

Report Lab 04: Inheritance and Polymorphism

1. Additional requirements of AIMS

```

Book.java
AIMSPJect > src > hust > soict > dsai > aims > media > Book.java > Book > addAuthor(String)
1  package hust.soict.dsai.aims.media;
2
3  import java.util.ArrayList;
4
5  public class Book extends Media {
6      private ArrayList<String> authors;
7
8      // Constructor
9      public Book(String title, String category, float cost) {
10         super(title, category, cost); // Gọi constructor của lớp cha
11         this.authors = new ArrayList<>();
12     }
13
14     // Thêm tác giả
15     public void addAuthor(String author) {
16         if (!authors.contains(author)) {
17             authors.add(author);
18         } else {
19             System.out.println("Author " + author + " already exists in the list.");
20         }
21     }
22
23     // Xóa tác giả
24     public void removeAuthor(String author) {
25         if (authors.contains(author)) {
26             authors.remove(author);
27         } else {
28             System.out.println("Author " + author + " is not in the list.");
29         }
30     }
31
32     public ArrayList<String> getAuthors() {
33         return authors;
34     }
35
36     @Override
37     public String toString() {

```

```

2
3     // Xóa tác giả
4     public void removeAuthor(String author) {
5         if (authors.contains(author)) {
6             authors.remove(author);
7         } else {
8             System.out.println("Author " + author + " is not in the list.");
9         }
10    }
11
12    public ArrayList<String> getAuthors() {
13        return authors;
14    }
15
16    @Override
17    public String toString() {
18        return "Book - " + getTitle() + " - " + getCategory() + " - Authors: " + String.join(delimiter:", ", authors) +
19            " - Cost: $" + getCost();
20    }
21
22    @Override
23    public void displayInfo() {
24        System.out.println("Book - " + getTitle() + " - " + getCategory() + " - Authors: " + String.join(delimiter:", ", authors)
25            + " - Cost: $" + getCost());
26    }
27 }
28

```

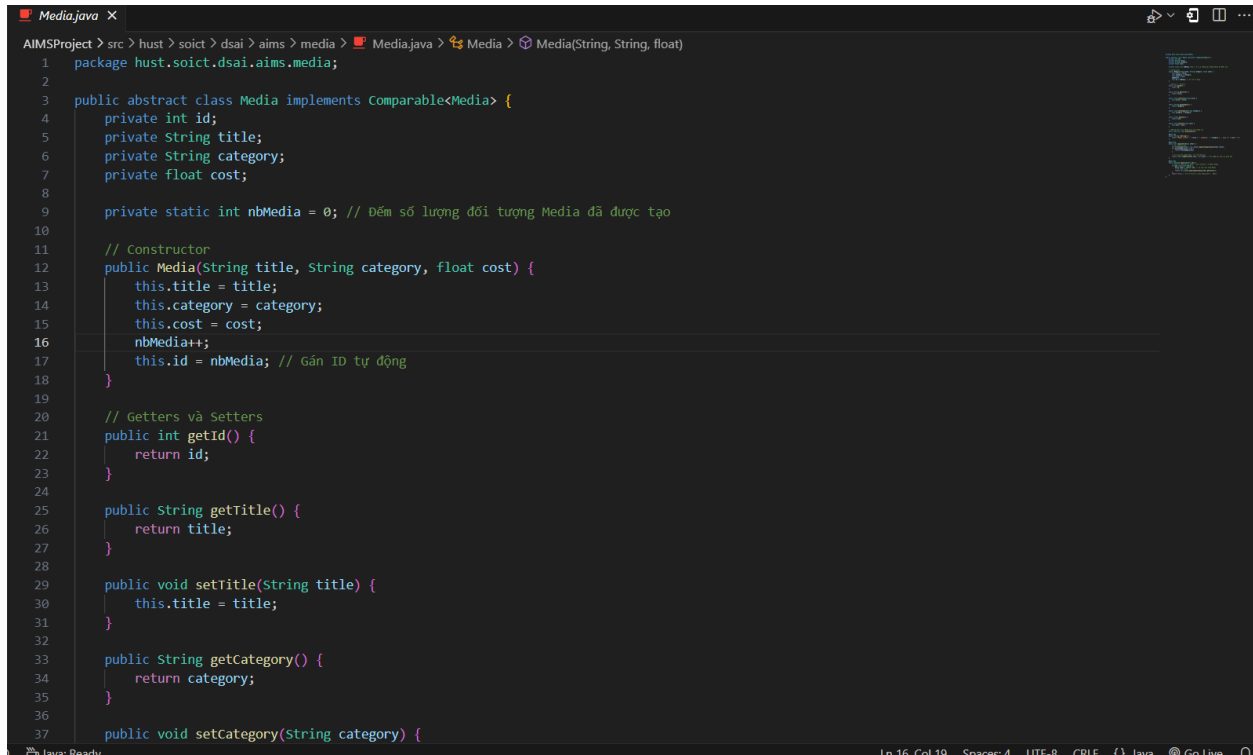
```

Book.java CompactDisc.java X
AIMSProject > src > hust > soict > dsai > aims > media > CompactDisc.java > CompactDisc > tracks
1 package hust.soict.dsai.aims.media;
2
3 import java.util.ArrayList;
4
5 public class CompactDisc extends Disc implements Playable {
6     private String artist; // Nghệ sĩ
7     private ArrayList<Track> tracks; // Danh sách bài hát
8
9     // Constructor
10    public CompactDisc(String title, String category, float cost, String artist, int length, String director) {
11        super(title, category, cost, length, director); // Gọi constructor của lớp cha (Disc)
12        this.artist = artist;
13        this.tracks = new ArrayList<>();
14    }
15
16    // Getter cho artist
17    public String getArtist() {
18        return artist;
19    }
20
21    // Thêm bài hát vào CD
22    public void addTrack(Track track) {
23        if (!tracks.contains(track)) {
24            tracks.add(track);
25            System.out.println("Added: " + track.getTitle());
26        } else {
27            System.out.println("Track " + track.getTitle() + " already exists in the CD.");
28        }
29    }
30
31    // Xóa bài hát khỏi CD
32    public void removeTrack(Track track) {
33        if (tracks.contains(track)) {
34            tracks.remove(track);
35        } else {
36            System.out.println("Track " + track.getTitle() + " is not in the CD.");
37        }
38    }
39
40    // Tính tổng độ dài của CD (tính tổng độ dài của tất cả các track)
41    public int getTotalLength() {
42        int totalLength = 0;
43        for (Track track : tracks) {
44            totalLength += track.getLength();
45        }
46        return totalLength;
47    }
48
49    @Override
50    public void play() {
51        System.out.println("Playing CompactDisc: " + getTitle());
52        System.out.println("Artist: " + getArtist());
53        System.out.println("CD length: " + getTotalLength() + " mins");
54
55        // Phát tất cả các bài hát
56        for (Track track : tracks) {
57            track.play();
58        }
59    }
60
61    @Override
62    public String toString() {
63        return "CompactDisc - " + getTitle() + " - " + getCategory() + " - Artist: " + getArtist() +
64            " - Director: " + getDirector() + " - Total Length: " + getTotalLength() + " mins - Cost: $"
65            + getCost();
66    }
67
68    // Hiển thị thông tin của CD
69    @Override
70    public void displayInfo() {
71        System.out.println("CompactDisc - " + getTitle() + " - " + getCategory() + " - Artist: " + artist
72            + " - Director: " + getDirector() + " - Total Length: " + getTotalLength() + " mins - Cost: $"
73            + getCost());
74    }

