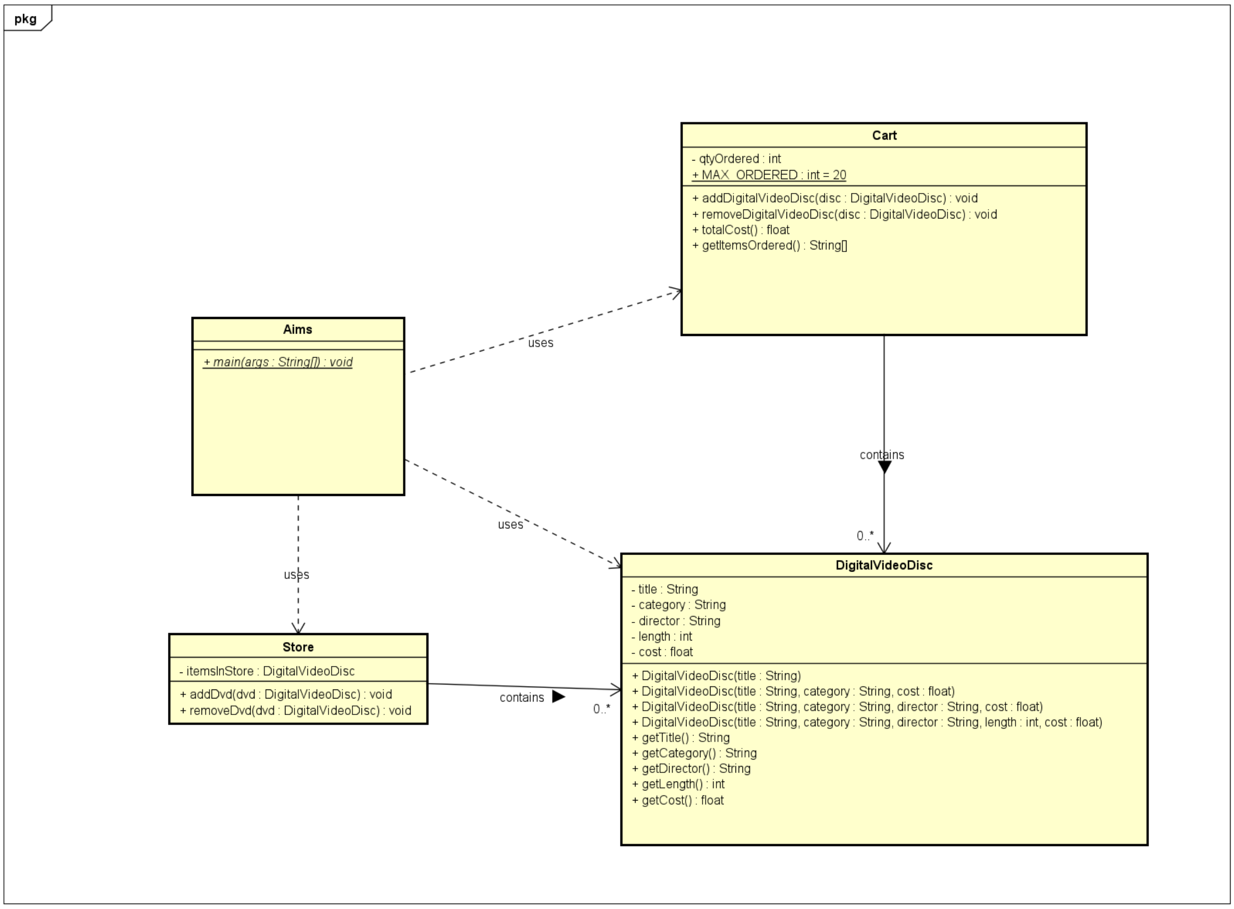
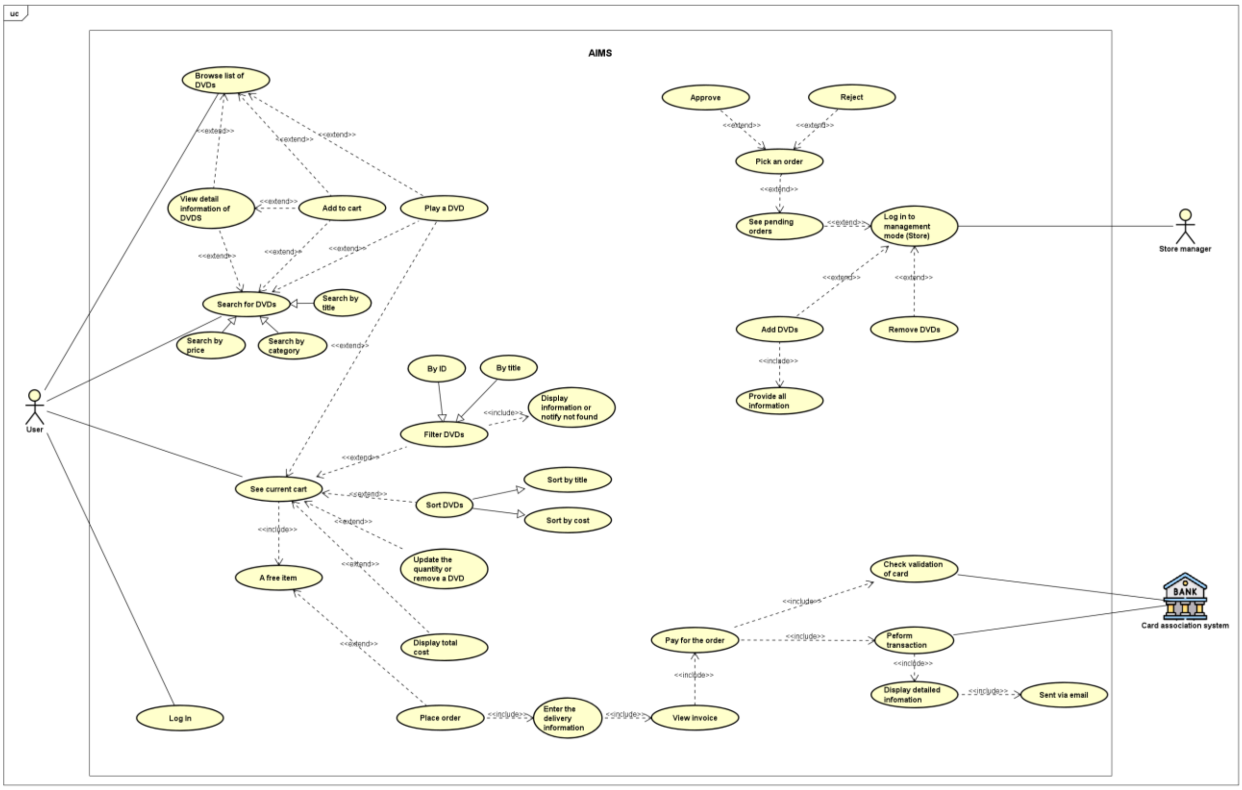
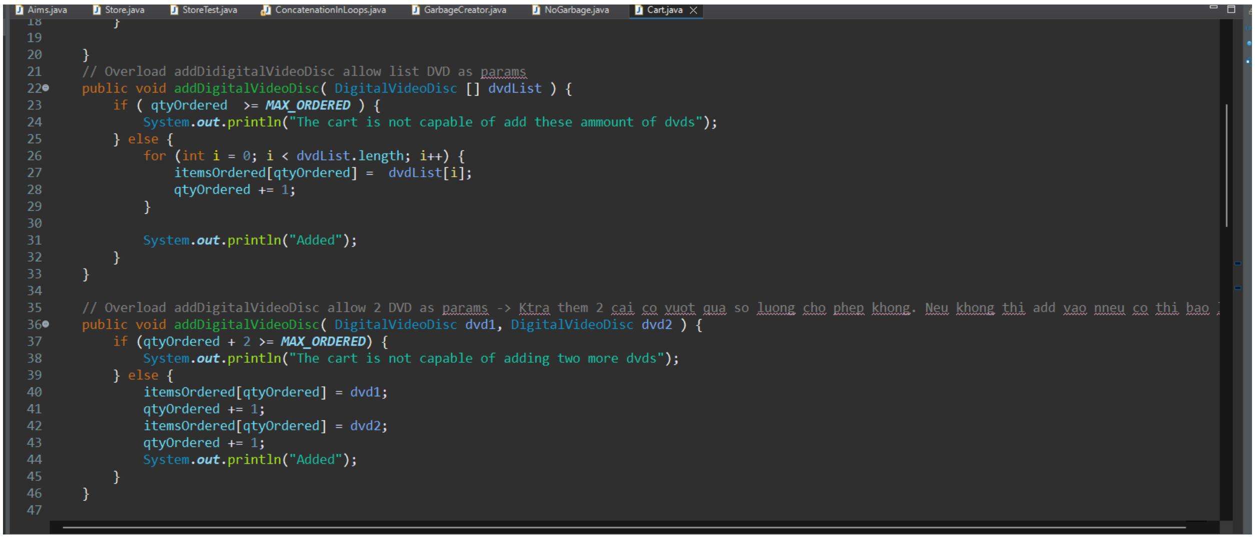
REPORT LAB 03

