Lab 03: Object-Oriented Techniques

Reading Assignment: Please answer three questions below:

- What are the advantages of Polymorphism?
  +Code linh hoạt: Cùng 1 method hoặc biến, có thể làm việc với nhiều kiểu đối tượng khác nhau.
  +Tái sử dụng code: Không cần viết lại method cho từng loại con. Ví dụ play() cho DVD, CD đều gọi chung Playable.play().
  +Dễ mở rộng: Thêm class mới không cần sửa code cũ, chỉ cần override hành vi.
  +Giảm coupling (liên kết chặt chẽ): Các module (class) ít phụ thuộc chi tiết vào nhau, dễ bảo trì, dễ test.
  +Tăng khả năng maintain: Dễ thêm, sửa đối tượng mới trong tương lai.
- How is Inheritance useful to achieve Polymorphism in Java?
  + Inheritance (Kế thừa) là nền tảng để có Polymorphism (Đa hình).
  + Kế thừa cho phép class con mở rộng từ class cha. Khi kế thừa, lớp con được coi như là lớp cha.
  + Một Media có thể là DigitalVideoDisc, CompactDisc, v.v.
  + Dùng polymorphism dễ dàng:
  + Media media1 = new DigitalVideoDisc(...);
  + Media media2 = new CompactDisc(...);
  + Khi gọi method, dùng tham chiếu Media nhưng hành vi thực tế sẽ là của DVD hay CD
  + Không có inheritance, sẽ không có khả năng đa hình kiểu này.
- What are the differences between Polymorphism and Inheritance in Java?

Tiêu chí	**Inheritance (Kế thừa)**	**Polymorphism (Đa hình)**
Định nghĩa	Một class kế thừa thuộc tính/method của class khác.	Một đối tượng có thể có nhiều hình thức khác nhau.
Mục tiêu chính	Tái sử dụng code, giảm lặp lại.	Cho phép code làm việc với nhiều kiểu đối tượng linh hoạt
Cách hoạt động	Dùng extends hoặc implements từ class/interface.	Dùng chung method reference, nhưng hành vi khác nhau tại runtime.
Quan hệ	Quan hệ cha - con (class hierarchy).	Quan hệ sử dụng (method call khác nhau tùy loại thực tế).
Ví dụ đơn giản	DigitalVideoDisc extends Disc.	Disc disc = new DigitalVideoDisc(); disc.play();
Cần cái gì để xảy ra?	Tính kế thừa (inheritance)	Tính kế thừa và override method

Question: Alternatively, to compare items in the cart, instead of using the Comparator class I have mentioned, you can use the Comparable interface 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?
- In those classes, how should you implement the compareTo()method to reflect the ordering that we want?
- 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?
- 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?

1.What class should implement the Comparable interface?

+ Class Media nên implement Comparable<Media>. Vì Cart chứa ArrayList<Media>.
+ Khi Collections.sort(cart), Java cần biết so sánh 2 Media như thế nào.
+ Nếu Media implements Comparable<Media>, mọi loại con (DVD, CD, Book) sẽ kế thừa sẵn so sánh mặc định.

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

+ Sắp xếp theo title (alphabetical), nếu title trùng, thì sắp giá cao hơn đứng trước.

```
@Override
public int compareTo(Media other) {
   int titleCompare = this.title.compareToIgnoreCase(other.title);
   if (titleCompare != 0) {
      return titleCompare;
   }
   return Float.compare(other.cost, this.cost);
}
```

3. Can we have two ordering rules (by title then cost and by cost then title) if we use this Comparable interface approach?

+ Không thể, vì comparable chỉ cho phép một quy tắc so sánh mặc định duy nhất trong compareTo().
+ Nếu muốn nhiều kiểu sắp xếp (title-cost hoặc cost-title), phải dùng Comparator thay vì Comparable.
+ Chỉ Comparator mới hỗ trợ đa dạng sorting tùy chọn.

4. 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?

+ Override compareTo() trong class DigitalVideoDisc riêng.

+ Không xài compareTo() kế thừa từ Media nữa.

```
@Override
public int compareTo(Media other) {
    if (other instanceof DigitalVideoDisc) {
        DigitalVideoDisc otherDVD = (DigitalVideoDisc) other;
        int titleCompare = this.title.compareToIgnoreCase(otherDVD.title);
        if (titleCompare != 0) {
            return titleCompare;
        }
        int lengthCompare = Integer.compare(otherDVD.length, this.length); // length giảm dần
        if (lengthCompare != 0) {
            return lengthCompare;
        }
        return Float.compare(this.cost, otherDVD.cost);
    } else {
        return super.compareTo(other);
    }
}
```

+ Nếu other là DVD:
+ So sánh title trước ➔ nếu trùng thì
+ So length giảm dần ➔ nếu trùng nữa thì
+ So cost tăng dần.
+ Nếu other không phải DVD, fallback gọi super.compareTo(other) (so title-cost bình thường).