```

```
58     }
59 }
60
61 @Override
62 public String toString() {
63     return "CompactDisc - " + getTitle() + " - " + getCategory() + " - Artist: " + getArtist() +
64         " - Director: " + getDirector() + " - Total Length: " + getTotalLength() + " mins - Cost: $"
65         + getCost();
66 }
67
68 // Hiển thị thông tin của CD
69 @Override
70 public void displayInfo() {
71     System.out.println("CompactDisc - " + getTitle() + " - " + getCategory() + " - Artist: " + artist
72         + " - Director: " + getDirector() + " - Total Length: " + getTotalLength() + " mins - Cost: $"
73         + getCost());
74     System.out.println(x:"Tracks:");
75     for (Track track : tracks) {
76         System.out.println("\t" + track.toString());
77     }
78 }
79 }
80
```

2. Creating the abstract **Media** class



```
Media.java x
AIMSProject > src > hust > soict > dsai > aims > media > Media.java > Media > Media(String, String, float)
1 package hust.soict.dsai.aims.media;
2
3 public abstract class Media implements Comparable<Media> {
4     private int id;
5     private String title;
6     private String category;
7     private float cost;
8
9     private static int nbMedia = 0; // Đếm số lượng đối tượng Media đã được tạo
10
11     // Constructor
12     public Media(String title, String category, float cost) {
13         this.title = title;
14         this.category = category;
15         this.cost = cost;
16         nbMedia++;
17         this.id = nbMedia; // Gán ID tự động
18     }
19
20     // Getters và Setters
21     public int getId() {
22         return id;
23     }
24
25     public String getTitle() {
26         return title;
27     }
28
29     public void setTitle(String title) {
30         this.title = title;
31     }
32
33     public String getCategory() {
34         return category;
35     }
36
37     public void setCategory(String category) {
```

```

37     public void setCategory(String category) {
38         this.category = category;
39     }
40
41     public float getCost() {
42         return cost;
43     }
44
45     public void setCost(float cost) {
46         this.cost = cost;
47     }
48
49     // Phương thức trừu tượng hiển thị thông tin
50     public abstract void displayInfo();
51
52     @Override
53     public String toString() {
54         return "Media [Title: " + title + ", Category: " + category + ", Cost: $" + cost + "];"
55     }
56
57     @Override
58     public int compareTo(Media other) {
59         // So sánh tiêu đề
60         int titleComparison = this.title.compareToIgnoreCase(other.title);
61         if (titleComparison != 0) {
62             return titleComparison;
63         }
64
65         // Nếu tiêu đề giống nhau, so sánh theo giá
66         return Float.compare(other.cost, this.cost); // Đảo ngược để sắp xếp giảm dần
67     }
68
69     @Override
70     public boolean equals(Object obj) {
71         // Kiểm tra xem obj có phải là một instance của Media không

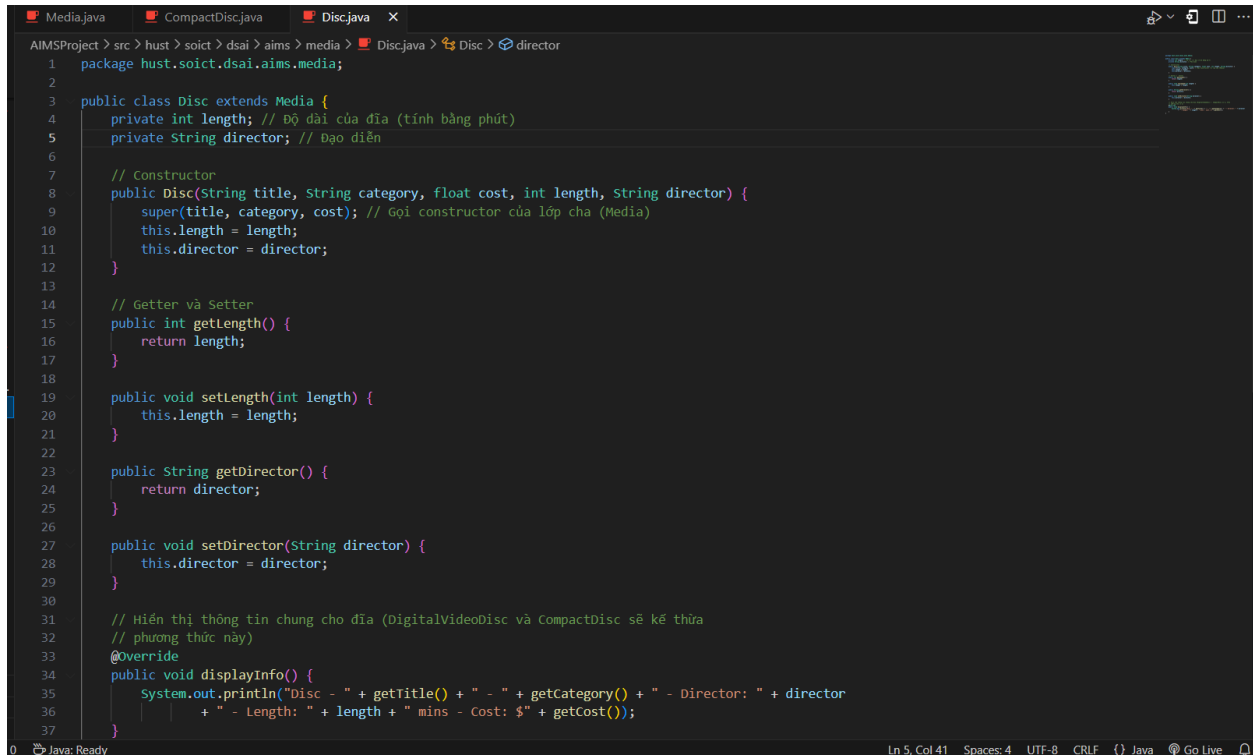
```

```

56
57     @Override
58     public int compareTo(Media other) {
59         // So sánh tiêu đề
60         int titleComparison = this.title.compareToIgnoreCase(other.title);
61         if (titleComparison != 0) {
62             return titleComparison;
63         }
64
65         // Nếu tiêu đề giống nhau, so sánh theo giá
66         return Float.compare(other.cost, this.cost); // Đảo ngược để sắp xếp giảm dần
67     }
68
69     @Override
70     public boolean equals(Object obj) {
71         // Kiểm tra xem obj có phải là một instance của Media không
72         if (obj instanceof Media) {
73             Media other = (Media) obj; // Ép kiểu obj sang Media
74             // So sánh title
75             return this.title.equalsIgnoreCase(other.getTitle());
76         }
77         return false; // Trả về false nếu obj không phải là Media
78     }
79 }
80