1. Update use-case diagram and class diagram


pkg
Cart
- qtyOrdered : int
+ MAX_ORDERED : int = 20
+ addDigitalVideoDisc(disc : DigitalVideoDisc) : void
+ removeDigitalVideoDisc(disc : DigitalVideoDisc) : void
+ totalCost() : float
+ getItemsOrdered() : String[]
Aims
+ main(args : String[]) : void
uses
uses
uses
contains
0..*
Store
- itemsInStore : DigitalVideoDisc
+ addDvd(dvd : DigitalVideoDisc) : void
+ removeDvd(dvd : DigitalVideoDisc) : void
contains
0..*
DigitalVideoDisc
- title : String
- category : String
- director : String
- length : int
- cost : float
+ DigitalVideoDisc(title : String)
+ DigitalVideoDisc(title : String, category : String, cost : float)
+ DigitalVideoDisc(title : String, category : String, director : String, cost : float)
+ DigitalVideoDisc(title : String, category : String, director : String, length : int, cost : float)
+ getTitle() : String
+ getCategory() : String
+ getDirector() : String
+ getLength() : int
+ getCost() : float



2. Working with method overloading

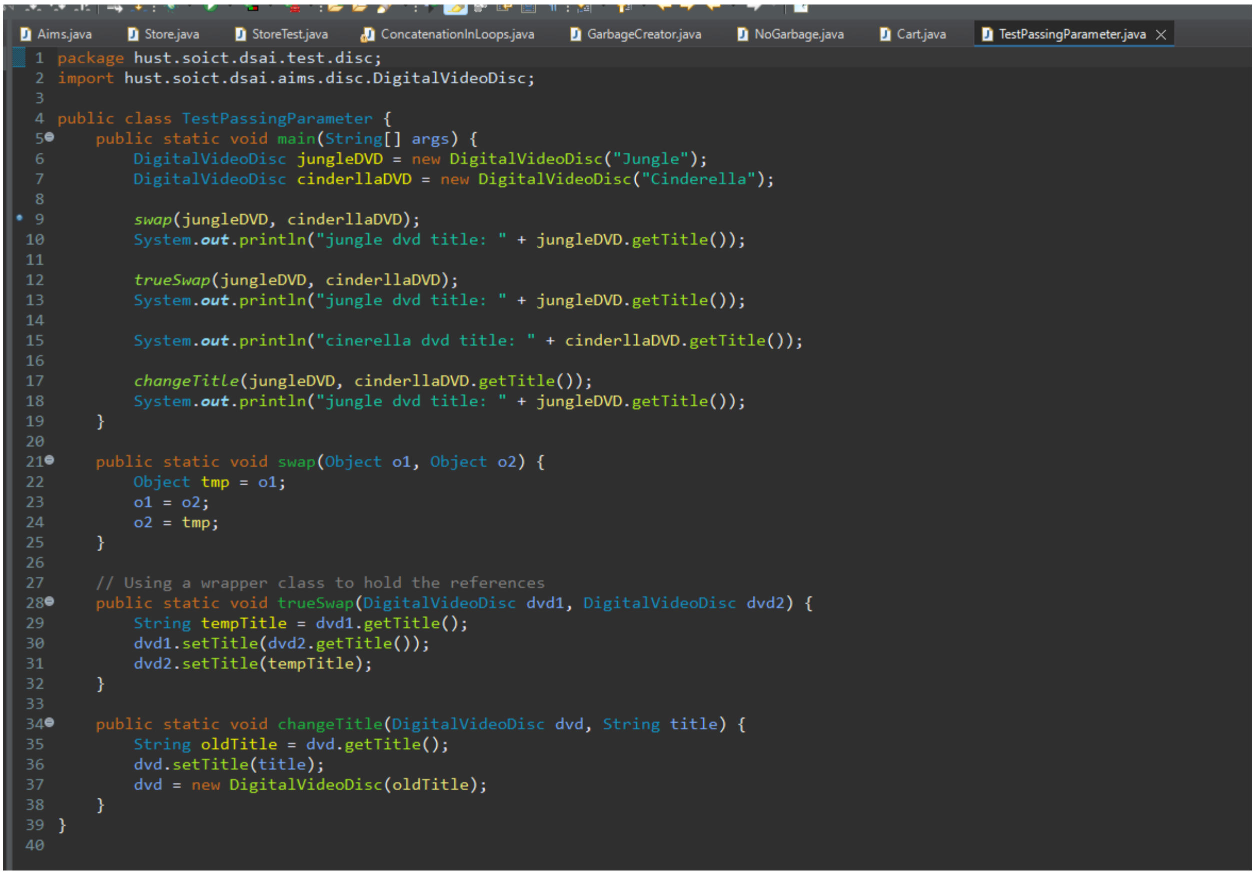


```java
        }

    }
    // Overload addDidigitalVideoDisc allow list DVD as params
    public void addDigitalVideoDisc( DigitalVideoDisc [] dvdList ) {
        if ( qtyOrdered  >= MAX_ORDERED ) {
            System.out.println("The cart is not capable of add these ammount of dvds");
        } else {
            for (int i = 0; i < dvdList.length; i++) {
                itemsOrdered[qtyOrdered] =  dvdList[i];
                qtyOrdered += 1;
            }

            System.out.println("Added");
        }
    }

    // Overload addDigitalVideoDisc allow 2 DVD as params -> Ktra them 2 cai co vuot qua so luong cho phep khong. Neu khong thi add vao nneu co thi bao
    public void addDigitalVideoDisc( DigitalVideoDisc dvd1, DigitalVideoDisc dvd2 ) {
        if (qtyOrdered + 2 >= MAX_ORDERED) {
            System.out.println("The cart is not capable of adding two more dvds");
        } else {
            itemsOrdered[qtyOrdered] = dvd1;
            qtyOrdered += 1;
            itemsOrdered[qtyOrdered] = dvd2;
            qtyOrdered += 1;
            System.out.println("Added");
        }
    }

