NGUYỄN TRUNG KIÊN 20226110

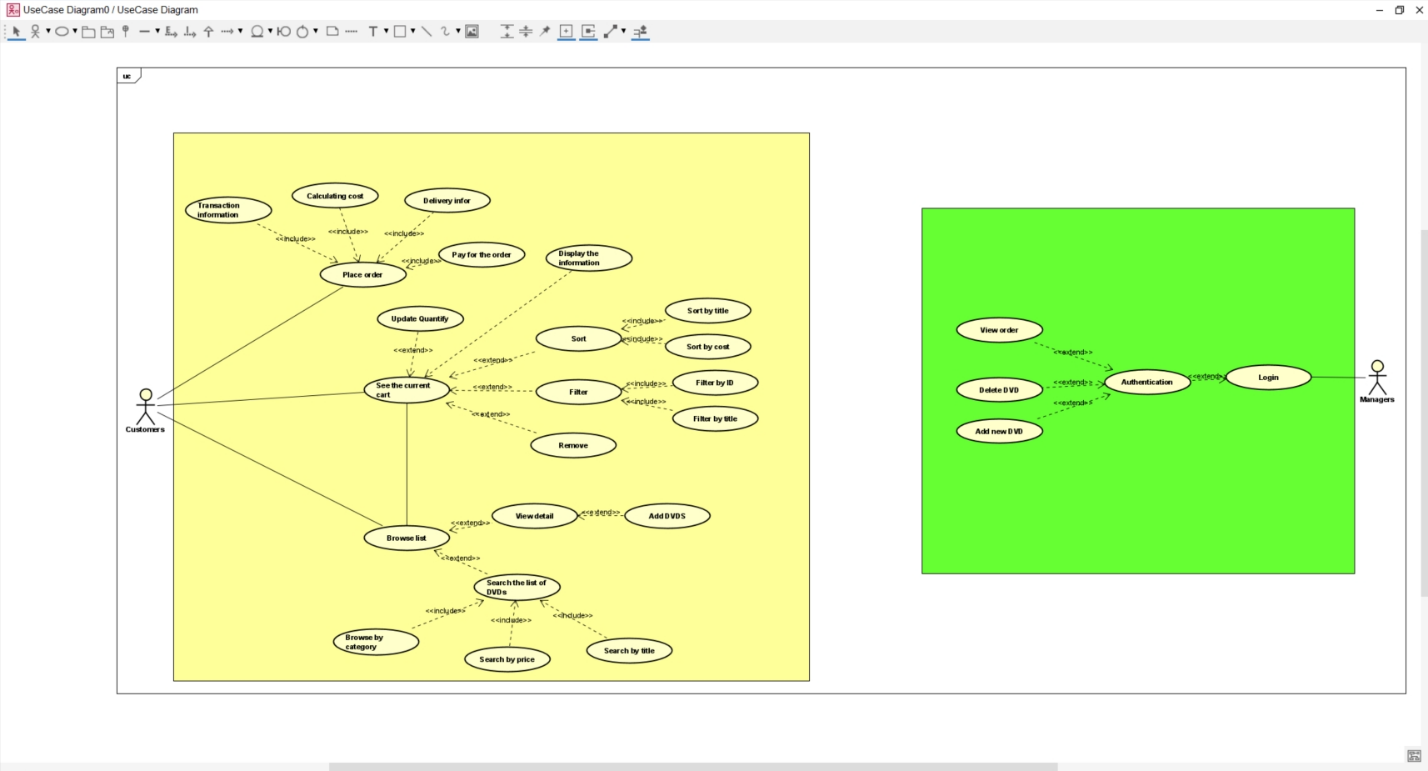
**Câu hỏi:**

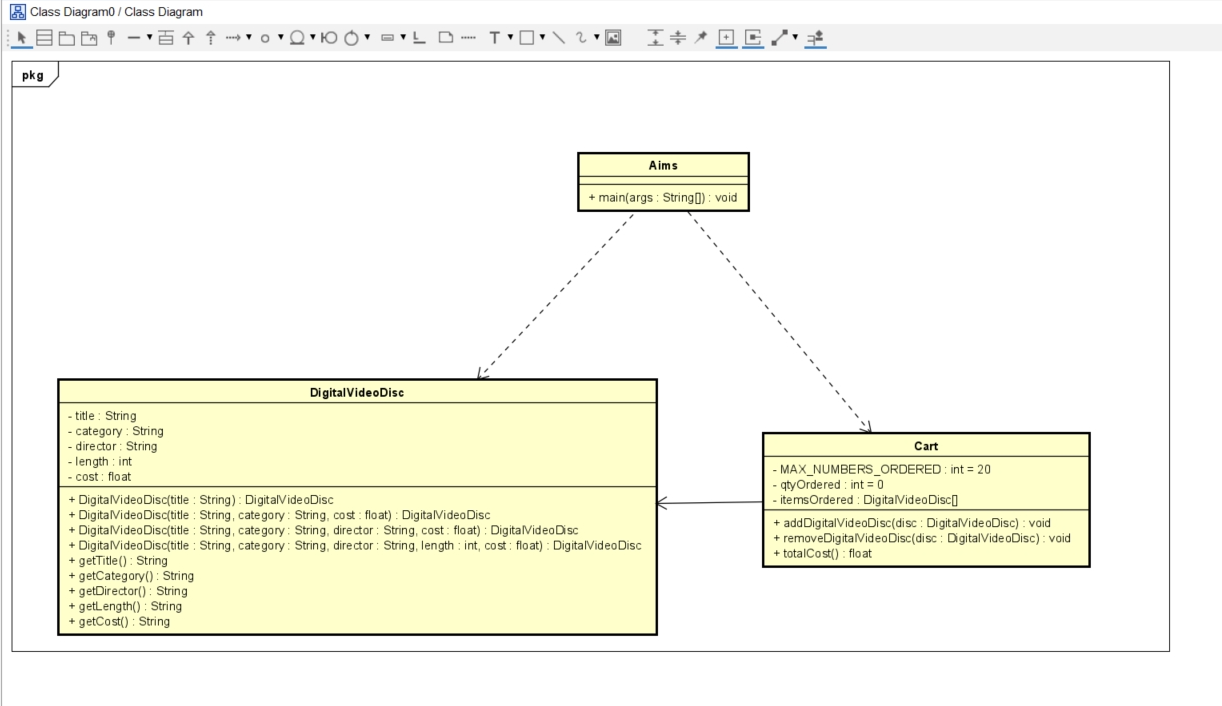
* Nếu bạn tạo một phương thức khởi tạo để xây dựng đối tượng DVD theo tiêu đề, sau đó tạo một phương thức khởi tạo để xây dựng đối tượng DVD theo thể loại, liệu Java có cho phép bạn thực hiện điều này không?

**Câu trả lời:**

* Java cho phép thực hiện điều này thông qua cơ chế nạp chồng phương thức (method overloading). Bạn có thể tạo nhiều phương thức khởi tạo với cùng tên nhưng có danh sách tham số khác nhau để xây dựng đối tượng DVD theo các cách khác nhau, ví dụ như theo tiêu đề hoặc theo thể loại.

**Các biểu đồ:**





**Phân tích về việc sử dụng Getter/Setter trong lập trình hướng đối tượng**

Biểu đồ nêu bật các quan điểm về việc dùng getter và setter trong lập trình hướng đối tượng (OOP). Mặc dù hai phương thức này khá phổ biến, chúng không hoàn toàn phù hợp với bản chất của lập trình OOP. Việc lạm dụng getter và setter có thể là dấu hiệu của thiết kế kém tinh tế, làm giảm tính trừu tượng của dữ liệu – một yếu tố quan trọng trong OOP.

1. **Bản chất của thiết kế**

