**ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

BÁO CÁO THỰC HÀNH IT3103 – 744530 - 2024.1

BÀI THỰC HÀNH 4

Họ và tên: NGUYEN TRI NAM

MSSV: 20226093

# Create the Book class

# 

*Figure 1.1: Book Class 1*

# Creating the abstract Media class

Đây sẽ là lớp cha để các lớp DigitalVideoDisc, Book kế thừa.

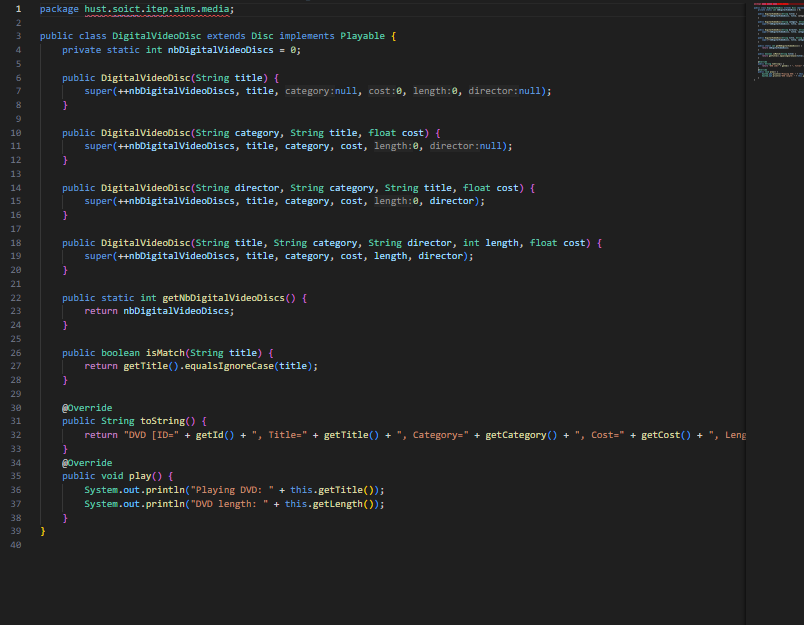


# Creating the CompactDisc class

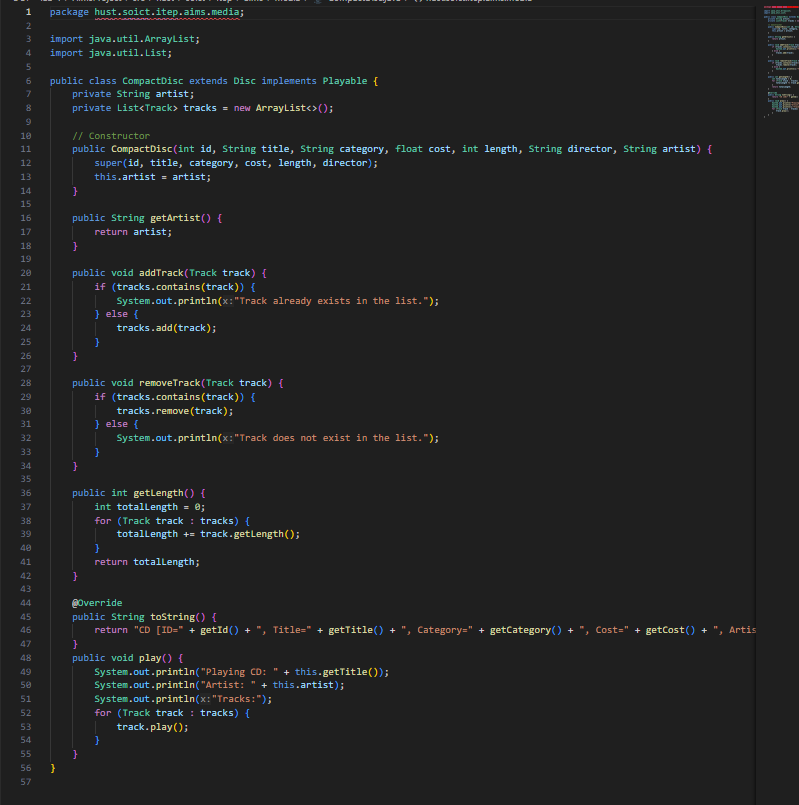
## Create the Disc class extending the Media clas

