OOP - Lab 03 Reading Assignment

Lê Minh Hiếu - 20235710
Mã lớp 750863

1. Write a toString() method for the DigitalVideoDisc class. What should be the return type of this method?

-> String

2. What are the advantages of Polymorphism?

-> Cho phép viết mã linh hoạt và mở rộng hơn, giảm sự phụ thuộc giữa các lớp, dễ bảo trì và mở rộng hệ thống khi thêm loại mới

3. How is Inheritance useful to achieve Polymorphism in Java?

-> Khi muốn xử lý chung cho nhiều đối tượng có bản chất khác nhau nhưng cùng kế thừa từ một lớp cha, trong design patterns như Strategy, Factory, hoặc khi làm việc với danh sách tổng quát như List<Media>

4. What are the differences between Polymorphism and Inheritance in Java?

-> Inheritance là cơ chế kế thừa từ lớp cha, tái sử dụng mã nguồn, dùng từ khóa extends hoặc implements. CÒn polymorphism là hành vi cho phép gọi đúng method, đảm bảo hành vi phù hợp tại thời điểm chạy chương trình và sử dụng cùng method ở các lớp con

5. Alternatively, to compare items in the cart, instead of using the Comparator class I have mentioned, you can use the Comparable interface[1] 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?

   -> Lớp `Media` (hoặc bất kỳ lớp nào đại diện cho các mục trong giỏ hàng mà bạn muốn so sánh) nên triển khai interface `Comparable<Media>`. Điều này cho phép các đối tượng của lớp đó có thể so sánh được với nhau.

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

   -> Phương thức `compareTo(Media other)` trong lớp `Media` nên chứa logic so sánh dựa trên tiêu chí sắp xếp mong muốn. Ví dụ, nếu muốn sắp xếp theo tiêu đề, bạn sẽ so sánh tiêu đề của đối tượng hiện tại (`this`) với tiêu đề của đối tượng `other`. Nếu muốn sắp xếp theo chi phí, em sẽ so sánh chi phí. Phương thức này nên trả về:

   + Một số âm nếu đối tượng hiện tại nhỏ hơn đối tượng `other`.
   + Số 0 nếu hai đối tượng bằng nhau.
   + Một số dương nếu đối tượng hiện tại lớn hơn đối tượng `other`. Để triển khai thứ tự theo tiêu đề trước rồi đến chi phí, em sẽ so sánh tiêu đề trước. Nếu tiêu đề bằng nhau, em sẽ tiếp tục so sánh chi phí.

- 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?
-> Không, với cách tiếp cận sử dụng interface `Comparable`, một lớp chỉ có thể có *một* cách triển khai `compareTo()`, do đó chỉ định nghĩa *một* quy tắc sắp xếp "tự nhiên" cho các đối tượng của lớp đó. Interface `Comparable` được thiết kế để cung cấp thứ tự mặc định hoặc chính yếu cho một lớp.

- Suppose the DVDs have 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?
-> Vì interface `Comparable` chỉ cho phép một quy tắc sắp xếp tự nhiên duy nhất cho một lớp nên không thể sử dụng chỉ mình `Comparable` để xử lý các quy tắc sắp xếp khác nhau cho các loại `Media` (như DVD so với các loại khác). Để xử lý các quy tắc sắp xếp đa dạng và cụ thể cho từng trường hợp (như quy tắc đặc biệt cho DVD), ta *vẫn* cần sử dụng interface `Comparator`. Em sẽ tạo các lớp `Comparator` riêng biệt cho từng quy tắc sắp xếp khác nhau (ví dụ: một `Comparator` cho sắp xếp chung, và một `Comparator` khác cho sắp xếp DVD theo tiêu đề, độ dài giảm dần, chi phí). Khi sắp xếp một danh sách chứa các loại `Media` khác nhau, em sẽ sử dụng phương thức `Collections.sort()` (hoặc các phương thức sắp xếp tương tự) và truyền vào instance của `Comparator` phù hợp với quy tắc sắp xếp mà em muốn áp dụng tại thời điểm đó.