

Báo cáo thực hành Lập trình hướng đối tượng Lab 4
Họ và tên: Nguyễn Lương Hoàng Tùng
Lớp: 744520
MSSV: 20226129

I. Create Book class

```
1 package hust.soict.dsai.aims.media;
2
3 import java.util.ArrayList;
4 import java.util.List;
5
6 public class Book {
7     private int id;
8     private String title;
9     private String category;
10    private float cost;
11    private List<String> authors = new ArrayList<String>();
12
13    public Book(int id, String title, String category, float cost, List<String> authors) {
14
15    }
16
17    public Book(int id, String title, String category, float cost) {
18
19    }
20
21    public List<String> getAuthors() {
22        return authors;
23    }
24
25    public void setAuthors(List<String> authors) {
26        this.authors = authors;
27    }
28
29    public void addAuthor(String authorName) {
30        if(authors.contains(authorName)) {
31            System.out.println(authorName + " is already in the list");
32        } else {
33            authors.add(authorName);
34            System.out.println("Author added");
35        }
36    }
37
38    public void removeAuthor(String authorName) {
39        if(authors.contains(authorName)) {
40            authors.remove(authorName);
41        } else {
42            System.out.println(authorName + " is not in the list");
43        }
44    }
45 }
```

II. Create Media class

1. Media class

```
1  package hust.soict.dsai.aims.media;
2
3  import java.util.*;
4
5  public abstract class Media {
6      private int id;
7      private String title;
8      private String category;
9      private float cost;
10
11     public Media(int id, String title, String category, float cost) {
12         this.setId(id);
13         this.setTitle(title);
14         this.setCategory(category);
15         this.setCost(cost);
16     }
17
18     public Media(String title){
19         this.setTitle(title);
20     }
21
22     public Media(String title, String category, float cost){
23         this(title);
24         this.setCategory(category);
25         this.setCost(cost);
26     }
27
28     public int getId() {
29         return id;
30     }
31     public void setId(int id) {
32         this.id = id;
33     }
34     public String getTitle() {
35         return title;
36     }
37     public void setTitle(String title) {
38         this.title = title;
39     }
40     public String getCategory() {
41         return category;
42     }
43     public void setCategory(String category) {
44         this.category = category;
45     }
46     public float getCost() {
47         return cost;
48     }
49     public void setCost(float cost) {
50         this.cost = cost;
51     }
52
53     public boolean equals(Media obj) {
54         return this.getTitle().equals(obj.getTitle());
55     }
56
57     @Override
58     public String toString() {
59         return "Media - " + this.getTitle() + " - " + this.getCategory() + " - " + this.getCost();
60     }
61 }
```

2. Fix Book class

```
1 package hust.soict.dsai.aims.media;
2
3 import java.util.*;
4
5 public class Book extends Media {
6     private List<String> authors = new ArrayList<String>();
7
8     public Book(int id, String title, String category, float cost, List<String> authors) {
9         super(id, title, category, cost);
10        this.authors = authors;
11    }
12
13    public Book(int id, String title, String category, float cost) {
14        super(id, title, category, cost);
15    }
16
17    public List<String> getAuthors() {
18        return authors;
19    }
20
21    public void setAuthors(List<String> authors) {
22        this.authors = authors;
23    }
24
25    public void addAuthor(String authorName) {
26        if(authors.contains(authorName)) {
27            System.out.println(authorName + " is already in the list");
28        } else {
29            authors.add(authorName);
30            System.out.println("Author added");
31        }
32    }
33
34    public void removeAuthor(String authorName) {
35        if(authors.contains(authorName)) {
36            authors.remove(authorName);
37        } else {
38            System.out.println(authorName + " is not in the list");
39        }
40    }
41
42    @Override
43    public String toString() {
44        return "Book - " + this.getTitle() + " - " + this.getCategory() + " - " + this.getCost() + " - " + this.getAuthors();
45    }
46 }
```