```

3. Creating the **CompactDisc** class



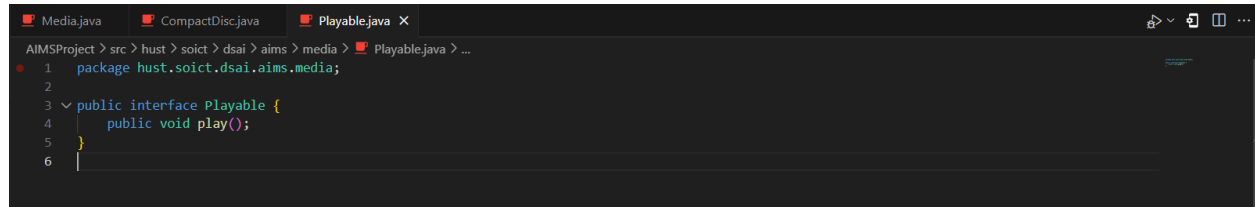
```
1 package hust.soict.dsai.aims.media;
2
3 public class Disc extends Media {
4     private int length; // Độ dài của đĩa (tính bằng phút)
5     private String director; // Đạo diễn
6
7     // Constructor
8     public Disc(String title, String category, float cost, int length, String director) {
9         super(title, category, cost); // Gọi constructor của lớp cha (Media)
10        this.length = length;
11        this.director = director;
12    }
13
14    // Getter và Setter
15    public int getLength() {
16        return length;
17    }
18
19    public void setLength(int length) {
20        this.length = length;
21    }
22
23    public String getDirector() {
24        return director;
25    }
26
27    public void setDirector(String director) {
28        this.director = director;
29    }
30
31    // Hiển thị thông tin chung cho đĩa (DigitalVideoDisc và CompactDisc sẽ kế thừa
32    // phương thức này)
33    @Override
34    public void displayInfo() {
35        System.out.println("Disc - " + getTitle() + " - " + getCategory() + " - Director: " + director
36        + " - Length: " + length + " mins - Cost: $" + getCost());
37    }
38 }
```

```

AIMSProject > src > hust > soict > dsai > aims > media > Track.java > Track
1  package hust.soict.dsai.aims.media;
2
3  public class Track implements Playable {
4      private String title;
5      private int length;
6
7      // Constructor
8      public Track(String title, int length) {
9          this.title = title;
10         this.length = length;
11     }
12
13     // Getter và Setter
14     public String getTitle() {
15         return title;
16     }
17
18     public int getLength() {
19         return length;
20     }
21
22     @Override
23     public void play() {
24         // Phát bài hát
25         System.out.println("Playing Track: " + title);
26         System.out.println("Track length: " + length + " mins");
27     }
28
29     // ToString
30     @Override
31     public String toString() {
32         return "Track: " + title + " - Length: " + length + " mins";
33     }
34
35     @Override
36     public boolean equals(Object obj) {
37         if (obj instanceof Track) {
38
39         }
40     }
41
42     // ToString
43     @Override
44     public String toString() {
45         return "Track: " + title + " - Length: " + length + " mins";
46     }
47
48     @Override
49     public boolean equals(Object obj) {
50         if (obj instanceof Track) {
51             Track other = (Track) obj;
52             return this.title.equalsIgnoreCase(other.getTitle()) && this.length == other.getLength(); // So sánh tiêu đề
53             // và độ dài
54         }
55         return false;
56     }
57 }

```

4. Create the Playable interface



The screenshot shows an IDE with three tabs: Media.java, CompactDisc.java, and Playable.java. The Playable.java tab is active, showing the following code:

```
AIMSProject > src > hust > soict > dsai > aims > media > Playable.java > ...
1 package hust.soict.dsai.aims.media;
2
3 public interface Playable {
4     public void play();
5 }
6 |
```