```

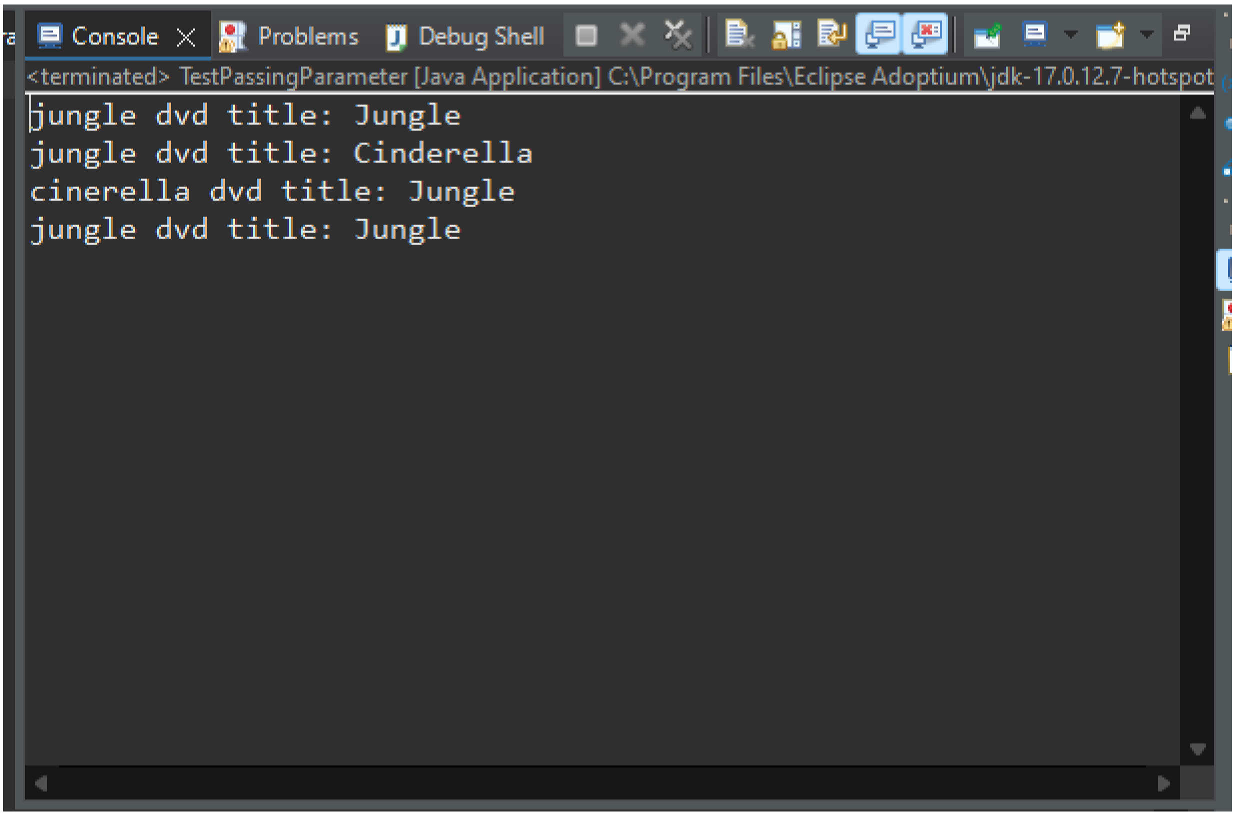
- Try to add a method addDigitalVideoDisc which allows to pass an arbitrary number of arguments for dvd. Compare to an array parameter. What do you prefer in this case?

+ I think I prefer the array parameter way as I can easily get the number of DVD I need to add to the cart to validate while it will be harder to handle if we just pass an arbitrary number of DVD.

3. Passing parameter



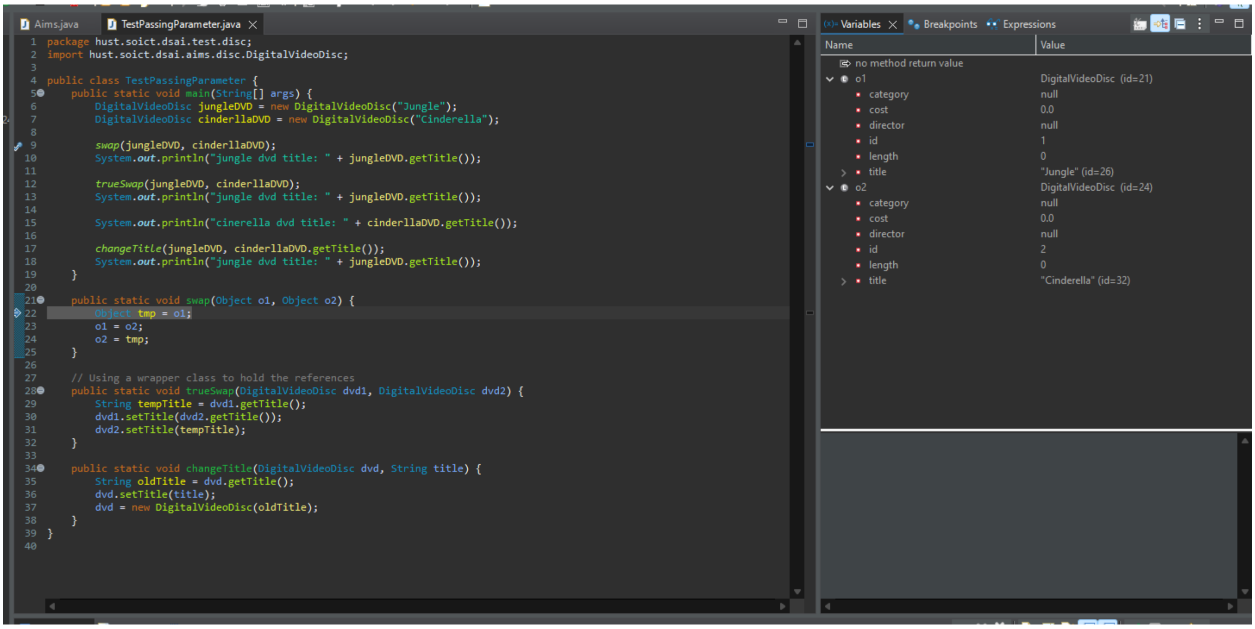
```java
package hust.soict.dsai.test.disc;
import hust.soict.dsai.aims.disc.DigitalVideoDisc;

public class TestPassingParameter {
    public static void main(String[] args) {
        DigitalVideoDisc jungleDVD = new DigitalVideoDisc("Jungle");
        DigitalVideoDisc cinderllaDVD = new DigitalVideoDisc("Cinderella");

        swap(jungleDVD, cinderllaDVD);
        System.out.println("jungle dvd title: " + jungleDVD.getTitle());

        trueSwap(jungleDVD, cinderllaDVD);
        System.out.println("jungle dvd title: " + jungleDVD.getTitle());

        System.out.println("cinerella dvd title: " + cinderllaDVD.getTitle());

        changeTitle(jungleDVD, cinderllaDVD.getTitle());
        System.out.println("jungle dvd title: " + jungleDVD.getTitle());
    }

    public static void swap(Object o1, Object o2) {
        Object tmp = o1;
        o1 = o2;
        o2 = tmp;
    }

    // Using a wrapper class to hold the references
    public static void trueSwap(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) {
        String tempTitle = dvd1.getTitle();
        dvd1.setTitle(dvd2.getTitle());
        dvd2.setTitle(tempTitle);
    }

    public static void changeTitle(DigitalVideoDisc dvd, String title) {
        String oldTitle = dvd.getTitle();
        dvd.setTitle(title);
        dvd = new DigitalVideoDisc(oldTitle);
    }
}

```



```
<terminated> TestPassingParameter [Java Application] C:\Program Files\Eclipse Adoptium\jdk-17.0.12.7-hotspot
jungle dvd title: Jungle
jungle dvd title: Cinderella
cinerella dvd title: Jungle
jungle dvd title: Jungle
```