III. Create CompactDisc class

1. Create Disc class extends Media class

```
1 package hust.soict.dsai.aims.media;
2
3 public class Disc extends Media {
4     private String director;
5     private float length;
6
7     public Disc(int id, String title, String category, float cost, String director, float length) {
8         super(id, title, category, cost);
9         this.director = director;
10        this.length = length;
11    }
12
13    public Disc(int id, String title, String category, float cost) {
14        super(id, title, category, cost);
15    }
16
17    public String getDirector() {
18        return director;
19    }
20
21    public void setDirector(String director) {
22        this.director = director;
23    }
24
25    public float getLength() {
26        return length;
27    }
28
29    public void setLength(float length) {
30        this.length = length;
31    }
32
33    @Override
34    public String toString() {
35        return "Disc - " + this.getTitle() + " - " + this.getCategory() + " - " + this.getCost() + " - " + this.getDirector() + " - " + this.getLength();
36    }
37 }
```

2. Fix DigitalVideoDisc class

```
1 package hust.soict.dsai.aims.media;
2
3 public class DigitalVideoDisc extends Disc {
4     private static int nbDigitalVideoDiscs = 0;
5
6     public DigitalVideoDisc(int id, String title, String category, float cost, String director, int length) {
7         super(id, title, category, cost, director, length);
8         nbDigitalVideoDiscs++;
9     }
10
11    public DigitalVideoDisc(int id, String title, String category, float cost) {
12        super(id, title, category, cost);
13        nbDigitalVideoDiscs++;
14    }
15
16    public void play() {
17        System.out.println("Playing DVD: " + this.getTitle());
18        System.out.println("DVD length: " + this.getLength());
19    }
20 }
```

3. Create Track class




```
1 package hust.soict.dsai.aims.media;
2
3 public class Track {
4     private String title;
5     private float length;
6
7     public float getLength() {
8         return length;
9     }
10
11     public String getTitle() {
12         return title;
13     }
14
15     public Track(String title, float length) {
16         this.title = title;
17         this.length = length;
18     }
19
20     public void play() {
21         System.out.println("Playing track: " + this.getTitle());
22         System.out.println("Track length: " + this.getLength());
23     }
24
25     public boolean equals(Track track) {
26         return this.getTitle().equals(track.getTitle());
27     }
28 }
```

4. Create CompactDisc class extends Disc class

```
1 package hust.soict.dsai.aims.media;
2
3 import java.util.ArrayList;
4
5 public class CompactDisc extends Disc {
6
7     private String artist;
8     private ArrayList<Track> tracks;
9
10    public CompactDisc(int id, String title, String category,
11        float cost, String artist, ArrayList<Track> tracks) {
12        super(id, title, category, cost);
13        this.tracks = tracks;
14        this.artist = artist;
15        this.setLength(getLength());
16    }
17
18    public CompactDisc(int id, String title, String category, float cost) {
19        super(id, title, category, cost);
20    }
21
22    public String getArtist() {
23        return artist;
24    }
25
26    public void addTrack(Track song) {
27        if (tracks.contains(song)) {
28            System.out.println(song.getTitle() + " is already in the CD");
29        } else {
30            tracks.add(song);
31            System.out.println("Track added");
32        }
33    }
34
35    public void removeTrack(Track song) {
36        if (tracks.contains(song)) {
37            tracks.remove(song);
38        } else {
39            System.out.println(song.getTitle() + " is not in the CD");
40        }
41    }
42
43    @Override
44    public String toString() {
45        return "CD - " + this.getTitle() + " - " + this.getCategory()
46            + " - " + this.getCost() + " - " + this.getDirector()
47            + " - " + this.getLength() + " - " + this.getArtist();
48    }
49
50 }
51
```

IV. Create interface Playable

1. Interface Playable



```
1 package hust.soict.dsai.aims.media;
2
3 public interface Playable {
4     public void play();
5 }
```

2. Fix CompactDisc



```
1 public class CompactDisc extends Disc implements Playable {
2
3     public void play() {
4         System.out.println("\nTitle: " + getTitle() + "\nArtist: " + getArtist());
5         for (Track track : tracks) {
6             track.play();
7         }
8     }
9 }
```

