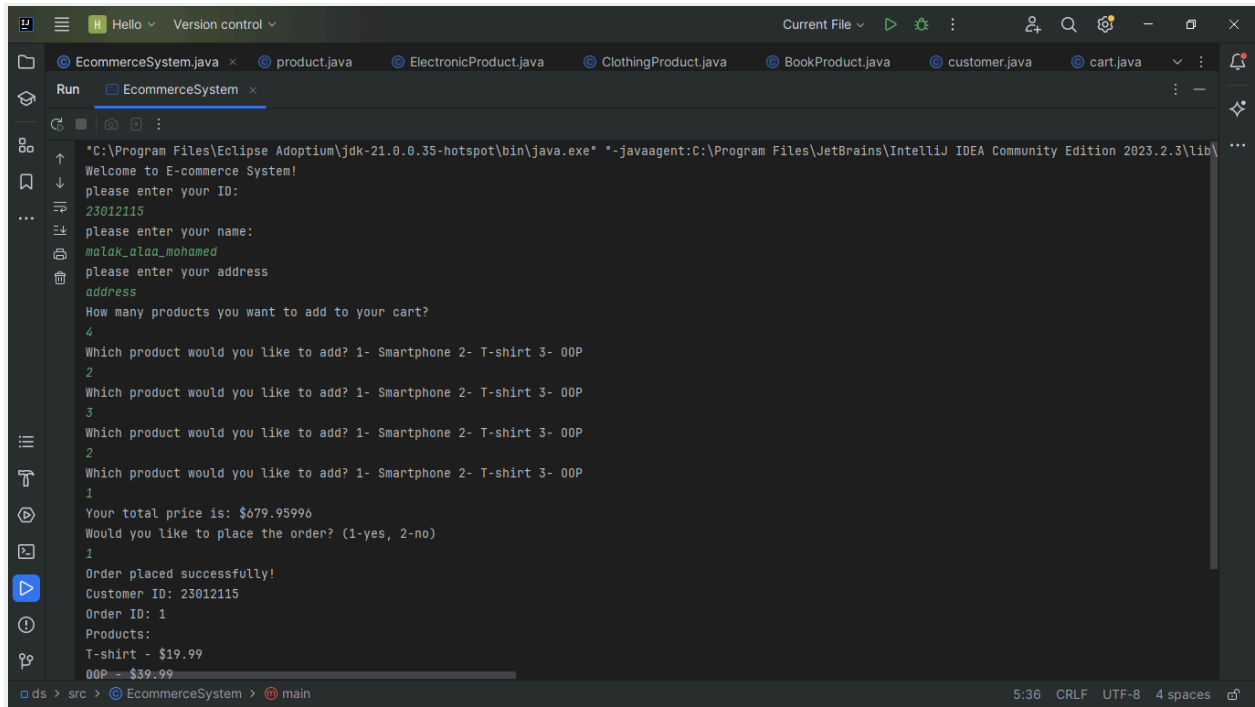
```
1 public class order {
2     int customerId;
3     int orderId;
4     product[] products;
5     float totalPrice;
6     public order(){
7
8     }
9     public order(int cId,int oId){
10         customerId=cId;
11         orderId=oId;
12     }
13     public void printOrderInfo() {
14         System.out.println("Customer ID: " + customerId);
15         System.out.println("Order ID: " + orderId);
16         System.out.println("Products:");
17         if (products.length == 0) {
18             System.out.println("No products in the order.");
19         } else {
20             for (int i = 0; i < products.length; i++) {
21                 product product = products[i];
22                 System.out.println(product.name + " - $" + product.price);
23             }
24         }
25         System.out.println("Total Price: $" + totalPrice);
26     }
27 }
28 }
```



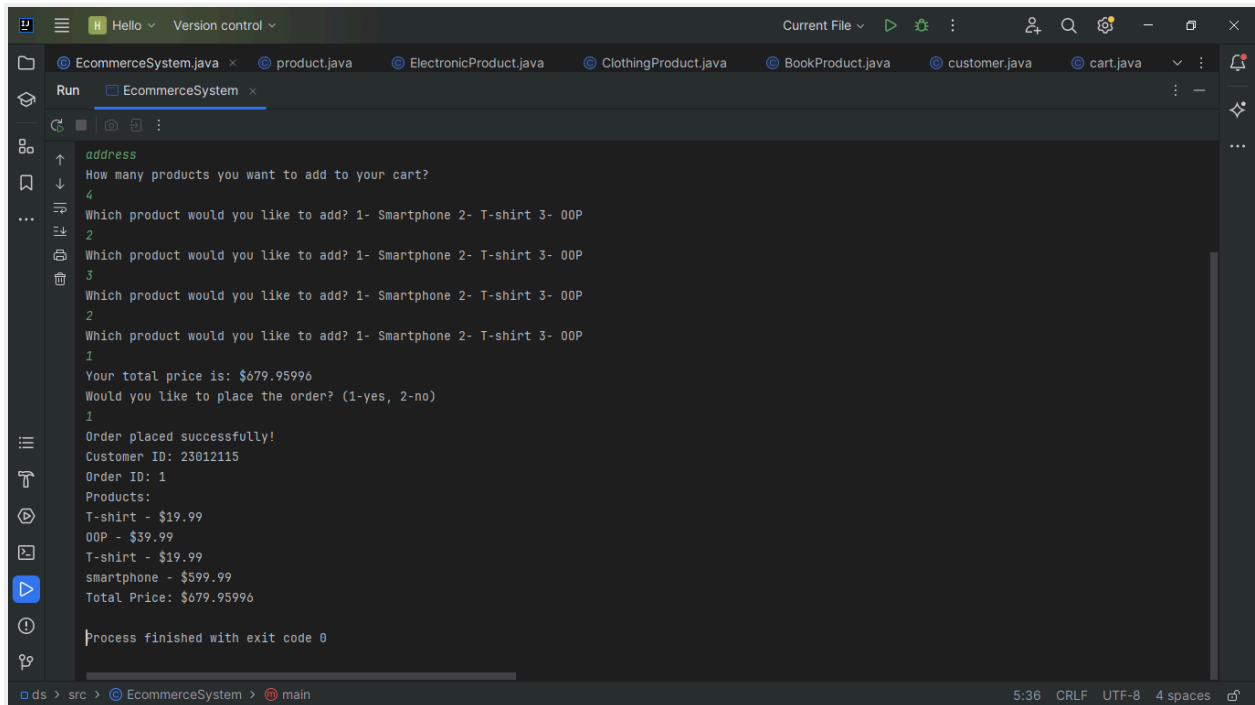
This screenshot shows the 'order.java' file in an IDE, similar to the previous one but with '4 usages' for the 'products' attribute. The code is identical to the previous screenshot, including the 'printOrderInfo()' method which prints the total price at the end.

```
4 product[] products;
5 float totalPrice;
6 public order(){
7
8 }
9 public order(int cId,int oId){
10     customerId=cId;
11     orderId=oId;
12 }
13 public void printOrderInfo() {
14     System.out.println("Customer ID: " + customerId);
15     System.out.println("Order ID: " + orderId);
16     System.out.println("Products:");
17     if (products.length == 0) {
18         System.out.println("No products in the order.");
19     } else {
20         for (int i = 0; i < products.length; i++) {