## 

*Figure 3.1: Disc Class*

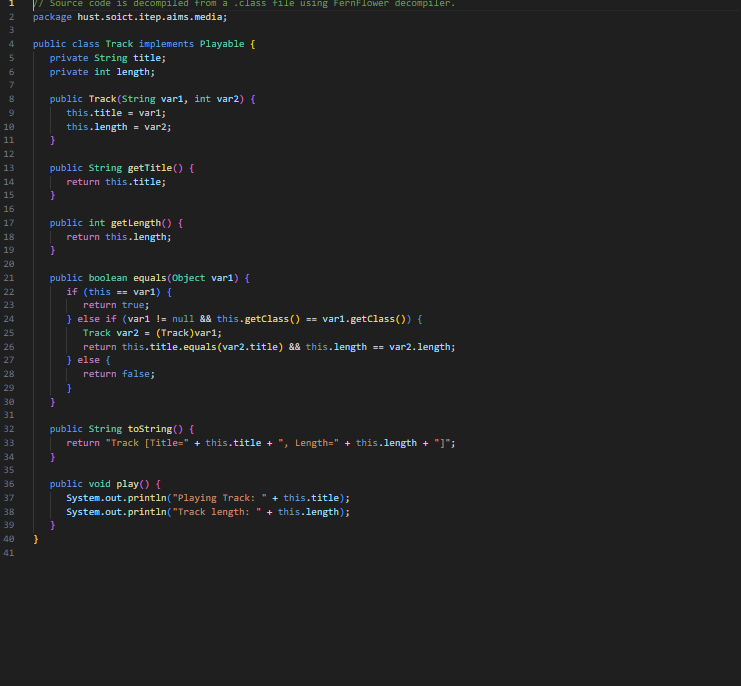


*Figure 3.2: DigitalVideoDisc Class*



*Figure 3.3: CompactDisc Class*

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

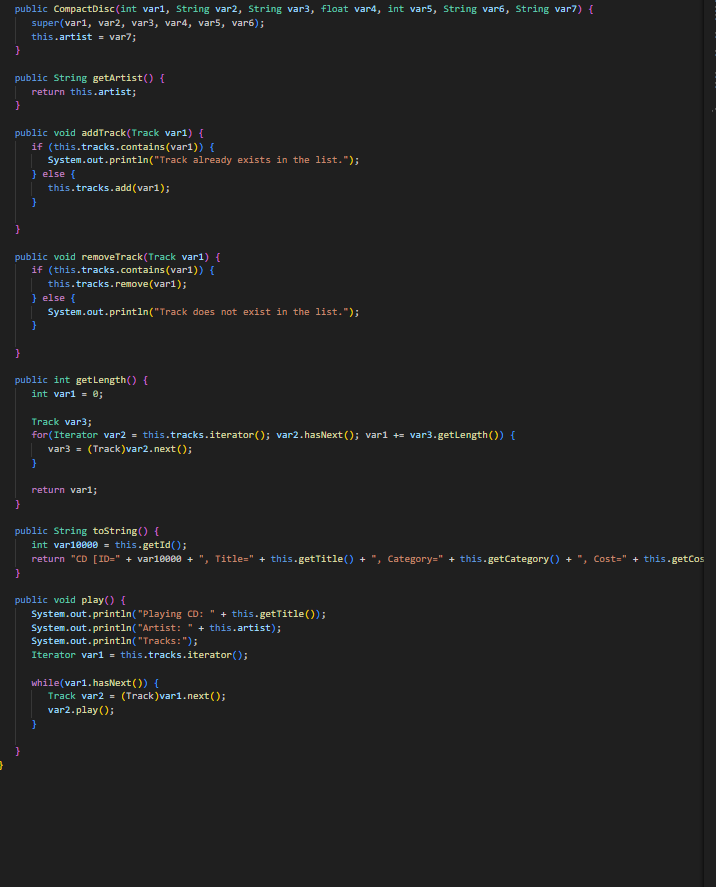


*Figure 3.4: Track Class*

## Open the CompactDisc class

## 

*Figure 3.5: CompactDisc Class 1*



*Figure 3.6: CompactDisc Class 2*

# Create the Playable interface

# 

*Figure 4.1: Playable interface*

Implement play() cho các class DigitalVideoDisc, Track, CompactDisc



*Figure 4.2: Method play() của DigitalVideoDisc*

**

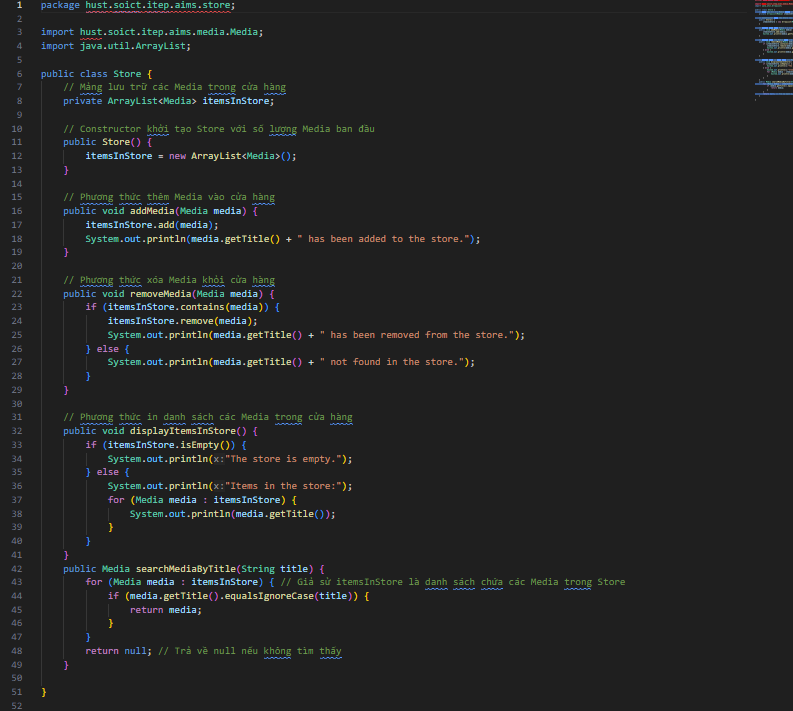
*Figure 4.3: Method play() của Track*

**

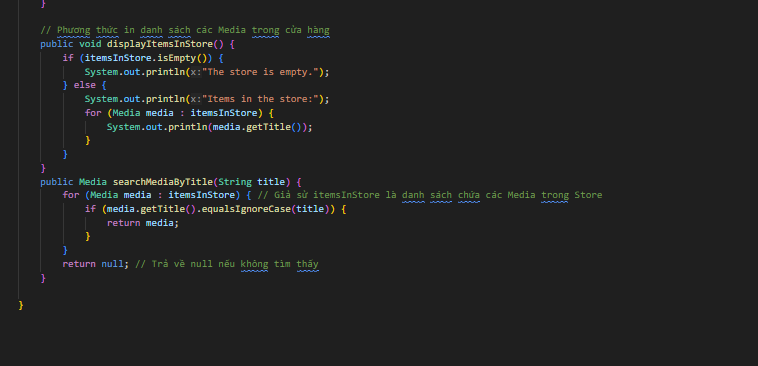
*Figure 4.4: Method play() của CompactDisc*

# Update the Cart class to work with Media

Lớp Cart bây giờ cần có khả năng tương tác với các đối tượng DVD, CD và Book. Vì các lớp DVD, CD và Book đều kế thừa từ lớp Media, nên thay vì làm việc trực tiếp với từng lớp con, lớp cart chỉ cần giao tiếp với lớp Media là có thể hoạt động được với tất cả.

