

BÁO CÁO THỰC HÀNH LAB 2 LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

Contents

1. Đề bài.....	2
2. Yêu cầu bài toán	2
3. Use case diagram.....	3
Usecase của khách hàng:.....	3
Usecase của Quản lí:.....	3
4. Class diagram	4
5. Mã nguồn	5
Main class:.....	5
DigitalVideoDisc class.....	6
Question:.....	6
Cart class:.....	9
6. Demo.....	11

Figures

Figure 1 Biểu đồ UseCase Khách hàng.....	3
Figure 2 Biểu đồ UseCase Quản lí	3
Figure 3 Biểu đồ Class	4
Figure 4 Code Main	5
Figure 5 DigitalVideoDisc (1)	6
Figure 6 DigitalVideoDisc (2)	7
Figure 7 DigitalVideoDisc (3)	8
Figure 8 Code Cart (1).....	9
Figure 9 Code Cart (2).....	10
Figure 10 demo without removeDigitalVideoDisc.....	11
Figure 11 demo with removeDigitalVideoDisc.....	12

1.Đề bài

Thiết kế hệ thống mới cho hệ thống AIMS. (Hiện tại chỉ có 1 phương tiện: DVD)

2.Yêu cầu bài toán

Đối với người mua hàng:

- Duyệt danh sách các DVD có sẵn trong cửa hàng
- Tìm kiếm DVD theo: tiêu đề, danh mục và giá cả
- Xem thông tin chi tiết của 1 DVD
- Thêm DVD vào giỏ hàng
- Xem giỏ hàng
- Sắp xếp DVD trong giỏ hàng theo tiêu đề hoặc chi phí
- Cập nhật số lượng DVD trong giỏ hàng
- Đặt hàng

Đối với quản lý cửa hàng

- Đăng nhập, kiểm tra quyền
- Xem danh sách các đơn hàng đang chờ xử lý
- Xem chi tiết đơn hàng và xử lý nốt(đồng ý/từ chối đặt hàng)
- Thêm(xóa) DVD mới vào(ra) danh sách sản phẩm

3. Use case diagram

Usecase của khách hàng:

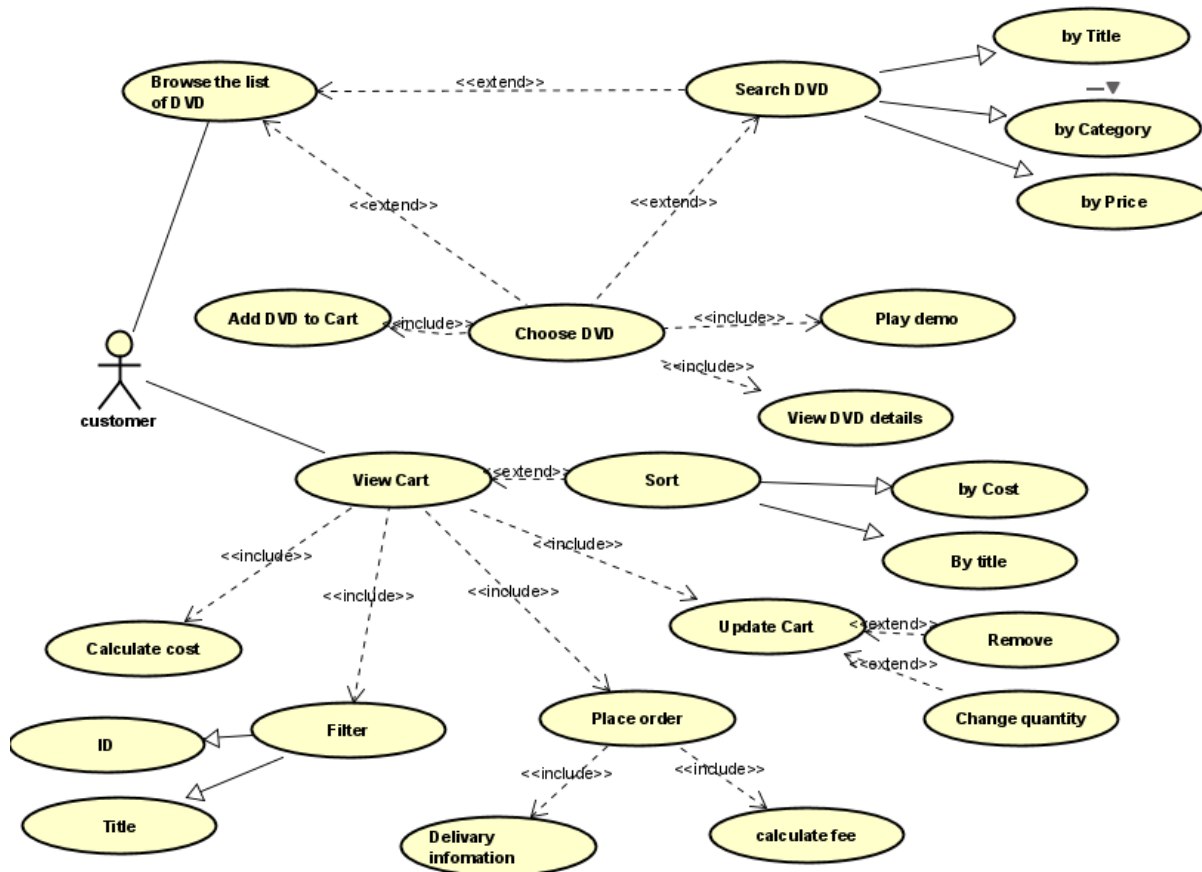


Figure 1 Biểu đồ UseCase Khách hàng

Usecase của Quản lí:

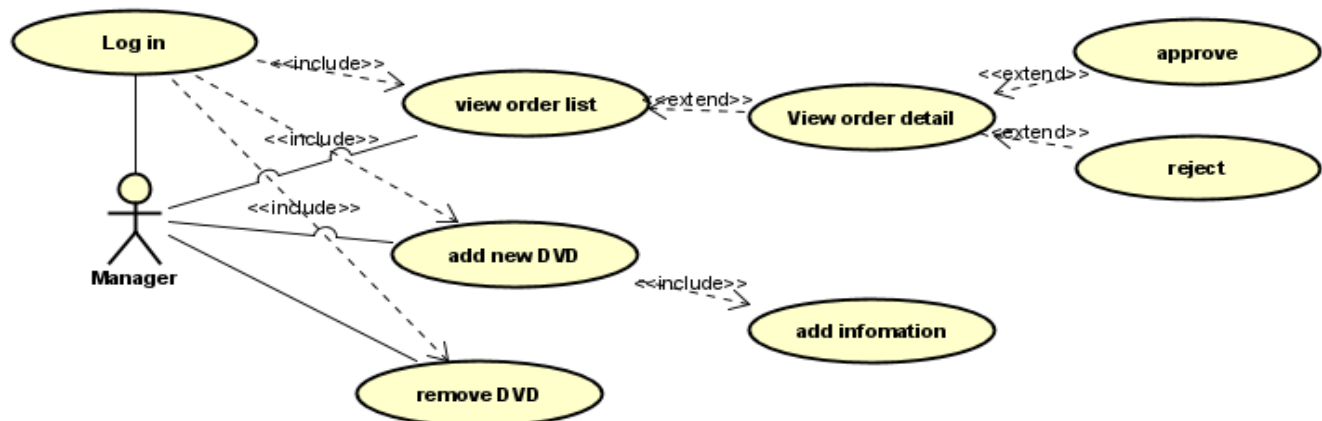


Figure 2 Biểu đồ UseCase Quản lí

4. Class diagram

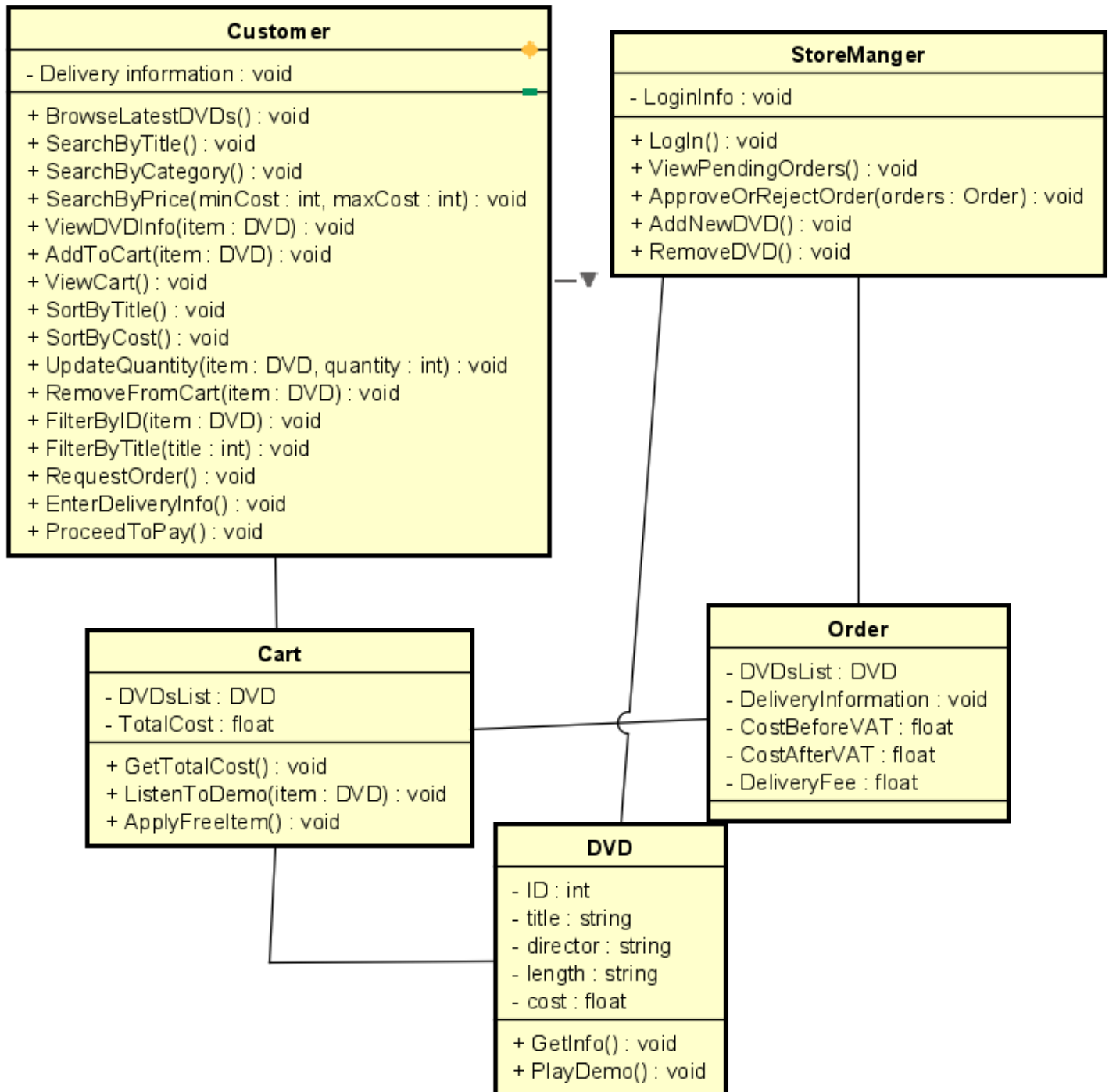


Figure 3 Biểu đồ Class

Miêu tả:

Customer: gồm thông tin giao hàng của khách hàng các chức năng khách hàng có thể sử dụng

Cart: gồm thông tin các món hàng trong giỏ hàng và giá tiền tổng món hàng

