

Báo cáo thực hành OOP Lab 03

Họ và tên: Phạm Đức Long

Mã số sinh viên: 20225737

Mã lớp học: 744520

1. Branch your repository	1
2. Working with method overloading	2
2.1. Overloading by differing types of parameter	2
2.2. Overloading by differing the number of parameters	5
3. Passing parameter	5
4. Use debug run	7
4.1. Debugging Java in Eclipse	7
4.2. Example of debug run for the swap method of TestPassingParameter	7
4.2.1. Setting, deleting & deactivate breakpoints:	7
4.2.2. Run in Debug mode:	8
4.2.3. Step Into, Step Over, Step Return, Resume:	8
4.2.4. Investigate value of variables:	9
4.2.5. Change value of variables:	10
5. Classifier Member and Instance Member	11
6. Open the Cart class	12
6.1. printCart():	12
6.2. searchCart()	13
7. Implement the Store class	14
7.1. addDVD():	14
7.2. removeDVD():	16
8. Re-organize your projects	17
9. String, StringBuilder and StringBuffer	17
9.1. So sánh String, StringBuilder and StringBuffer	17
9.2. GarbageCreator và NoGarbage	17
10. Release flow demonstration	20

[google docs](#)

1. Branch your repository

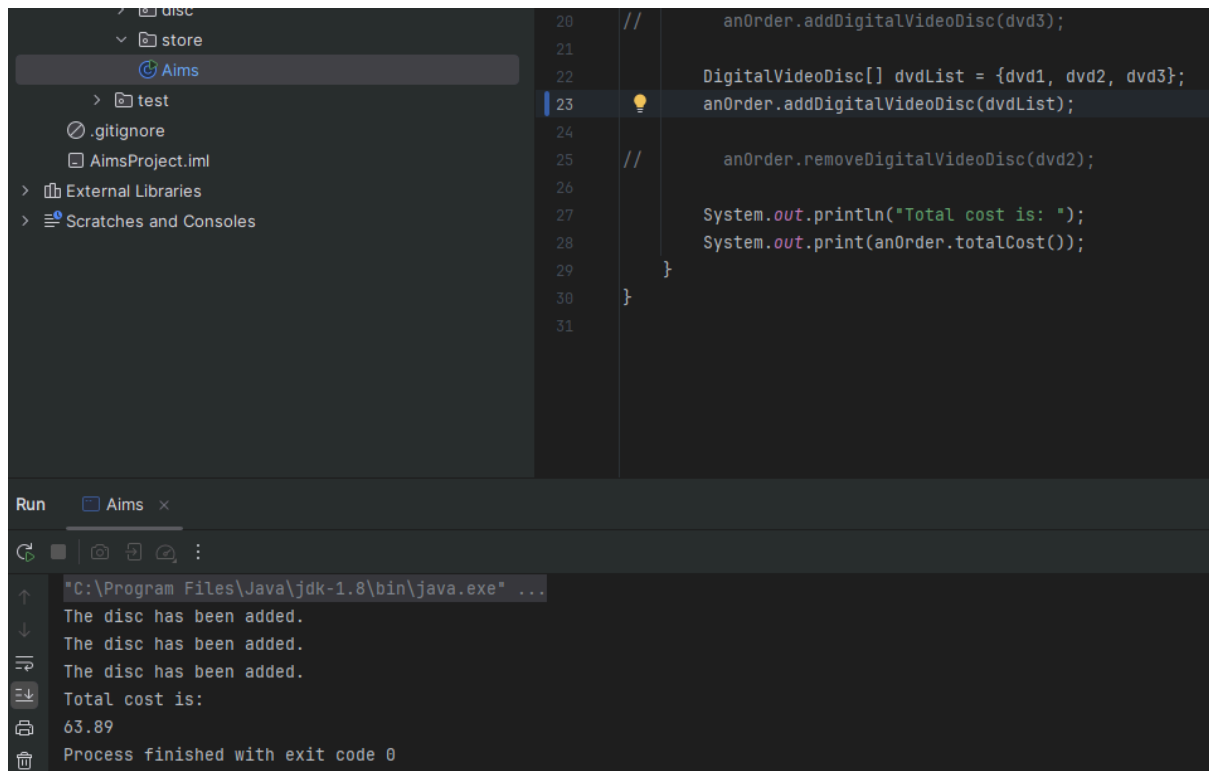
- Github: [OOP_Lab_744520](#)

2. Working with method overloading

2.1. Overloading by differing types of parameter

```
public void addDigitalVideoDisc(DigitalVideoDisc[] dvdList) {  
    for (int i = 0; i < dvdList.length; i++) {  
        if (qtyOrdered < MAX_NUMBERS_ORDERED) {  
            itemsOrdered[qtyOrdered] = dvdList[i];  
            qtyOrdered++;  
            System.out.println("The disc has been added.");  
        } else {  
            System.out.println("The cart is almost full.");  
        }  
    }  
}
```

Kết quả:



The screenshot shows an IDE with a project named 'Aims'. The left sidebar shows the project structure with folders 'disc', 'store', 'test', and files '.gitignore', 'AimsProject.iml', 'External Libraries', and 'Scratches and Consoles'. The main editor displays Java code for an order management system. The code includes methods for adding and removing digital video discs, and a method to print the total cost. The output window shows the execution results, including the total cost of 63.89.

```
20 // anOrder.addDigitalVideoDisc(dvd3);
21
22 DigitalVideoDisc[] dvdList = {dvd1, dvd2, dvd3};
23 anOrder.addDigitalVideoDisc(dvdList);
24
25 // anOrder.removeDigitalVideoDisc(dvd2);
26
27 System.out.println("Total cost is: ");
28 System.out.print(anOrder.totalCost());
29 }
30 }
31
```

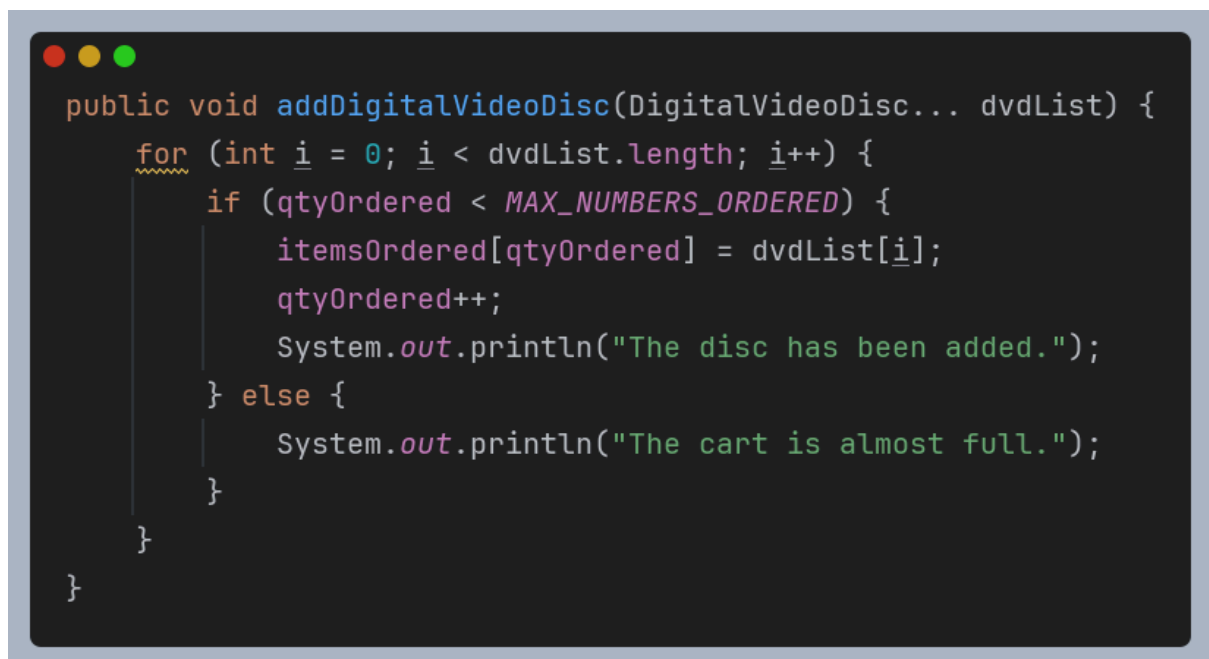
Run Aims x

"C:\Program Files\Java\jdk-1.8\bin\java.exe" ...

