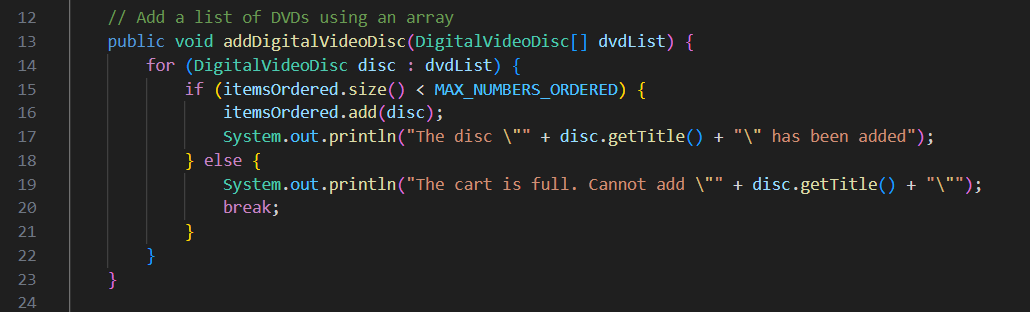
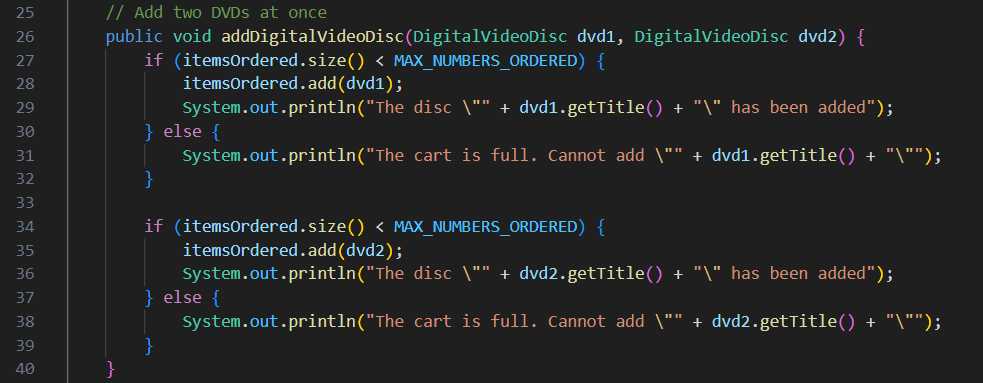
**Report Lab 03: Basic Object-Oriented Techniques**