DVD: Thông tin của tất cả món hàng trong kho

Order: Thông tin đơn hàng gồm thông tin giao hàng của khách, các loại hàng và giá trị đơn hàng

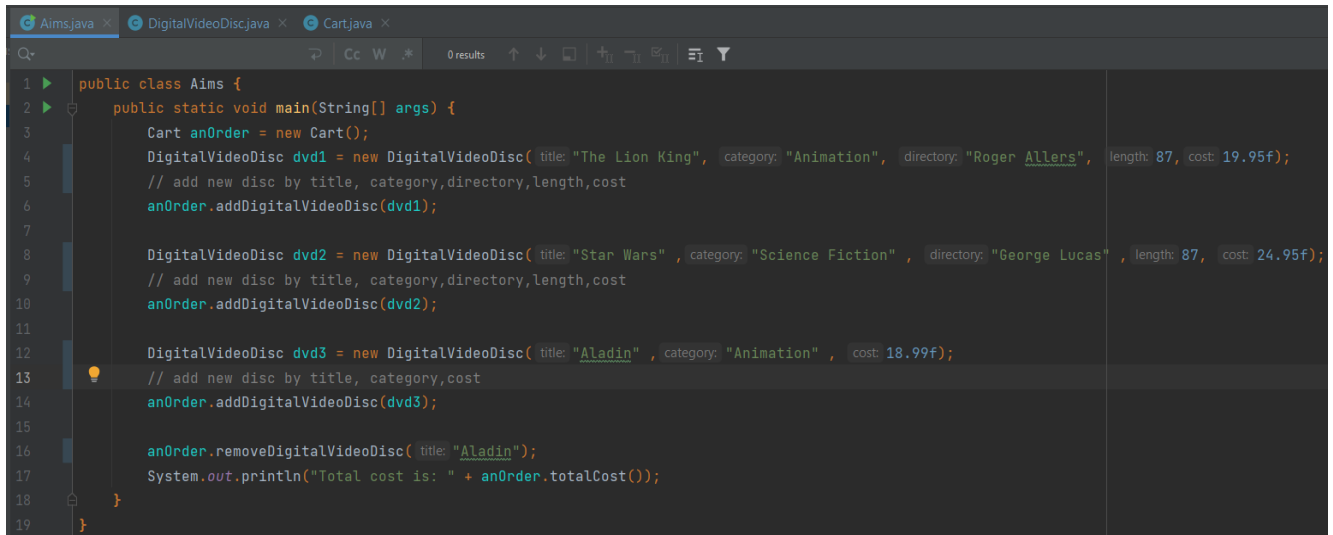
StoreManger: Thông tin đăng nhập của quản lí, thêm mặt hàng vào DVD, xóa mặt hàng, xác nhận hoặc từ chối đơn hàng