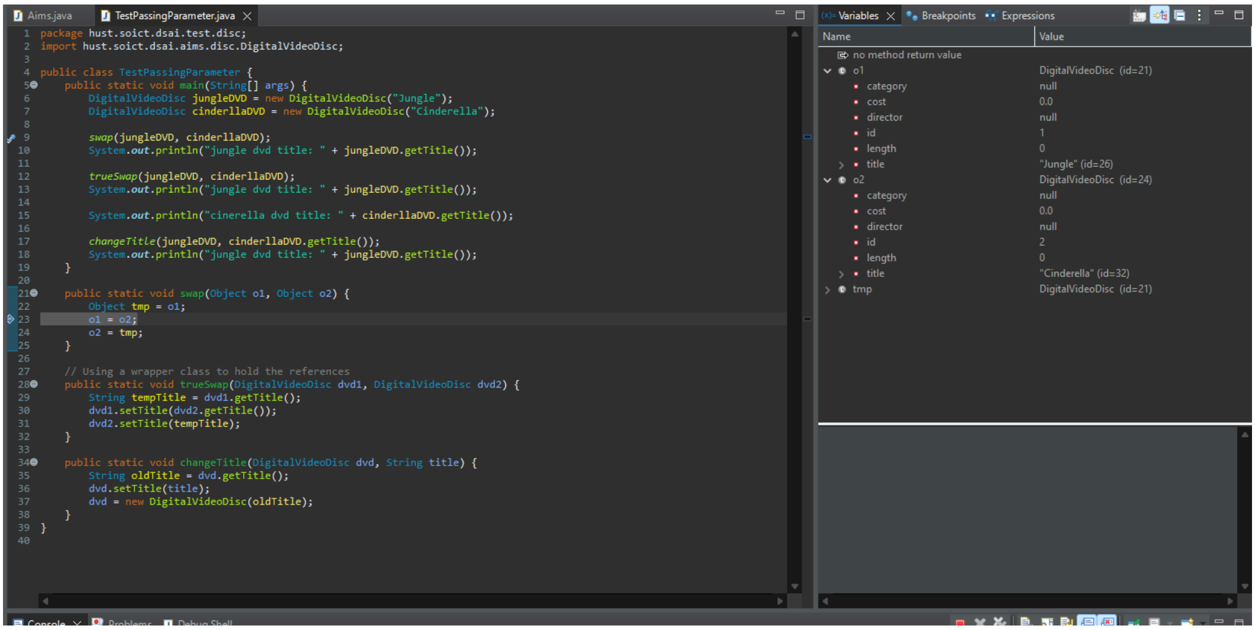
- ***Is JAVA a Pass by Value or a Pass by Reference programming language?***
  - Java Pass by Value. For example, if we pass an object into a method in Java (swap(DVD dvd1, DVD dvd2)) then in the method swap only recieve the address value point to the dvd1 Object and dvd2 object in the memory so if we try to swap by Obj tmp = dvd1; dvd1 = dvd2; dvd2 = tmp then it won't work. Because it is only dvd1 and dvd2 in the method change value for each other which does not affect original 2 objects.
- After the call of **swap(jungleDVD, cinderellaDVD)** why does the title of these two objects still remain?
  - As i said earlier, o1 and o2 are just local variables of method swap so swap value of o1 and o2 does not affect the value of original objects which are jungleDVD and cinderellaDVD. So the value of jungleDVD and cinderellaDVD still remain
- After the call of **changeTitle(jungleDVD, cinderellaDVD.getTitle())** why is the title of the JungleDVD changed?
  - In changeTitle, we have passed down the address of jungleDVD object so when we modify the title of dvd (which is the jungleDVD object) it also changes the title of jungleDVD because both point to the same Object.

4. Debugging Java in Eclipse



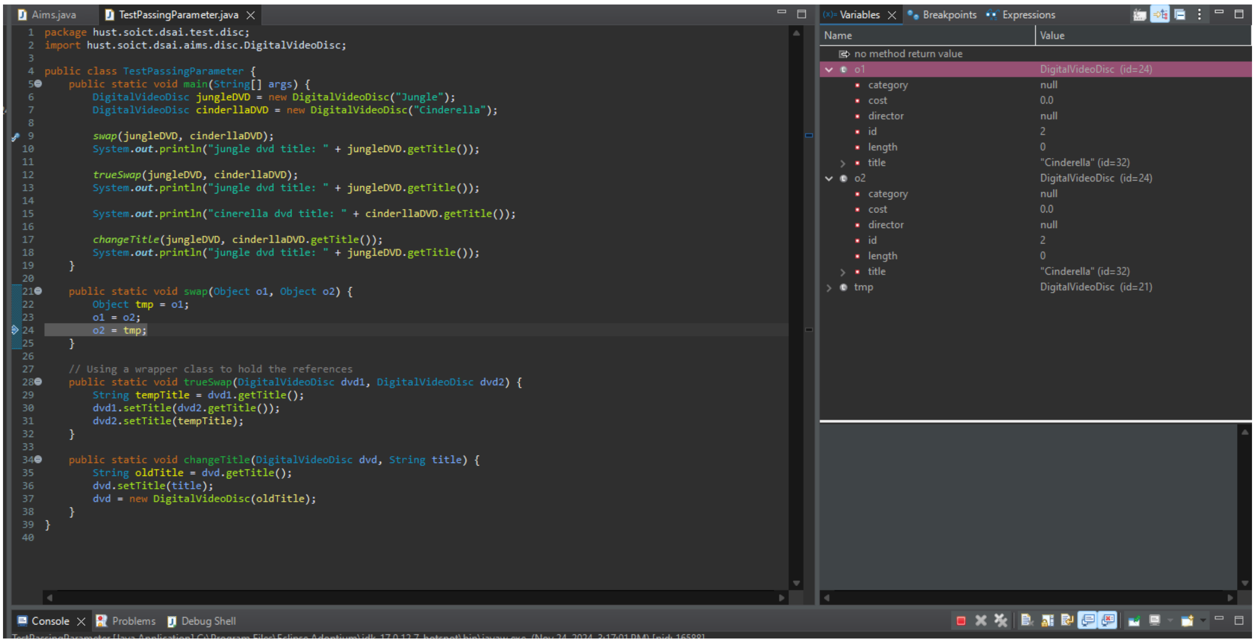
```java
package hust.soict.dsai.test.disc;
import hust.soict.dsai.aims.disc.DigitalVideoDisc;

public class TestPassingParameter {
    public static void main(String[] args) {
        DigitalVideoDisc jungleDVD = new DigitalVideoDisc("Jungle");
        DigitalVideoDisc cinderllaDVD = new DigitalVideoDisc("Cinderella");

        swap(jungleDVD, cinderllaDVD);
        System.out.println("jungle dvd title: " + jungleDVD.getTitle());

        trueSwap(jungleDVD, cinderllaDVD);
        System.out.println("jungle dvd title: " + jungleDVD.getTitle());

        System.out.println("cinerella dvd title: " + cinderllaDVD.getTitle());

        changeTitle(jungleDVD, cinderllaDVD.getTitle());
        System.out.println("jungle dvd title: " + jungleDVD.getTitle());
    }

    public static void swap(Object o1, Object o2) {
        Object tmp = o1;
        o1 = o2;
        o2 = tmp;
    }

    // Using a wrapper class to hold the references
    public static void trueSwap(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) {
        String tempTitle = dvd1.getTitle();
        dvd1.setTitle(dvd2.getTitle());
        dvd2.setTitle(tempTitle);
    }

    public static void changeTitle(DigitalVideoDisc dvd, String title) {
        String oldTitle = dvd.getTitle();
        dvd.setTitle(title);
        dvd = new DigitalVideoDisc(oldTitle);
    }
}
```

```
Name                        Value
no method return value
o1                          DigitalVideoDisc  (id=21)
    category                null
    cost                    0.0
    director                null
    id                      1
    length                  0
    title                   "Jungle" (id=26)
o2                          DigitalVideoDisc  (id=24)
    category                null
    cost                    0.0
    director                null
    id                      2
    length                  0
    title                   "Cinderella" (id=32)
```



```java
package hust.soict.dsai.test.disc;
import hust.soict.dsai.aims.disc.DigitalVideoDisc;

public class TestPassingParameter {
    public static void main(String[] args) {
        DigitalVideoDisc jungleDVD = new DigitalVideoDisc("Jungle");
        DigitalVideoDisc cinderllaDVD = new DigitalVideoDisc("Cinderella");

        swap(jungleDVD, cinderllaDVD);
        System.out.println("jungle dvd title: " + jungleDVD.getTitle());

        trueSwap(jungleDVD, cinderllaDVD);
        System.out.println("jungle dvd title: " + jungleDVD.getTitle());

        System.out.println("cinerella dvd title: " + cinderllaDVD.getTitle());

        changeTitle(jungleDVD, cinderllaDVD.getTitle());
        System.out.println("jungle dvd title: " + jungleDVD.getTitle());
    }

    public static void swap(Object o1, Object o2) {
        Object tmp = o1;
        o1 = o2;
        o2 = tmp;
    }

    // Using a wrapper class to hold the references
    public static void trueSwap(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) {
        String tempTitle = dvd1.getTitle();
        dvd1.setTitle(dvd2.getTitle());
        dvd2.setTitle(tempTitle);
    }

    public static void changeTitle(DigitalVideoDisc dvd, String title) {
        String oldTitle = dvd.getTitle();
        dvd.setTitle(title);
        dvd = new DigitalVideoDisc(oldTitle);
    }
}
```