**2. Working with method overloading**

***New code:***

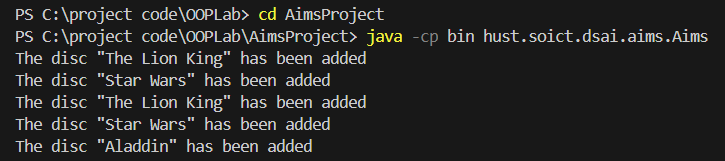
******

******

***Add some DVDs:***

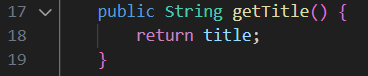
****

***Result:***

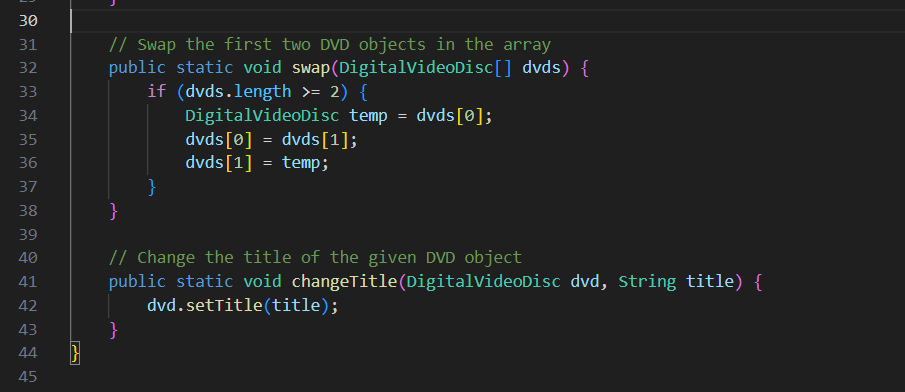
****

**3. Passing parameter**

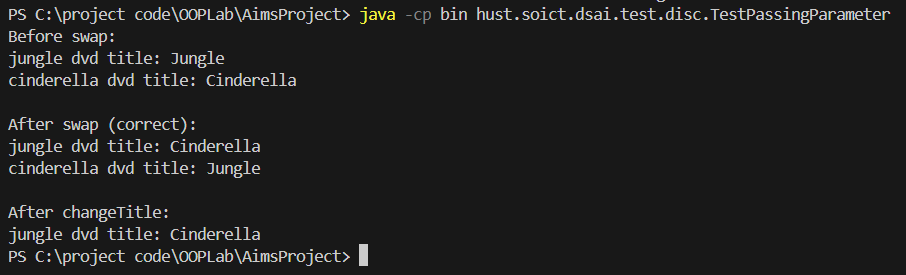
***Update new code in DigitalVideoDisc.java:***

****

***Create TestPassingParameter.java:***

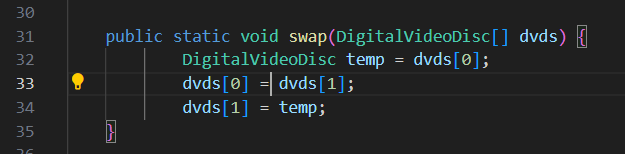
****

***Result:***

****

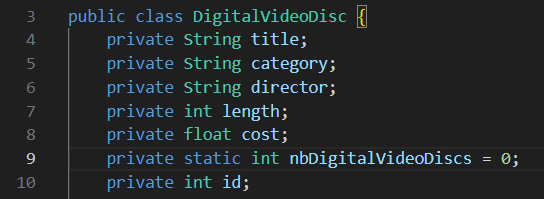
**4. Use debug run:**

***Debugging successful:***

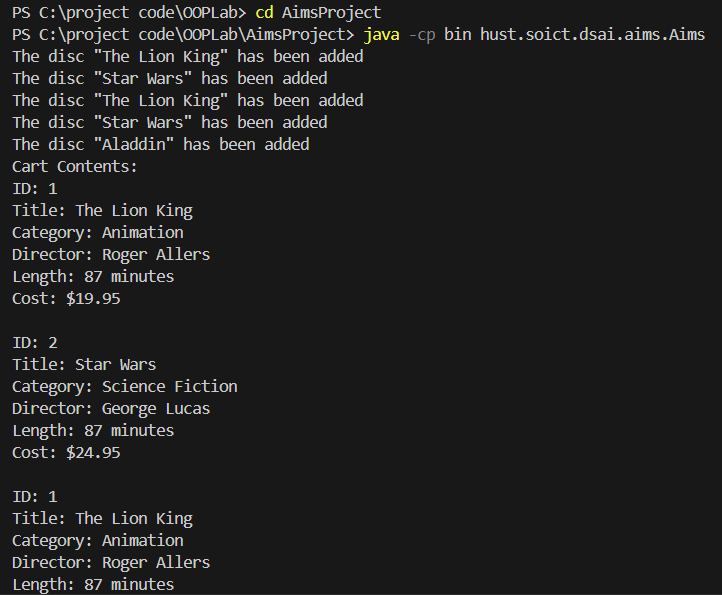
****

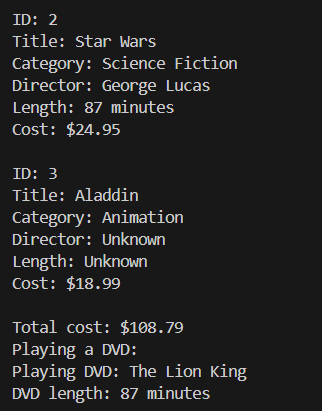
**5. Classifier Member and Instance Member**