PaymentSystem: Xác thực thông tin thanh toán của đơn hàng

AimSoftware: Nếu xác thực thành công sẽ hiển thị thông tin và gửi email

5. Mã nguồn

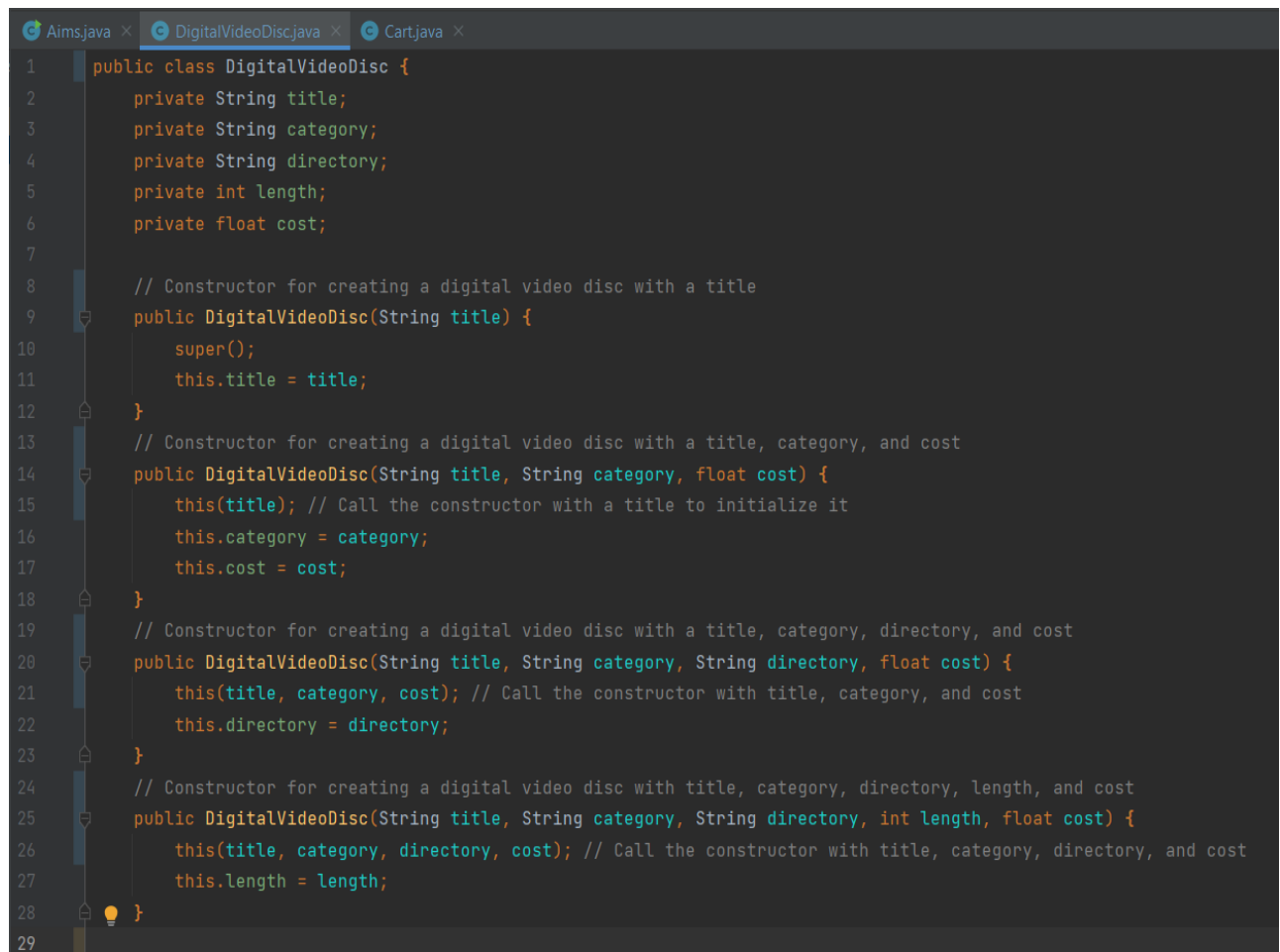
Main class:



```
1 public class Aims {
2     public static void main(String[] args) {
3         Cart anOrder = new Cart();
4         DigitalVideoDisc dvd1 = new DigitalVideoDisc( title: "The Lion King", category: "Animation", directory: "Roger Allers", length: 87, cost: 19.95f);
5         // add new disc by title, category,directory,length,cost
6         anOrder.addDigitalVideoDisc(dvd1);
7
8         DigitalVideoDisc dvd2 = new DigitalVideoDisc( title: "Star Wars" , category: "Science Fiction" , directory: "George Lucas" , length: 87, cost: 24.95f);
9         // add new disc by title, category,directory,length,cost
10        anOrder.addDigitalVideoDisc(dvd2);
11
12        DigitalVideoDisc dvd3 = new DigitalVideoDisc( title: "Aladin" , category: "Animation" , cost: 18.99f);
13        // add new disc by title, category,cost
14        anOrder.addDigitalVideoDisc(dvd3);
15
16        anOrder.removeDigitalVideoDisc( title: "Aladin");
17        System.out.println("Total cost is: " + anOrder.totalCost());
18    }
19 }
```

Figure 4 Code Main

DigitalVideoDisc class

The image shows a screenshot of a Java IDE with three tabs: Aims.java, DigitalVideoDisc.java (selected), and Cart.java. The code in DigitalVideoDisc.java defines a public class DigitalVideoDisc with private attributes: String title, String category, String directory, int length, and float cost. It includes four constructors: a no-argument constructor, a constructor with a title, a constructor with title, category, and cost, and a constructor with title, category, directory, length, and cost. The constructors use this() to call other constructors for initialization.

