

Báo cáo thực hành Lập trình hướng đối tượng Lab 04

Họ và tên: Phạm Đức Long

Mã số sinh viên: 20225737

Mã lớp: 744520

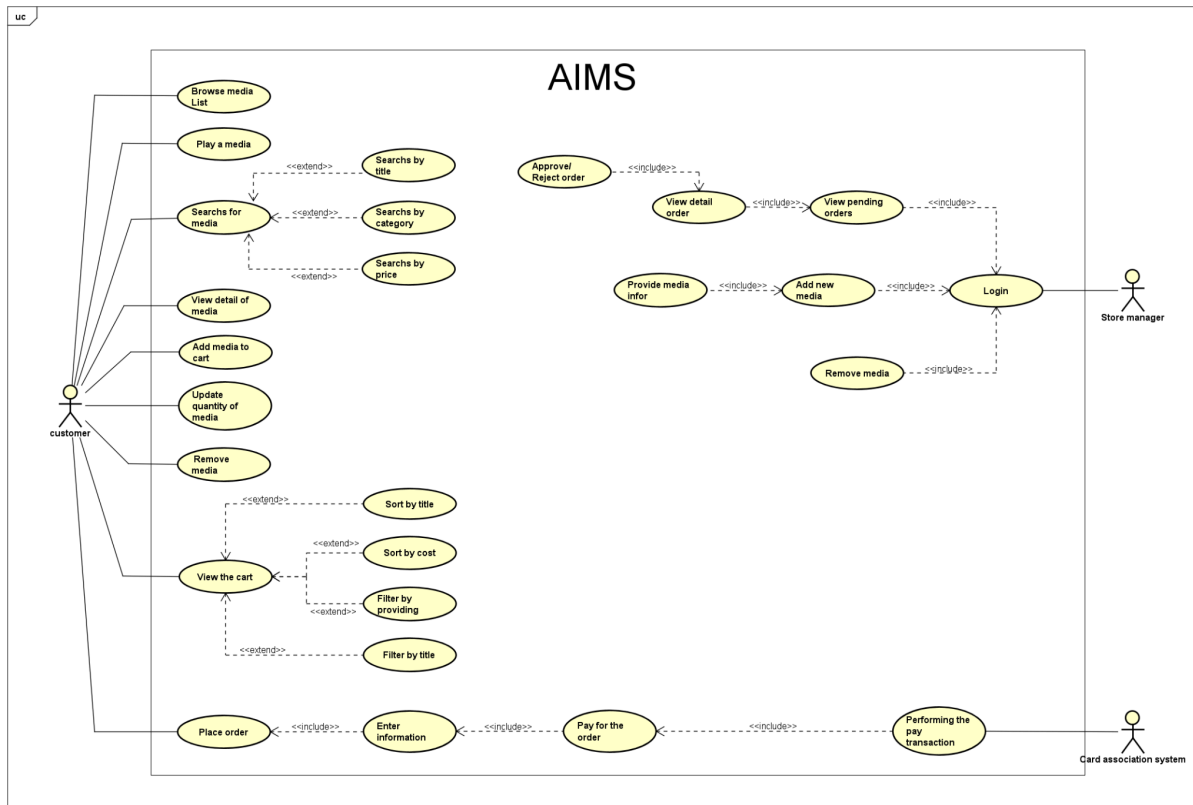
[google docs](#)

[github](#)

1. Import the existing project into the workspace of Eclipse	2
2. Additional requirements of AIMS	2
3. Creating the Book class	2
4. Creating the abstract Media class (Gộp phần 3 và 4 làm một)	2
5. Creating the CompactDisc class	5
5.1. Create the Disc class extending the Media class	5
5.2. Create the Track class which models a track on a compact disc and will store information including the title and length of the track	6
5.3. Open the CompactDisc class	7
6. Create the Playable interface	8
7. Update the Cart class to work with Media	9
8. Update the Store class to work with Media	11
9. Constructors of whole classes and parent classes	12
10. Unique item in a list	14
11. Polymorphism with toString() method	16
12. Sort media in the cart	18
13. Create a complete console application in the Aims class	21

