LAB 04 - Tạ Hải Luân

9. Constructors of whole classes and parent classes

- Which classes are aggregates of other classes? Checking all constructors of whole classes if they initialize for their parts?

- Aggregates:

+ **Store** aggregates **Media**.
+ **Cart** aggregates **Media**.
+ **CompactDisc** aggregates **Track**.

· **Store Class**

+ **Attributes**: Likely contains a collection of Media objects.
+ **Constructor**: Initializes the list of Media.
+ **Aggregation**: The Store class aggregates Media objects because Media can exist independently of the Store.

· **Cart Class**

+ **Attributes**: Likely contains a collection of Media objects.
+ **Constructor**: Initializes the list of Media.
+ **Aggregation**: The Cart class aggregates Media objects for the same reason as Store.

· **Disc Class**

+ **Attributes**: May contain additional details like length and director.
+ **Constructor**: Sets properties for Disc, and indirectly via inheritance, initializes Media attributes.
+ **Aggregation**: Aggregates no separate objects but inherits from Media.

· **CompactDisc Class**

+ **Attributes**: Contains a List<Track> and an artist.
+ **Constructor**: Likely initializes the List<Track>.
+ **Aggregation**: The CompactDisc aggregates Track because Track instances can exist independently of a CompactDisc.

· **Track Class**

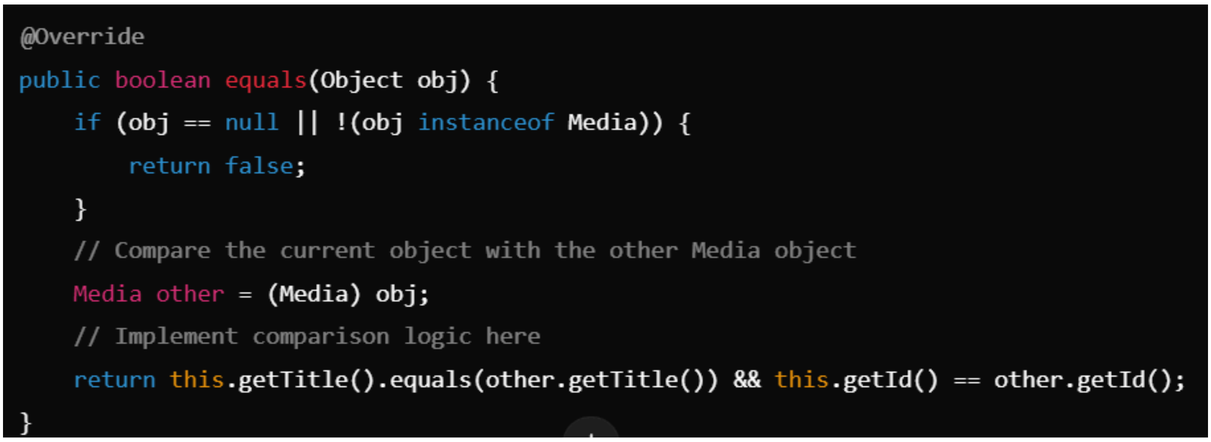
+ **Attributes**: Title and length.

  - **Constructor**: Initializes these properties.
  - **Aggregation**: Not an aggregate class since it contains no other objects.

- **DigitalVideoDisc Class**
  - **Attributes**: Inherits Disc attributes and methods.
  - **Constructor**: Sets properties specific to DigitalVideoDisc and initializes inherited ones.
  - **Aggregation**: None; it directly inherits from Disc.

10. If the passing object is not an instance of Media, what happens?

- If an object passed to the equals() method is not an instance of Media or Track, it will return false. This ensures type safety, preventing the equals() method from throwing a ClassCastException. The check could be implemented as follows:



```java
@Override
public boolean equals(Object obj) {
    if (obj == null || !(obj instanceof Media)) {
        return false;
    }
    // Compare the current object with the other Media object
    Media other = (Media) obj;
    // Implement comparison logic here
    return this.getTitle().equals(other.getTitle()) && this.getId() == other.getId();
}
```

12. Sort media in the cart

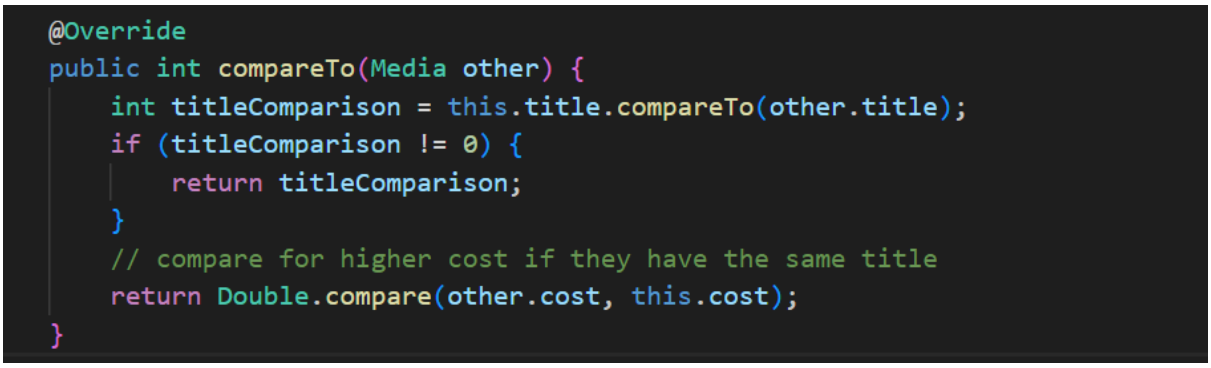
Question: *Alternatively, to compare items in the cart, instead of using Comparator, we can use the Comparable interface and override the compareTo()method. You can refer to the Java docs to see the information of this interface.*