Phương thức getter và setter có thể hữu ích hoặc không, tùy thuộc vào bối cảnh. Theo nguyên tắc OOP, đối tượng không nên để lộ chi tiết thực thi của nó. Nếu có quá nhiều getter và setter, hệ thống có thể trở nên cứng nhắc, khó bảo trì. Nên tập trung thiết kế đối tượng sao cho các thay đổi nội bộ không ảnh hưởng đến các phần khác của mã nguồn.

1. **Trừu tượng hóa dữ liệu**

Hệ thống OOP chuẩn cần bảo vệ đối tượng khỏi sự phụ thuộc vào chi tiết thực thi. Sử dụng nhiều accessor sẽ phá vỡ tính trừu tượng, làm giảm tính hướng đối tượng của hệ thống. Getter và setter khi bị lạm dụng sẽ chỉ làm lộ chi tiết thực thi thay vì bảo vệ dữ liệu.

1. **Khi nào nên sử dụng accessor**

Có lúc việc dùng getter và setter là không thể tránh khỏi, nhưng một lập trình viên kinh nghiệm có thể loại bỏ tới 99% các phương thức này mà vẫn giữ được sự toàn vẹn của hệ thống. Người thiết kế cần biết khi nào nên che giấu chi tiết thực thi để tránh phụ thuộc vào nó.

**Kết luận**

Getter và setter không phải lúc nào cũng xấu, nhưng nếu lạm dụng chúng, tính trừu tượng và chất lượng của hệ thống OOP sẽ bị ảnh hưởng đáng kể.

