**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 OOP LAB

Họ và tên sv: Nguyễn Thanh Hưng

MSSV: 20225633  
Lớp: Việt Nhật 03 K67  
GVHD: Lê Thị Hoa

HTGD: Bùi Trọng Dũng

Hà Nội 11/2024

Table of Contents

[1. Working with method overloading 4](#_Toc183030181)

[1.1. Overloading by differing types of parameter 4](#_Toc183030182)

[1.2. Overloading by differing the number of parameters 5](#_Toc183030187)

[1.3. Explain 5](#_Toc183030190)

[2. Passing parameter 6](#_Toc183030191)

[2.1. Question: *Is JAVA a Pass by Value or a Pass by Reference programming language?* 6](#_Toc183030192)

[2.2. Trước khi sửa lại Swap 6](#_Toc183030193)

[2.3. Cách giải quyết 8](#_Toc183030196)

[3. Classifier Member and Instance Member 8](#_Toc183030199)

[4. Open the Cart class 10](#_Toc183030202)

[5. Implement the Store class 13](#_Toc183030209)

[6. String, StringBuilder and StringBuffer 15](#_Toc183030213)

[7. Class Diagram 17](#_Toc183030218)

Table of Figures

[Figure 1: Method add Digital Video Disc (dvdList) 4](#_Toc183030183)

[Figure 2: Result of method add Digital Video Disc (dvdList) 4](#_Toc183030184)

[Figure 3: Method add Digital Video Disc (discs) 4](#_Toc183030185)

[Figure 4: Result of method add Digital Video Disc (discs) 5](#_Toc183030186)

[Figure 5: Method add Digital Video Disc (dvd1, dvd2) 5](#_Toc183030188)

[Figure 6: Result of method add Digital Video Disc (dvd1, dvd2) 5](#_Toc183030189)

[Figure 7: Passing parameter code 7](#_Toc183030194)

[Figure 8: Result of passing parameter code 7](#_Toc183030195)

[Figure 9: Swap code 8](#_Toc183030197)

[Figure 10: Result of swap code 8](#_Toc183030198)

[Figure 11: Class member and Instance member 10](#_Toc183030200)

[Figure 12: Result of class member and Instance member 10](#_Toc183030201)

[Figure 13: Code method to print the content in Cart 11](#_Toc183030203)

[Figure 14: Code method to search by ID in Cart 11](#_Toc183030204)

[Figure 15: Code method to search by Title in Cart 11](#_Toc183030205)

[Figure 16: IsMatch() in DigitalVideoDisc class 12](#_Toc183030206)

[Figure 17: CartTest class 12](#_Toc183030207)

[Figure 17: CartTest class 13](#_Toc183030208)

[Figure 18: Store class 13](#_Toc183030210)

[Figure 19: StoreTest class 14](#_Toc183030211)

[Figure 20: Result of Store 15](#_Toc183030212)

[Figure 21: ConcatenationInLoops Class 15](#_Toc183030214)

[Figure 22: Result of ConcatenationInLoops Class 15](#_Toc183030215)

[Figure 23: GarbageCreator Class 16](#_Toc183030216)

[Figure 24: NoGarbage Class 16](#_Toc183030217)

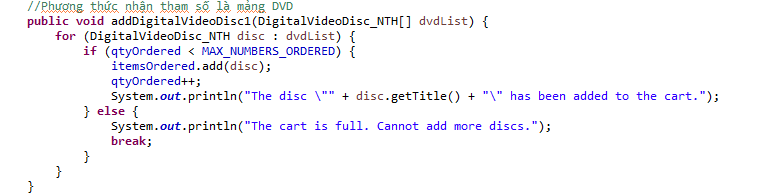
[Figure 24: Class Diagram 17](#_Toc183030219)

BÁO CÁO THỰC HÀNH LAB03

LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

1. Working with method overloading
   1. Overloading by differing types of parameter

Code\_1: Phương thức nhận tham số là mảng DVD:



### Figure 1: Method add Digital Video Disc (dvdList)

Result\_1:

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, hàng

Mô tả được tạo tự động

### Figure 2: Result of method add Digital Video Disc (dvdList)

Code\_2: Phương thức nhận tham số là danh sách số lượng tùy ý (varargs):