3. Fix DigitalVideoDisc



```
1 public class DigitalVideoDisc extends Disc implements Playable {  
2     public void play() {  
3         System.out.println("Playing DVD: " + this.getTitle());  
4         System.out.println("DVD length: " + this.getLength());  
5     }  
6 }
```

V. Update Cart class to work with Media


```

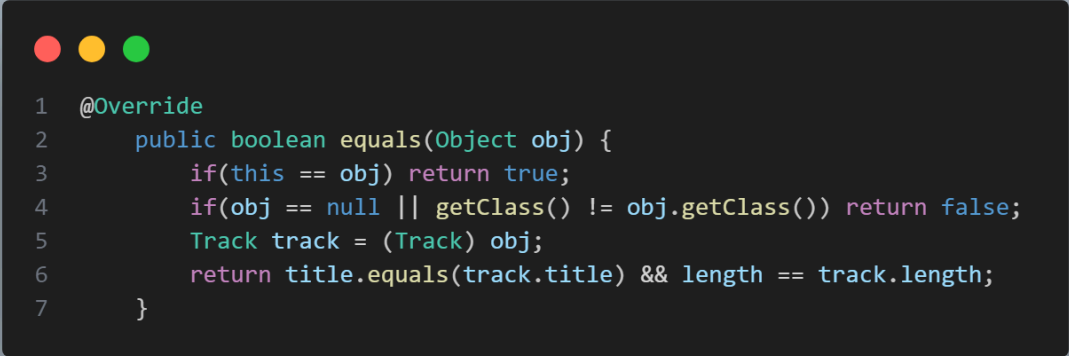
1 package hust.soict.dsai.aims.cart;
2
3 import hust.soict.dsai.aims.media.DigitalVideoDisc;
4 import hust.soict.dsai.aims.media.Media;
5 import java.lang.module.ModuleDescriptor;
6 import java.util.*;
7
8 public class Cart {
9     private ArrayList<Media> itemsOrdered = new ArrayList<Media>();
10
11     public void addMedia(Media item) {
12         if(itemsOrdered.contains(item)) {
13             System.out.println(item.getTitle() + " is already in the cart");
14         } else {
15             itemsOrdered.add(item);
16             System.out.println("Added " + item.getTitle() + " to cart");
17         }
18     }
19
20     public void removeMedia(Media item) {
21         if(itemsOrdered.contains(item)) {
22             itemsOrdered.remove(item);
23             System.out.println("Removed " + item.getTitle() + " from cart");
24         } else {
25             System.out.println(item.getTitle() + "is not in cart");
26         }
27     }
28
29     public float totalCost() {
30         float result = 0f;
31         for(Media item : itemsOrdered) {
32             result += item.getCost();
33         }
34         return result;
35     }
36
37     public void searchById(int id) {
38         if(id < 0 || id >= itemsOrdered.size()) {
39             System.out.println("Invalid id");
40             return;
41         } else {
42             System.out.println("Disc found: " + itemsOrdered.get(id).getTitle()
43 + " - " + itemsOrdered.get(id).getCategory()
44 + " - " + itemsOrdered.get(id).getCost() + "$\n");
45         }
46     }
47
48     public void searchByTitle(String title) {
49         for(Media item : itemsOrdered) {
50             if(item.getTitle().equals(title)) {
51                 System.out.println("Disc found: " + item.getTitle()
52 + " - " + item.getCategory()
53 + " - " + item.getCost() + "$\n");
54                 return;
55             }
56         }
57         System.out.println("Disc not found\n");
58     }
59
60     public void displayCart() {
61         System.out.println("*****CART*****");
62         for(Media item : itemsOrdered) {
63             System.out.println(item.toString());
64         }
65         System.out.println("Total cost: " + totalCost());
66         System.out.println("*****");
67     }
68
69     public Media findMedia(String title) {
70         for(Media item : itemsOrdered) {
71             if(item.getTitle().equals(title)) {
72                 return item;
73             }
74         }
75         return null;
76     }
77
78     public void emptyCart() {
79         itemsOrdered.removeAll(itemsOrdered);
80     }
81
82     public void sortByTitleCost() {
83         Collections.sort(itemsOrdered, Media.COMPARE_BY_TITLE_COST);
84     }
85
86     public void sortByCostTitle() {
87         Collections.sort(itemsOrdered, Media.COMPARE_BY_COST_TITLE);
88     }
89 }