*For this **Comparable** interface approach.*

- What class should implement the Comparable interface?

  - The Media class should implement the Comparable interface since we want to define a default ordering for media objects.

- In those classes, how should you implement the compareTo() method be to reflect the ordering that we want?



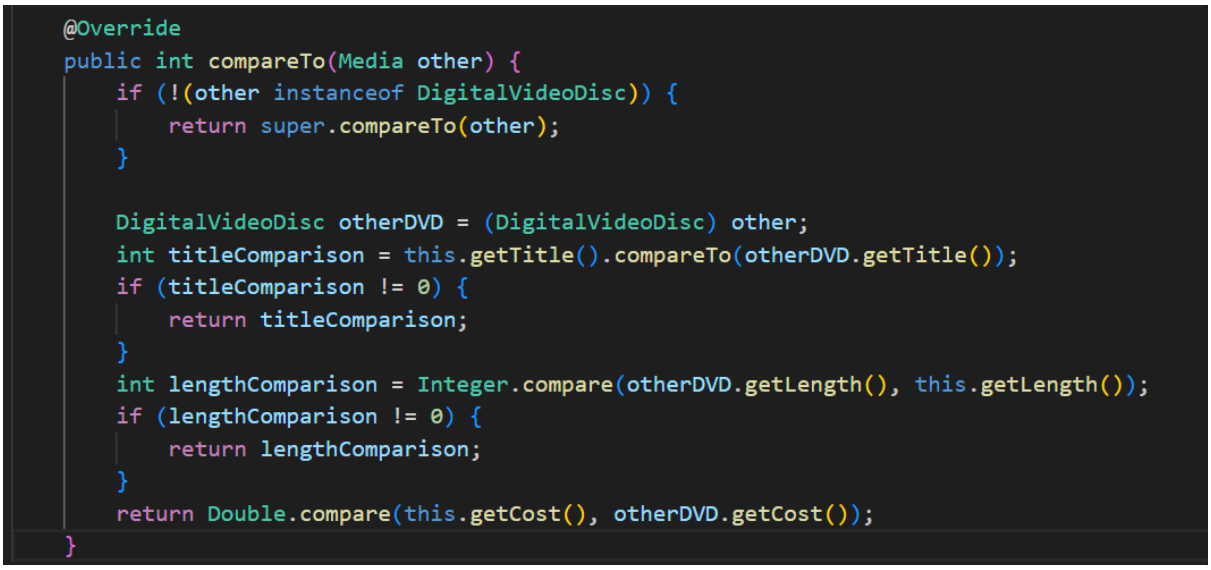
```
@Override
public int compareTo(Media other) {
    int titleComparison = this.title.compareTo(other.title);
    if (titleComparison != 0) {
        return titleComparison;
    }
    // compare for higher cost if they have the same title
    return Double.compare(other.cost, this.cost);
}
```

- 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?

- No, the Comparable interface allows only one natural ordering for the class. This means that you cannot have multiple sorting rules with the Comparable interface itself. If you want to define multiple sorting criteria, like sorting first by title then cost, and sorting by cost then title, you would need to use a Comparator instead.
- The Comparator interface allows you to define multiple sorting rules. You can have different Comparator implementations for different sorting strategies.