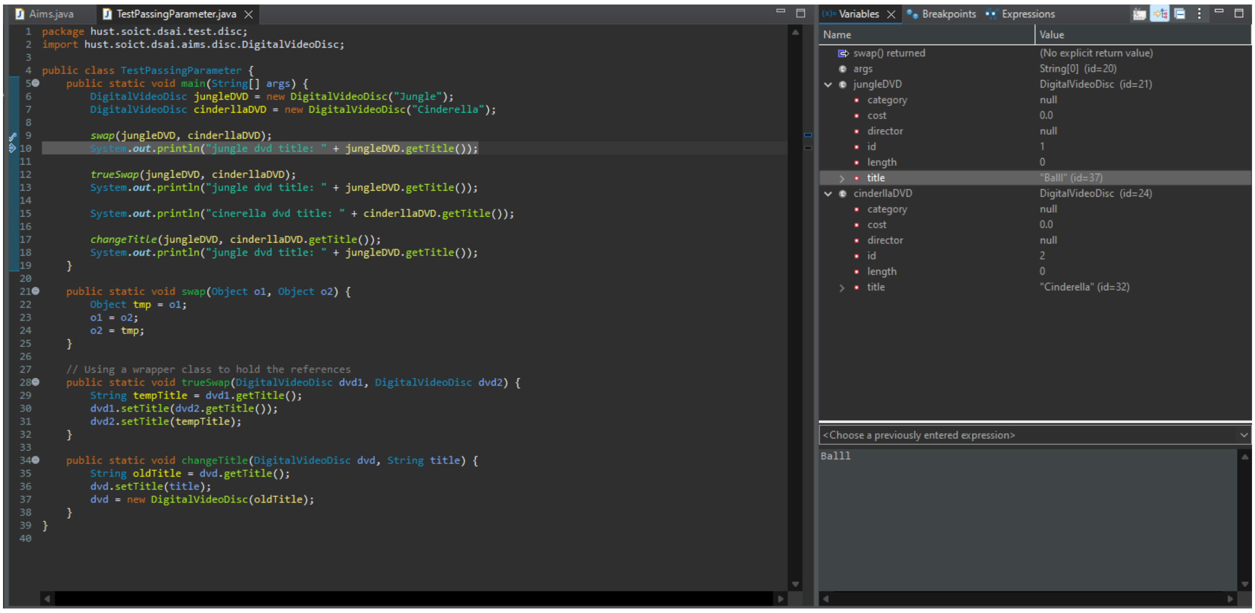
```
Name                        Value
no method return value
o1                          DigitalVideoDisc  (id=21)
    category                null
    cost                    0.0
    director                null
    id                      1
    length                  0
    title                   "Jungle" (id=26)
o2                          DigitalVideoDisc  (id=24)
    category                null
    cost                    0.0
    director                null
    id                      2
    length                  0
    title                   "Cinderella" (id=32)
tmp                         DigitalVideoDisc  (id=21)
```



```java
package hust.soict.dsai.test.disc;
import hust.soict.dsai.aims.disc.DigitalVideoDisc;

public class TestPassingParameter {
    public static void main(String[] args) {
        DigitalVideoDisc jungleDVD = new DigitalVideoDisc("Jungle");
        DigitalVideoDisc cinderllaDVD = new DigitalVideoDisc("Cinderella");

        swap(jungleDVD, cinderllaDVD);
        System.out.println("jungle dvd title: " + jungleDVD.getTitle());

        trueSwap(jungleDVD, cinderllaDVD);
        System.out.println("jungle dvd title: " + jungleDVD.getTitle());

        System.out.println("cinerella dvd title: " + cinderllaDVD.getTitle());

        changeTitle(jungleDVD, cinderllaDVD.getTitle());
        System.out.println("jungle dvd title: " + jungleDVD.getTitle());
    }

    public static void swap(Object o1, Object o2) {
        Object tmp = o1;
        o1 = o2;
        o2 = tmp;
    }

    // Using a wrapper class to hold the references
    public static void trueSwap(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) {
        String tempTitle = dvd1.getTitle();
        dvd1.setTitle(dvd2.getTitle());
        dvd2.setTitle(tempTitle);
    }

    public static void changeTitle(DigitalVideoDisc dvd, String title) {
        String oldTitle = dvd.getTitle();
        dvd.setTitle(title);
        dvd = new DigitalVideoDisc(oldTitle);
    }
}
```

```
Name                        Value
no method return value
o1                          DigitalVideoDisc  (id=24)
    category                null
    cost                    0.0
    director                null
    id                      2
    length                  0
    title                   "Cinderella" (id=32)
o2                          DigitalVideoDisc  (id=24)
    category                null
    cost                    0.0
    director                null
    id                      2
    length                  0
    title                   "Cinderella" (id=32)
tmp                         DigitalVideoDisc  (id=21)
```

```
TestPassingParamter [Java Application] C:\Program Files\Eclipse Adoptium\jdk-17.0.12.7-hotspot\bin\javaw.exe (Nov 24, 2024, 3:17:01 PM) [pid: 16588]
```