```

VI. Update Store class to work with Media

```
1  package hust.soict.dsai.aims.store;
2
3  import java.util.ArrayList;
4  import hust.soict.dsai.aims.media.*;
5
6  public class Store {
7      private ArrayList<Media> itemsInStore = new ArrayList<Media>();
8
9      private boolean checkMedia(Media disc) {
10         for(Media item : itemsInStore) {
11             if(item.equals(disc)) {
12                 return true;
13             }
14         }
15         return false;
16     }
17
18     public Media findMedia(String title) {
19         for(Media item : itemsInStore) {
20             if(item.getTitle().equals(title)) {
21                 return item;
22             }
23         }
24         return null;
25     }
26
27     public ArrayList<Media> getItemsInStore() {
28         return itemsInStore;
29     }
30
31     public void addMedia(Media disc) {
32         if(checkMedia(disc)) {
33             System.out.println("The disc " + disc.getTitle() + " is already in store!");
34         } else {
35             itemsInStore.add(disc);
36             System.out.println("The disc " + disc.getTitle() + " has been added!");
37         }
38     }
39
40     public void removeMedia(Media disc) {
41         if(itemsInStore.remove(disc)) {
42             System.out.println("The disc " + disc.getTitle() + " has been removed!");
43         } else {
44             System.out.println("Could not find " + disc.getTitle() + " in store!");
45         }
46     }
47
48     @Override
49     public String toString() {
50         StringBuilder string = new StringBuilder("\n*****STORE*****\nItems in the store: \n");
51         if(itemsInStore.isEmpty()) {
52             string.append("Empty\n");
53         } else {
54             for(Media item : itemsInStore) {
55                 string.append(item.toString()).append("\n");
56             }
57         }
58         string.append("*****\n");
59         return string.toString();
60     }
61 }
62
```

VII. Unique item in a list



```
1  @Override
2      public boolean equals(Object obj) {
3          if(this == obj) return true;
4          if(obj == null || getClass() != obj.getClass()) return false;
5          Track track = (Track) obj;
6          return title.equals(track.title) && length == track.length;
7      }
```

2. Sửa lại Media

```
1 package hust.soict.dsai.aims.media;
2
3 import java.util.*;
4
5 public abstract class Media {
6     private int id;
7     private String title;
8     private String category;
9     private float cost;
10
11     public static final Comparator<Media> COMPARE_BY_TITLE_COST = new MediaComparatorByTitleCost();
12     public static final Comparator<Media> COMPARE_BY_COST_TITLE = new MediaComparatorByCostTitle();
13
14
15     public Media(int id, String title, String category, float cost) {
16         this.setId(id);
17         this.setTitle(title);
18         this.setCategory(category);
19         this.setCost(cost);
20     }
21
22     public Media(String title){
23         this.setTitle(title);
24     }
25
26     public Media(String title, String category, float cost){
27         this(title);
28         this.setCategory(category);
29         this.setCost(cost);
30     }
31
32     public int getId() {
33         return id;
34     }
35     public void setId(int id) {
36         this.id = id;
37     }
38     public String getTitle() {
39         return title;
40     }
41     public void setTitle(String title) {
42         this.title = title;
43     }
44     public String getCategory() {
45         return category;
46     }
47     public void setCategory(String category) {
48         this.category = category;
49     }
50     public float getCost() {
51         return cost;
52     }
53     public void setCost(float cost) {
54         this.cost = cost;
55     }
56
57     public boolean equals(Media obj) {
58         return this.getTitle().equals(obj.getTitle());
59     }
60
61     @Override
62     public String toString() {
63         return "Media - " + this.getTitle() + " - " + this.getCategory() + " - " + this.getCost();
64     }
65 }
```