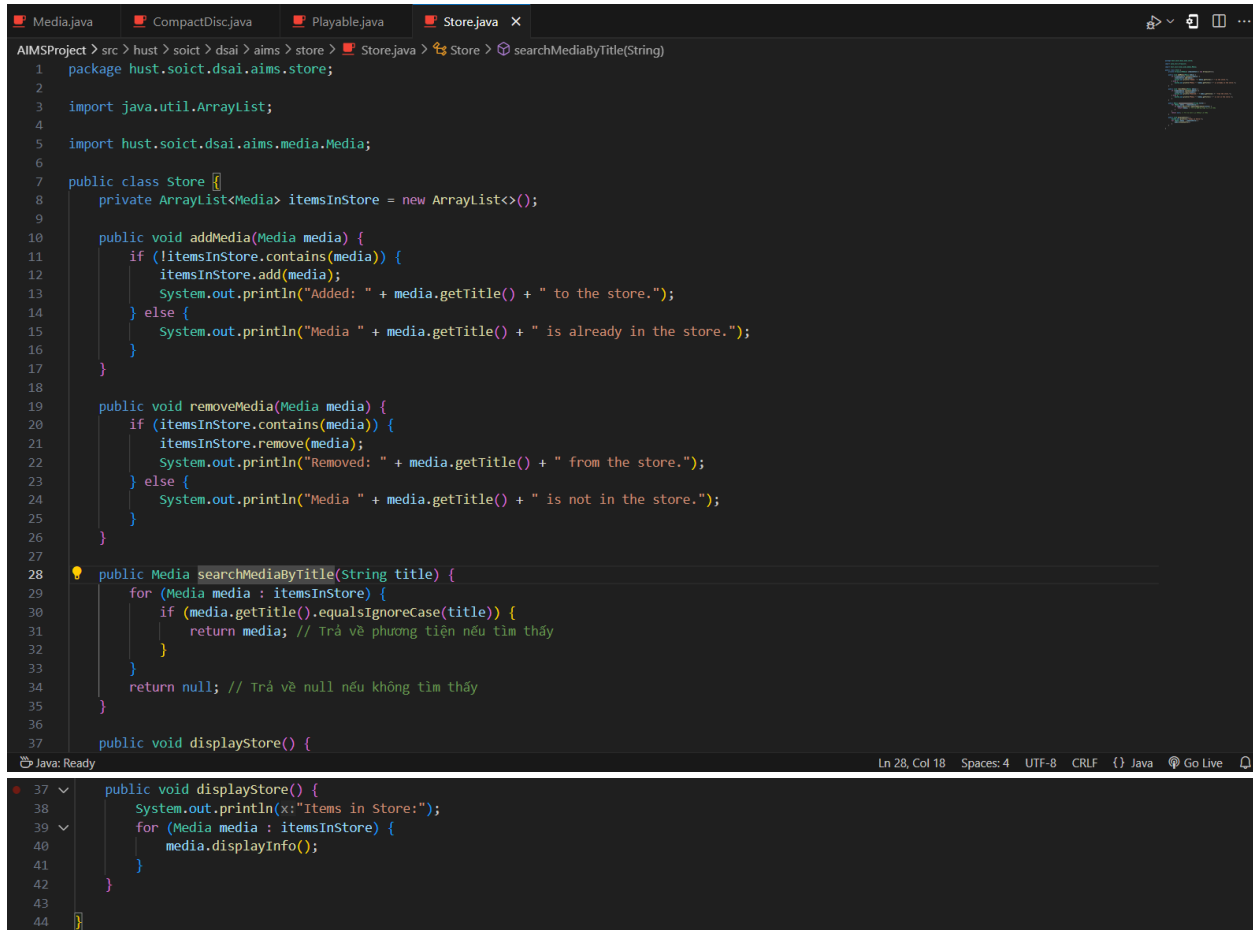

5. Update the `cart` class to work with `media`

```

Media.java CompactDisc.java Playable.java Cart.java x
AIMSProject > src > hust > dsai > aims > cart > Cart.java > Cart > getItemsOrdered()
1 package hust.soict.dsai.aims.cart;
2
3 import java.util.ArrayList;
4 import java.util.Collections;
5
6 import hust.soict.dsai.aims.media.CompareByCostThenTitle;
7 import hust.soict.dsai.aims.media.CompareByTitleThenCost;
8 import hust.soict.dsai.aims.media.Media;
9
10 public class Cart {
11     private ArrayList<Media> itemsOrdered = new ArrayList<>(); // Danh sách các phương tiện trong giỏ hàng
12
13     // add
14     public void addMedia(Media media) {
15         if (!itemsOrdered.contains(media)) {
16             itemsOrdered.add(media);
17             System.out.println("Added: " + media.getTitle());
18         } else {
19             System.out.println("Media " + media.getTitle() + " is already in the cart.");
20         }
21     }
22
23     // remove
24     public void removeMedia(Media media) {
25         if (itemsOrdered.contains(media)) {
26             itemsOrdered.remove(media);
27             System.out.println("Removed: " + media.getTitle());
28         } else {
29             System.out.println("Media " + media.getTitle() + " is not in the cart.");
30         }
31     }
32
33     // totalCost
34     public float totalCost() {
35         float total = 0;
36         for (Media media : itemsOrdered) {
37             total += media.getCost();
38         }
39         return Math.round(total * 100) / 100.0f; // làm tròn đến 2 chữ số thập phân
40     }
41
42     // display Cart
43     public void displayCart() {
44         System.out.println(x:"Items in Cart:");
45         for (Media media : itemsOrdered) {
46             media.displayInfo();
47         }
48         System.out.println("Total Cost: $" + totalCost());
49     }
50
51     // Sort Cart by Title then Cost
52     public void sortByTitleThenCost() {
53         Collections.sort(itemsOrdered, new CompareByTitleThenCost());
54         System.out.println(x:"\nCart sorted by Title then Cost:");
55         displayCart();
56     }
57
58     // Sort Cart by Cost then Title
59     public void sortByCostThenTitle() {
60         Collections.sort(itemsOrdered, new CompareByCostThenTitle());
61         System.out.println(x:"\nCart sorted by Cost then Title:");
62         displayCart();
63     }
64
65     // Phương thức trả về danh sách các phương tiện trong giỏ hàng
66     public ArrayList<Media> getItemsOrdered() {
67         return itemsOrdered;
68     }
69 }

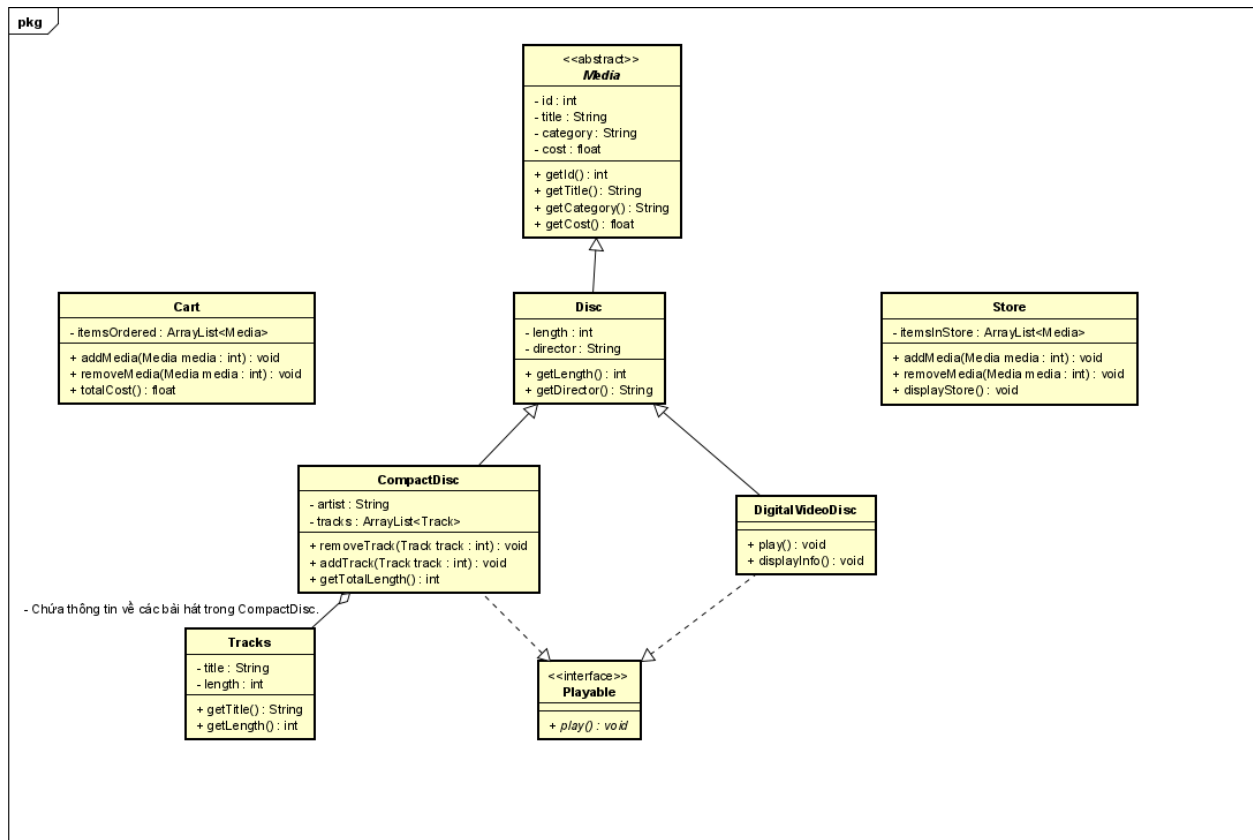
```