1. Import the existing project into the workspace of Eclipse

2. Additional requirements of AIMS



3. Creating the Book class

4. Creating the abstract Media class (Gộp phần 3 và 4 làm một)

- Book class:

```

public class Book extends Media { 8 usages  ⬆ Long Pham +1
    private List<String> authors = new ArrayList<String>(); 6 usages

    public Book() { super(); }

    public Book(int id, String title, String category, float cost, List<String> authors) { 3 usages  ⬆ Lon
        super(id, title, category, cost);
        this.authors = authors;
    }

    public Book(int id, String title, String category, float cost) { super(id, title, category, cost); }

    public Book(String title, String category, float cost) { super(nbMedia++, title, category, cost); }

    public void addAuthor(String authorName) { 4 usages  ⬆ Long Pham +1
        if(!authors.contains(authorName)) {
            authors.add(authorName);
            System.out.println("Author added");
        }
        else System.out.println("Author is already in the list");
    }

    public void removeAuthor(String authorName) { no usages  ⬆ Long Pham +1
        if(authors.contains(authorName)) {
            authors.remove(authorName);
            System.out.println("Author removed");
        }
        else System.out.println("Author is not in the list");
    }

    @Override  ⬆ Long Pham +1
    public String toString() {
        return "Book: " +
            "id = " + getId() +
            " - title = " + getTitle() + '\'' +
            " - category = " + getCategory() + '\'' +
            " - cost = " + getCost() +
            " - (List of) authors = " + authors;
    }
}

```

- Media class:

```

public abstract class Media implements Comparable<Media> { 4 inheritors ± Long Pham +1
    public static final Comparator<Media> COMPARE_BY_TITLE_COST = new MediaComparatorByTitleCost();
    public static final Comparator<Media> COMPARE_BY_COST_TITLE = new MediaComparatorByCostTitle();
    private int id; 4 usages
    private String title; 8 usages
    private String category; 4 usages
    private float cost; 6 usages

    public static int nbMedia = 1; 6 usages

    public Media() { ± Long Pham
    }

    public Media(int id, String title, String category, float cost) { ± Long Pham
        this.id = id;
        this.title = title;
        this.category = category;
        this.cost = cost;
    }

    public int getId() { 6 usages ± Long Pham
        return id;
    }

    public void setId(int id) { this.id = id; }

    public String getTitle() { ± Long Pham
        return title;
    }

    public void setTitle(String title) { this.title = title; }

    public String getCategory() { return category; }

    public void setCategory(String category) { this.category = category; }

    public float getCost() { 11 usages ± Long Pham
        return cost;
    }

    public void setCost(float cost) { this.cost = cost; }

    @Override 4 overrides ± Long Pham
    public String toString() {
        return "Media: " +
            "id=" + id +
            " - title='" + title + '\'' +
            " - category='" + category + '\'' +
            " - cost=" + cost;
    }

    @Override ± Long Pham
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Media media = (Media) o;
        return Objects.equals(title, media.title);
    }

    @Override 1 override ± Long Pham
    public int compareTo(Media other) {
        int titleComparison = this.title.compareTo(other.title);
        if (titleComparison != 0) return titleComparison;

        return Float.compare(other.cost, this.cost);
    }
}

```

5. Creating the CompactDisc class

5.1. Create the Disc class extending the Media class

```
public class Disc extends Media { 2 usages 2 inheritors Long Pham
    private int length; 3 usages
    private String director; 3 usages

    public Disc() { 1 usage Long Pham
        super();
    }

    public Disc(int id, String title, String category, float cost, int length, String director) {
        super(id, title, category, cost);
        this.length = length;
        this.director = director;
    }

    public int getLength() { 11 usages 1 override Long Pham
        return length;
    }

    public void setLength(int length) { no usages Long Pham
        this.length = length;
    }

    public String getDirector() { 3 usages Long Pham
        return director;
    }

    public void setDirector(String director) { no usages Long Pham
        this.director = director;
    }

    @Override 2 overrides Long Pham
    public String toString() {
        return "Disc: " +
            "id=" + getId() +
            " - title = " + getTitle() + '\'' +
            " - category = " + getCategory() + '\'' +
            " - cost = " + getCost() +
            " - length = " + getLength() +
            " - director = " + getDirector() + '\'';
    }
}
```