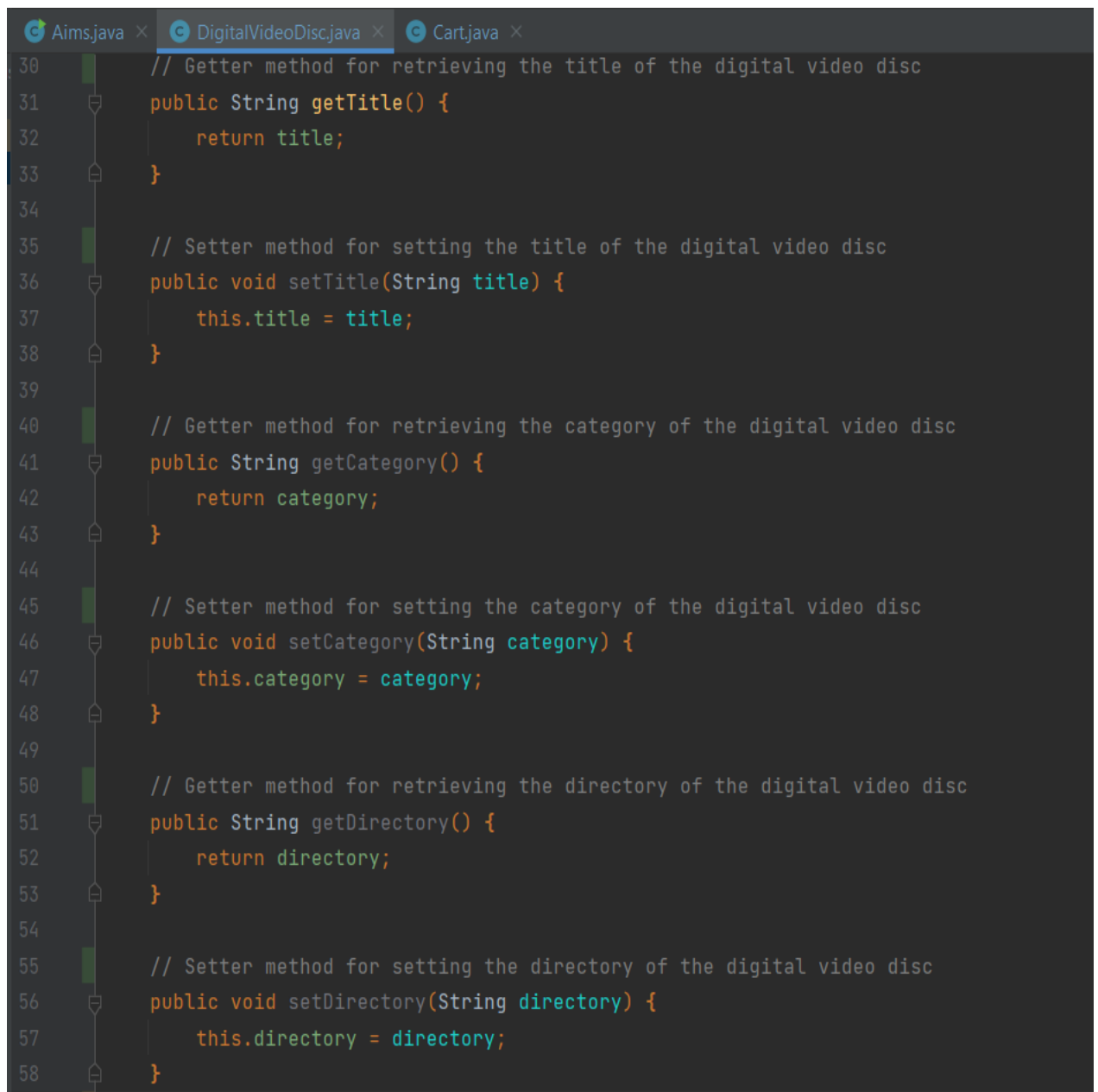
```
1 public class DigitalVideoDisc {
2     private String title;
3     private String category;
4     private String directory;
5     private int length;
6     private float cost;
7
8     // Constructor for creating a digital video disc with a title
9     public DigitalVideoDisc(String title) {
10         super();
11         this.title = title;
12     }
13     // Constructor for creating a digital video disc with a title, category, and cost
14     public DigitalVideoDisc(String title, String category, float cost) {
15         this(title); // Call the constructor with a title to initialize it
16         this.category = category;
17         this.cost = cost;
18     }
19     // Constructor for creating a digital video disc with a title, category, directory, and cost
20     public DigitalVideoDisc(String title, String category, String directory, float cost) {
21         this(title, category, cost); // Call the constructor with title, category, and cost
22         this.directory = directory;
23     }
24     // Constructor for creating a digital video disc with title, category, directory, length, and cost
25     public DigitalVideoDisc(String title, String category, String directory, int length, float cost) {
26         this(title, category, directory, cost); // Call the constructor with title, category, directory, and cost
27         this.length = length;
28     }
29 }
```

Figure 5 DigitalVideoDisc (1)

Question:

If you create a constructor method to build a DVD by title then create a constructor method to build a DVD by category. Does JAVA allow you to do this?