***Update new code in DigitalVideoDisc.java:***

****

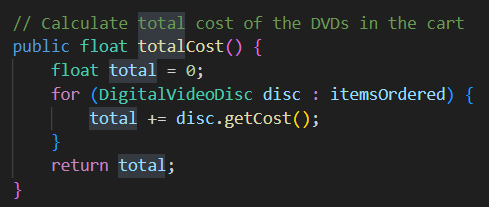
***Result:***

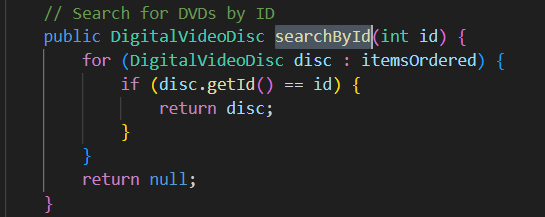
******

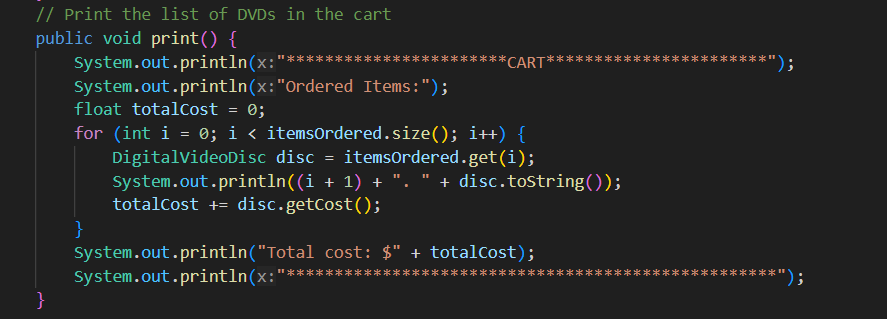
******

**6. Open the Cart class**

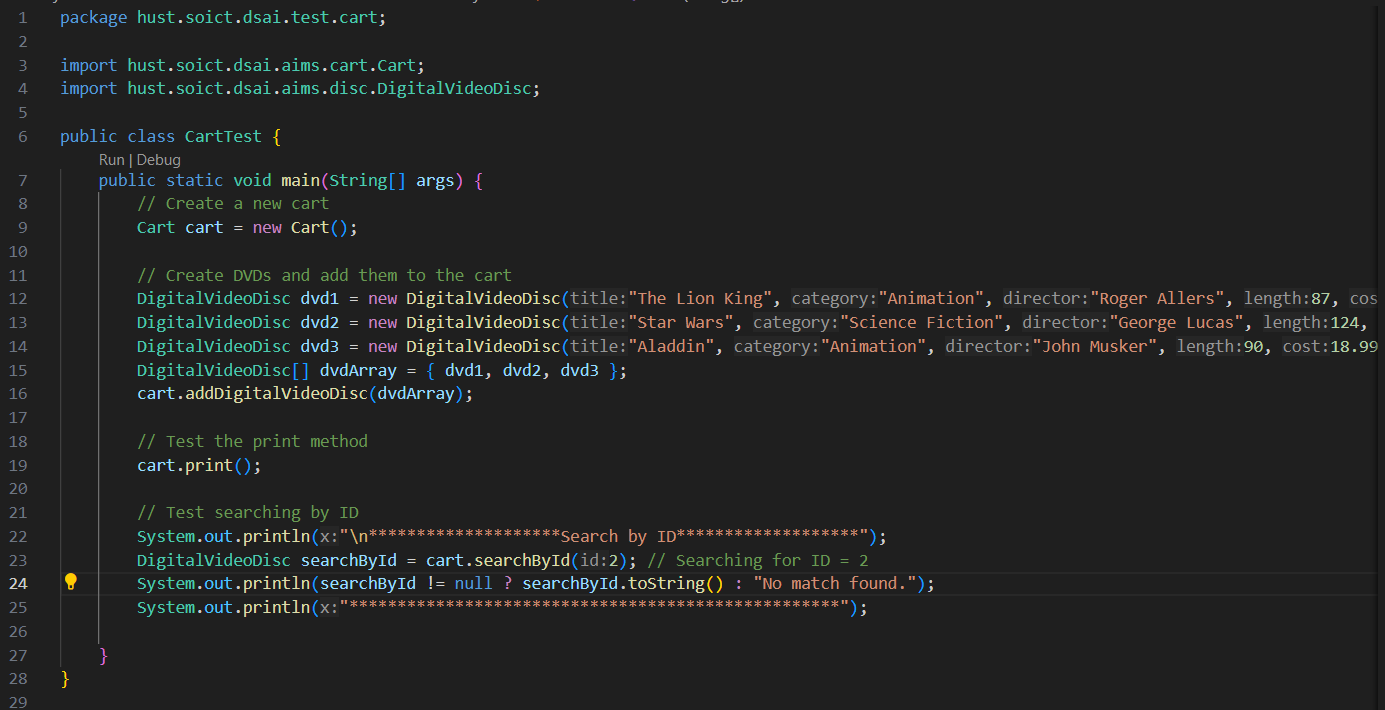
***Update totalCost, searchById and print in Cart.java:***