5.2. Create the Track class which models a track on a compact disc and will store information including the title and length of the track

```
public class Track implements Playable { 26 usages  ⚡ Long Pham +1
    private String title; 6 usages
    private int length; 6 usages

    public Track() { no usages  ⚡ Long Pham
        super();
    }

    public Track(String title, int length) { 9 usages  ⚡ Long Pham
        this.title = title;
        this.length = length;
    }

    public String getTitle() { ⚡ Long Pham
        return title;
    }

    public void setTitle(String title) { ⚡ Long Pham
        this.title = title;
    }

    public int getLength() { 3 usages  ⚡ Long Pham
        return length;
    }

    public void setLength(int length) { no usages  ⚡ Long Pham
        this.length = length;
    }

    @Override 7 usages  ⚡ Long Pham +1
    public void play() {
        System.out.println("Playing track: " + getTitle());
        if (getLength() == 0) System.out.println("Track cannot be played.");
        else System.out.println("Track length: " + getLength());
    }

    @Override ⚡ Long Pham
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Track track = (Track) o;
        return length == track.length && Objects.equals(title, track.title);
    }

    @Override ⚡ Long Pham
    public String toString() {
        return "Track: " +
            "title = '" + title + '\'' +
            " - length = " + length;
    }
}
```

5.3. Open the CompactDisc class

```
public class CompactDisc extends Disc implements Playable { 17 usages ± Long Pham +1
    private String artist; 5 usages
    private ArrayList<Track> tracks = new ArrayList<Track>(); 10 usages

    public CompactDisc() { 1 usage ± Long Pham
        super();
    }

    public CompactDisc(int id, String title, String category, float cost, String director, String artist) { 1 usage ± Long Pham
        super(id, title, category, cost, length: 0, director);
        this.artist = artist;
    }

    public CompactDisc(int id, String title, String category, float cost, String director, String artist, ArrayList<Track> tracks) {
        super(id, title, category, cost, length: 0, director);
        this.artist = artist;
        this.tracks = tracks;
    }

    public CompactDisc(String title, String category, String artist, float cost) { 5 usages ± Long Pham +1
        super(nbMedia++, title, category, cost, length: 0, director: null);
        this.artist = artist;
        this.tracks = tracks;
    }

    public String getArtist() { return artist; }

    public void addTrack(Track track) { 9 usages ± Long Pham
        if (tracks.contains(track)) System.out.println("Track is already in the list");
        else {
            tracks.add(track);
            System.out.println("Track added");
        }
    }

    public void removeTrack(Track track) { no usages ± Long Pham
        if (tracks.contains(track)) {
            tracks.remove(track);
            System.out.println("Track removed");
        } else System.out.println("Track is not in the list");
    }

    public int getLength() { 11 usages ± Long Pham
        int length = 0;
        for (Track track : tracks) {
            length += track.getLength();
        }
        return length;
    }

    @Override 7 usages ± Long Pham +1
    public void play() {
        System.out.println("Playing CompactDisc: " + this.getTitle());

        if (getLength() == 0) System.out.println("CD cannot be played!");
        else {
            System.out.println("CD length: " + getLength());
            System.out.println("Artist: " + this.getArtist());

            for (Track track : tracks) track.play();
        }
    }

    @Override ± Long Pham
    public String toString() {
        return "CompactDisc: " +
            "id = " + getId() +
            " - title = " + getTitle() + '\'' +
            " - category = " + getCategory() + '\'' +
            " - cost = " + getCost() +
            " - length = " + getLength() +
            " - director = " + getDirector() + '\'' +
            " - artist = " + artist + '\'' +
            " - tracks = " + tracks;
    }
}
```