6. Update the `store` class to work with `Media`



```
Media.java CompactDisc.java Playable.java Store.java X
AIMSProject > src > hust > soict > dsai > aims > store > Store.java > Store > searchMediaByTitle(String)
1 package hust.soict.dsai.aims.store;
2
3 import java.util.ArrayList;
4
5 import hust.soict.dsai.aims.media.Media;
6
7 public class Store {
8     private ArrayList<Media> itemsInStore = new ArrayList<>();
9
10    public void addMedia(Media media) {
11        if (!itemsInStore.contains(media)) {
12            itemsInStore.add(media);
13            System.out.println("Added: " + media.getTitle() + " to the store.");
14        } else {
15            System.out.println("Media " + media.getTitle() + " is already in the store.");
16        }
17    }
18
19    public void removeMedia(Media media) {
20        if (itemsInStore.contains(media)) {
21            itemsInStore.remove(media);
22            System.out.println("Removed: " + media.getTitle() + " from the store.");
23        } else {
24            System.out.println("Media " + media.getTitle() + " is not in the store.");
25        }
26    }
27
28    public Media searchMediaByTitle(String title) {
29        for (Media media : itemsInStore) {
30            if (media.getTitle().equalsIgnoreCase(title)) {
31                return media; // Trả về phương tiện nếu tìm thấy
32            }
33        }
34        return null; // Trả về null nếu không tìm thấy
35    }
36
37    public void displayStore() {
38
39        public void displayStore() {
40            System.out.println(x:"Items in Store:");
41            for (Media media : itemsInStore) {
42                media.displayInfo();
43            }
44        }
45    }
46}
```

7. Constructors of whole classes and parent classes



8. Unique item in a list

```

@Override
public boolean equals(Object obj) {
    // Kiểm tra xem obj có phải là một instance của Media không
    if (obj instanceof Media) {
        Media other = (Media) obj; // Ép kiểu obj sang Media
        // So sánh title
        return this.title.equalsIgnoreCase(other.getTitle());
    }
    return false; // Trả về false nếu obj không phải là Media
}

```

```

@Override
public boolean equals(Object obj) {
    if (obj instanceof Track) {
        Track other = (Track) obj;
        return this.title.equalsIgnoreCase(other.getTitle()) && this.length == other.getLength(); // So sánh tiêu đề
        // và độ dài
    }
    return false;
}

```

9. Polymorphism with toString() method

```

@Override
public String toString() {
    return "Media [Title: " + title + ", Category: " + category + ", Cost: $" + cost + "];"
}

```

```

@Override
public String toString() {
    return "DVD - " + getTitle() + " - " + getCategory() + " - Director: " + getDirector() +
        " - Length: " + getLength() + " mins - Cost: $" + getCost();
}

```

```

@Override
public String toString() {
    return "Book - " + getTitle() + " - " + getCategory() + " - Authors: " + String.join(delimiter:", ", authors) +
        " - Cost: $" + getCost();
}

```

```

@Override
public String toString() {
    return "CompactDisc - " + getTitle() + " - " + getCategory() + " - Artist: " + getArtist() +
        " - Director: " + getDirector() + " - Total Length: " + getTotalLength() + " mins - Cost: $"
        + getCost();
}

```

10. Sort media in the cart

```

@Override
public int compareTo(Media other) {
    // So sánh tiêu đề
    int titleComparison = this.title.compareToIgnoreCase(other.title);
    if (titleComparison != 0) {
        return titleComparison;
    }

    // Nếu tiêu đề giống nhau, so sánh theo giá
    return Float.compare(other.cost, this.cost); // Đảo ngược để sắp xếp giảm dần
}

```

```

@Override
public int compareTo(Media other) {
    if (other instanceof DigitalVideoDisc) {
        DigitalVideoDisc dvd = (DigitalVideoDisc) other;
        int titleComparision = this.getTitle().compareToIgnoreCase(dvd.getTitle());
        if (titleComparision != 0) {
            return titleComparision;
        }
        // so sánh độ dài
        int lengthComparision = Integer.compare(dvd.getLength(), this.getLength());
        if (lengthComparision != 0) {
            return lengthComparision;
        }
        // so sánh giá
        return Float.compare(dvd.getCost(), this.getCost());
    }
    return 0;
}