Có, Overload trong JAVA cho phép tạo 1 method khác chồng lên 1 method đã tồn tại trước đó.



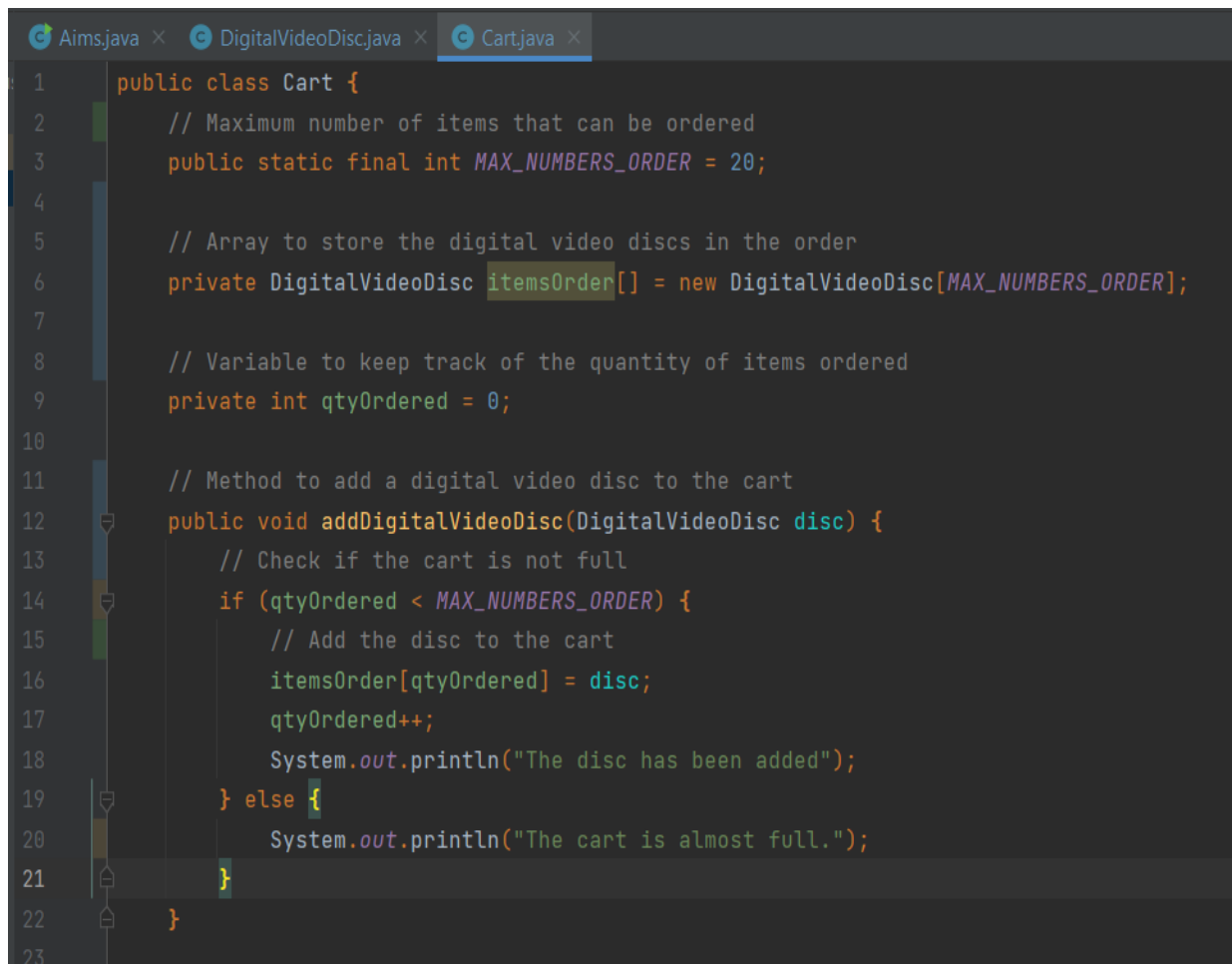
```
30 // Getter method for retrieving the title of the digital video disc
31 public String getTitle() {
32     return title;
33 }
34
35 // Setter method for setting the title of the digital video disc
36 public void setTitle(String title) {
37     this.title = title;
38 }
39
40 // Getter method for retrieving the category of the digital video disc
41 public String getCategory() {
42     return category;
43 }
44
45 // Setter method for setting the category of the digital video disc
46 public void setCategory(String category) {
47     this.category = category;
48 }
49
50 // Getter method for retrieving the directory of the digital video disc
51 public String getDirectory() {
52     return directory;
53 }
54
55 // Setter method for setting the directory of the digital video disc
56 public void setDirectory(String directory) {
57     this.directory = directory;
58 }
```

Figure 6 DigitalVideoDisc (2)

```
59
60 // Getter method for retrieving the length of the digital video disc
61 public int getLength() {
62     return length;
63 }
64
65 // Setter method for setting the length of the digital video disc
66 public void setLength(int length) {
67     this.length = length;
68 }
69
70 // Getter method for retrieving the cost of the digital video disc
71 public float getCost() {
72     return cost;
73 }
74
75 // Setter method for setting the cost of the digital video disc
76 public void setCost(float cost) { this.cost = cost; }
77
78
79 }
```