VIII. Polymorphism with toString method;

```
1 package hust.soict.dsai.aims.media;
2
3 import java.util.List;
4 import java.util.ArrayList;
5
6 public class Polymorphism {
7
8     public static void main(String[] args) {
9         List<Media> mediae = new ArrayList<Media>();
10        ArrayList<Track> tracks = new ArrayList<Track>();
11        tracks.add(new Track("Track 1", 1.0f));
12        tracks.add(new Track("Track 2", 2.0f));
13        CompactDisc cd = new CompactDisc(1, "Nhac bat hu", "Music", 10.0f, "Artist", tracks);
14        DigitalVideoDisc dvd = new DigitalVideoDisc(2, "DVD", "Movie", 20.0f, "Director", 120);
15        Book book = new Book(3, "Book", "Book", 5.0f, List.of("Author 1", "Author 2"));
16        mediae.add(cd);
17        mediae.add(dvd);
18        mediae.add(book);
19
20        for(Media m: mediae) {
21            System.out.println(m.toString());
22        }
23    }
24
25 }
```

result:

```
<terminated> Polymorphism (1) [Java Application] C:\Program Files\Java\jdk-
CD - Nhac bat hu - Music - 10.0 - 0.0 - Artist - 2 tracks
Disc - DVD - Movie - 20.0 - Director - 120.0
Book - Book - Book - 5.0 - [Author 1, Author 2]
```

IX. Sort media in the cart

```

1 package hust.soict.dsai.aims.media;
2
3 import java.util.Comparator;
4
5 public class MediaComparatorByCostTitle implements Comparator<Media>
6 {
7     @Override
8     public int compare(Media m1, Media m2) {
9         if(m1.getCost() != m2.getCost()) {
10             return Float.compare(m1.getCost(), m2.getCost());
11         }
12         return m1.getTitle().compareTo(m2.getTitle());
13     }
14
15 }

```

```

1 package hust.soict.dsai.aims.media;
2
3 import java.util.Comparator;
4 public class MediaComparatorByTitleCost implements Comparator<Media> {
5     @Override
6     public int compare(Media m1, Media m2) {
7         if(m1.getTitle().compareTo(m2.getTitle()) != 0) {
8             return m1.getTitle().compareTo(m2.getTitle());
9         } else {
10             if(m1.getCost() > m2.getCost()) return 1;
11             else return -1;
12         }
13     }
14
15 }

```

X. Create a complete console application in the Aims class

```

package hust.soict.dsai.aims;
import hust.soict.dsai.aims.cart.Cart;
import hust.soict.dsai.aims.store.Store;
import hust.soict.dsai.aims.media.*;
import java.util.*;

```

```

public class Aims {
    public static void main(String[] args) {
        DigitalVideoDisc dvd = new
DigitalVideoDisc(1,"Cinderella","Fantasy", 18.5f,"Tung",97);
        ArrayList<Track> tracks = new ArrayList<Track>();
        tracks.add(new Track("Happy new year",3));
        tracks.add(new Track("i want it that way",4));
        CompactDisc cd = new CompactDisc(2,"Nhac bat hu ngay xua","Nhac
Au My Latin",25.5f,"Various artist",tracks);
        List<String> authors = new ArrayList<String>();
        authors.add("Phung Quan");
        authors.add("Gia Huy");
        Book book    = new Book(3,"Tuoi tho du
doi","novel",25.2f,authors);
        Store store = new Store();
        store.addMedia(cd);
        store.addMedia(dvd);
        store.addMedia(book);
        Cart cart = new Cart();
        Scanner scanner = new Scanner(System.in);
        showMenu(scanner, store, cart);
    }

    public static void showMenu(Scanner scanner, Store store, Cart
cart) {
        while (true) {
            System.out.println(
                """

                AIMS:
                -----
                1. View store
                2. Update store
                3. See current cart
                0. Exit
                -----
                Please choose a number: 0-1-2-3
                """);
            int option = scanner.nextInt();
            switch (option) {
                case 0 -> {
                    scanner.close();
                    System.exit(0);
                }
            }
        }
    }
}