Ảnh có chứa văn bản, Phông chữ, hàng, ảnh chụp màn hình

Mô tả được tạo tự động

### Figure 3: Method add Digital Video Disc (discs)

Result\_2:

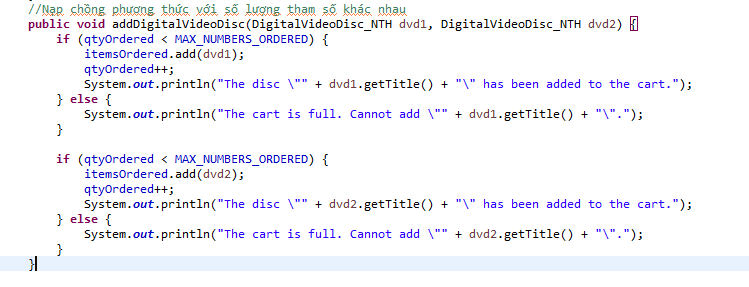
Ảnh có chứa văn bản, Phông chữ, ảnh chụp màn hình, hàng

Mô tả được tạo tự động

### Figure 4: Result of method add Digital Video Disc (discs)

* 1. Overloading by differing the number of parameters

Code:



### Figure 5: Method add Digital Video Disc (dvd1, dvd2)

Result:

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, hàng

Mô tả được tạo tự động

### Figure 6: Result of method add Digital Video Disc (dvd1, dvd2)

* 1. Explain

**So sánh giữa mảng và varargs**:

* **Mảng**: Cần thiết khi bạn đã có một mảng các đối tượng DVD từ trước.
* **Varargs**: Linh hoạt hơn, giúp người dùng truyền bao nhiêu tham số tùy ý mà không cần tạo mảng trước. **Ưu tiên sử dụng varargs** vì tính tiện lợi và dễ đọc.

1. Passing parameter
   1. Question: *Is JAVA a Pass by Value or a Pass by Reference programming language?*

* **Pass by Value**: Giá trị của tham số được sao chép sang một biến khác, và bản sao đó được truyền vào phương thức. Vì vậy, bất kỳ thay đổi nào trong phương thức đều không ảnh hưởng đến giá trị gốc.
* **Pass by Reference**: Tham chiếu đến tham số thực tế được truyền vào phương thức, nên bất kỳ thay đổi nào trong phương thức sẽ ảnh hưởng trực tiếp đến tham số gốc.
  1. Trước khi sửa lại Swap

Code:

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

### Figure 7: Passing parameter code

Result:

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

### Figure 8: Result of passing parameter code

* After the call of swap(jungleDVD, cinderellaDVD) why does the title of these two objects still remain?
  + - Java truyền tham số theo giá trị (Pass by Value). Ở đây, giá trị là tham chiếu (reference) đến đối tượng. Khi thực hiện swap(), các tham chiếu được hoán đổi cục bộ trong phương thức, nhưng không ảnh hưởng đến tham chiếu gốc bên ngoài.
* After the call of changeTitle(jungleDVD, cinderellaDVD.getTitle()) why is the title of the JungleDVD changed?
  + - Khi truyền tham số jungleDVD vào phương thức changeTitle, giá trị của tham chiếu đến đối tượng được truyền. Mặc dù tham số này được sao chép, phương thức có thể thay đổi nội dung của đối tượng mà tham chiếu trỏ đến (ở đây là thay đổi title của jungleDVD).
  1. Cách giải quyết

Sử dụng một mảng để hoán đổi tham chiếu thực tế.