****

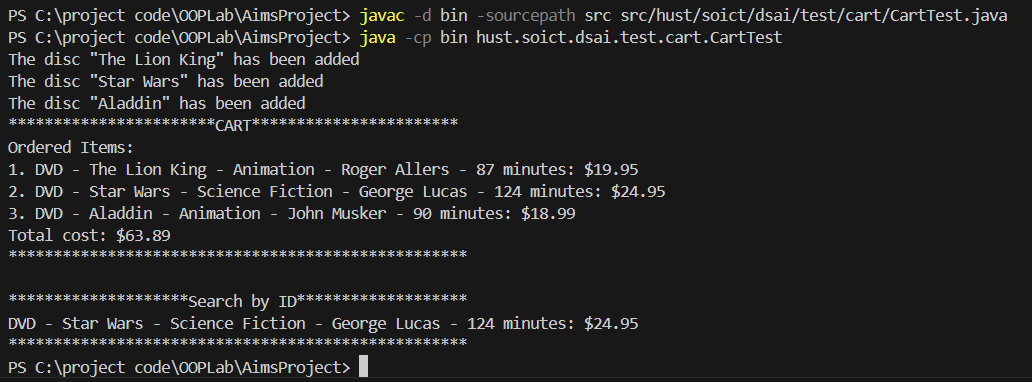
****

****

***Create new file CartTest.java:***

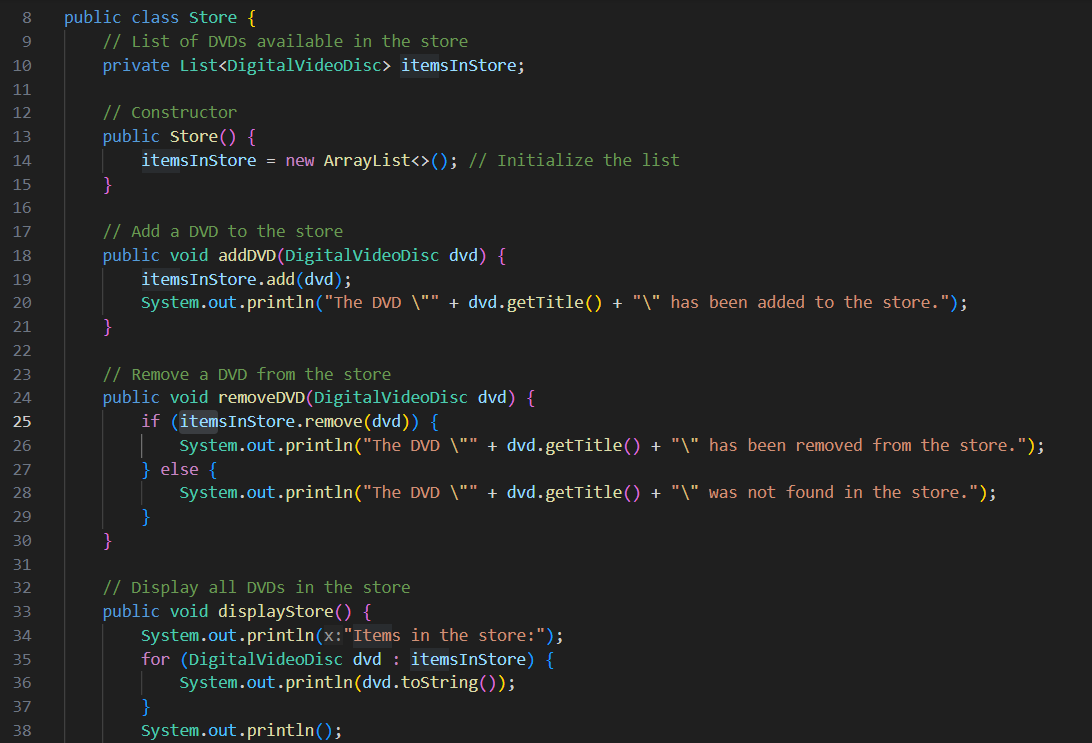
******

***Result:***

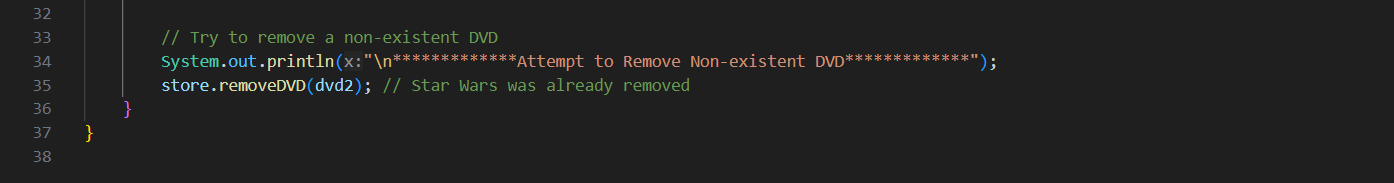
****

**7. Implement the Store class**

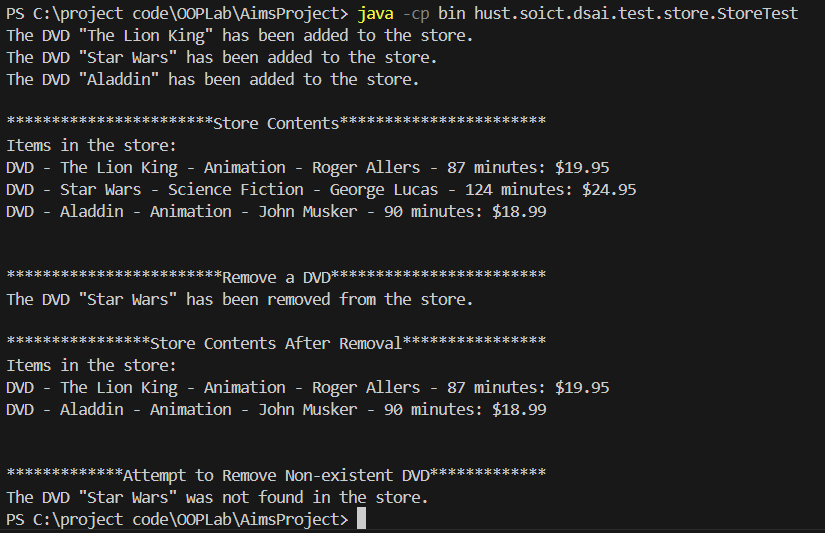
***Create Store.java:***

****

***Create StoreTest.java:***

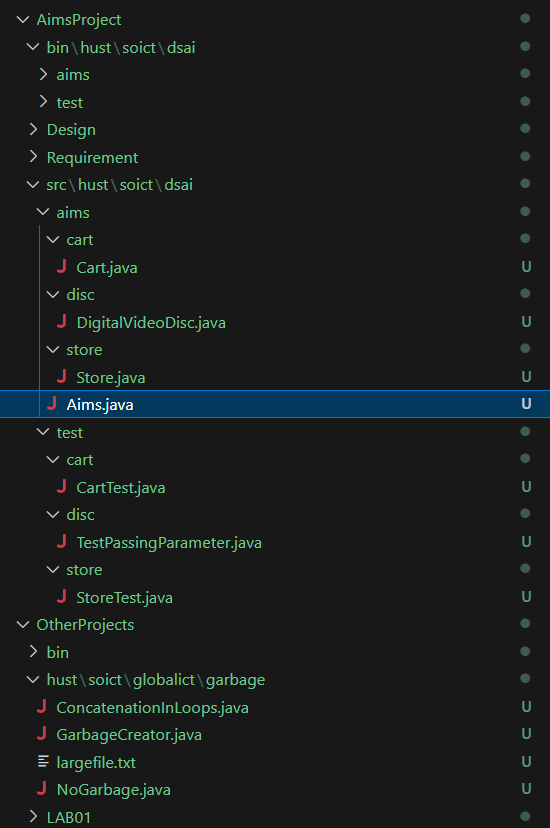
******

***Result:***

******

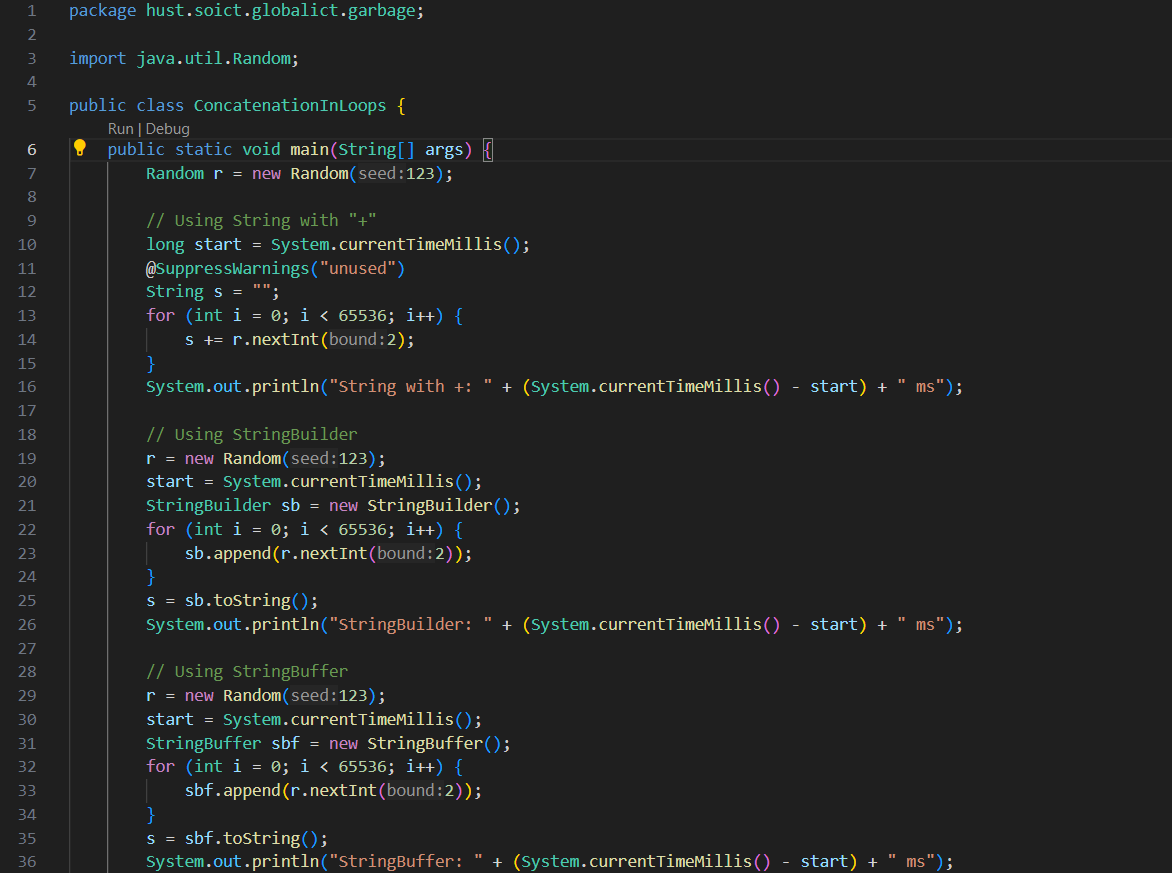