6. Create the Playable interface

```
public interface Playable {  
    public void play();  
}
```

- CompactDisc class:

```
@Override 7 usages Long Pham +1  
public void play() {  
    System.out.println("Playing CompactDisc: " + this.getTitle());  
  
    if (getLength() == 0) System.out.println("CD cannot be played!");  
    else {  
        System.out.println("CD length: " + getLength());  
        System.out.println("Artist: " + this.getArtist());  
  
        for (Track track : tracks) track.play();  
    }  
}
```

- DigitalVideoDisc class:

```
@Override 7 usages Long Pham +1  
public void play() {  
    System.out.println("Playing DVD: " + getTitle());  
  
    if (getLength() == 0) System.out.println("DVD cannot be played.");  
    else System.out.println("DVD length: " + getLength());  
}
```

- Track class:


```

@Override 7 usages Long Pham +1
public void play() {
    System.out.println("Playing track: " + getTitle());
    if (getLength() == 0) System.out.println("Track cannot be played.");
    else System.out.println("Track length: " + getLength());
}

```

7. Update the Cart class to work with Media

```

public class Cart { 6 usages Long Pham +1
    public static final int MAX_NUMBERS_ORDERED = 20; 1 usage
    private ArrayList<Media> itemsOrdered = new ArrayList<>();

    public void addMedia(Media media) { 4 usages Long Pham
        if (itemsOrdered.size() < MAX_NUMBERS_ORDERED) {
            itemsOrdered.add(media);
            System.out.println("The media has been added");
        } else {
            System.out.println("The media is almost full");
        }
    }

    public float totalCost() { 1 usage Long Pham
        float totalCost = 0;
        for (Media media : itemsOrdered) {
            totalCost += media.getCost();
        }

        return totalCost;
    }
}

```

```
public Media searchByTitle(String title) { 3 usages Long Pham
    for (Media media : itemsOrdered) {
        if (media.getTitle().equals(title))
            return media;
    }

    return null;
}

public Media searchById(int id) { 1 usage Long-1911 +1
    for (Media media : itemsOrdered) {
        if (media.getId() == id)
            return media;
    }

    return null;
}

public void sortByTitle() { 1 usage Long Pham
    Collections.sort(itemsOrdered, Media.COMPARE_BY_TITLE_COST);
    Iterator<Media> iterator = itemsOrdered.iterator();

    while (iterator.hasNext()) {
        System.out.println((iterator.next()).toString());
    }
}

public void sortByCost() { 1 usage Long Pham
    Collections.sort(itemsOrdered, Media.COMPARE_BY_COST_TITLE);
    Iterator<Media> iterator = itemsOrdered.iterator();

    while (iterator.hasNext()) System.out.println((iterator.next()).toString());
}
```

```

public void removeMedia(Media media) { 1 usage  Long Pham
    if (itemsOrdered.contains(media)) {
        itemsOrdered.remove(media);
        System.out.println("The media has been removed");
    } else System.out.println("The media is not in the cart");
}

public void empty() { itemsOrdered.clear(); }
}

```

8. Update the Store class to work with Media

```

public class Store { 6 usages  Long Pham +1
    private ArrayList<Media> itemsInStore = new ArrayList<Media>(); {

    public void addMedia(Media media) { 15 usages  Long Pham +1
        boolean existed = false;
        for (Media item : itemsInStore) {
            if (item.getTitle().equals(media.getTitle())) {
                existed = true;
                break;
            }
        }

        if (!existed) {
            itemsInStore.add(media);
            System.out.println("The media has been added in Store.");
        } else {
            System.out.println("The media is already in the store.");
        }
    }
}