The disc has been added.
The disc has been added.
The disc has been added.
Total cost is:
63.89
Process finished with exit code 0

- Try to add a method `addDigitalVideoDisc` which allows to pass an arbitrary number of arguments for dvd. Compare to an array parameter. What do you prefer in this case?

Thay thế [] bằng ... (varargs)



The screenshot shows a code editor with a Java method `addDigitalVideoDisc` that uses varargs. The method takes a `DigitalVideoDisc...` array and iterates through it, adding discs to the order if the quantity is less than the maximum allowed. The code is as follows:

```
public void addDigitalVideoDisc(DigitalVideoDisc... dvdList) {
    for (int i = 0; i < dvdList.length; i++) {
        if (qtyOrdered < MAX_NUMBERS_ORDERED) {
            itemsOrdered[qtyOrdered] = dvdList[i];
            qtyOrdered++;
            System.out.println("The disc has been added.");
        } else {
            System.out.println("The cart is almost full.");
        }
    }
}
```

Kết quả:

```
.gitignore
AimsProject.iml
> External Libraries
> Scratches and Consoles

20 // anOrder.addDigitalVideoDisc(dvd3);
21
22 DigitalVideoDisc[] dvdList = {dvd1, dvd2, dvd3};
23 anOrder.addDigitalVideoDisc(dvdList);
24
25 // anOrder.removeDigitalVideoDisc(dvd2);
26
27 System.out.println("Total cost is: ");
28 System.out.print(anOrder.totalCost());
29
30 }
```

Run Aims x

"C:\Program Files\Java\jdk-1.8\bin\java.exe" ...

The disc has been added.
The disc has been added.
The disc has been added.
Total cost is:
63.89
Process finished with exit code 0

```
> External Libraries
> Scratches and Consoles

21
22 DigitalVideoDisc[] dvdList = {dvd1, dvd2, dvd3};
23 anOrder.addDigitalVideoDisc(dvd1);
24
25 // anOrder.removeDigitalVideoDisc(dvd2);
26
27 System.out.println("Total cost is: ");
28 System.out.print(anOrder.totalCost());
29
30 }
```

Run Aims x

"C:\Program Files\Java\jdk-1.8\bin\java.exe" ...

The disc has been added.
Total cost is:
19.95
Process finished with exit code 0

```
AimsProject.iml
> External Libraries
> Scratches and Consoles

20 // anOrder.addDigitalVideoDisc(dvd3);
21
22 DigitalVideoDisc[] dvdList = {dvd1, dvd2, dvd3};
23 anOrder.addDigitalVideoDisc(dvd1, dvd2);
24
25 // anOrder.removeDigitalVideoDisc(dvd2);
26
27 System.out.println("Total cost is: ");
28 System.out.print(anOrder.totalCost());
29
30 }
```

Run Aims x

"C:\Program Files\Java\jdk-1.8\bin\java.exe" ...

The disc has been added.
The disc has been added.
Total cost is:
44.9
Process finished with exit code 0

So sánh:

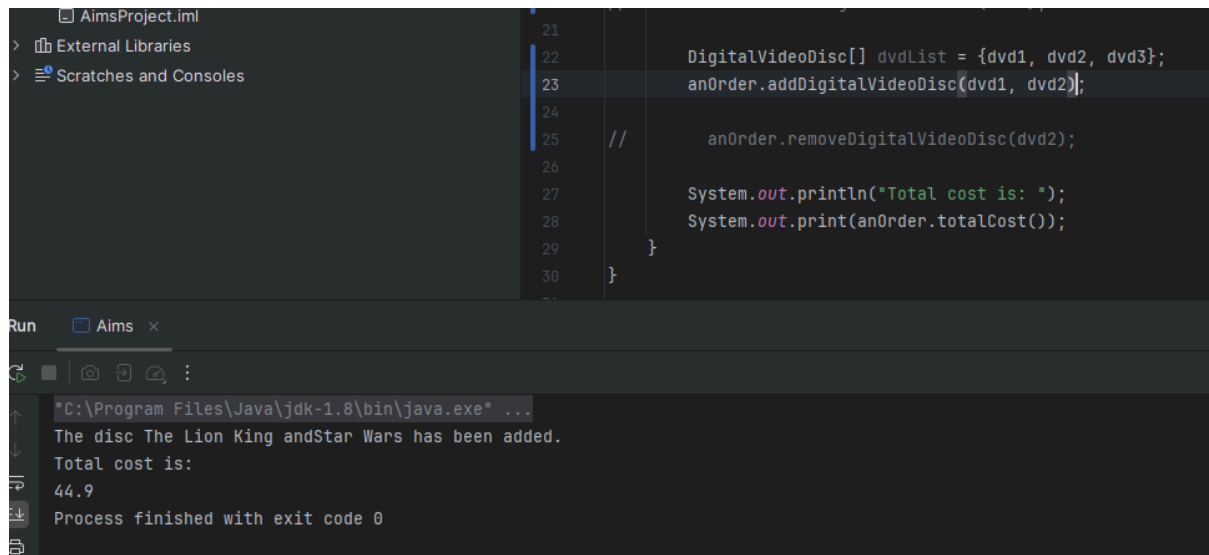
...	[]
Dùng để truyền số lượng tham số không xác định vào một phương thức.	Dùng để lưu trữ một tập hợp các giá trị cố định với kích thước xác định.
Thích hợp cho các phương thức linh hoạt với số lượng tham số không biết trước.	Phù hợp cho các trường hợp mà ta biết chính xác số lượng phần tử ngay từ đầu.
Tiện lợi hơn khi gọi phương thức với nhiều tham số mà không cần tạo mảng thủ công.	Ít tiện lợi hơn vì luôn cần tạo mảng trước khi truyền.

Cá nhân em thích dùng ... hơn vì như đã trình bày ở trên, nó tiện lợi hơn khá nhiều so với []. Nhưng có lẽ do thói quen, em vẫn dùng [] nhiều hơn.

2.2. Overloading by differing the number of parameters

```
public void addDigitalVideoDisc(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) { no usages Long Pham +1
    if(qtyOrdered < MAX_NUMBERS_ORDERED) {
        itemsOrdered[qtyOrdered] = dvd1;
        itemsOrdered[qtyOrdered + 1] = dvd2;
        qtyOrdered += 2;
        System.out.println("The disc " + dvd1.getTitle() + " and " + dvd2.getTitle() + " has been added.");
    } else {
        System.out.println("The cart is almost full.");
    }
}
```

Kết quả;



The screenshot shows an IDE with a project named 'AimsProject'. The code editor displays the following Java code:

```
21
22     DigitalVideoDisc[] dvdList = {dvd1, dvd2, dvd3};
23     anOrder.addDigitalVideoDisc(dvd1, dvd2);
24
25     //         anOrder.removeDigitalVideoDisc(dvd2);
26
27     System.out.println("Total cost is: ");
28     System.out.print(anOrder.totalCost());
29 }
30 }
```

The Run window shows the output of the program:

```
"C:\Program Files\Java\jdk-1.8\bin\java.exe" ...
The disc The Lion King andStar Wars has been added.
Total cost is:
44.9
Process finished with exit code 0
```

3. Passing parameter

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

- Java truyền tham trị và không thực sự truyền tham chiếu (Như C hay C++) nhưng khi làm việc với **đối tượng**, tham trị này lại là **tham trị của tham chiếu**. Khi truyền đối tượng (Ví dụ như DigitalVideoDisc hay Cart trong bài này), giá trị của tham chiếu (địa chỉ bộ nhớ) được sao chép và truyền vào phương thức, điều này cho phép ta thay đổi thuộc tính của đối tượng, nhưng không thể thay đổi tham chiếu (Ví dụ ta có thể lấy setter của dvd để thay đổi giá trị của nó như ở phương thức changeTitle bên dưới).

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

- Khi ta gọi swap(jungleDVD, cinderellaDVD), giá trị của các tham chiếu (jungleDVD và cinderellaDVD) được sao chép và truyền vào tham số o1 và o2 trong phương thức swap.
- Bất kỳ thay đổi nào đối với o1 và o2 bên trong phương thức **chỉ ảnh hưởng đến bản sao** của các tham chiếu này, không ảnh hưởng đến bản gốc (jungleDVD và cinderellaDVD).