Figure 7 DigitalVideoDisc (3)

Cart class:



```
1 public class Cart {
2     // Maximum number of items that can be ordered
3     public static final int MAX_NUMBERS_ORDER = 20;
4
5     // Array to store the digital video discs in the order
6     private DigitalVideoDisc itemsOrder[] = new DigitalVideoDisc[MAX_NUMBERS_ORDER];
7
8     // Variable to keep track of the quantity of items ordered
9     private int qtyOrdered = 0;
10
11     // Method to add a digital video disc to the cart
12     public void addDigitalVideoDisc(DigitalVideoDisc disc) {
13         // Check if the cart is not full
14         if (qtyOrdered < MAX_NUMBERS_ORDER) {
15             // Add the disc to the cart
16             itemsOrder[qtyOrdered] = disc;
17             qtyOrdered++;
18             System.out.println("The disc has been added");
19         } else {
20             System.out.println("The cart is almost full.");
21         }
22     }
23 }
```

Figure 8 Code Cart (1)

```
23
24 // Method to remove a digital video disc from the cart
25 public void removeDigitalVideoDisc(String title) {
26     for (int i = 0; i < qtyOrdered; i++) {
27         if (itemsOrder[i] != null && itemsOrder[i].getTitle().equals(title)) { //search the title name in the list
28             // Shift items in the array to remove the specified disc
29             while (i < qtyOrdered - 1) {
30                 itemsOrder[i] = itemsOrder[i + 1];
31                 i++;
32             }
33             itemsOrder[qtyOrdered - 1] = null; // Set the last element to null
34             qtyOrdered--; // Decrement the quantity ordered
35             System.out.println("The disc with title '" + title + "' has been removed.");
36             return;
37         }
38     }
39     System.out.println("Disc with title '" + title + "' not found in the cart.");
40 }
41
42 // Method to calculate the total cost of the items in the cart
43 public float totalCost() {
44     float total = 0.0f;
45     for (int i = 0; i < qtyOrdered; i++) {
46         // Accumulate the cost of each item in the cart
47         total += itemsOrder[i].getCost();
48     }
49     return total;
50 }
51 }
```

Figure 9 Code Cart (2)

6. Demo

- Khi **không** có hàm **removeDigitalVideoDisc()** trong main()