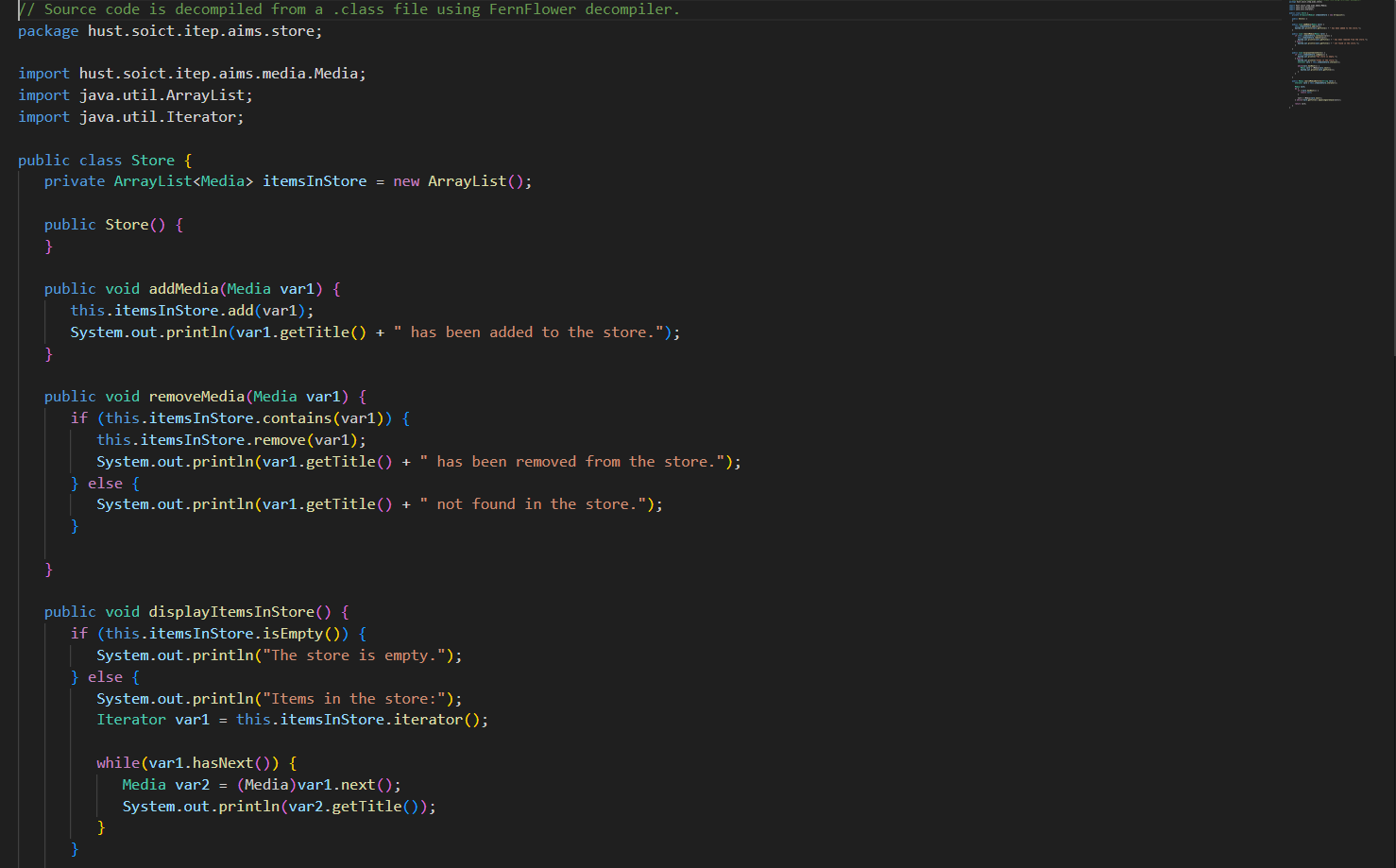


*Figure 5.1: Cart Class 1*

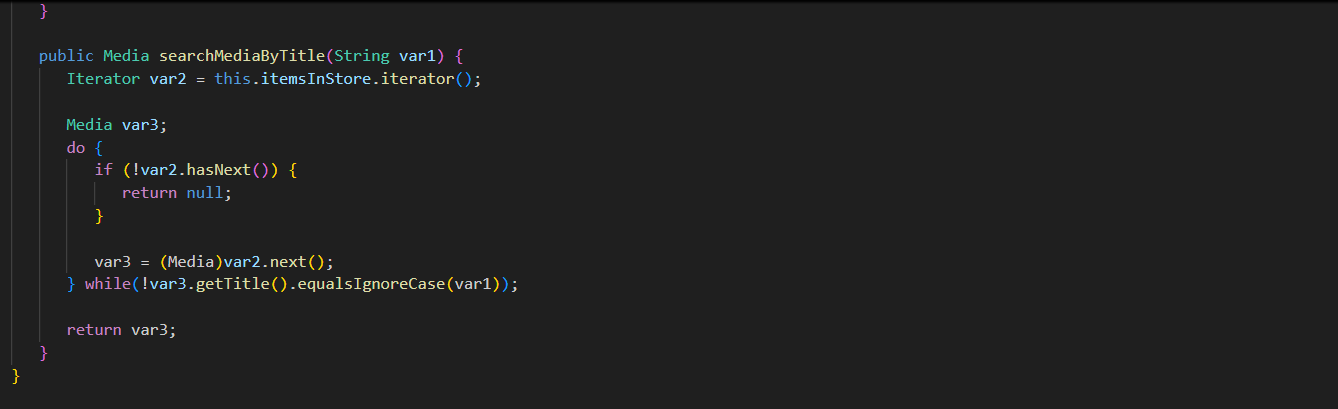


*Figure 5.2: Cart Class 2*

# Update the Store class to work with Media

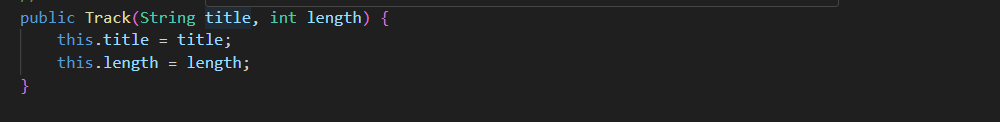


*Figure 6.1: Store Class 1*

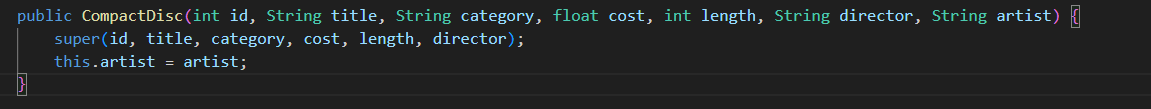


*Figure 6.2: Store Class 2*

# Constructors of whole classes and parent classes

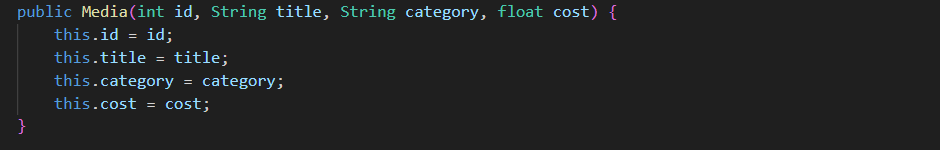


*Figure 7.1: Constructor Track Class*

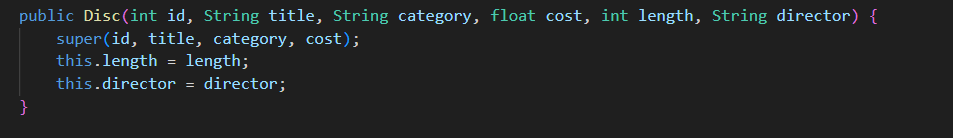
**

*Figure 7.2: Constructor CompactDisc Class*

Lớp Disc kế thừa lớp Media, khi đó lớp Media là lớp cha, lớp Disc là lớp con.



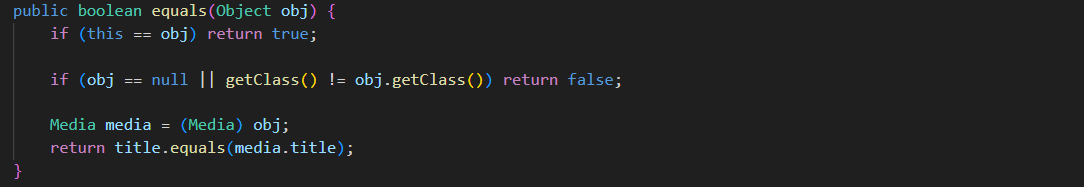
*Figure 7.3: Constructor Media Class*

**

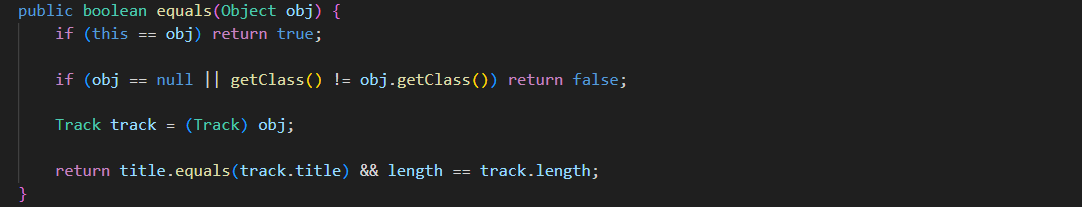
*Figure 7.4: Constructor Disc Class*