21             product product = products[i];
22             System.out.println(product.name + " - $" + product.price);
23         }
24     }
25     System.out.println("Total Price: $" + totalPrice);
26 }
27 }
28 }
```

Output:



```
"C:\Program Files\Eclipse Adoptium\jdk-21.0.0-hotspot\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.3\lib\idea_rt.jar=536:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.3\bin" -Dfile.encoding=UTF-8
Welcome to E-commerce System!
please enter your ID:
23012115
please enter your name:
malak_alaa_mohamed
please enter your address
address
How many products you want to add to your cart?
4
Which product would you like to add? 1- Smartphone 2- T-shirt 3- OOP
2
Which product would you like to add? 1- Smartphone 2- T-shirt 3- OOP
3
Which product would you like to add? 1- Smartphone 2- T-shirt 3- OOP
2
Which product would you like to add? 1- Smartphone 2- T-shirt 3- OOP
1
Your total price is: $679.95996
Would you like to place the order? (1=yes, 2=no)
1
Order placed successfully!
Customer ID: 23012115
Order ID: 1
Products:
T-shirt - $19.99
OOP - $39.99
```



```
address
How many products you want to add to your cart?
4
Which product would you like to add? 1- Smartphone 2- T-shirt 3- OOP
2
Which product would you like to add? 1- Smartphone 2- T-shirt 3- OOP
3
Which product would you like to add? 1- Smartphone 2- T-shirt 3- OOP
2
Which product would you like to add? 1- Smartphone 2- T-shirt 3- OOP
1
Your total price is: $679.95996
Would you like to place the order? (1=yes, 2=no)
1
Order placed successfully!
Customer ID: 23012115
Order ID: 1
Products:
T-shirt - $19.99
OOP - $39.99
T-shirt - $19.99
Smartphone - $599.99
Total Price: $679.95996

Process finished with exit code 0
```