```

```

        }
        case 1 -> storeMenu(scanner, store, cart);
        case 2 -> updateStoreMenu(scanner, store);
        case 3 -> {
            cart.displayCart();
            cartMenu(scanner, cart);
        }
    }
}

}

}

public static void updateStoreMenu(Scanner scanner, Store store) {
    System.out.println("""
        =====
        1. add Media
        2. delete Media
        3. update Media in Store
        0. Back
        =====
        Option: """);
    int option = scanner.nextInt();
    switch (option) {
        case 1 -> {
            System.out.println("""
                1.DigitalVideoDisc
                2.CompactDisc
                3.Book
                -----
                -> Your type: """);
            int option2 = scanner.nextInt();
            System.out.print("Enter id: ");
            int id = scanner.nextInt();
            scanner.nextLine();
            System.out.print("Enter title: ");
            String title = scanner.nextLine();
            System.out.print("Enter category: ");
            String category = scanner.nextLine();
            System.out.print("Enter cost: ");
            float cost = scanner.nextFloat();
            scanner.nextLine();
            switch (option2) {
                case 1 ->
                {

```



```

        System.out.print("Enter director's name: ");
        String director = scanner.nextLine();
        System.out.print("Enter dvd's length: ");
        int length = scanner.nextInt();
        scanner.nextLine();
        store.addMedia(new
DigitalVideoDisc(id,title,category,cost,director,length));
    }
    case 3 -> {
        System.out.print("Enter author's name (Enter
nothing to skip): ");
        StringBuilder author = new
StringBuilder(scanner.nextLine());
        ArrayList<String> authors = new
ArrayList<String>();
        while (!author.isEmpty()) {
            authors.add(author.toString());
            System.out.print("Enter author's name
(Enter nothing to skip): ");
        }
        store.addMedia(new
Book(id,title,category,cost,authors));
    }
    case 2 -> {
        System.out.print("Enter artist's name: ");
        StringBuffer artist = new
StringBuffer(scanner.nextLine());
        System.out.print("Enter number of track: ");
        int nbTrack =
scanner.nextInt();scanner.nextLine();
        ArrayList<Track> tracks = new
ArrayList<Track>();
        StringBuilder name = new StringBuilder();
        for(int i = 0;i< nbTrack;i++) {
            System.out.print("Enter Track[" + i + "]"s
name: ");
            name.replace(0,name.length(),scanner.nextLine());
            System.out.print("Enter Track[" + i + "]"s
length: ");
            int length = scanner.nextInt();

```

```

        tracks.add(new Track(name.toString(),
length));

        scanner.nextLine();

    }
    store.addMedia(new
CompactDisc(id,title,category,cost,artist.toString(),tracks));
    }
}
case 2 -> {
    System.out.println("Enter item's title: ");
    scanner.nextLine();
    String title = scanner.nextLine();
    Iterator<Media> iter =
store.getItemsInStore().iterator();
    while (iter.hasNext()) {
        Media item = iter.next();
        if(item.getTitle().equals(title)) {
            iter.remove();

System.out.println(item.getClass().getSimpleName() + " " +
item.getTitle() + "'ve been deleted from the store !");
        }
    }

}
case 3 -> {
    System.out.println("Enter item's id: ");
    int id = scanner.nextInt();
    scanner.nextLine();
    System.out.print("Enter title: ");
    String title = scanner.nextLine();
    System.out.print("Enter category: ");
    String category = scanner.nextLine();
    System.out.print("Enter cost: ");
    float cost = scanner.nextFloat();
    store.getItemsInStore().get(id).setCost(cost);
    store.getItemsInStore().get(id).setTitle(title);
    store.getItemsInStore().get(id).setCategory(category);
    System.out.println(store);
}
}