```

Media.java DigitalVideoDisc.java CompareByCostThenTitle.java X

AIMSProject > src > hust > soict > dsai > aims > media > CompareByCostThenTitle.java > ...

```

1  package hust.soict.dsai.aims.media;
2
3  import java.util.Comparator;
4
5  public class CompareByCostThenTitle implements Comparator<Media> {
6
7      @Override
8      public int compare(Media media1, Media media2) {
9          int costComparision = Float.compare(media1.getCost(), media2.getCost());
10         if (costComparision != 0) {
11             return costComparision;
12         }
13         // Nếu giá bằng nhau
14         return media1.getTitle().compareToIgnoreCase(media2.getTitle());
15     }
16 }

```

Media.java DigitalVideoDisc.java CompareByCostThenTitle.java CompareByTitleThenCost.java X

AIMSProject > src > hust > soict > dsai > aims > media > CompareByTitleThenCost.java > ...

```

1  package hust.soict.dsai.aims.media;
2
3  import java.util.Comparator;
4
5  public class CompareByTitleThenCost implements Comparator<Media> {
6
7      @Override
8      public int compare(Media media1, Media media2) {
9          // So sánh tiêu đề
10         int titleComparision = media1.getTitle().compareToIgnoreCase(media2.getTitle());
11         if (titleComparision != 0) {
12             return titleComparision;
13         }
14         // Nếu tiêu đề giống nhau
15         return Float.compare(media1.getCost(), media2.getCost()); // Đảo ngược để sắp xếp giảm dần
16     }
17 }

```

11. Create a complete console application in the Aims class

```

Aims.java X
AIMSProject > src > hust > soict > dsai > aims > Aims.java > ...
1 package hust.soict.dsai.aims;
2
3 import hust.soict.dsai.aims.cart.Cart;
4 import hust.soict.dsai.aims.media.*;
5 import hust.soict.dsai.aims.store.Store;
6
7 import java.util.Collections;
8 import java.util.Scanner;
9
10 public class Aims {
11     Run | Debug
12     public static void main(String[] args) {
13         Scanner scanner = new Scanner(System.in);
14
15         // Khởi tạo kho và giỏ hàng
16         Store store = new Store();
17         Cart cart = new Cart();
18
19         // Thêm vài phương tiện mẫu vào kho
20         store.addMedia(new Book(title:"The Hobbit", category:"Fantasy", cost:10.99f));
21         store.addMedia(new DigitalVideoDisc(title:"The Lion King", category:"Animation", director:"Roger Allers", length:87, cost:19.95f));
22         store.addMedia(new CompactDisc(title:"Thriller", category:"Pop", cost:15.99f, artist:"Michael Jackson", length:42, director:"Quincy Jones");
23         store.addMedia(new Book(title:"1984", category:"Dystopian", cost:15.99f));
24
25         // Menu chính
26         int choice = -1;
27         while (choice != 0) {
28             showMenu();
29             choice = scanner.nextInt();
30             scanner.nextLine(); // đọc dòng dư
31             switch (choice) {
32                 case 1: // View store
33                     storeMenu(store, cart, scanner);
34                     break;
35                 case 2: // Update Store
36                     updateStore(store, scanner);
37                     break;
38                 case 3: // See current cart
39                     cartMenu(cart, scanner);
40                     break;
41                 case 0:
42                     System.out.println(x:"Exiting...");
43                     break;
44                 default:
45                     System.out.println(x:"Invalid option. Please try again.");
46             }
47         }
48
49         // Menu chính
50         public static void showMenu() {
51             System.out.println(x:"AIMS: ");
52             System.out.println(x:"-----");
53             System.out.println(x:"1. View store");
54             System.out.println(x:"2. Update store");
55             System.out.println(x:"3. See current cart");
56             System.out.println(x:"0. Exit");
57             System.out.println(x:"-----");
58             System.out.print(s:"Please choose a number: 0-1-2-3: ");
59         }
60
61         // Menu quản lý kho
62         public static void storeMenu(Store store, Cart cart, Scanner scanner) {
63             int choice = -1;
64             while (choice != 0) {
65                 System.out.println(x:"Options: ");
66                 System.out.println(x:"-----");
67                 System.out.println(x:"1. See a media's details");
68                 System.out.println(x:"2. Add a media to cart");
69             }
70         }
71     }
72 }

```

```

68     System.out.println(x:"2. Add a media to cart");
69     System.out.println(x:"3. Play a media");
70     System.out.println(x:"4. See current cart");
71     System.out.println(x:"0. Back");
72     System.out.println(x:"-----");
73     System.out.print(s:"Please choose a number: 0-1-2-3-4: ");
74     choice = scanner.nextInt();
75     scanner.nextLine(); // Đọc dòng dư
76     switch (choice) {
77         case 1: // See media details
78             viewMediaDetails(store, scanner, cart);
79             break;
80         case 2: // Add media to cart
81             addMediaToCart(store, cart, scanner);
82             break;
83         case 3: // Play media
84             playMedia(store, scanner);
85             break;
86         case 4: // See current cart
87             cart.displayCart();
88             break;
89         case 0:
90             return;
91         default:
92             System.out.println(x:"Invalid option. Please try again.");
93     }
94 }
95 }
96
97 // Xem chi tiết phương tiện
98 public static void viewMediaDetails(Store store, Scanner scanner, Cart cart) {
99     System.out.print(s:"Enter the title of the media: ");
100     String title = scanner.nextLine();
101     Media media = store.searchMediaByTitle(title);
102     if (media != null) {
103         if (media != null) {
104             media.displayInfo();
105             mediaDetailsMenu(scanner, media, cart);
106         } else {
107             System.out.println(x:"Media not found!");
108         }
109     }
110
111 // Menu chi tiết phương tiện
112 public static void mediaDetailsMenu(Scanner scanner, Media media, Cart cart) {
113     int choice = -1;
114     while (choice != 0) {
115         System.out.println(x:"Options: ");
116         System.out.println(x:"-----");
117         System.out.println(x:"1. Add to cart");
118         System.out.println(x:"2. Play");
119         System.out.println(x:"0. Back");
120         System.out.println(x:"-----");
121         System.out.print(s:"Please choose a number: 0-1-2: ");
122         choice = scanner.nextInt();
123         scanner.nextLine(); // Đọc dòng dư
124         switch (choice) {
125             case 1:
126                 addMediaToCartHelper(cart, media);
127                 break;
128             case 2:
129                 if (media instanceof Playable) {
130                     ((Playable) media).play();
131                 } else {
132                     System.out.println(x:"This media cannot be played.");
133                 }
134                 break;
135             case 0:
136                 return;