**8. Re-organize your projects**

***Project’s entire tree:***

****

**8.1. String, StringBuilder and StringBuffer**

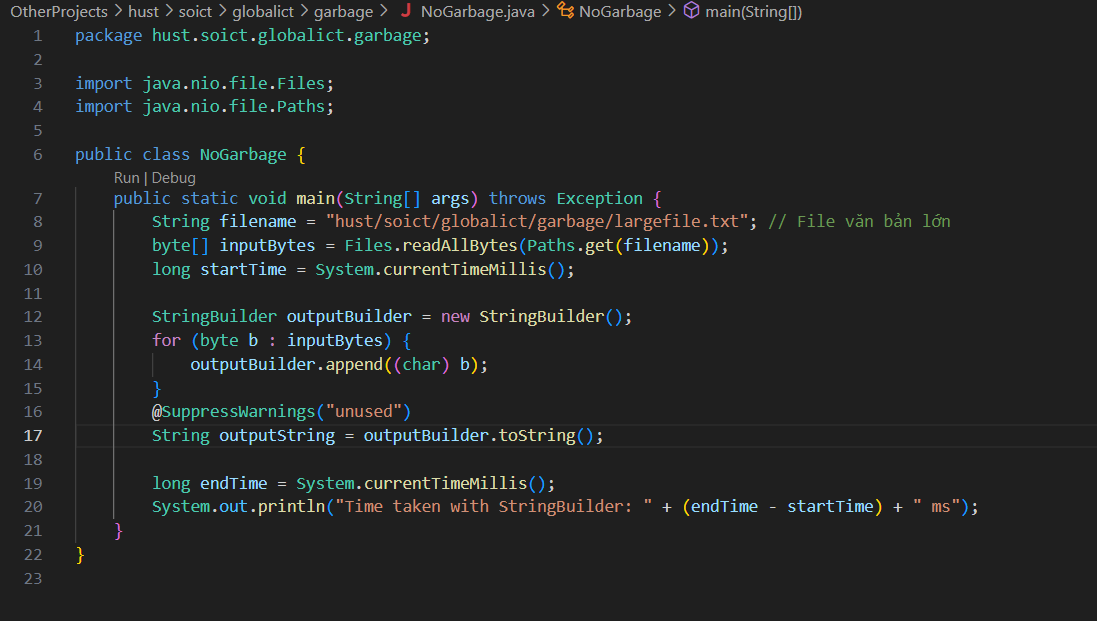
***Create ConcetanationInLoops.java:***

******

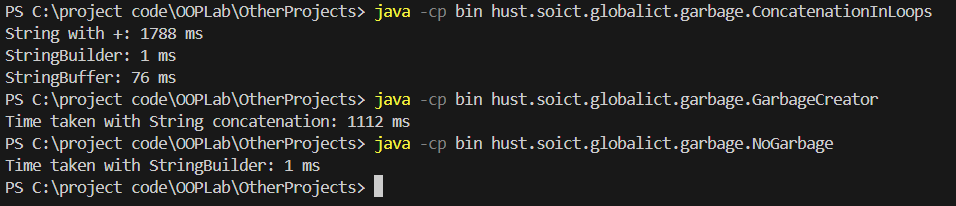
***Create GarbageCreator.java:***

******

***Create NoGarbage.java:***

******

***Results:***

****

**Answer the questions in the section:**

***English version:***

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

**Answer:**

Java is a Pass by Value programming language.

* *For primitive types:* Java passes the actual value of the variable, meaning any changes inside the method do not affect the variable outside.
* *For objects:* Java passes a copy of the reference to the object. This allows changes to the object's attributes, but the reference itself cannot be altered (e.g., swapping objects does not work).

***Question 2: After the call of swap(jungleDVD, cinderellaDVD), why does the title of these two objects still remain unchanged?***

**Answer:**

When the swap method is called:

* The references to the objects jungleDVD and cinderellaDVD are copied.
* Inside the method, o1 and o2 are the local copies of the original references.
* Assigning o1 = o2 only modifies these local copies and does not affect the original references outside the method.