```

```

    }

    public static void mediaDetailsMenu(Scanner scanner, Store store,
    Cart cart) {
        System.out.print("Enter media's title: ");
        String title = scanner.nextLine();
        Media item = store.findMedia(title);
        if(item == null) {
            System.out.println("There is no such media !");
            return;
        }
        System.out.println(item);
        while (true) {
            System.out.println("""
                                Options:
                                -----
                                1. Add to cart
                                2. Play
                                0. Back
                                -----
                                Please choose a number: 0-1-2""");
            int option = scanner.nextInt();
            scanner.nextLine();
            switch (option) {
                case 1 -> {
                    cart.addMedia(item);
                }
                case 2 -> {
                    if (item.getClass().getSimpleName().equals("Book"))
                    {
                        System.out.println("This media is unplayable");
                    } else {
                        if (item instanceof DigitalVideoDisc dvd) {
                            dvd.play();
                        }
                        if (item instanceof CompactDisc cd) {
                            cd.play();
                        }
                    }
                }
                case 0 -> {
                    return;
                }
            }
        }
    }
}

```

```

    }
}

}

    public static void storeMenu(Scanner scanner, Store store, Cart
cart) {
    System.out.println(store);
    while (true) {
        System.out.println("Options: ");
        System.out.println("-----");
        System.out.println("1. See a media's details");
        System.out.println("2. Add a media to cart");
        System.out.println("3. Play a media");
        System.out.println("4. See current cart");
        System.out.println("0. Back");
        System.out.println("-----");
        System.out.println("Please choose a number: 0-1-2-3-4");
        int option = scanner.nextInt();
        scanner.nextLine();
        switch (option) {
            case 1 -> mediaDetailsMenu(scanner, store, cart);
            case 0 -> {
                return;
            }
            case 2 -> {
                System.out.print("Enter media's title: ");
                String title = scanner.nextLine();
                Media item = store.findMedia(title);
                if (item == null) {
                    System.out.println("There is no such media !");
                } else {
                    cart.addMedia(item);
                }
            }
            case 3 -> {
                System.out.print("Enter media's title: ");
                String title = scanner.nextLine();
                Media item = store.findMedia(title);
                if (item == null) {
                    System.out.println("There is no such media !");
                } else {
                    if
(item.getClass().getSimpleName().equals("Book")) {

```

```

        System.out.println("This media is
unplayable");
    } else {
        if (item instanceof DigitalVideoDisc dvd) {
            dvd.play();
        }
        if (item instanceof CompactDisc cd) {
            cd.play();
        }
    }
}
}
case 4 -> {
    cart.displayCart();
    cartMenu(scanner, cart);
}
}
}

public static void cartMenu(Scanner scanner, Cart cart) {
    while (true) {
        System.out.println("""
Options:
-----
1. Filter medias in cart
2. Sort medias in cart
3. Remove media from cart
4. Play a media
5. Place order
0. Back
-----
Please choose a number: 0-1-2-3-4-5""");
        int option = scanner.nextInt();
        scanner.nextLine();
        switch (option) {
            case 0 -> {
                return;
            }
            case 1 -> {
                System.out.println("""
1. Filter by title
2. Filter by id

```

```

        -----
        your option:
        """);
    int option2 = scanner.nextInt();
    scanner.nextLine();
    if (option2 == 1) {
        int id = scanner.nextInt();
        cart.searchById(id);
    } else {
        String title = scanner.nextLine();
        cart.searchByTitle(title);
    }
}
case 2 -> {
    System.out.println("""
        1. sort by title cost
        2. sort by cost title
        -----
        your option:
        """);
    int option2 = scanner.nextInt();
    scanner.nextLine();
    if (option2 == 1) {
        cart.sortByTitleCost();
        cart.displayCart();
    } else {
        cart.sortByCostTitle();
        cart.displayCart();
    }
}
case 3 -> {
    System.out.print("Enter media's title: ");
    String title = scanner.nextLine();
    Media item = cart.findMedia(title);
    if (item == null) {
        System.out.println("There is no such media !");
    } else {
        cart.removeMedia(item);
    }
}
case 4 -> {
    System.out.print("Enter media's title: ");
    String title = scanner.nextLine();