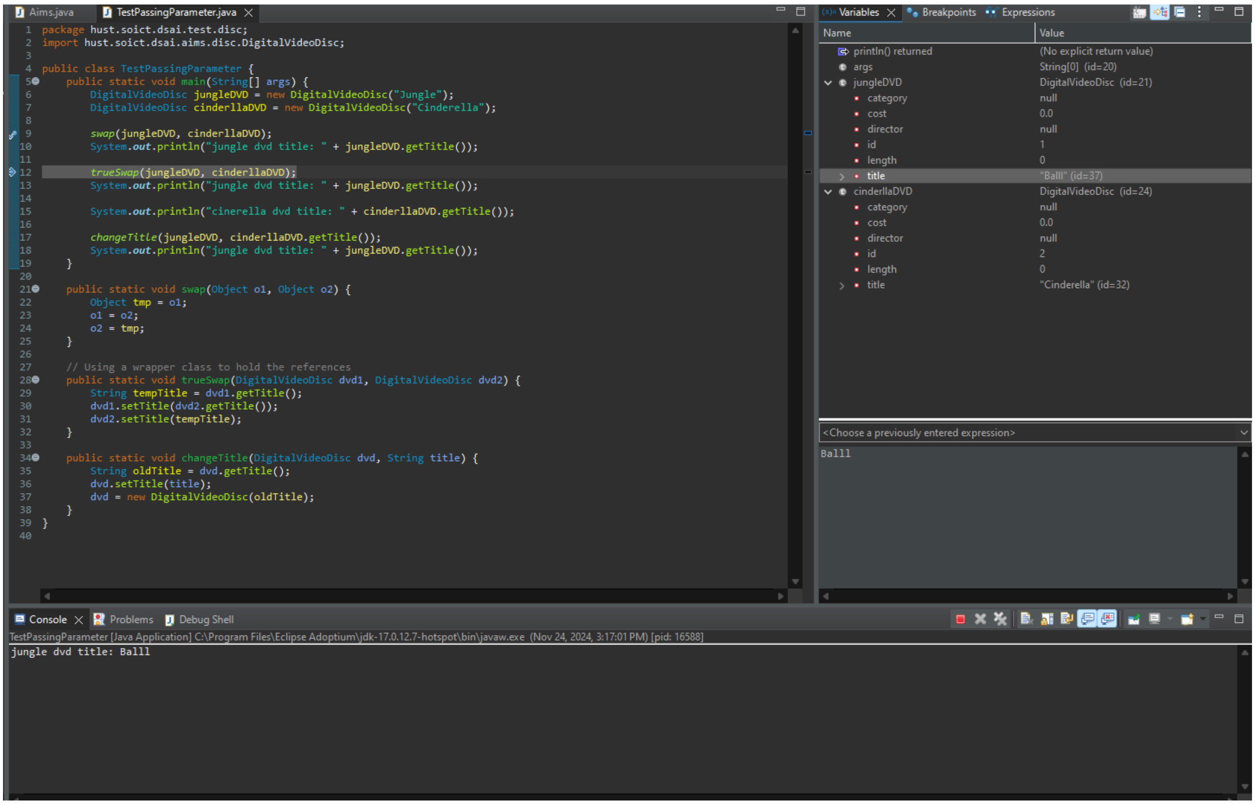
```java
package hust.soict.dsai.test.disc;
import hust.soict.dsai.aims.disc.DigitalVideoDisc;

public class TestPassingParameter {
    public static void main(String[] args) {
        DigitalVideoDisc jungleDVD = new DigitalVideoDisc("Jungle");
        DigitalVideoDisc cinderllaDVD = new DigitalVideoDisc("Cinderella");

        swap(jungleDVD, cinderllaDVD);
        System.out.println("jungle dvd title: " + jungleDVD.getTitle());

        trueSwap(jungleDVD, cinderllaDVD);
        System.out.println("jungle dvd title: " + jungleDVD.getTitle());

        System.out.println("cinerella dvd title: " + cinderllaDVD.getTitle());

        changeTitle(jungleDVD, cinderllaDVD.getTitle());
        System.out.println("jungle dvd title: " + jungleDVD.getTitle());
    }

    public static void swap(Object o1, Object o2) {
        Object tmp = o1;
        o1 = o2;
        o2 = tmp;
    }

    // Using a wrapper class to hold the references
    public static void trueSwap(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) {
        String tempTitle = dvd1.getTitle();
        dvd1.setTitle(dvd2.getTitle());
        dvd2.setTitle(tempTitle);
    }

    public static void changeTitle(DigitalVideoDisc dvd, String title) {
        String oldTitle = dvd.getTitle();
        dvd.setTitle(title);
        dvd = new DigitalVideoDisc(oldTitle);
    }
}
```

Name	Value
swap() returned	(No explicit return value)
args	String[0] (id=20)
jungleDVD	DigitalVideoDisc (id=21)
category	null
cost	0.0
director	null
id	1
length	0
title	"Balll" (id=37)
cinderllaDVD	DigitalVideoDisc (id=24)
category	null
cost	0.0
director	null
id	2
length	0
title	"Cinderella" (id=32)

<Choose a previously entered expression>

```
Balll
```

Result:



```java
package hust.soict.dsai.test.disc;
import hust.soict.dsai.aims.disc.DigitalVideoDisc;

public class TestPassingParameter {
    public static void main(String[] args) {
        DigitalVideoDisc jungleDVD = new DigitalVideoDisc("Jungle");
        DigitalVideoDisc cinderllaDVD = new DigitalVideoDisc("Cinderella");

        swap(jungleDVD, cinderllaDVD);
        System.out.println("jungle dvd title: " + jungleDVD.getTitle());

        trueSwap(jungleDVD, cinderllaDVD);
        System.out.println("jungle dvd title: " + jungleDVD.getTitle());

        System.out.println("cinerella dvd title: " + cinderllaDVD.getTitle());

        changeTitle(jungleDVD, cinderllaDVD.getTitle());
        System.out.println("jungle dvd title: " + jungleDVD.getTitle());
    }

    public static void swap(Object o1, Object o2) {
        Object tmp = o1;
        o1 = o2;
        o2 = tmp;
    }

    // Using a wrapper class to hold the references
    public static void trueSwap(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) {
        String tempTitle = dvd1.getTitle();
        dvd1.setTitle(dvd2.getTitle());
        dvd2.setTitle(tempTitle);
    }

    public static void changeTitle(DigitalVideoDisc dvd, String title) {
        String oldTitle = dvd.getTitle();
        dvd.setTitle(title);
        dvd = new DigitalVideoDisc(oldTitle);
    }
}
```

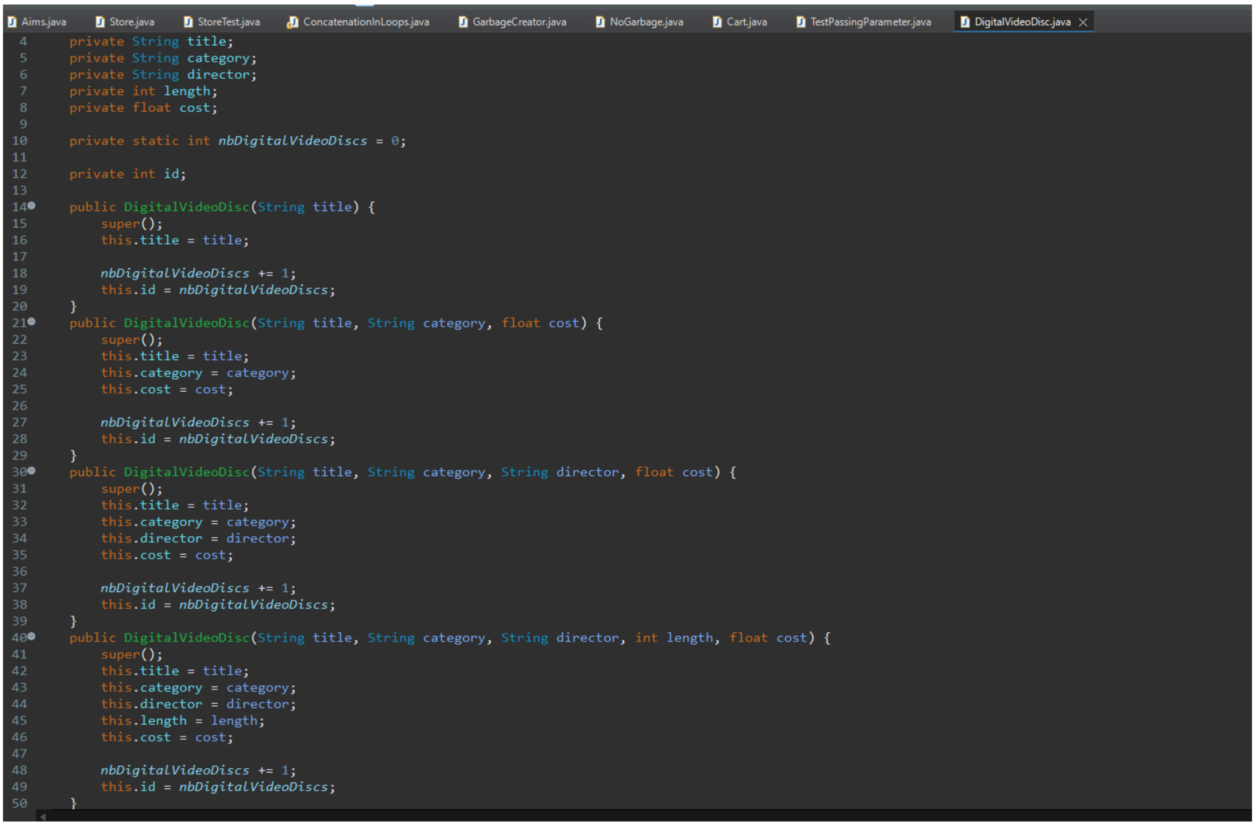
Name	Value
println() returned	(No explicit return value)
args	String[0] (id=20)
jungleDVD	DigitalVideoDisc (id=21)
category	null
cost	0.0
director	null
id	1
length	0
title	"Balll" (id=37)
cinderllaDVD	DigitalVideoDisc (id=24)
category	null
cost	0.0
director	null
id	2
length	0
title	"Cinderella" (id=32)