# Unique item in a list

Để tránh trùng lặp các phần tử media trong giỏ hàng hoặc các track trong một đĩa CD, chúng ta có thể ghi đè lại phương thức equals() mặc định kế thừa từ lớp Object. Việc này cho phép so sánh bản chất thay vì so sánh vị trí ô nhớ của các đối tượng, qua đó ngăn chặn thêm các phần tử bị trùng lắp vào danh sách.



*Figure 8.1: Override equals in Media Class*

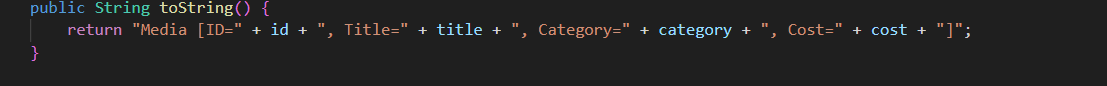
**

*Figure 8.2: Override equals in Track Class*

# Polymorphism with toString() method



*Figure 9.1: Code mô phỏng Polymorphism*



*Figure 9.2: Override toString() in Media Class*

Lớp Media là lớp cơ sở được kế thừa bởi các lớp cụ thể hơn là CompactDisc, DigitalVideoDisc và Book. Khi khởi tạo các đối tượng cd, dvd, book thuộc lớp con rồi gán chúng cho biến kiểu Media, ta áp dụng kỹ thuật gọi là upcasting.