```

```

public void removeMedia(Media media) { 2 usages Long Pham +1
    boolean existed = false;
    for (Media item : itemsInStore) {
        if (item.getTitle().equals(media.getTitle())) {
            itemsInStore.remove(item);
            System.out.println("The media has been removed from Store.");
            existed = true;
            break;
        }
    }

    if (!existed) {
        System.out.println("The media is not in the store.");
    }
}

public void printStore() { 7 usages Long Pham
    for (Media media : itemsInStore) {
        System.out.println(media.toString());
    }
}

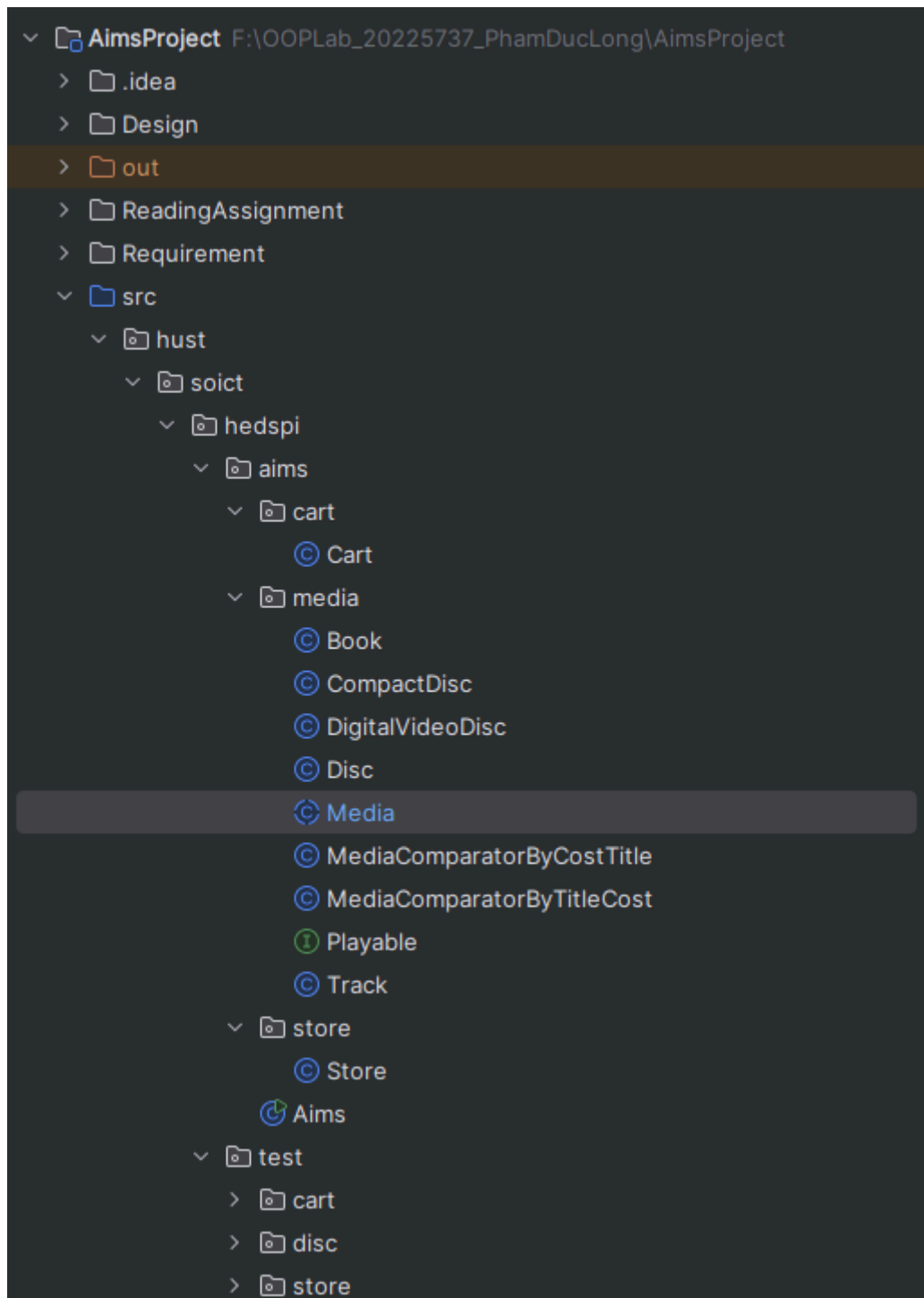
public Media searchByTitle(String title) { 4 usages Long Pham
    for (Media media : itemsInStore) {
        if (media.getTitle().equals(title)) {
            return media;
        }
    }

    return null;
}
}