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

- Khi truyền một biến có kiểu object vào một hàm thì lúc đó có nghĩa ta đã truyền giá trị của biến đó để sử dụng trong hàm, chứ không phải truyền đối tượng được biến đó tham chiếu tới. Vậy nên việc thay đổi

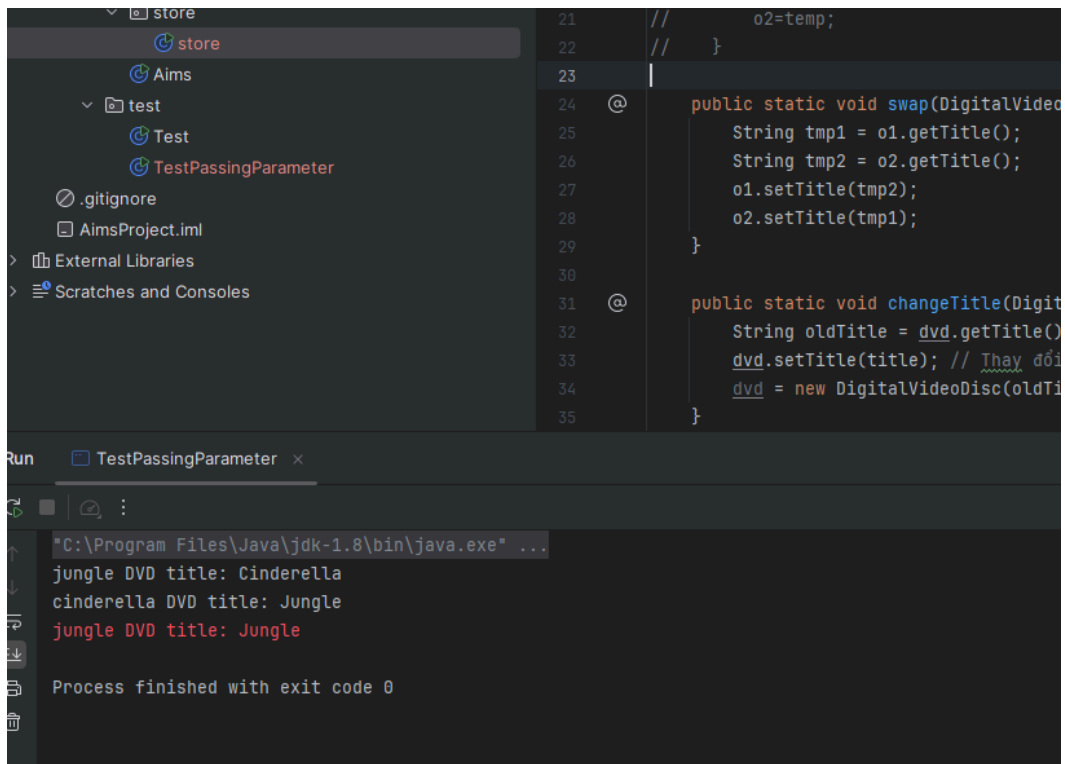
giá trị của biến có kiểu object này bằng cách gán cho một biến kiểu object khác thì object được tham chiếu đến lúc đầu không bị ảnh hưởng, chỉ khi sử dụng các method (ở đây là Setter của Disc) của chính các object được tham chiếu này thì dữ liệu của object mới được thay đổi sau khi ra khỏi hàm.

```
public static void changeTitle(DigitalVideoDisc dvd, String title){ 1 usage
    String oldTitle = dvd.getTitle(); // Lấy tiêu đề cũ của dvd
    dvd.setTitle(title); // Thay đổi tiêu đề của đối tượng dvd bằng setter
    dvd = new DigitalVideoDisc(oldTitle); // Thay đổi tham chiếu của đối tượng dvd (không ảnh hưởng đến đối tượng ban đầu)
}
```

Please write a swap() method that can correctly swap the two objects:

```
public static void swap(DigitalVideoDisc o1, DigitalVideoDisc o2){
    String tmp1 = o1.getTitle();
    String tmp2 = o2.getTitle();
    o1.setTitle(tmp2);
    o2.setTitle(tmp1);
}
```

Kết quả:



The screenshot shows an IDE with a project structure on the left, code in the center, and a run console at the bottom. The project structure includes 'store', 'Aims', 'test', and 'TestPassingParameter'. The code in the center shows two methods: 'swap' and 'changeTitle'. The 'swap' method uses temporary variables to swap the titles of two objects. The 'changeTitle' method creates a new object and assigns it to the parameter variable. The run console shows the output of the 'TestPassingParameter' test, which prints the titles of two objects before and after the swap method is called, demonstrating that the swap method works correctly.

```
21 // o2=temp;
22 // }
23
24 @
25 public static void swap(DigitalVideoDisc o1, DigitalVideoDisc o2){
26     String tmp1 = o1.getTitle();
27     String tmp2 = o2.getTitle();
28     o1.setTitle(tmp2);
29     o2.setTitle(tmp1);
30 }
31
32 @
33 public static void changeTitle(DigitalVideoDisc dvd, String title){
34     String oldTitle = dvd.getTitle();
35     dvd.setTitle(title); // Thay đổi tiêu đề của đối tượng dvd bằng setter
36     dvd = new DigitalVideoDisc(oldTitle); // Thay đổi tham chiếu của đối tượng dvd (không ảnh hưởng đến đối tượng ban đầu)
37 }
```

Run TestPassingParameter x

```
"C:\Program Files\Java\jdk-1.8\bin\java.exe" ...
jungle DVD title: Cinderella
cinderella DVD title: Jungle
jungle DVD title: Jungle

Process finished with exit code 0
```

4. Use debug run

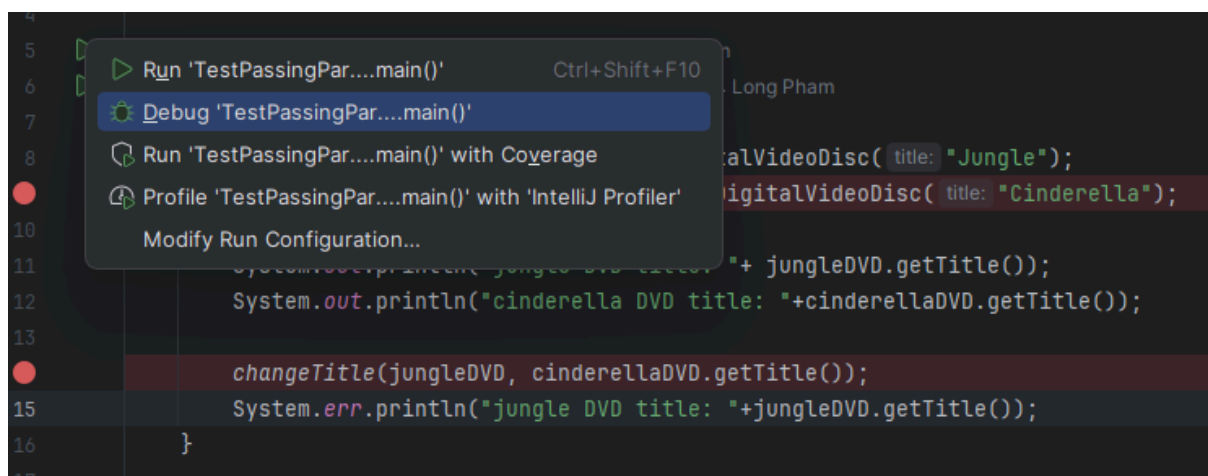
4.1. Debugging Java in Eclipse

4.2. Example of debug run for the swap method of TestPassingParameter

4.2.1. Setting, deleting & deactivate breakpoints:

```
1 package hust.soict.hedspi.test.disc;
2
3 import hust.soict.hedspi.aims.disc.DigitalVideoDisc;
4
5 public class TestPassingParameter {  Long Pham
6     public static void main(String[] args){  Long Pham
7         //TODO Auto-generated method stub
8         DigitalVideoDisc jungleDVD = new DigitalVideoDisc( title: "Jungle");
9         DigitalVideoDisc cinderellaDVD = new DigitalVideoDisc( title: "Cinderella");
10        swap(jungleDVD, cinderellaDVD);
11        System.out.println("jungle DVD title: "+ jungleDVD.getTitle());
12        System.out.println("cinderella DVD title: "+cinderellaDVD.getTitle());
13
14        changeTitle(jungleDVD, cinderellaDVD.getTitle());
15        System.err.println("jungle DVD title: "+jungleDVD.getTitle());
16    }
17 }
```