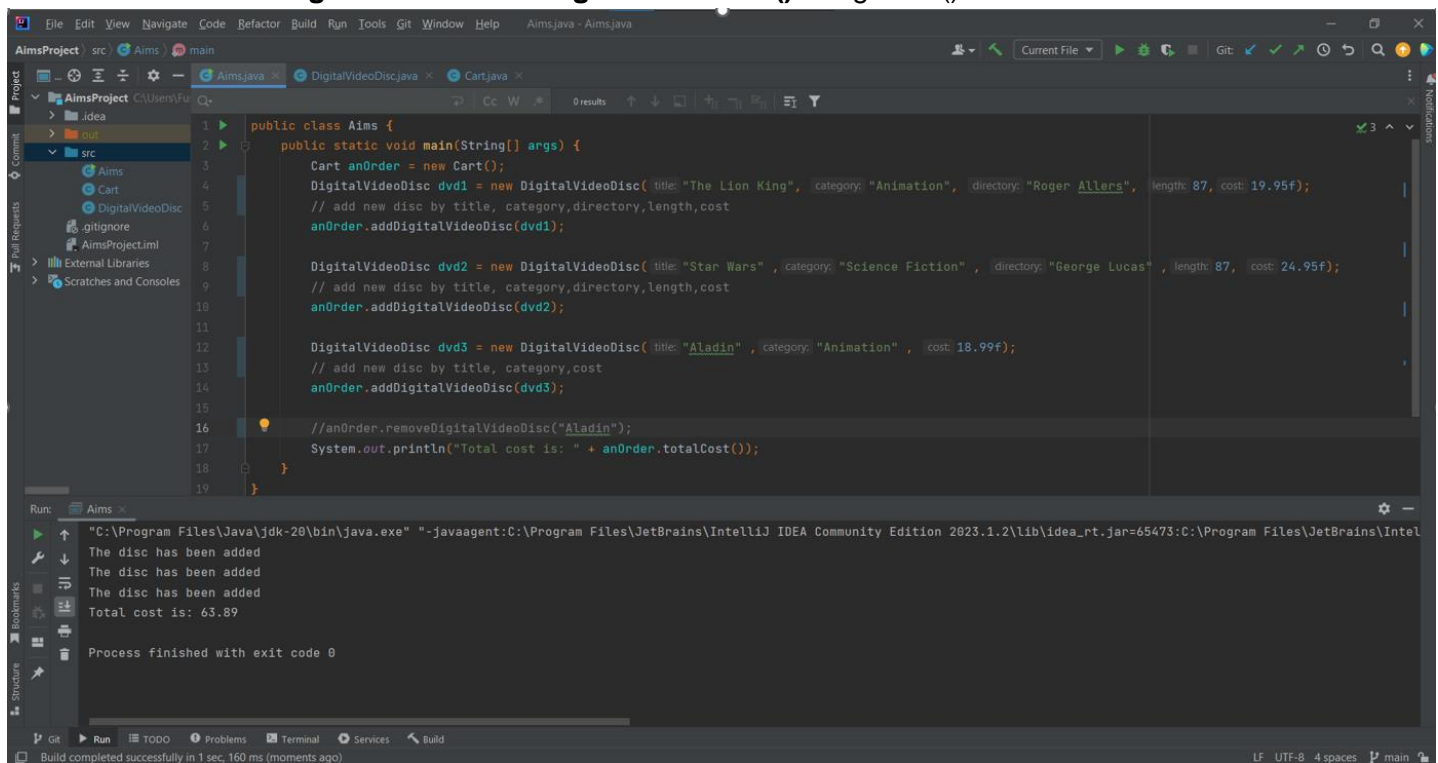
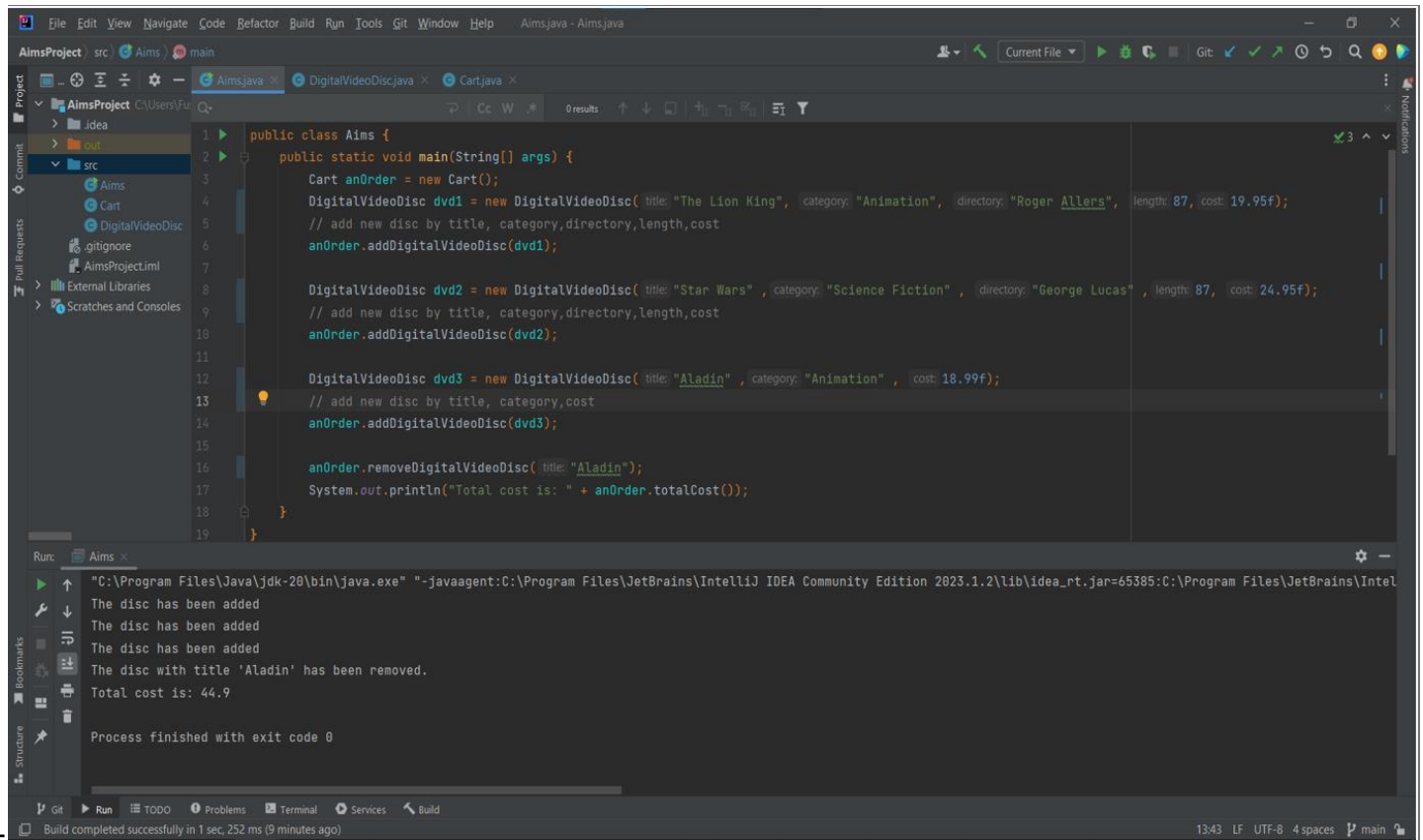


Figure 10 demo without removeDigitalVideoDisc

- Khi có hàm **removeDigitalVideoDisc()** trong main()

Figure 11 demo with `removeDigitalVideoDisc`