```



```

// Constructor
public Media(int id, String title, String category, float cost) {
    this.id = id;
    this.title = title;
    this.category = category;
    this.cost = cost;
}

// Getters
public int getId() { return id; }
public String getTitle() { return title; }
public String getCategory() { return category; }
public float getCost() { return cost; }
}

Lớp Book (Lớp Con)package hust.soict.dsai.aims.media;

import java.util.ArrayList;

public class Book extends Media {
    private ArrayList<String> authors;

    // Constructor
    public Book(int id, String title, String category, float cost) {
        super(id, title, category, cost);
        this.authors = new ArrayList<>();
    }

    // Thêm tác giả
    public void addAuthor(String authorName) {
        if (!authors.contains(authorName)) {
            authors.add(authorName);
        } else {
            System.out.println("Author already exists.");
        }
    }

    // Xóa tác giả
    public void removeAuthor(String authorName) {
        if (authors.contains(authorName)) {
            authors.remove(authorName);
        } else {
            System.out.println("Author not found.");
        }
    }
}

```


Question 2: Alternatively, to compare items in the cart, instead of using Comparator, we can use the Comparable interface and override the compareTo() method. You can refer to the Java docs to see the information of this interface.

Suppose we are taking this Comparable interface approach.

- What class should implement the Comparable interface?
- In those classes, how should you implement the compareTo() method to reflect the ordering that we want?
- Can we have two ordering rules of the item (by title then cost and by cost then title) if we use this Comparable interface approach?
- Suppose the DVDs has a different ordering rule from the other media types, that is by title, then decreasing length, then cost. How would you modify your code to allow this?

+ Lớp Media nên triển khai Comparable<Media> vì tất cả các loại media (Book, CompactDisc, DigitalVideoDisc) đều kế thừa từ Media. Điều này cho phép các đối tượng thuộc các lớp con có thể so sánh được dựa trên tiêu chí mặc định.

+ @Override

```
public int compareTo(Media other) {  
    // So sánh theo title  
    int titleComparison = this.title.compareTo(other.title);  
    if (titleComparison != 0) {  
        return titleComparison; // Trả về kết quả so sánh title  
    }  
    // Nếu title giống nhau, so sánh theo cost  
    return Float.compare(this.cost, other.cost);  
}
```

+ Không, nếu chỉ dùng Comparable, chỉ có thể áp dụng một quy tắc sắp xếp mặc định. Nếu muốn hỗ trợ nhiều quy tắc sắp xếp (ví dụ: title → cost và cost → title), nên sử dụng Comparator để có tính linh hoạt hơn.

+ Cần override lại phương thức compareTo() trong lớp con DigitalVideoDisc

```
package hust.soict.dsai.aims.media;  
  
public class DigitalVideoDisc extends Media implements Comparable<Media> {  
    private int length;  
  
    public DigitalVideoDisc(int id, String title, String category, float cost, int length) {  
        super(id, title, category, cost);  
        this.length = length;  
    }  
  
    public int getLength() {  
        return length;  
    }  
  
    @Override  
    public int compareTo(Media other) {  
        if (other instanceof DigitalVideoDisc) {  
            DigitalVideoDisc otherDVD = (DigitalVideoDisc) other;
```

```

// So sánh theo title
int titleComparison = this.getTitle().compareTo(otherDVD.getTitle());
if (titleComparison != 0) {
    return titleComparison;
}

// Nếu title giống nhau, so sánh length (giảm dần)
int lengthComparison = Integer.compare(otherDVD.getLength(), this.getLength());
if (lengthComparison != 0) {
    return lengthComparison;
}

// Nếu length giống nhau, so sánh cost
return Float.compare(this.getCost(), otherDVD.getCost());
}

// Nếu không phải DVD, sử dụng quy tắc của Media
return super.compareTo(other);
}
}

```

XII. Update class diagram