Việc thêm chúng vào danh sách media và duyệt danh sách để in ra thông tin mỗi phần tử bằng phương thức toString() là ví dụ điển hình cho tính đa hình động. Mỗi lớp con có thể cài đặt riêng toString() nên kết quả sẽ khác nhau dựa theo loại đối tượng, mà không cần quan tâm đến kiểu cụ thể của từng phần tử.

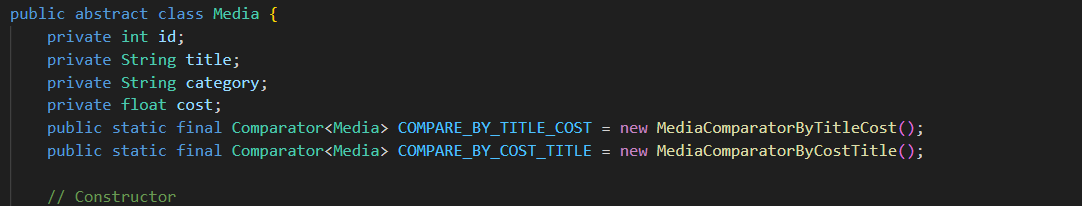
# Sort media in the car

Sắp xếp các media trong giỏ hàng theo hai tiêu chí:

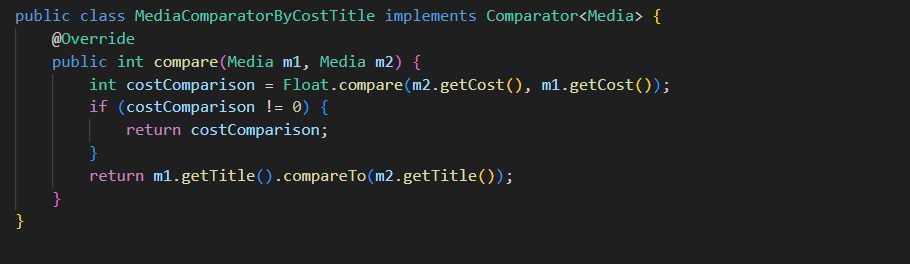
* + Bằng title: Hiển thị tất cả các media theo thứ tự bảng chữ cái. Trong trường hợp cùng title,

media có cost cao hơn sẽ được hiển thị trước.

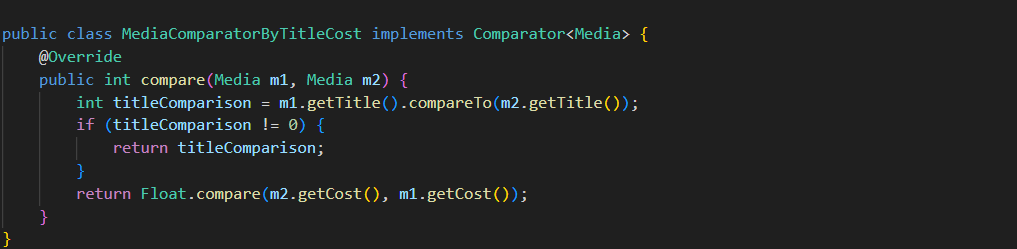
* + Bằng cost: Hiển thị theo thứ tự cost giảm dần. Trong trường hợp cost như nhau, sắp xếp media theo thứ tự bảng chữ cái



*Figure 10.1: Add the comparators as attributes of the Media class*



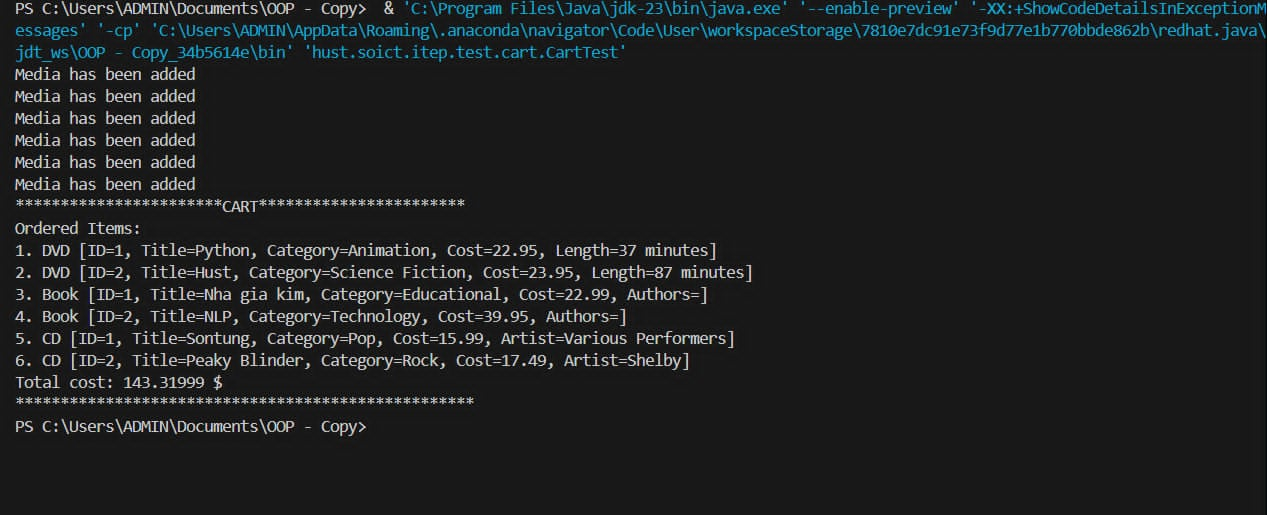
*Figure 10.2: MediaComparatorByCostTitle Class*