```

9. Constructors of whole classes and parent classes

- Cấu trúc project Lab04:



- Class Diagram sau Lab04:

- Cart và Store đã khởi tạo các thành phần của chúng (ArrayList<Media>) một cách rõ ràng tại điểm khai báo. Điều này đảm bảo constructor không cần thực hiện thêm thao tác khởi tạo nào, trừ khi logic khởi tạo phức tạp hơn.

10. Unique item in a list

- Media class:

```
@Override  Long Pham
public boolean equals(Object o) {
    if (this == o) return true;
    if (o == null || getClass() != o.getClass()) return false;
    Media media = (Media) o;
    return Objects.equals(title, media.title);
}
```

- Track class:

```
@Override  Long Pham
public boolean equals(Object o) {
    if (this == o) return true;
    if (o == null || getClass() != o.getClass()) return false;
    Track track = (Track) o;
    return length == track.length && Objects.equals(title, track.title);
}
```

- Kiểm tra:

```
Media book1 = new Book( id: 5, title: "Book1", category: "Horror", cost: 4.5f);
// anOrder.addMedia(book1);

Media book2 = new Book( id: 6, title: "Book1", category: "Scientific", cost: 5.5f);
```

```
if (book1.equals(book2)) System.out.println("Two objects are equal"); // test equals
else System.out.println("Two objects are not equal");
```

```
"C:\Program Files\Java\jdk-1.8\bin\java.exe" ...  
Two objects are equal  
  
Process finished with exit code 0
```

- If the passing object is not an instance of Media, what happens?
Nếu đối tượng truyền vào không phải là một instance của Media thì sẽ không thực hiện được phép so sánh.
Nhưng nếu đối tượng truyền là instance con của Media thì ta vẫn sẽ thực hiện được phép so sánh, như ở ví dụ trên.

11. Polymorphism with toString() method

```
List<Media> mediae = new ArrayList<Media>();  
ArrayList<Track> tracks = new ArrayList<Track>();  
tracks.add(new Track( title: "Track 1", length: 1));  
tracks.add(new Track( title: "Track 2", length: 2));  
Media cd = new CompactDisc( id: 7, title: "CD2", category: "Music", cost: 10.3f, director: "Director 2", artist: "Long", tracks);  
  
Media dvd = new DigitalVideoDisc( title: "The Lion King",  
category: "Animation", director: "Roger Allers", length: 87, cost: 19.95f);  
  
List<String> authors = new ArrayList<>();  
authors.add("Author 1");  
authors.add("Author 2");  
Media book = new Book( id: 8, title: "Book2", category: "Horror", cost: 4.5f, authors);  
  
mediae.add(cd);  
mediae.add(dvd);  
mediae.add(book);  
  
for (Media m : mediae) {  
    System.out.println(m.toString());  
}
```

- Kết quả:

```
"C:\Program Files\Java\jdk-1.8\bin\java.exe" ...  
CompactDisc: id = 7 - title = 'CD2' - category = 'Music' - cost = 10.3 - length = 3 - director = 'Director 2' - artist = 'Long' - tracks = [Track: title = 'Track 1' - length = 1, Track: title = 'Track 2' - length = 2]  
DigitalVideoDisc: id = 1 - title = 'The Lion King' - category = 'Animation' - cost = 19.95 - length = 87 - director = 'Roger Allers'  
Book: id = 8 - title = 'Book2' - category = 'Horror' - cost = 4.5 - (List of) authors = [Author 1, Author 2]  
  
Process finished with exit code 0
```