```

```

136         default:
137             System.out.println(x:"Invalid option. Please try again.");
138         }
139     }
140 }
141
142 // Thêm phương tiện vào giỏ hàng
143 public static void addMediaToCartHelper(Cart cart, Media media) {
144     cart.addMedia(media); // Thêm phương tiện vào giỏ hàng thực tế
145     System.out.println("Media " + media.getTitle() + " added to cart.");
146 }
147
148 // Thêm phương tiện vào giỏ hàng
149 public static void addMediaToCart(Store store, Cart cart, Scanner scanner) {
150     System.out.print(s:"Enter the title of the media to add to cart: ");
151     String title = scanner.nextLine();
152     Media media = store.searchMediaByTitle(title);
153     if (media != null) {
154         cart.addMedia(media);
155     } else {
156         System.out.println(x:"Media not found!");
157     }
158 }
159
160 // Phát phương tiện
161 public static void playMedia(Store store, Scanner scanner) {
162     System.out.print(s:"Enter the title of the media to play: ");
163     String title = scanner.nextLine();
164     Media media = store.searchMediaByTitle(title);
165     if (media != null) {
166         if (media instanceof Playable) {
167             ((Playable) media).play();
168         } else {
169             System.out.println(x:"This media cannot be played.");
170         }
171     } else {
172         System.out.println(x:"Media not found!");
173     }
174 }
175
176 // Cập nhật kho (thêm/xóa phương tiện)
177 public static void updateStore(Store store, Scanner scanner) {
178     System.out.println(x:"Options: ");
179     System.out.println(x:"-----");
180     System.out.println(x:"1. Add a media");
181     System.out.println(x:"2. Remove a media");
182     System.out.println(x:"0. Back");
183     System.out.println(x:"-----");
184     System.out.print(s:"Please choose a number: 0-1-2: ");
185     int choice = scanner.nextInt();
186     scanner.nextLine(); // Đọc dòng dư
187     switch (choice) {
188         case 1: // Add media
189             System.out.print(s:"Enter media type (Book/DVD/CD): ");
190             String mediaType = scanner.nextLine();
191             if (mediaType.equalsIgnoreCase(anotherString:"Book")) {
192                 System.out.print(s:"Enter title, category, cost: ");
193                 String title = scanner.nextLine();
194                 String category = scanner.nextLine();
195                 float cost = scanner.nextFloat();
196                 store.addMedia(new Book(title, category, cost));
197             } else if (mediaType.equalsIgnoreCase(anotherString:"DVD")) {
198                 System.out.print(s:"Enter title, category, director, length, cost: ");
199                 String titleDvd = scanner.nextLine();
200                 String categoryDvd = scanner.nextLine();
201                 String director = scanner.nextLine();
202                 int length = scanner.nextInt();
203                 float costDvd = scanner.nextFloat();
204                 store.addMedia(new DigitalVideoDisc(titleDvd, categoryDvd, director, length, costDvd));
205             } else if (mediaType.equalsIgnoreCase(anotherString:"CD")) {

```



```

206         System.out.print(s:"Enter title, category, artist, director, length, cost: ");
207         String titleCd = scanner.nextLine();
208         String categoryCd = scanner.nextLine();
209         String artist = scanner.nextLine();
210         String directorCd = scanner.nextLine();
211         int lengthCd = scanner.nextInt();
212         float costCd = scanner.nextFloat();
213         store.addMedia(new CompactDisc(titleCd, categoryCd, costCd, artist, lengthCd, directorCd));
214     } else {
215         System.out.println(x:"Invalid media type!");
216     }
217     break;
218 case 2: // Remove media
219     System.out.print(s:"Enter the title of the media to remove: ");
220     String titleToRemove = scanner.nextLine();
221     Media mediaToRemove = store.searchMediaByTitle(titleToRemove);
222     if (mediaToRemove != null) {
223         store.removeMedia(mediaToRemove);
224     } else {
225         System.out.println(x:"Media not found!");
226     }
227     break;
228 case 0:
229     return;
230 default:
231     System.out.println(x:"Invalid option.");
232 }
233 }
234
235 // Menu giỏ hàng
236 public static void cartMenu(Cart cart, Scanner scanner) {
237     int choice = -1;
238     while (choice != 0) {
239         System.out.println(x:"Options: ");
240         System.out.println(x:"-----");
241
242         System.out.println(x:"2. Sort medias in cart");
243         System.out.println(x:"3. Remove media from cart");
244         System.out.println(x:"4. Play a media");
245         System.out.println(x:"5. Place order");
246         System.out.println(x:"0. Back");
247         System.out.println(x:"-----");
248         System.out.print(s:"Please choose a number: 0-1-2-3-4-5: ");
249         choice = scanner.nextInt();
250         scanner.nextLine(); // Đọc dòng dư
251         switch (choice) {
252             case 1:
253                 filterCart(cart, scanner);
254                 break;
255             case 2:
256                 sortCart(cart, scanner);
257                 break;
258             case 3:
259                 removeMediaFromCart(cart, scanner);
260                 break;
261             case 4:
262                 playMediaFromCart(cart, scanner);
263                 break;
264             case 5:
265                 placeOrder(cart);
266                 break;
267             case 0:
268                 return;
269             default:
270                 System.out.println(x:"Invalid option.");
271         }
272     }
273 }
274
275 // Lưu giỏ hàng (theo ID hoặc tiêu đề)

```

```

275 // Lọc giỏ hàng (theo ID hoặc tiêu đề)
276 public static void filterCart(Cart cart, Scanner scanner) {
277     System.out.println(x:"Filter by: ");
278     System.out.println(x:"1. ID");
279     System.out.println(x:"2. Title");
280     int filterChoice = scanner.nextInt();
281     scanner.nextLine(); // Đọc dòng dư
282     switch (filterChoice) {
283     case 1:
284         System.out.print(s:"Enter media ID: ");
285         int id = scanner.nextInt();
286         Media media = cart.getItemsOrdered().stream()
287             .filter(m -> m.getId() == id)
288             .findFirst()
289             .orElse(other:null);
290         if (media != null) {
291             media.displayInfo();
292         } else {
293             System.out.println(x:"Media not found.");
294         }
295         break;
296     case 2:
297         System.out.print(s:"Enter media title: ");
298         String title = scanner.nextLine();
299         cart.getItemsOrdered().stream()
300             .filter(m -> m.getTitle().contains(title))
301             .forEach(Media::displayInfo);
302         break;
303     default:
304         System.out.println(x:"Invalid choice.");
305     }
306 }
307
308 // Sắp xếp giỏ hàng (theo tiêu đề hoặc giá)
309 public static void sortCart(Cart cart, Scanner scanner) {