**

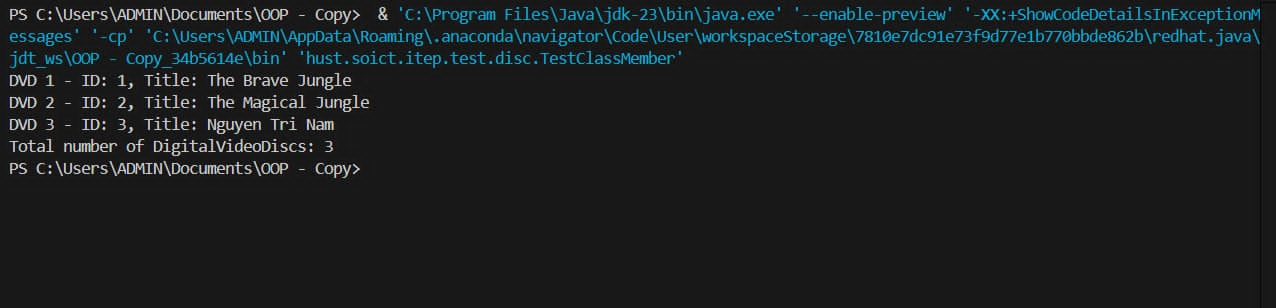
*Figure 10.3: MediaComparatorByTitleCost Class*