- ***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?***

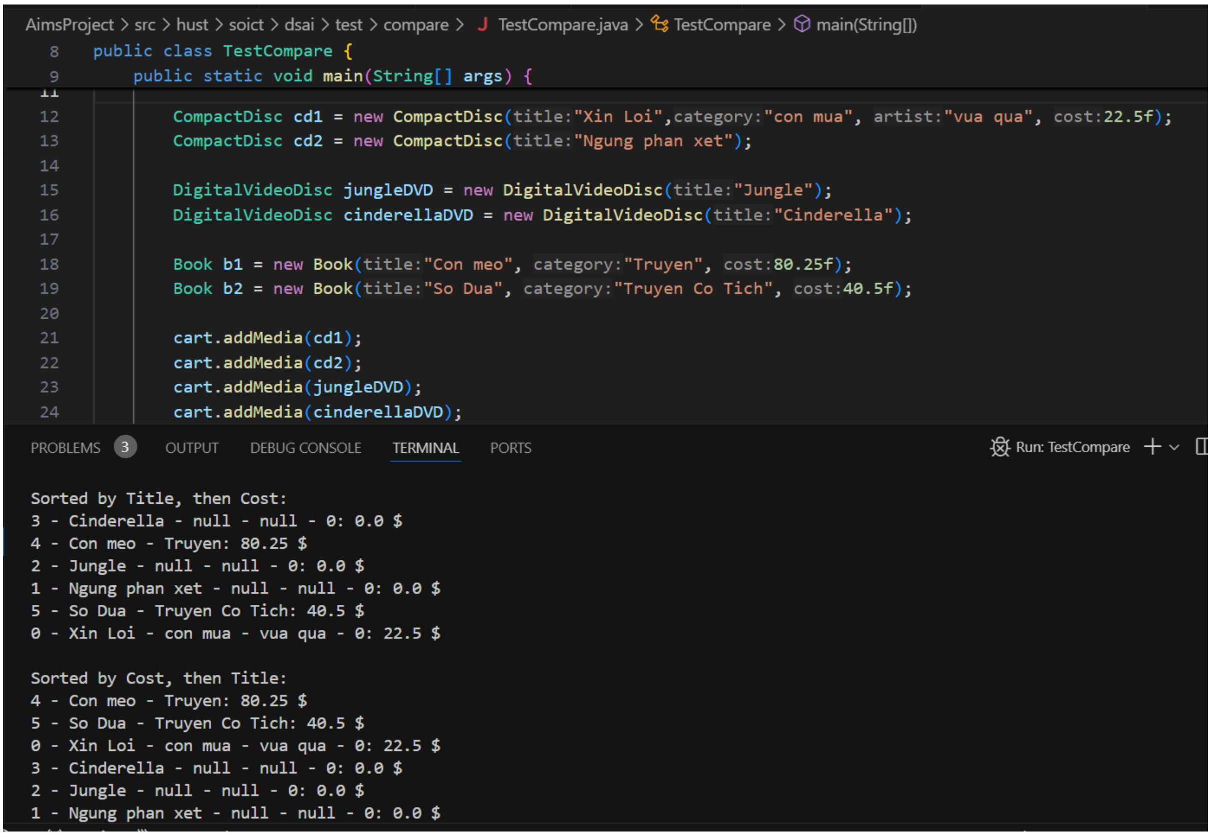
- To allow this, we will override compareTo() method in DigitalVideoDisc class. By this way, DVDs will use the overridden compareTo() method



```
@Override
public int compareTo(Media other) {
    if (!(other instanceof DigitalVideoDisc)) {
        return super.compareTo(other);
    }

    DigitalVideoDisc otherDVD = (DigitalVideoDisc) other;
    int titleComparison = this.getTitle().compareTo(otherDVD.getTitle());
    if (titleComparison != 0) {
        return titleComparison;
    }
    int lengthComparison = Integer.compare(otherDVD.getLength(), this.getLength());
    if (lengthComparison != 0) {
        return lengthComparison;
    }
    return Double.compare(this.getCost(), otherDVD.getCost());
}
```

Test compare:


AimsProject > src > hust > soict > dsai > test > compare > TestCompare.java > TestCompare > main(String[])
PROBLEMS 3
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
Run: TestCompare

```java
public class TestCompare {
    public static void main(String[] args) {
        CompactDisc cd1 = new CompactDisc(title:"Xin Loi",category:"con mua", artist:"vua qua", cost:22.5f);
        CompactDisc cd2 = new CompactDisc(title:"Ngung phan xet");

        DigitalVideoDisc jungleDVD = new DigitalVideoDisc(title:"Jungle");
        DigitalVideoDisc cinderellaDVD = new DigitalVideoDisc(title:"Cinderella");

        Book b1 = new Book(title:"Con meo", category:"Truyen", cost:80.25f);
        Book b2 = new Book(title:"So Dua", category:"Truyen Co Tich", cost:40.5f);

        cart.addMedia(cd1);
        cart.addMedia(cd2);
        cart.addMedia(jungleDVD);
        cart.addMedia(cinderellaDVD);
```

```
Sorted by Title, then Cost:
3 - Cinderella - null - null - 0: 0.0 $
4 - Con meo - Truyen: 80.25 $
2 - Jungle - null - null - 0: 0.0 $
1 - Ngung phan xet - null - null - 0: 0.0 $
5 - So Dua - Truyen Co Tich: 40.5 $
0 - Xin Loi - con mua - vua qua - 0: 22.5 $

Sorted by Cost, then Title:
4 - Con meo - Truyen: 80.25 $
5 - So Dua - Truyen Co Tich: 40.5 $
0 - Xin Loi - con mua - vua qua - 0: 22.5 $
3 - Cinderella - null - null - 0: 0.0 $
2 - Jungle - null - null - 0: 0.0 $
1 - Ngung phan xet - null - null - 0: 0.0 $
```