<Choose a previously entered expression>

```
Balll
```

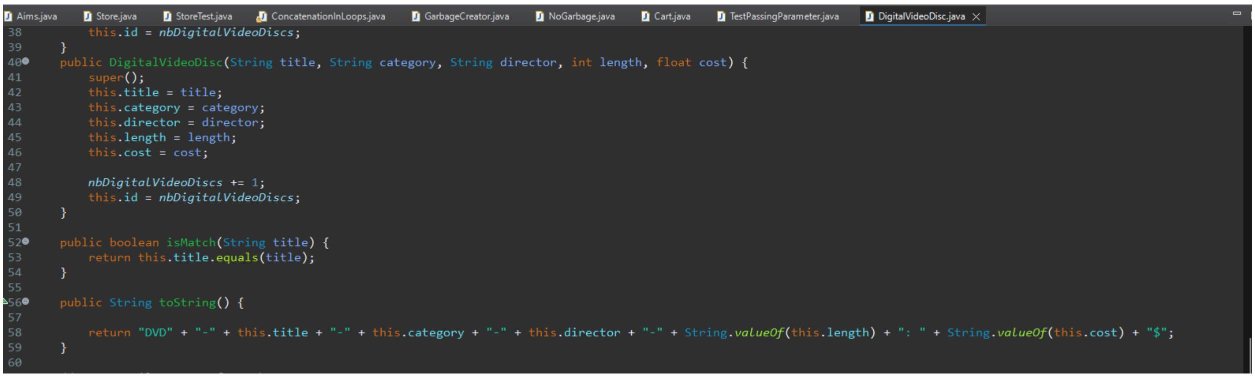
```
TestPassingParameter [Java Application] C:\Program Files\Eclipse Adoptium\jdk-17.0.12.7-hotspot\bin\javaw.exe  (Nov 24, 2024, 3:17:01 PM) [pid: 16588]
jungle dvd title: Balll
```

5. Classifier Member and Instance Member



```java
    private String title;
    private String category;
    private String director;
    private int length;
    private float cost;

    private static int nbDigitalVideoDiscs = 0;

    private int id;

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

        nbDigitalVideoDiscs += 1;
        this.id = nbDigitalVideoDiscs;
    }
    public DigitalVideoDisc(String title, String category, float cost) {
        super();
        this.title = title;
        this.category = category;
        this.cost = cost;

        nbDigitalVideoDiscs += 1;
        this.id = nbDigitalVideoDiscs;
    }
    public DigitalVideoDisc(String title, String category, String director, float cost) {
        super();
        this.title = title;
        this.category = category;
        this.director = director;
        this.cost = cost;

        nbDigitalVideoDiscs += 1;
        this.id = nbDigitalVideoDiscs;
    }
    public DigitalVideoDisc(String title, String category, String director, int length, float cost) {
        super();
        this.title = title;
        this.category = category;
        this.director = director;
        this.length = length;
        this.cost = cost;

        nbDigitalVideoDiscs += 1;
        this.id = nbDigitalVideoDiscs;
    }
```