# Create a complete console application in the Aims class

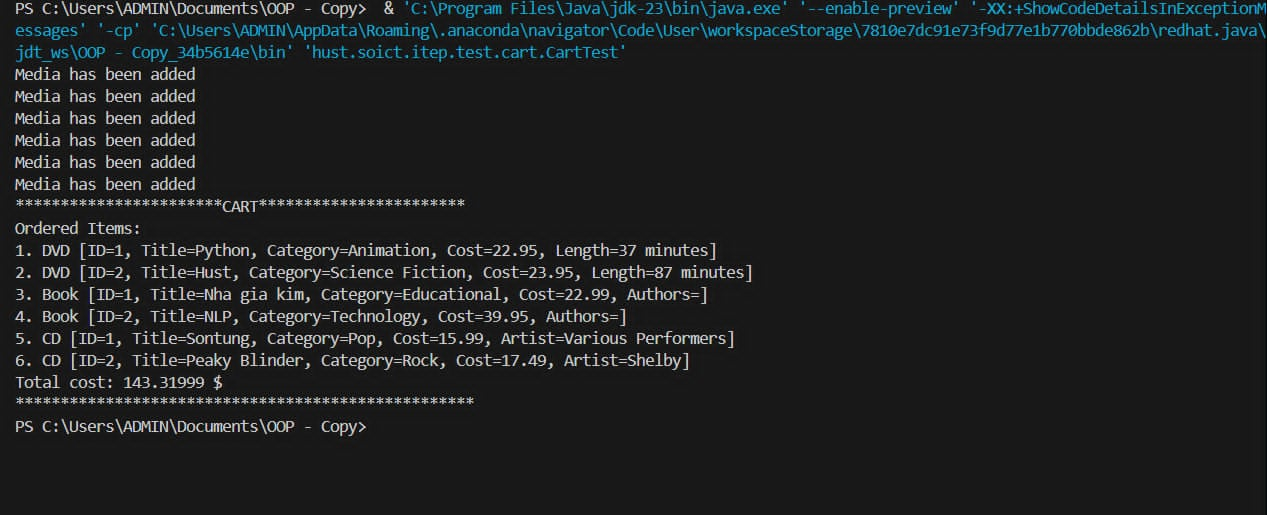
1 cartest.java



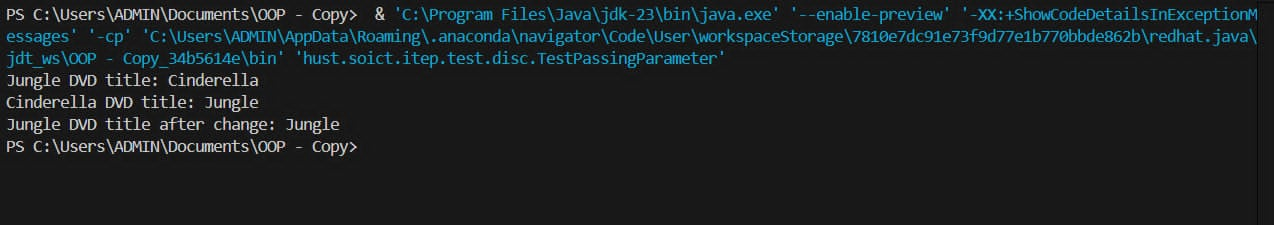
2 testclassmember.java



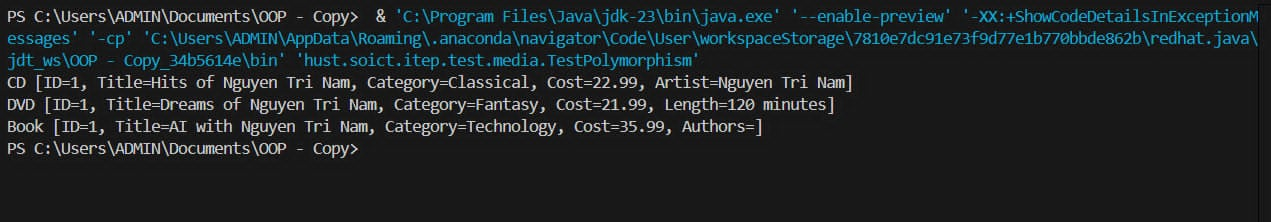
3 cartest.java



4 testpassingparameter.java



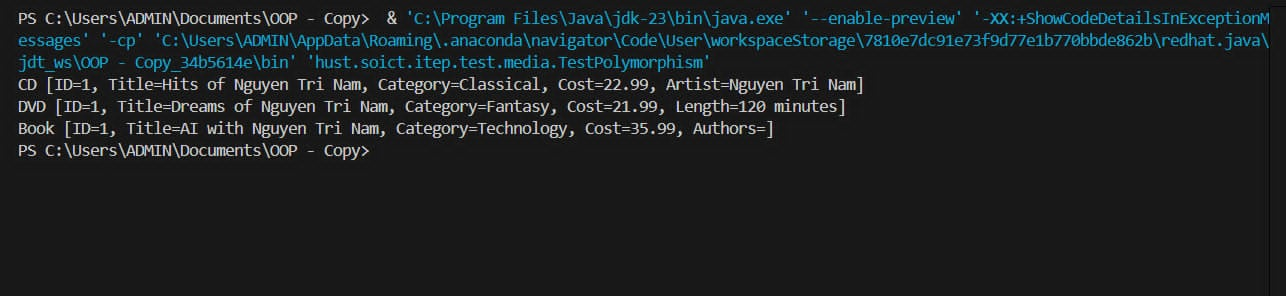
5 testphorlymorphism.java



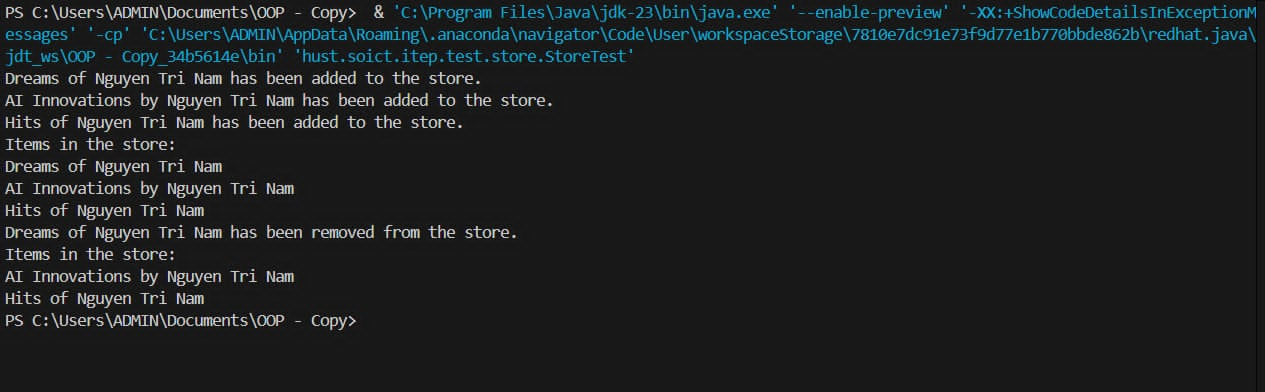
*6 testplayable.java*



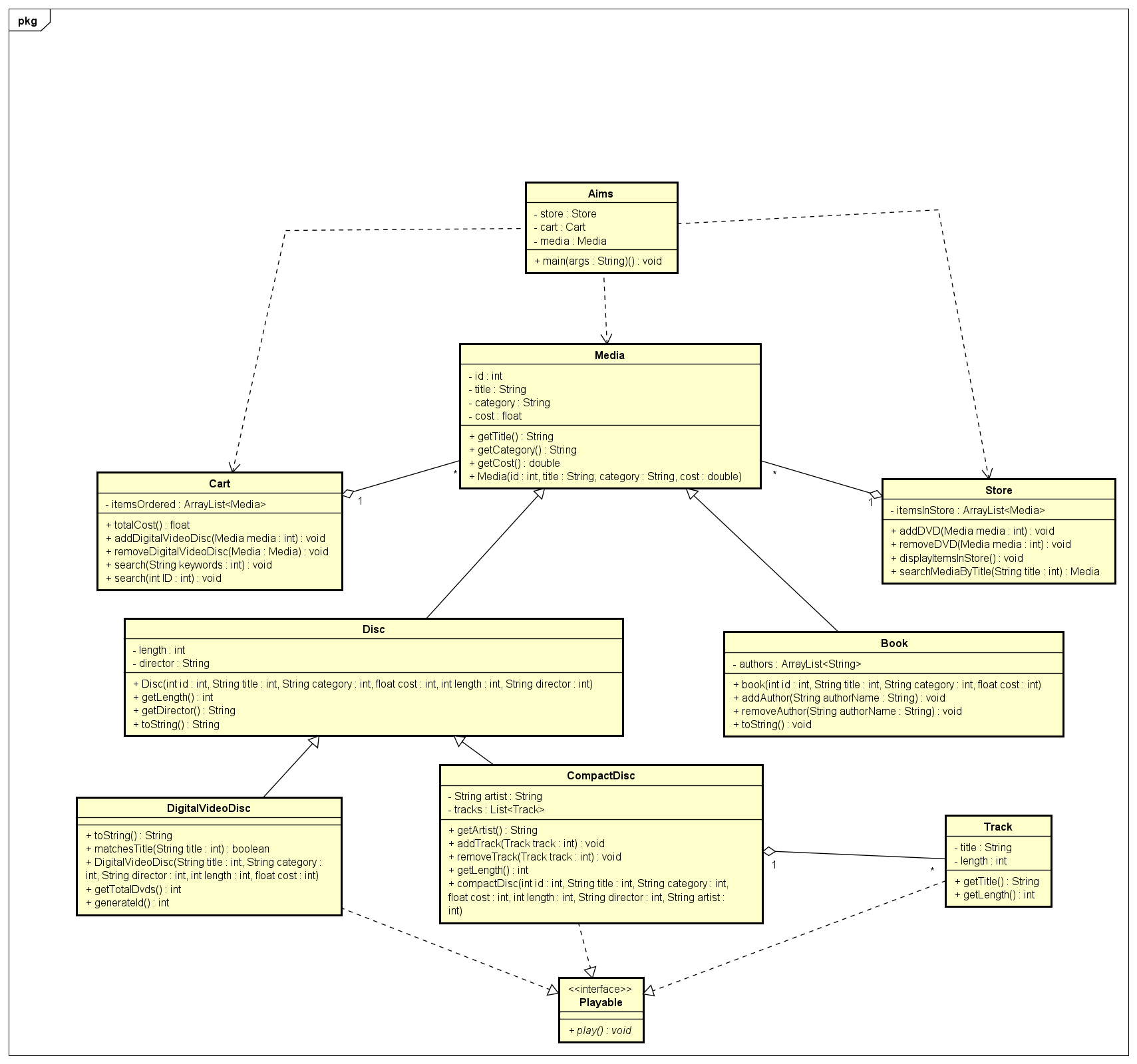
*7 testsort.java*



8 storetest.java



# Class Diagram



*Figure 12.1: Class Diagram*

# UseCase Diagram

# 

*Figure 13.1: UseCase Diagram*

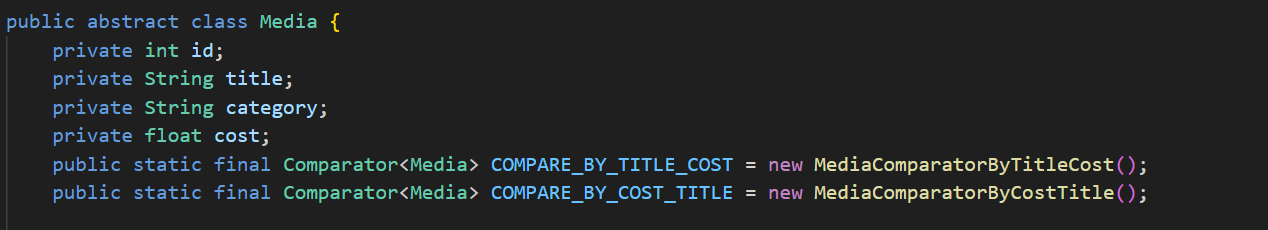
# Answer Questions

Trong trường hợp muốn so sánh các đối tượng Media với nhau bằng cách sử dụng Comparable thay vì Comparator, thì thay vì tạo ra các lớp riêng cho từng Comparator, chúng ta cần để lớp Media triển khai interface Comparable.

* Lớp nào nên triển khai giao diện Comparable?

Trả lời: Lớp nên triển khai giao diện Comparable là lớp chứa đối tượng mà bạn muốn so sánh, trong