4.2.2. Run in Debug mode:

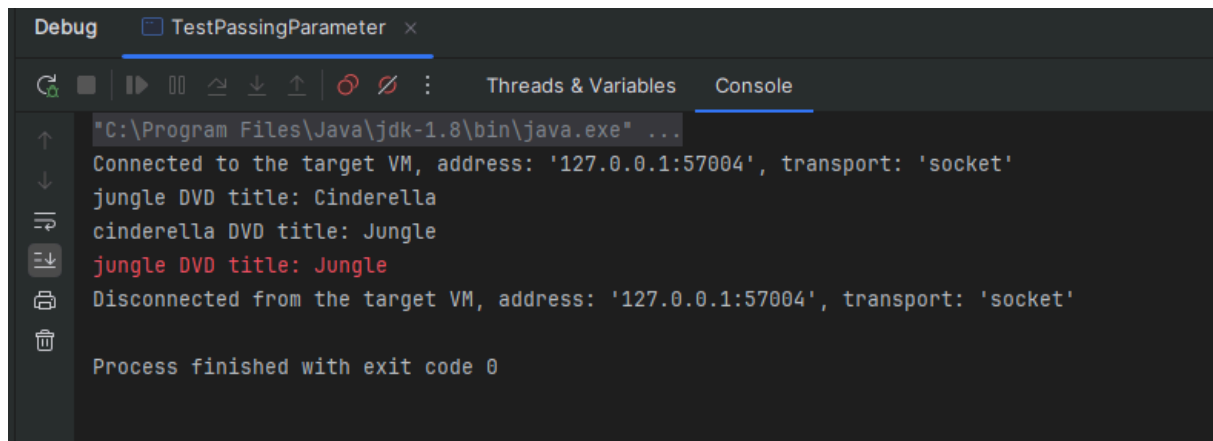
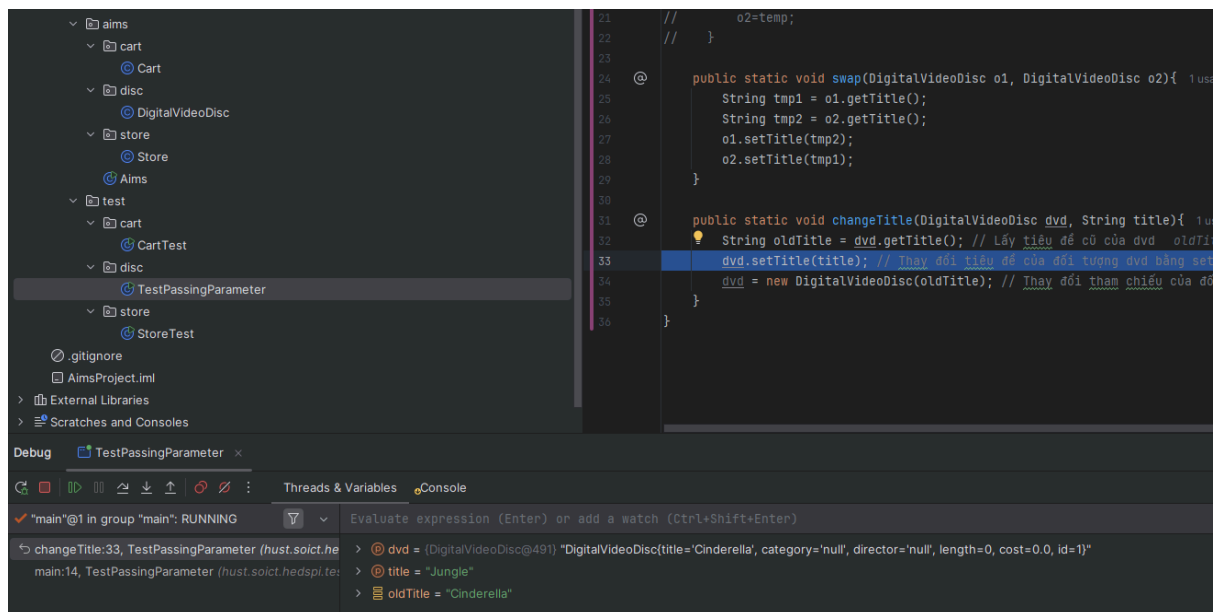
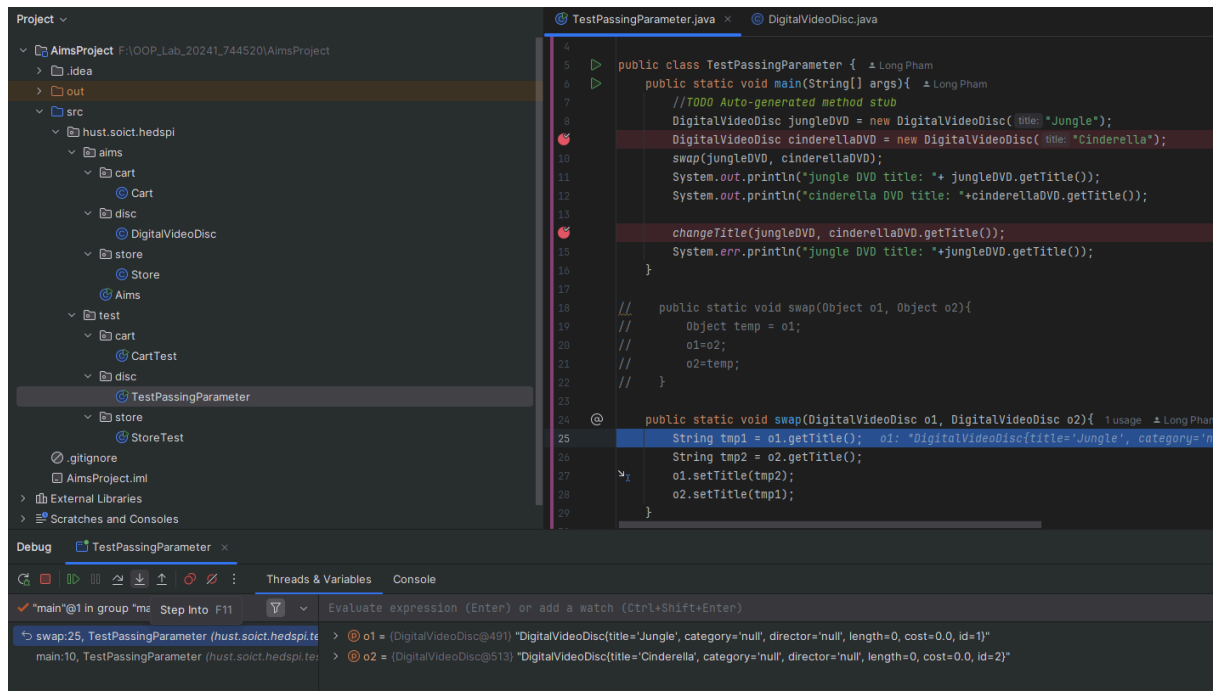


The screenshot shows the Eclipse IDE with the 'TestPassingParameter' class open. The 'Run' menu is open, and the 'Debug' option is selected. The menu items are:

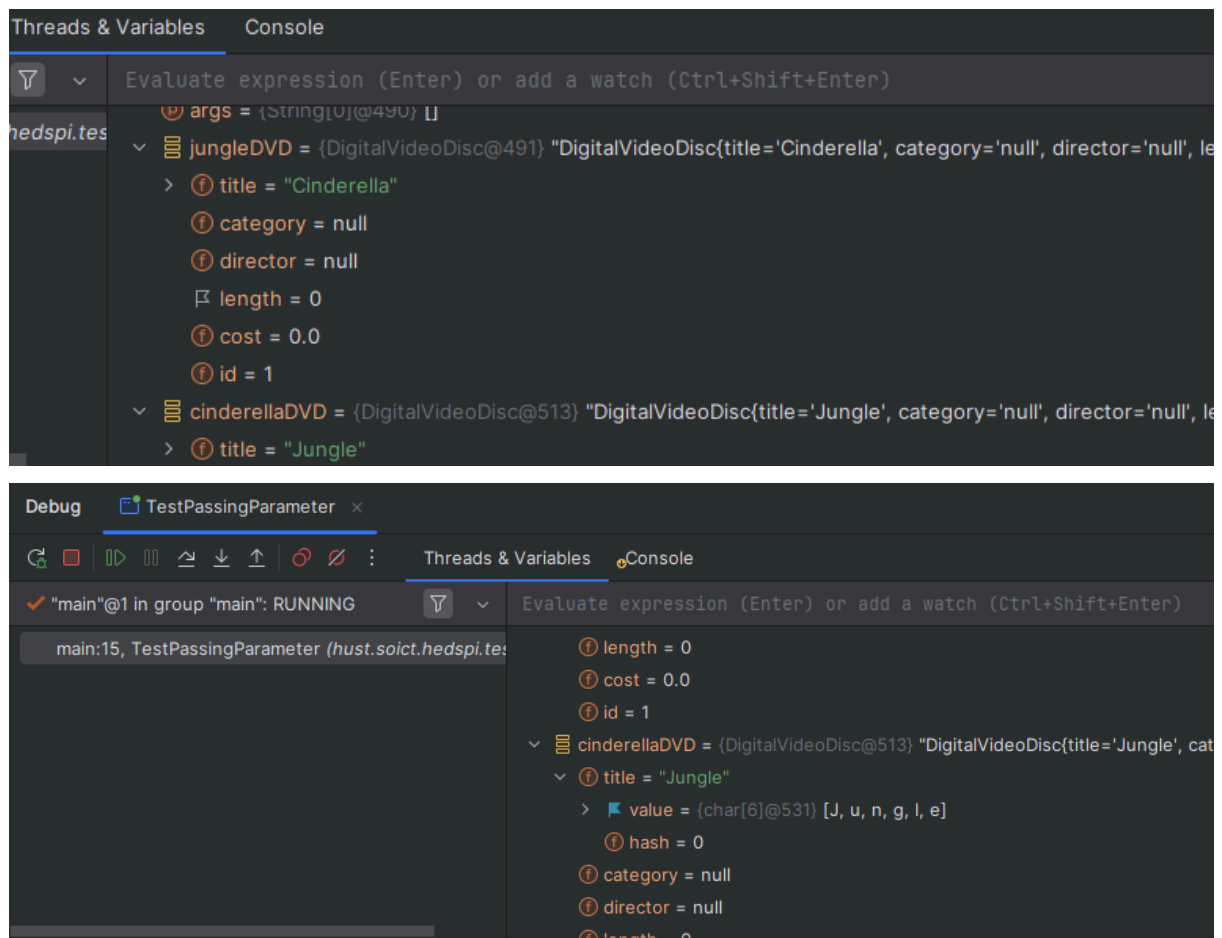
- Run 'TestPassingPar....main()' (Ctrl+Shift+F10)
- Debug 'TestPassingPar....main()'**
- Run 'TestPassingPar....main()' with Coverage
- Profile 'TestPassingPar....main()' with 'IntelliJ Profiler'
- Modify Run Configuration...

The background code is the same as in the previous image, showing the swap method and the main method.

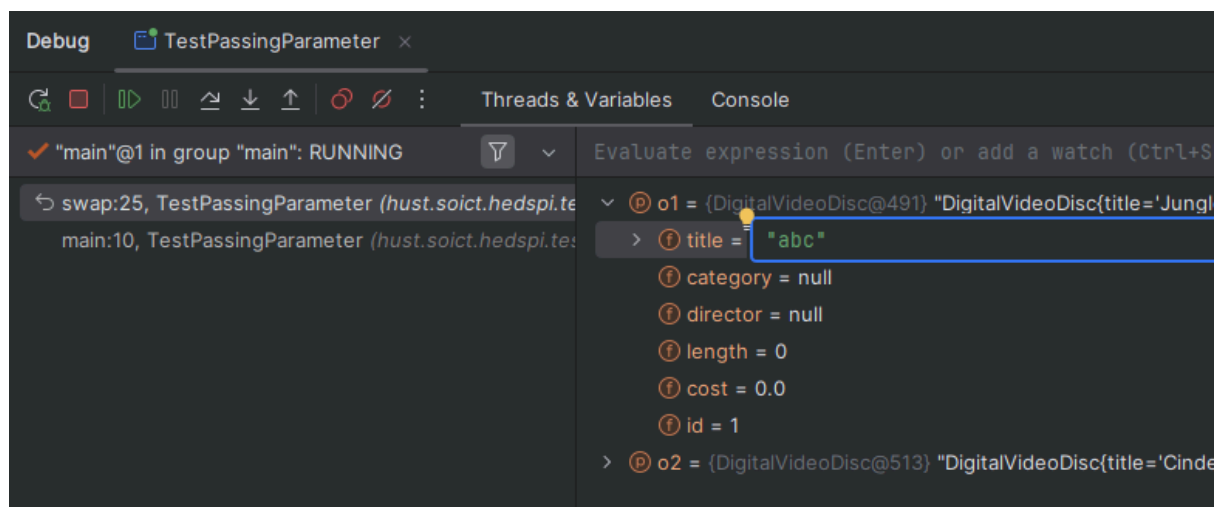
4.2.3. Step Into, Step Over, Step Return, Resume:

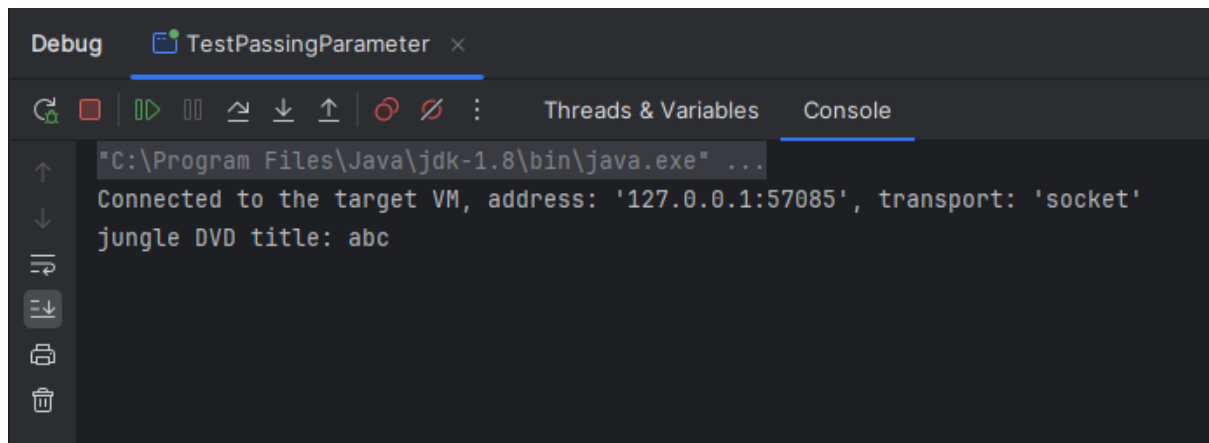


4.2.4. Investigate value of variables:



4.2.5. Change value of variables:





5. Classifier Member and Instance Member

```
private String title; 6 usages
private String category; 5 usages
private String director; 4 usages
private int length; 3 usages
private float cost; 5 usages
private int id; 6 usages

private static int nbDigitalVideoDiscs = 0;
```

```
public DigitalVideoDisc(String title) {
    this.title = title;

    nbDigitalVideoDiscs++;
    this.id = nbDigitalVideoDiscs;
}
```

```

public DigitalVideoDisc(String title, String category, float cost) {
    this.title = title;
    this.category = category;
    this.cost = cost;

    nbDigitalVideoDiscs++;
    this.id = nbDigitalVideoDiscs;
}

```

...(Các constructor khác tương tự)

Kết quả:

The screenshot shows an IDE with a project named 'Aims'. The left sidebar shows the project structure with folders like 'src', 'test', and 'External Libraries'. The main editor displays a Java file with the following code:

```

18 DigitalVideoDisc dvd3 = new DigitalVideoDisc("Aladdin",
19 category: "Animation", cost: 18.99f);
20 // anOrder.addDigitalVideoDisc(dvd3);
21
22 DigitalVideoDisc[] dvdList = {dvd1, dvd2, dvd3};
23 anOrder.addDigitalVideoDisc(dvdList);
24
25 // anOrder.removeDigitalVideoDisc(dvd2);
26
27 // System.out.println("Total cost is: ");
28 // System.out.print(anOrder.totalCost());
29
30 System.out.println("The id of " + dvd1.getTitle() + " is " + dvd1.getId());
31 System.out.println("The id of " + dvd2.getTitle() + " is " + dvd2.getId());
32 System.out.println("The id of " + dvd3.getTitle() + " is " + dvd3.getId());
33 }
34
35

```

The bottom panel shows the 'Run' output for 'Aims':

```

C:\Program Files\Java\jdk-1.8\bin\java.exe ...
The disc has been added.
The disc has been added.
The disc has been added.
The id of The Lion King is 1
The id of Star Wars is 2
The id of Aladdin is 3

```

6. Open the Cart class

6.1. printCart():

```

public void printCart() { 1 usage new *
    System.out.println("*****CART*****");
    for (int i = 0; i < qtyOrdered; i++)
        System.out.println(i + 1 + ". DVD - " + itemsOrdered[i].getTitle() + " - " + itemsOrdered[i].getCategory()

    System.out.println("Total cost: " + totalCost());
    System.out.println("*****");
}

```

Kết quả trong CartTest.java:

```

The disc has been added.
The disc has been added.
The disc has been added.
*****CART*****
1. DVD - The Lion King - Animation - Roger Allers - 87 - 19.95
2. DVD - Star Wars - Science Fiction - George Lucas - 87 - 24.95
3. DVD - Aladdin - Animation - null - 0 - 18.99
Total cost: 63.89
*****

```

6.2. searchCart()

- Theo id:

```

public boolean search(int id) {
    for (int i = 0; i < qtyOrdered; i++) {
        if (itemsOrdered[i].getId() == id)
            return true;
    }

    return false;
}

```

Kết quả:

```

"C:\Program Files\Java\jdk-1.8\bin\java.exe" ...
The disc has been added.
The disc has been added.
The disc has been added.
Enter the id of DVD you want to search:
3
Found DVD with id 3 and the title is Aladin

Process finished with exit code 0
|

```

- Theo title:

```
public boolean search(String title) { no usages new *  
    for (int i = 0; i < qtyOrdered; i++) {  
        if (itemsOrdered[i].getTitle().equals(title))  
            return true;  
    }  
  
    return false;  
}
```

Kết quả:

```
"C:\Program Files\Java\jdk-1.8\bin\java.exe" ...  
The disc has been added.  
The disc has been added.  
The disc has been added.  
Enter the title of DVD you want to search:  
The Lion King  
Found DVD with title The Lion King and the id of this DVD is 1  
  
Process finished with exit code 0
```

7. Implement the Store class

7.1. addDVD():

```
private DigitalVideoDisc[] itemsInStore = new DigitalVideoDisc[40];  
private int qtyInStore = 0; 3 usages
```

```

public void addDVD(DigitalVideoDisc disc) { 4 usages new *
    boolean existed = false;
    for (int i = 0; i < qtyInStore; i++) {
        if (itemsInStore[i].getTitle().equals(disc.getTitle())) {
            existed = true;
            break;
        }
    }

    if (!existed) {
        itemsInStore[qtyInStore] = disc;
        qtyInStore++;
        System.out.println("The disc has been added in Store.");
    } else {
        System.out.println("The disc is already in the store.");
    }
}

```

Kiểm tra:

The screenshot shows an IDE with the following components:

- Project Explorer (Left):** Displays a project structure with folders like 'hust.solict.hedspi', 'aims', 'cart', 'disc', 'store', and 'test'. The 'StoreTest' class is selected under the 'store' folder.
- Source Editor (Center):** Shows the code for the 'StoreTest' class. It includes a 'main' method that creates a 'Store' object and adds four DVD objects: 'The Lion King' (Animation), 'Star Wars' (Science Fic), 'Aladin' (Animation), and 'The Lion King' (Animation). The 'addDVD' method is called for each object.
- Run Console (Bottom):** Shows the output of the program execution. It displays four lines of text: 'The disc has been added in Store.' (three times) and 'The disc is already in the store.' (once), corresponding to the four DVD objects added. The console also shows the process finished with exit code 0.

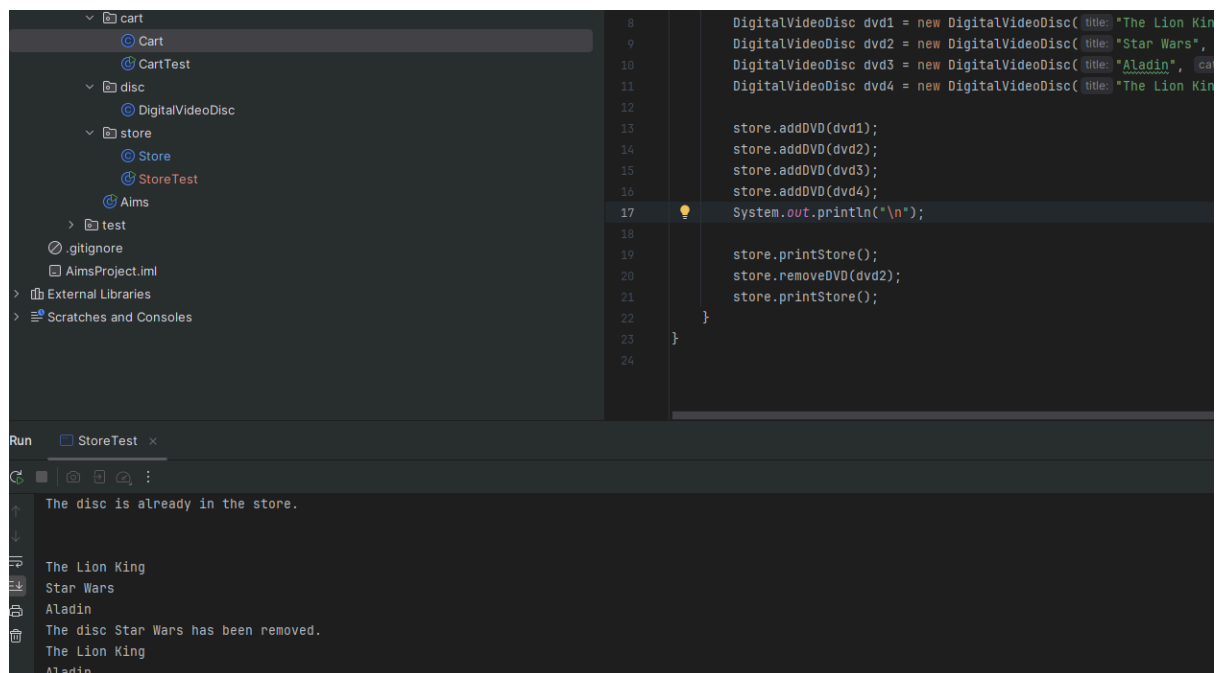
7.2. removeDVD():

```
public void removeDVD(DigitalVideoDisc disc) { 1usage new *
    int check = 41;
    for (int i = 0; i < qtyInStore; i++) {
        if (itemsInStore[i].getTitle().equals(disc.getTitle()))
            check = i;
    }

    if (check != 41) {
        for (int i = check; i < qtyInStore; i++)
            itemsInStore[i] = itemsInStore[i + 1];

        System.out.println("The disc " + disc.getTitle() + " has been removed.");
        qtyInStore--;
    } else
        System.out.println("The disc is not in the store.");
}
```

Kết quả:



The screenshot displays an IDE with a project structure on the left, Java code in the center, and a Run console at the bottom. The project structure includes folders for 'cart', 'disc', 'store', and 'test', along with files like 'Cart', 'CartTest', 'DigitalVideoDisc', 'Store', 'StoreTest', and 'Aims'. The code in the center defines a 'removeDVD' method that iterates through a store's inventory to find and remove a DVD by title. The Run console shows the output of a test run, indicating that the DVD 'Star Wars' was successfully removed from the store.

```
8      DigitalVideoDisc dvd1 = new DigitalVideoDisc( title: "The Lion Kin
9      DigitalVideoDisc dvd2 = new DigitalVideoDisc( title: "Star Wars",
10     DigitalVideoDisc dvd3 = new DigitalVideoDisc( title: "Aladin", loca
11     DigitalVideoDisc dvd4 = new DigitalVideoDisc( title: "The Lion Kin
12
13     store.addDVD(dvd1);
14     store.addDVD(dvd2);
15     store.addDVD(dvd3);
16     store.addDVD(dvd4);
17     System.out.println("\n");
18
19     store.printStore();
20     store.removeDVD(dvd2);
21     store.printStore();
22 }
23
24
```

Run StoreTest x

The disc is already in the store.