Code:

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, hàng

Mô tả được tạo tự động

### Figure 9: Swap code

Result:

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

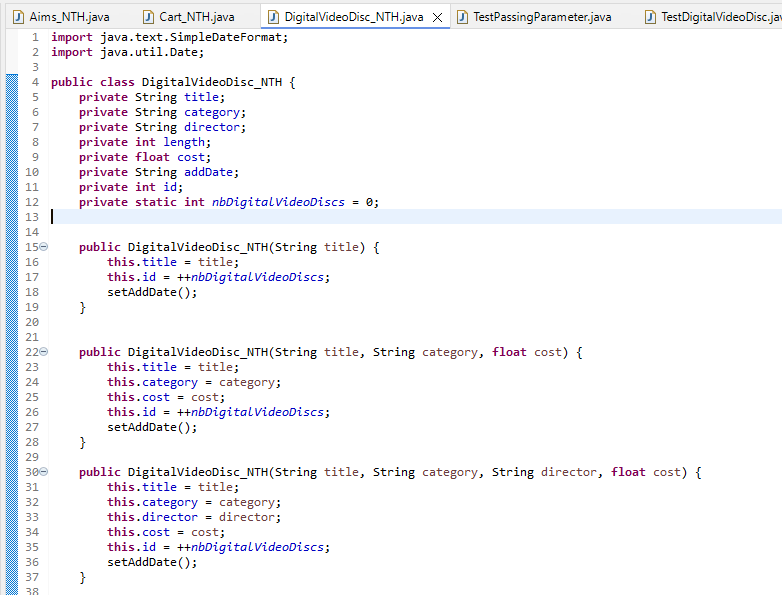
Mô tả được tạo tự động

### Figure 10: Result of swap code

* + - Java là ngôn ngữ Pass by Value, ngay cả khi làm việc với tham chiếu đối tượng. Tham chiếu được sao chép và truyền vào phương thức, nhưng không thay đổi tham chiếu gốc.
    - Để hoán đổi hai đối tượng thực sự, cần sử dụng cấu trúc bên ngoài (như mảng) để thay đổi tham chiếu thực tế.

1. Classifier Member and Instance Member

Code:



Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

### Figure 11: Class member and Instance member

Result:

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

### Figure 12: Result of class member and Instance member

1. Open the Cart class

Code:

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, hàng

Mô tả được tạo tự động

### Figure 13: Code method to print the content in Cart

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, hàng

Mô tả được tạo tự động

### Figure 14: Code method to search by ID in Cart

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, hàng

Mô tả được tạo tự động

### Figure 15: Code method to search by Title in Cart

Ảnh có chứa văn bản, Phông chữ, hàng, ảnh chụp màn hình

Mô tả được tạo tự động

### Figure 16: IsMatch() in DigitalVideoDisc class

Ảnh có chứa văn bản, ảnh chụp màn hình, hàng, Phông chữ

Mô tả được tạo tự động

### Figure 17: CartTest class

Result:

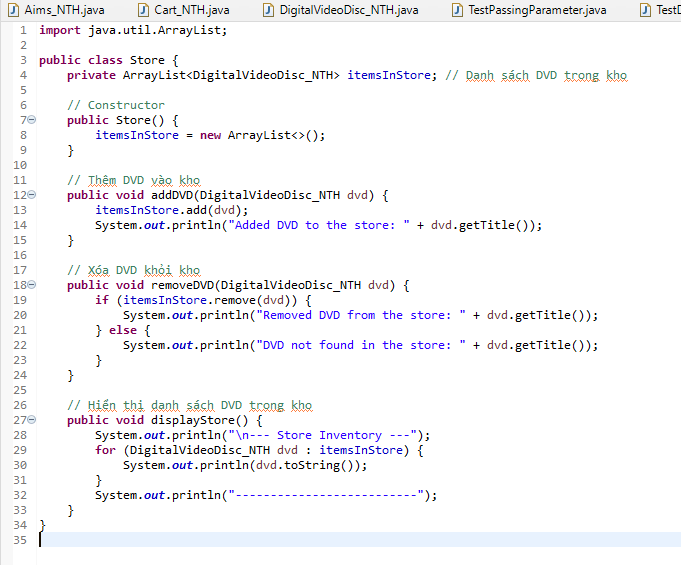
Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

### Figure 17: CartTest class

1. Implement the Store class

Code:



### Figure 18: Store class

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, phần mềm

Mô tả được tạo tự động

### Figure 19: StoreTest class

Result:

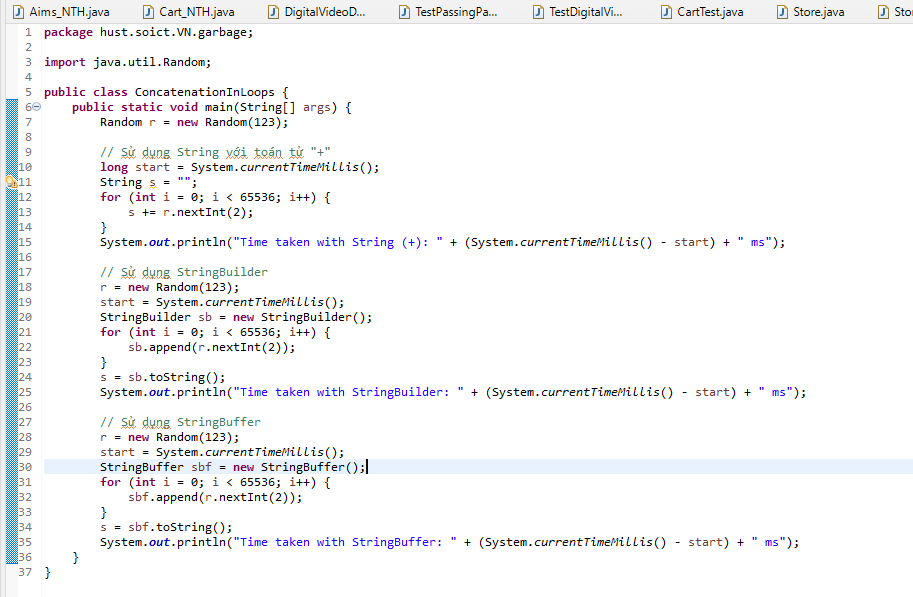
Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

### Figure 20: Result of Store

1. String, StringBuilder and StringBuffer

Code: ConcatenationInLoops

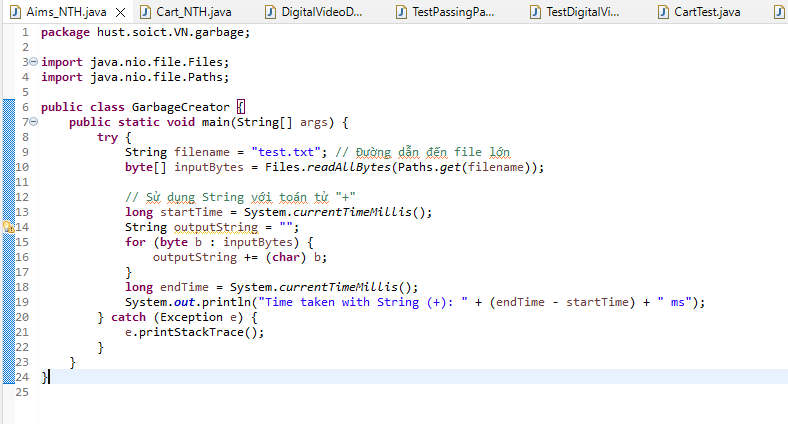


### Figure 21: ConcatenationInLoops Class

Result:

### Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số Mô tả được tạo tự động Figure 22: Result of ConcatenationInLoops Class

Code: GarbageCreator: tạo "rác" bằng cách đọc một file lớn và sử dụng *String* với toán tử + để nối chuỗi.



### Figure 23: GarbageCreator Class

Code: NoGarbage: sử dụng *StringBuffer* thay vì *String* với toán tử *+* để cải thiện hiệu năng

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Phông chữ

Mô tả được tạo tự động

### Figure 24: NoGarbage Class

Result:

Conclusion:

* So sánh:
  + *String* với toán tử *+* tạo nhiều đối tượng tạm thời, gây giảm hiệu năng.
  + *StringBuilder* và *StringBuffer* hiệu quả hơn, trong đó StringBuilder nhanh hơn do không hỗ trợ đồng bộ (*non-synchronized*).
* Ứng dụng:
  + Sử dụng *StringBuilder* trong hầu hết các trường hợp khi cần xử lý chuỗi lớn
  + Sử dụng *StringBuffer* nếu cần đồng bộ trong môi trường đa luồng.

1. Class Diagram

**Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, biểu đồ

Mô tả được tạo tự động**

Figure 24: Class Diagram