trường hợp này, là abstract class “Media”.



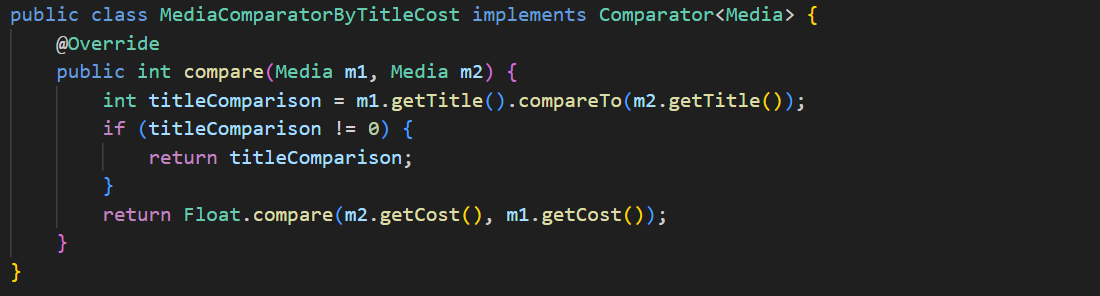
*Figure 14.1: Triển khai Comparable trong lớp trừu tượng Media*

* Trong các lớp đó, phải triển khai phương thức compareTo() như thế nào để phản ánh đúng thứ tự mà chúng ta muốn?

Trả lời: Để triển khai phương thức compareTo() thì cần so sánh các thuộc tính của đối tượng. Đối với yêu cầu sắp xếp theo title sau đó là cost, có thể triển khai như sau:



*Figure 14.2: Mở rộng để so sánh nhiều thuộc tính hơn*

**

*Figure 14.3: Triển khai tại lớp con (Disc)*

Cách triển khai này giúp chúng ta linh hoạt hơn khi so sánh các đối tượng Media và cung cấp khả năng

mở rộng cho các lớp con khác nếu cần thiết.

Liệu có thể có hai quy tắc sắp xếp (theo title sau đó là cost và theo cost sau đó là title) nếu sử dụng cách tiếp cận này với giao diện Comparable?

Trả lời: Không, với giao diện Comparable, bạn chỉ có thể có một quy tắc sắp xếp cho mỗi lớp.

Điều này là do phương thức compareTo() chỉ trả về một giá trị int.