CompactDisc: id = 7 - title = 'CD2' - category = 'Music' - cost = 10.3 - length = 3 - director = 'Director 2' - artist = 'Long' - tracks = [Track: title = 'Track 1' - length = 1, Track: title = 'Track 2' - length = 2]

DigitalVideoDisc: id = 1 - title = 'The Lion King' - category = 'Animation' - cost = 19.95 - length = 87 - director = 'Roger Allers'

Book: id = 8 - title = 'Book2' - category = 'Horror' - cost = 4.5 - (List of) authors = [Author 1, Author 2]

- Giải thích: Mặc dù trong class Media cũng đã có phương thức toString() nhằm hiển thị ra các thuộc tính và giá trị của các thuộc tính:

```
@Override 4 overrides Long Pham
public String toString() {
    return "Media: " +
        "id=" + id +
        " - title='" + title + '\'' +
        " - category='" + category + '\'' +
        " - cost=" + cost;
}
```

Nhưng toString() cũng được override để hiển thị chi tiết các thuộc tính hơn trong các lớp CD, DVD và Book. Do đó java sẽ sử dụng toString() trong các lớp này:

- DVD:

```
@Override Long Pham
public String toString() {
    return "DigitalVideoDisc: " +
        "id = " + getId() +
        " - title = '" + getTitle() + '\'' +
        " - category = '" + getCategory() + '\'' +
        " - cost = " + getCost() +
        " - length = " + getLength() +
        " - director = '" + getDirector() + '\'';
}
```

- CD:

```
@Override  Long Pham
public String toString() {
    return "CompactDisc: " +
        "id = " + getId() +
        " - title = '" + getTitle() + '\'" +
        " - category = '" + getCategory() + '\'" +
        " - cost = " + getCost() +
        " - length = " + getLength() +
        " - director = '" + getDirector() + '\'" +
        " - artist = '" + artist + '\'" +
        " - tracks = " + tracks;
}
```

- Book:

```
@Override  Long Pham +1
public String toString() {
    return "Book: " +
        "id = " + getId() +
        " - title = '" + getTitle() + '\'" +
        " - category = '" + getCategory() + '\'" +
        " - cost = " + getCost() +
        " - (List of) authors = " + authors;
}
```

12. Sort media in the cart

- MediaComparatorByCostTitle:

```
public class MediaComparatorByCostTitle implements Comparator<Media> {

    @Override // Long Pham
    public int compare(Media o1, Media o2) {
        int costComparison = Float.compare(o2.getCost(), o1.getCost());
        if (costComparison != 0) {
            return costComparison;
        }

        return o1.getTitle().compareTo(o2.getTitle());
    }
}
```

- MediaComparatorByTitleCost:

```
public class MediaComparatorByTitleCost implements Comparator<Media> {

    @Override // Long Pham
    public int compare(Media o1, Media o2) {
        int titleComparison = o1.getTitle().compareTo(o2.getTitle());
        if (titleComparison != 0) {
            return titleComparison;
        }

        return Float.compare(o2.getCost(), o1.getCost());
    }
}
```