As a result, the objects jungleDVD and cinderellaDVD outside the method are not swapped, and their titles remain unchanged.

***Question 3: After the call of changeTitle(jungleDVD, cinderellaDVD.getTitle()), why is the title of the jungleDVD changed?***

**Answer:**

* When jungleDVD is passed to the changeTitle method:
* The reference to jungleDVD is copied.
* Inside the method, calling dvd.setTitle(title) modifies the title attribute of the object that dvd references, which is the same object as jungleDVD.

The creation of a new object inside the method (e.g., new DigitalVideoDisc) does not affect the original jungleDVD because dvd is only a copy of the reference.

**Conclusion:**

Java is a Pass by Value programming language:

* For objects: Java passes the value of the reference (a copy of the reference), not the actual reference itself. This means:
* The contents of the object being referenced (e.g., title) can be modified.
* The reference itself cannot be modified (e.g., swapping two objects does not work).
* For primitive types: Java passes the actual value of the variable, and changes inside the method do not affect the variable outside.

***Vietnamese version:***

***Câu hỏi 1: Java là ngôn ngữ lập trình truyền tham trị (Pass by Value) hay truyền tham chiếu (Pass by Reference)?***

Java là một ngôn ngữ lập trình truyền tham trị (Pass by Value).

* Với kiểu dữ liệu nguyên thủy (**primitive types**): Java truyền giá trị thực của biến, nghĩa là mọi thay đổi trong phương thức không ảnh hưởng đến biến bên ngoài.
* Với đối tượng (**objects**): Java truyền bản sao của tham chiếu đến đối tượng. Điều này cho phép thay đổi các thuộc tính của đối tượng, nhưng tham chiếu gốc không thể bị thay đổi (ví dụ: không thể hoán đổi hai đối tượng).

***Câu hỏi 2: Tại sao sau khi gọi swap(jungleDVD, cinderellaDVD), tiêu đề của hai đối tượng vẫn không thay đổi?***

Khi phương thức swap được gọi:

* Tham chiếu đến các đối tượng jungleDVD và cinderellaDVD được sao chép.
* Trong phương thức, o1 và o2 là các bản sao cục bộ của tham chiếu gốc.
* Việc gán o1 = o2 chỉ thay đổi các bản sao này, không ảnh hưởng đến tham chiếu gốc bên ngoài phương thức.

=> Do đó, các đối tượng jungleDVD và cinderellaDVD bên ngoài phương thức không bị hoán đổi, và tiêu đề của chúng không thay đổi.

***Câu hỏi 3: Tại sao sau khi gọi changeTitle(jungleDVD, cinderellaDVD.getTitle()), tiêu đề của jungleDVD lại thay đổi?***

Khi jungleDVD được truyền vào phương thức changeTitle:

* Tham chiếu đến jungleDVD được sao chép.
* Trong phương thức, khi gọi dvd.setTitle(title), thuộc tính title của đối tượng mà dvd tham chiếu được thay đổi. Đây chính là đối tượng jungleDVD bên ngoài.

Việc tạo một đối tượng mới trong phương thức (ví dụ: new DigitalVideoDisc) không ảnh hưởng đến đối tượng jungleDVD ban đầu, vì dvd chỉ là một bản sao của tham chiếu.

***Kết luận:***

Java là một ngôn ngữ lập trình truyền tham trị (Pass by Value):

* **Với object:** Java truyền giá trị của tham chiếu (bản sao của tham chiếu), không phải tham chiếu thực sự. Điều này có nghĩa là:
* Nội dung của đối tượng được tham chiếu (ví dụ: thuộc tính title) có thể thay đổi.
* Tham chiếu gốc không thể thay đổi (ví dụ: không thể hoán đổi hai đối tượng).
* **Với kiểu dữ liệu nguyên thủy**: Java truyền giá trị thực của biến, và mọi thay đổi trong phương thức không ảnh hưởng đến giá trị bên ngoài.