```

```

309 public static void sortCart(Cart cart, Scanner scanner) {
310     System.out.println(x:"Sort by: ");
311     System.out.println(x:"1. Title");
312     System.out.println(x:"2. Cost");
313     int sortChoice = scanner.nextInt();
314     scanner.nextLine(); // Đọc dòng dư
315     switch (sortChoice) {
316     case 1:
317         Collections.sort(cart.getItemsOrdered(), new CompareByTitleThenCost());
318         break;
319     case 2:
320         Collections.sort(cart.getItemsOrdered(), new CompareByCostThenTitle());
321         break;
322     default:
323         System.out.println(x:"Invalid choice.");
324         return;
325     }
326     cart.displayCart();
327 }
328
329 // Xóa phương tiện khỏi giỏ hàng
330 public static void removeMediaFromCart(Cart cart, Scanner scanner) {
331     System.out.print(s:"Enter media title to remove: ");
332     String title = scanner.nextLine();
333     Media mediaToRemove = cart.getItemsOrdered().stream()
334         .filter(m -> m.getTitle().equalsIgnoreCase(title))
335         .findFirst()
336         .orElse(other:null);
337     if (mediaToRemove != null) {
338         cart.removeMedia(mediaToRemove);
339     } else {
340         System.out.println(x:"Media not found.");
341     }
342 }
343
344 // Phát phương tiện từ giỏ hàng

```

```

344 // Phát phương tiện từ giỏ hàng
345 public static void playMediaFromCart(Cart cart, Scanner scanner) {
346     System.out.print(s:"Enter media title to play: ");
347     String title = scanner.nextLine();
348     Media mediaToPlay = cart.getItemsOrdered().stream()
349         .filter(m -> m.getTitle().equalsIgnoreCase(title))
350         .findFirst()
351         .orElse(other:null);
352     if (mediaToPlay != null && mediaToPlay instanceof Playable) {
353         ((Playable) mediaToPlay).play();
354     } else {
355         System.out.println(x:"Media not found or cannot be played.");
356     }
357 }
358
359 // Đặt hàng
360 public static void placeOrder(Cart cart) {
361     System.out.println("Order placed! Total Cost: $" + cart.totalCost());
362     cart.getItemsOrdered().clear(); // Empty the cart after placing the order
363     System.out.println(x:"Cart has been cleared.");
364 }
365 }
366

```

12. Question

1. Lớp nào nên triển khai interface Comparable?

Bất kỳ lớp nào cần được sắp xếp dựa trên các thuộc tính của nó nên triển khai interface Comparable. Ví dụ, nếu bạn có một lớp gọi là Item đại diện cho các mục trong giỏ hàng, nó nên triển khai Comparable<Item>.

```
public class Item implements Comparable<Item> {
```

```
    private String title;
```

```
    private double cost;
```

```
// Constructor, getters, và setters bị bỏ qua để tiết kiệm không gian
```

```
@Override
```

```
public int compareTo(Item other) {
```

```
    // Logic so sánh sẽ được thực hiện ở đây
```

```
}
```

```
}
```

2. Trong những lớp đó, bạn nên triển khai phương thức `compareTo()` như thế nào để phản ánh thứ tự mà chúng ta muốn?

Phương thức `compareTo()` phải trả về một số nguyên cho biết thứ tự của đối tượng hiện tại so với một đối tượng khác:

- Một số nguyên âm nếu đối tượng này nhỏ hơn đối tượng đã chỉ định.
- Zero nếu đối tượng này bằng với đối tượng đã chỉ định.
- Một số nguyên dương nếu đối tượng này lớn hơn đối tượng đã chỉ định.

Nếu bạn muốn sắp xếp các mục trước tiên theo tiêu đề và sau đó theo chi phí, việc triển khai của bạn có thể như sau:

@Override

```
public int compareTo(Item other) {  
    int titleComparison = this.title.compareTo(other.title);  
    if (titleComparison != 0) {  
        return titleComparison; // Sắp xếp theo tiêu đề  
    }  
    return Double.compare(this.cost, other.cost); // Sắp xếp theo chi phí  
}
```

3. Chúng ta có thể có hai quy tắc sắp xếp của mục (theo tiêu đề rồi đến chi phí và theo chi phí rồi đến tiêu đề) nếu chúng ta sử dụng cách tiếp cận interface `Comparable` không?

Sử dụng interface `Comparable`, bạn chỉ có thể định nghĩa một thứ tự tự nhiên cho mỗi lớp. Tuy nhiên, nếu bạn cần nhiều quy tắc sắp xếp (như sắp xếp theo tiêu đề rồi đến chi phí và theo chi phí rồi đến tiêu đề), bạn thường sẽ sử dụng interface `Comparator` thay thế.

Ví dụ:

```
Comparator<Item> byCostThenTitle = Comparator.comparingDouble(Item::getCost)
        .thenComparing(Item::getTitle);
```

4. Giả sử các DVD có quy tắc sắp xếp khác với các loại phương tiện khác, đó là theo tiêu đề, sau đó là độ dài giảm dần, rồi đến chi phí. Bạn sẽ sửa đổi mã của mình như thế nào để cho phép điều này?

Nếu các DVD có quy tắc sắp xếp cụ thể (theo tiêu đề, sau đó là độ dài giảm dần, rồi đến chi phí), bạn có thể đạt được điều này bằng cách tạo một lớp riêng cho DVD mà triển khai Comparable:

```
public class DVD implements Comparable<DVD> {
    private String title;
    private double length; // Giả sử độ dài tính bằng phút
    private float cost;

    @Override
    public int compareTo(Media other) {
        if (other instanceof DigitalVideoDisc) {
            DigitalVideoDisc dvd = (DigitalVideoDisc) other;
            int titleComparision = this.getTitle().compareToIgnoreCase(dvd.getTitle());
            if (titleComparision != 0) {
                return titleComparision;
            }
            // so sánh độ dài
```

```
int lengthComparision = Integer.compare(dvd.getLength(), this.getLength());  
if (lengthComparision != 0)  
    return lengthComparision;  
    // so sánh giá  
    return Float.compare(dvd.getCost(), this.getCost());  
}  
return 0;  
}  
}
```