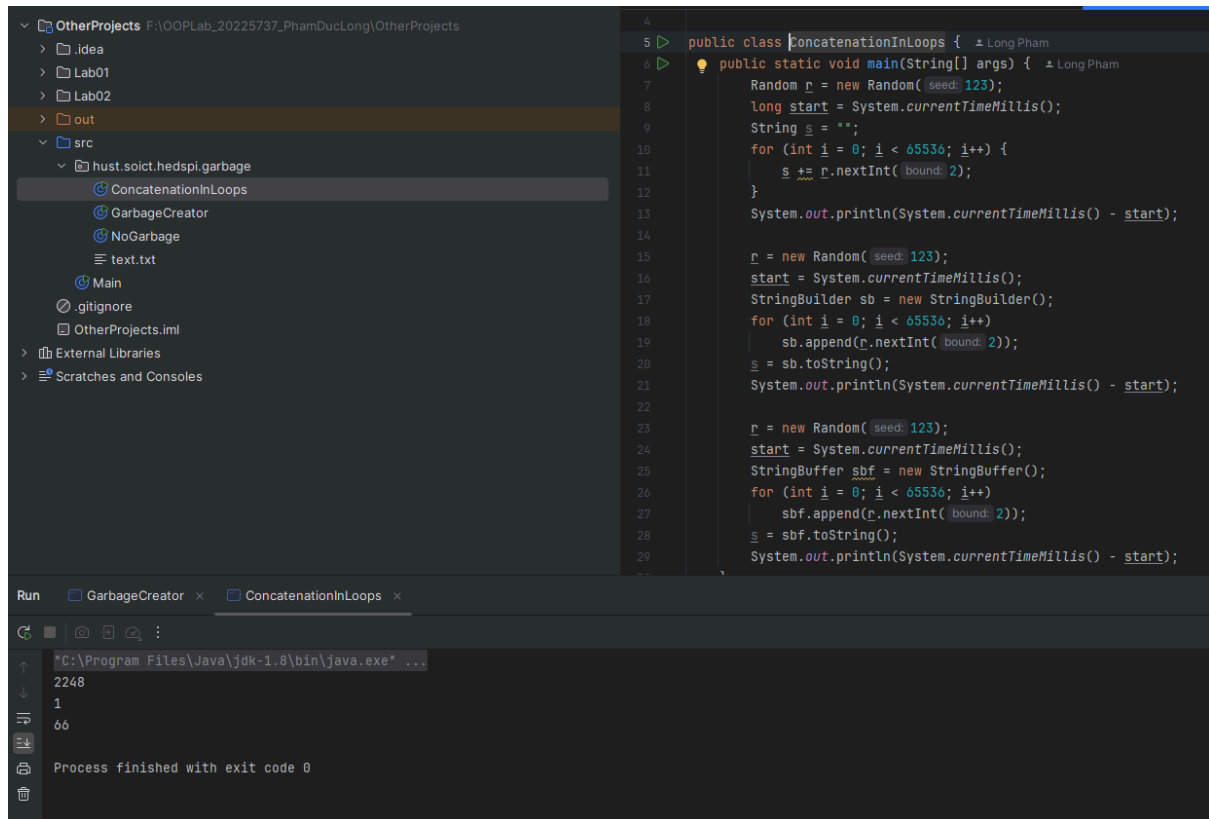
The Lion King
Star Wars
Aladin
The disc Star Wars has been removed.
The Lion King
Aladin

8. Re-organize your projects

9. String, StringBuilder and StringBuffer

9.1. So sánh String, StringBuilder and StringBuffer

Kết quả:



The screenshot shows an IDE with a project structure on the left and a code editor on the right. The project structure includes a package `hust.soict.hedspl.garbage` with classes `ConcatenationInLoops`, `GarbageCreator`, and `NoGarbage`. The code editor displays the `ConcatenationInLoops` class, which compares three methods of string concatenation: using `String`, `StringBuilder`, and `StringBuffer`. The `String` method uses `+` for concatenation, while `StringBuilder` and `StringBuffer` use `append`. The `String` method is the slowest, while `StringBuilder` is the fastest. The `StringBuffer` method is slightly slower than `StringBuilder` but more thread-safe.

```
4 public class ConcatenationInLoops {
5     public static void main(String[] args) {
6         Random r = new Random( seed: 123);
7         long start = System.currentTimeMillis();
8         String s = "";
9         for (int i = 0; i < 65536; i++) {
10             s += r.nextInt( bound: 2);
11         }
12         System.out.println(System.currentTimeMillis() - start);
13
14         r = new Random( seed: 123);
15         start = System.currentTimeMillis();
16         StringBuilder sb = new StringBuilder();
17         for (int i = 0; i < 65536; i++)
18             sb.append(r.nextInt( bound: 2));
19         s = sb.toString();
20         System.out.println(System.currentTimeMillis() - start);
21
22         r = new Random( seed: 123);
23         start = System.currentTimeMillis();
24         StringBuffer sbf = new StringBuffer();
25         for (int i = 0; i < 65536; i++)
26             sbf.append(r.nextInt( bound: 2));
27         s = sbf.toString();
28         System.out.println(System.currentTimeMillis() - start);
29     }
30 }
```

- Nhận xét:

- String kết hợp “+” sẽ tạo một đối tượng mới mỗi lần nối chuỗi, do đó hiệu suất kém nhất. Phương pháp này chỉ tốt khi xử lý ít chuỗi hoặc đoạn mã đơn giản.
- StringBuilder nhanh nhất vì không tạo ra đối tượng mới mỗi lần nối chuỗi.
- StringBuffer có hiệu suất thấp hơn StringBuilder một tý, nhưng có vẻ thích hợp hơn khi xử lý đa luồng (Do đôi khi kết quả nhận được của StringBuffer là 4 - 5 chữ không phải gần như 0 giống StringBuilder).

9.2. GarbageCreator và NoGarbage

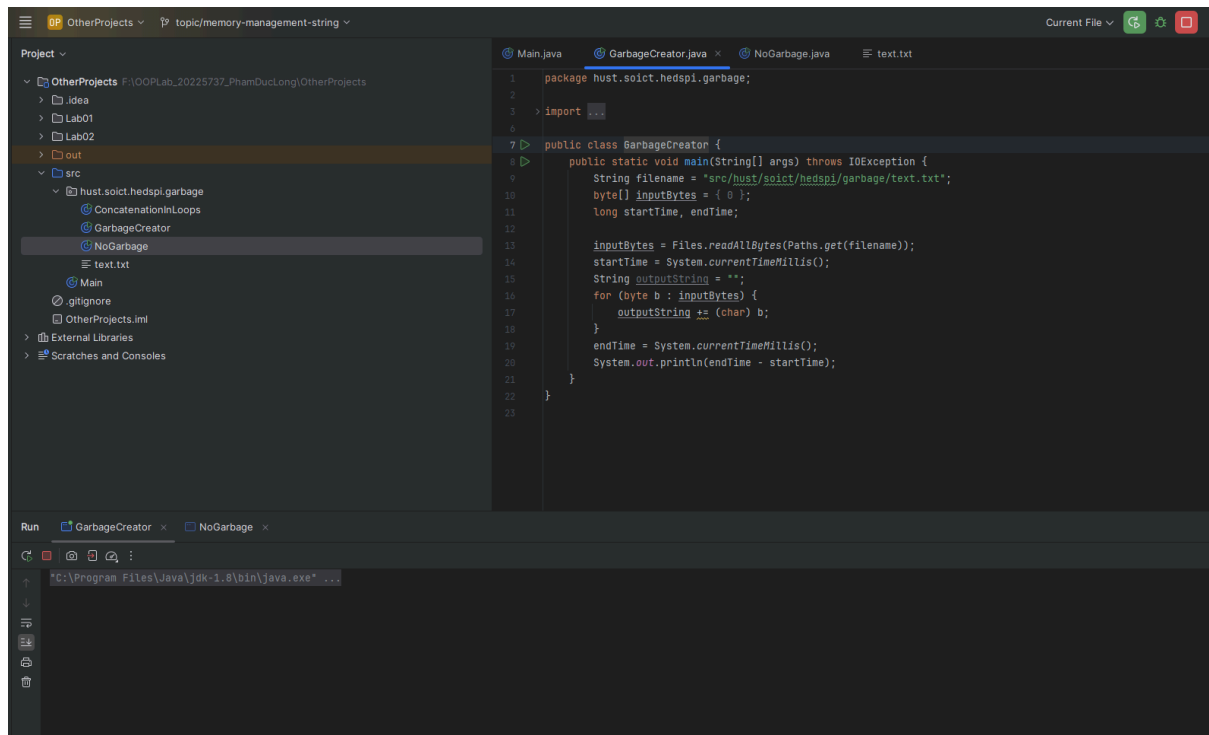
Tạo đoạn code sử dụng String với toán tử “+” để đọc một file txt có khoảng hơn 200000 dòng:

200432	abc
200433	abc
200434	abc
200435	abc
200436	abc
200437	abc
200438	abc
200439	abc
200440	abc
200441	abc
200442	abc
200443	abc
200444	abc
200445	abc
200446	abc
200447	abc
200448	abc

```
String filename = "src/hust/soict/hedspi/garbage/text.txt";
byte[] inputBytes = { 0 };
long startTime, endTime;

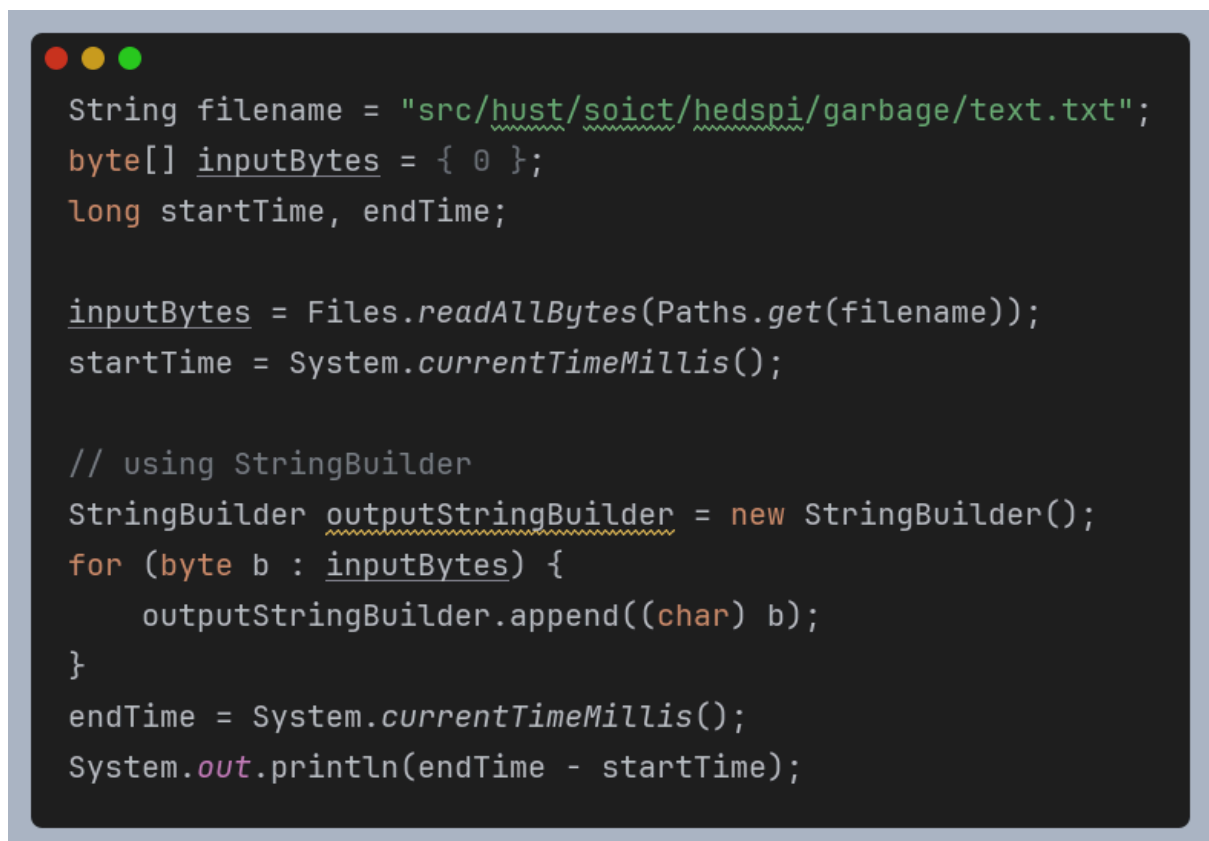
inputBytes = Files.readAllBytes(Paths.get(filename));
startTime = System.currentTimeMillis();
String outputString = "";
for (byte b : inputBytes) {
    outputString += (char) b;
}
endTime = System.currentTimeMillis();
System.out.println(endTime - startTime);
```

Kết quả thu được:

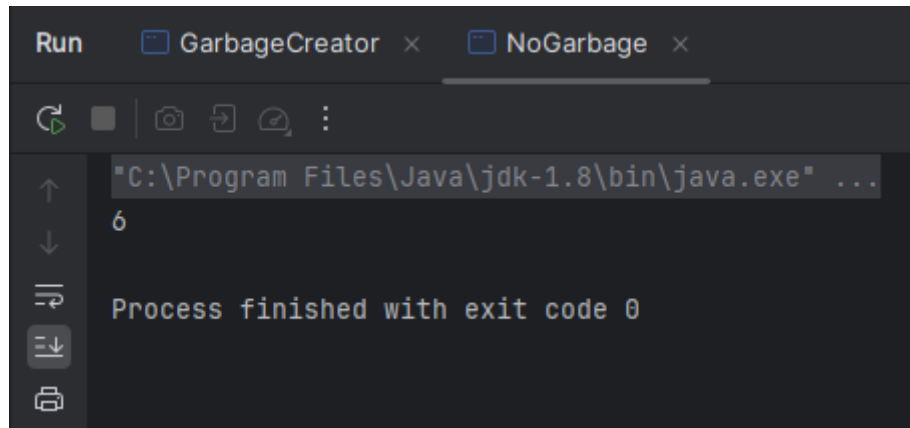


(Chương trình bị đơ)

Xử lý với NoGarbage sử dụng StringBuilder:



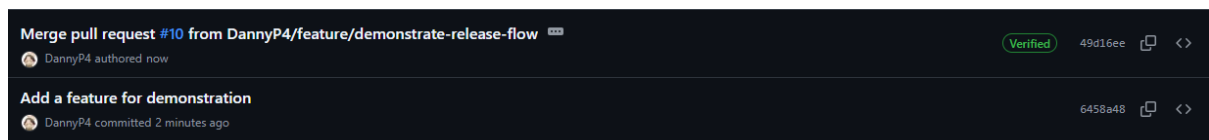
Kết quả thu được:



```
Run  GarbageCreator x  NoGarbage x
6
Process finished with exit code 0
```

Như đã giải thích ở trên, String sẽ tạo đối tượng mới do đó số lượng đối tượng String tăng lên nhanh chóng, đặc biệt là trong các vòng lặp lớn (200000 dòng, mỗi dòng chứa “abc\n” tương đương với 800000 vòng lặp) gây mất thời gian. Còn StringBuilder chỉ thay đổi nội dung của nó không có sự sao chép chuỗi không cần thiết nên tốc độ xử lý rất nhanh, lý tưởng.

10. Release flow demonstration



```
Merge pull request #10 from DannyP4/feature/demonstrate-release-flow
Verified 49d16ee
Add a feature for demonstration
6458a48
```