- Chạy thử:

```

public static void main(String[] args) { new *
    ArrayList<Media> mediaList = new ArrayList<>();
    mediaList.add(new DigitalVideoDisc( title: "The Lion King", category: "Animation", cost: 10.5f));
    mediaList.add(new DigitalVideoDisc( title: "The Lion King", category: "Animation", cost: 15.0f));
    mediaList.add(new DigitalVideoDisc( title: "Aladdin", category: "Animation", cost: 12.0f));

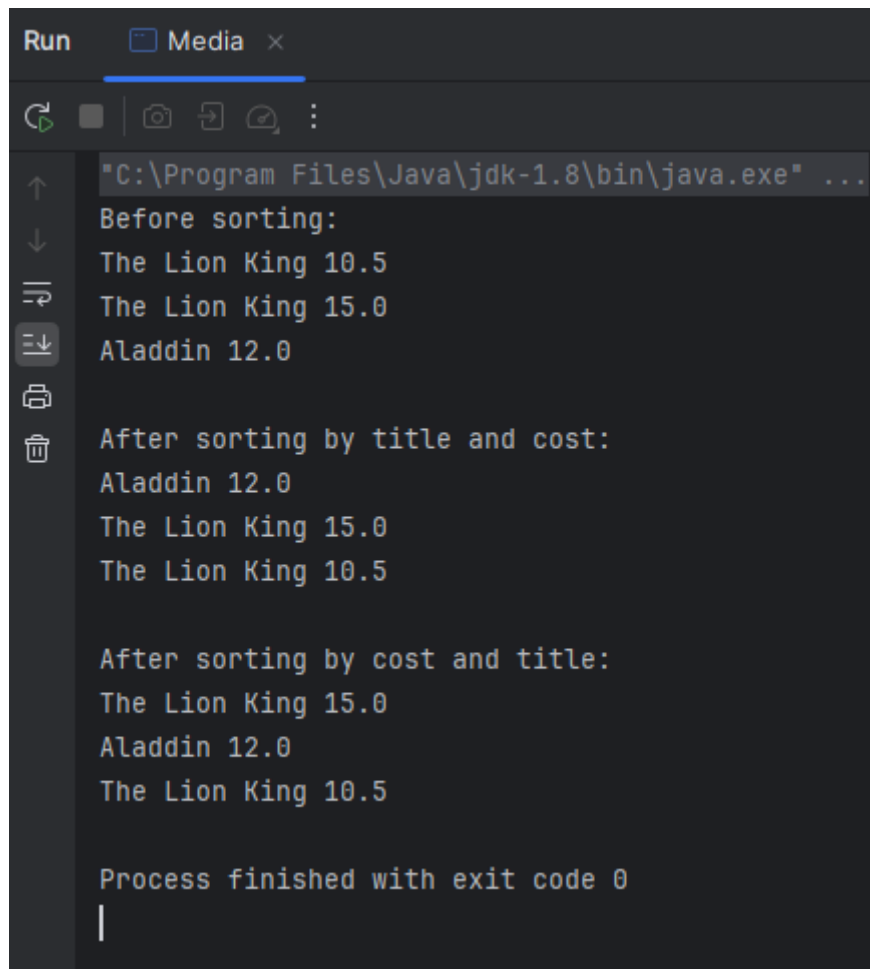
    System.out.println("Before sorting:");
    for (Media media : mediaList) {
        System.out.println(media.title + " " + media.cost);
    }

    Collections.sort(mediaList, Media.COMPARE_BY_TITLE_COST);
    System.out.println("\nAfter sorting by title and cost:");
    for (Media media : mediaList) {
        System.out.println(media.title + " " + media.cost);
    }

    Collections.sort(mediaList, Media.COMPARE_BY_COST_TITLE);
    System.out.println("\nAfter sorting by cost and title:");
    for (Media media : mediaList) {
        System.out.println(media.title + " " + media.cost);
    }
}

```

- Kết quả:



The screenshot shows a 'Run' window in an IDE with a tab labeled 'Media'. The console output is as follows:

```
"C:\Program Files\Java\jdk-1.8\bin\java.exe" ...  
Before sorting:  
The Lion King 10.5  
The Lion King 15.0  
Aladdin 12.0  
  
After sorting by title and cost:  
Aladdin 12.0  
The Lion King 15.0  
The Lion King 10.5  
  
After sorting by cost and title:  
The Lion King 15.0  
Aladdin 12.0  
The Lion King 10.5  
  
Process finished with exit code 0  
|
```

Các câu hỏi phần này đã được em trả lời trong file [answer.txt](#), thầy (và anh trợ giảng) có thể xem trực tiếp trên [docs](#), ở thanh bên trái ạ

13. Create a complete console application in the Aims class

- Do phần này khá dài nên em xin phép để trên [github](#) ạ.