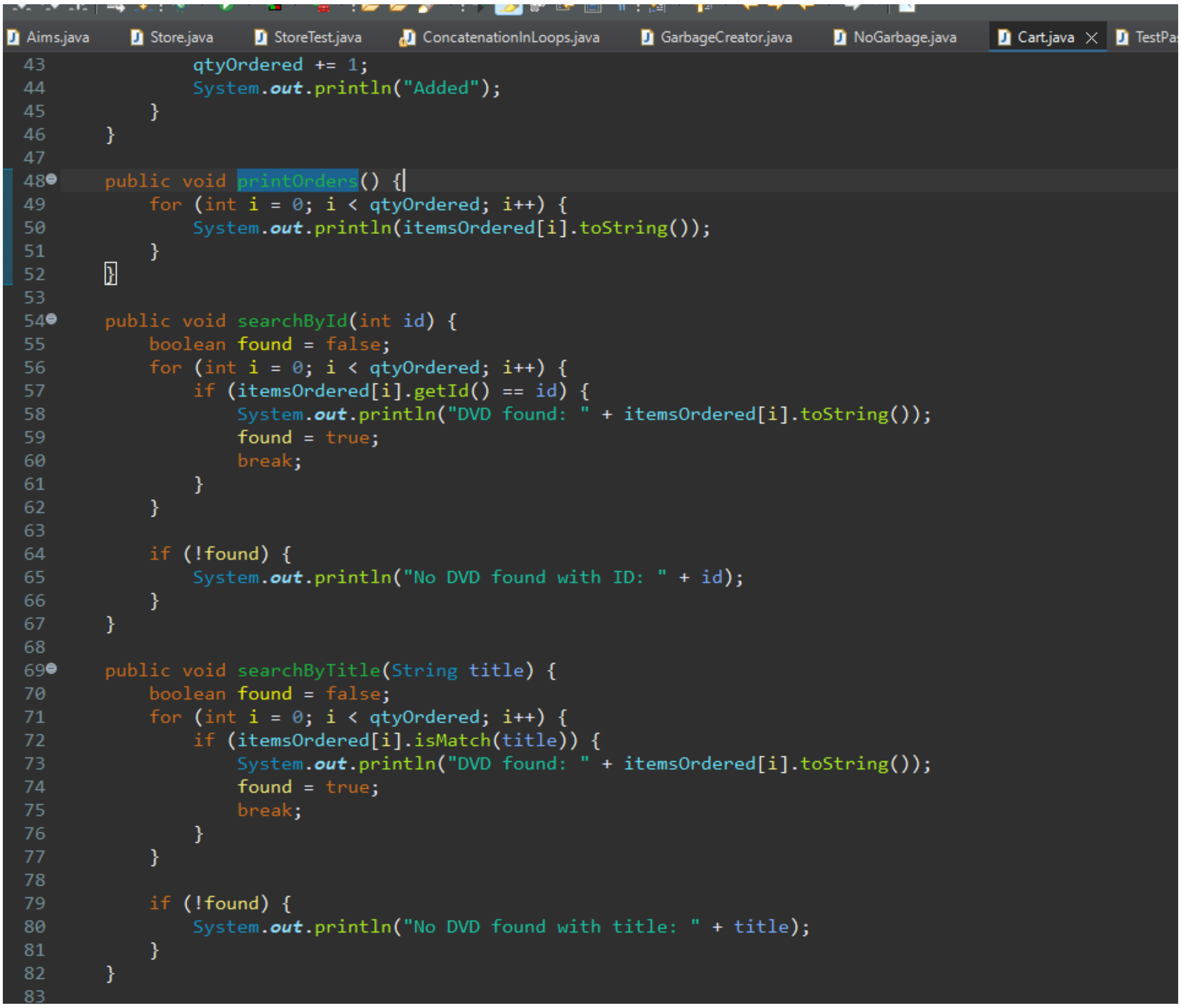
6. Open the **Cart** class



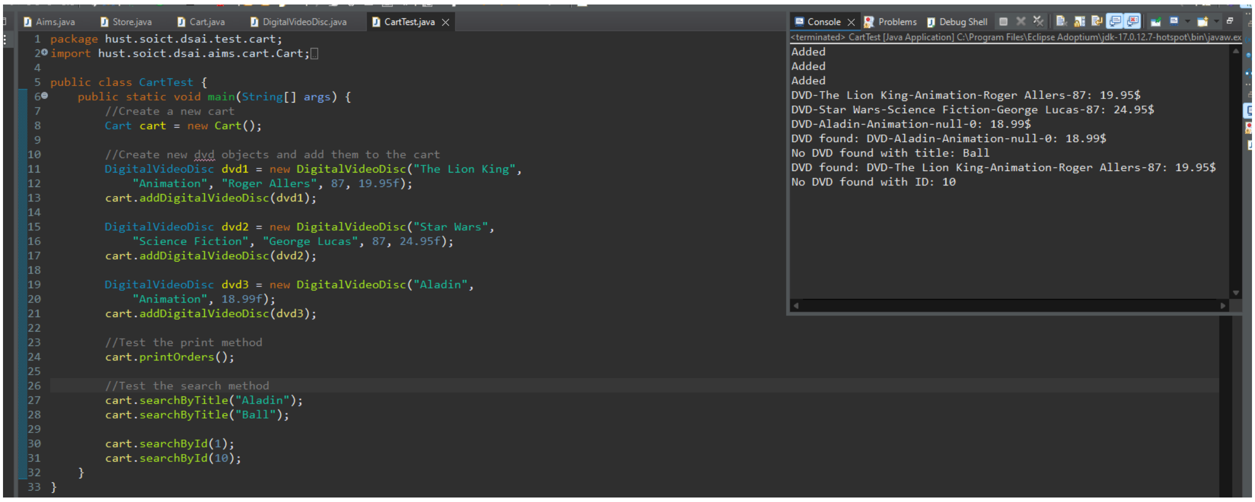
```java
        this.id = nbDigitalVideoDiscs;
    }
    public DigitalVideoDisc(String title, String category, String director, int length, float cost) {
        super();
        this.title = title;
        this.category = category;
        this.director = director;
        this.length = length;
        this.cost = cost;

        nbDigitalVideoDiscs += 1;
        this.id = nbDigitalVideoDiscs;
    }

    public boolean isMatch(String title) {
        return this.title.equals(title);
    }

    public String toString() {

        return "DVD" + "-" + this.title + "-" + this.category + "-" + this.director + "-" + String.valueOf(this.length) + ": " + String.valueOf(this.cost) + "$";
    }

```

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

  + The method should return a String.



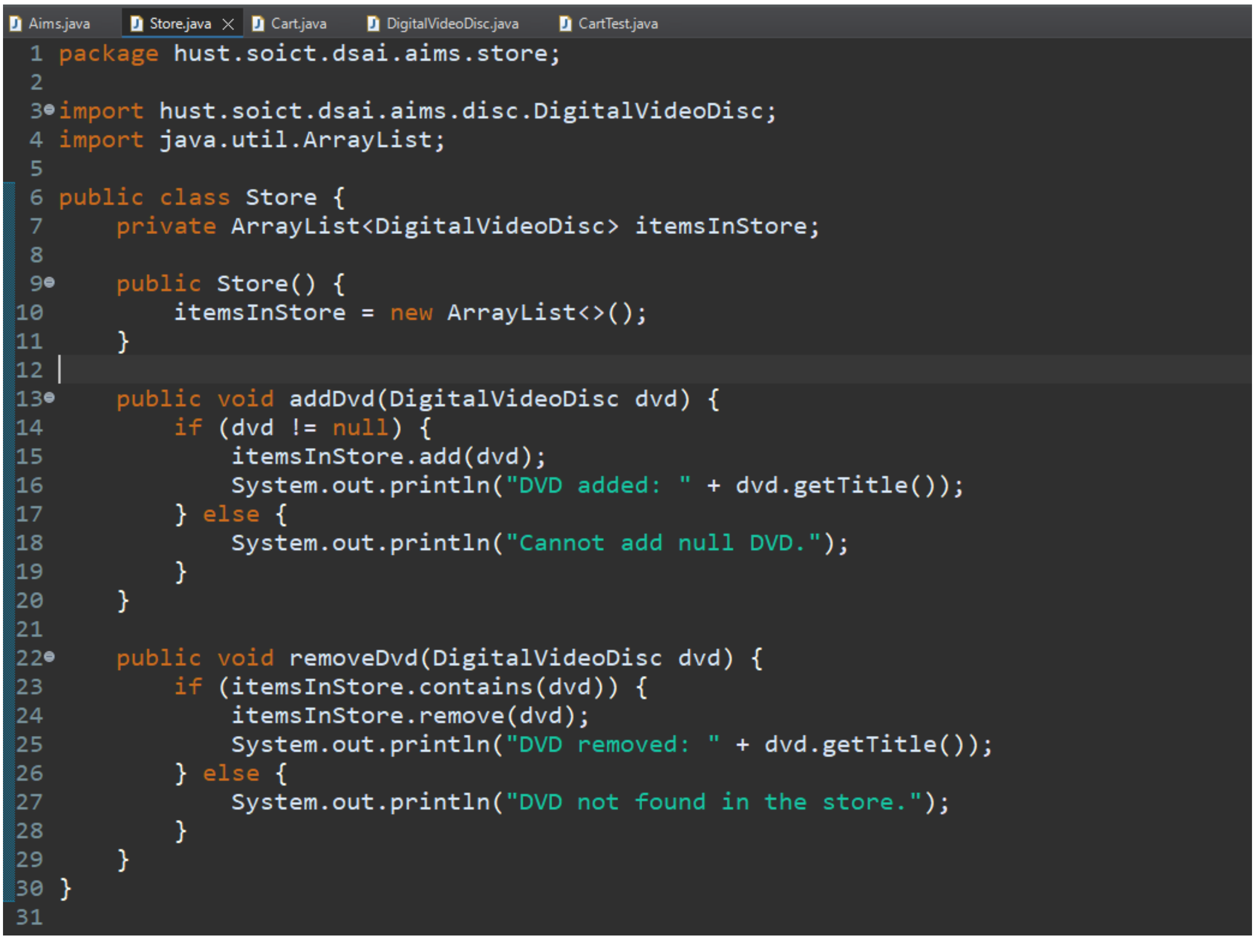
```java
            qtyOrdered += 1;
            System.out.println("Added");
        }
    }

    public void printOrders() {
        for (int i = 0; i < qtyOrdered; i++) {
            System.out.println(itemsOrdered[i].toString());
        }
    }

    public void searchById(int id) {
        boolean found = false;
        for (int i = 0; i < qtyOrdered; i++) {
            if (itemsOrdered[i].getId() == id) {
                System.out.println("DVD found: " + itemsOrdered[i].toString());
                found = true;
                break;
            }
        }

        if (!found) {
            System.out.println("No DVD found with ID: " + id);
        }
    }

    public void searchByTitle(String title) {
        boolean found = false;
        for (int i = 0; i < qtyOrdered; i++) {
            if (itemsOrdered[i].isMatch(title)) {
                System.out.println("DVD found: " + itemsOrdered[i].toString());
                found = true;
                break;
            }
        }

        if (!found) {
            System.out.println("No DVD found with title: " + title);
        }
    }

```



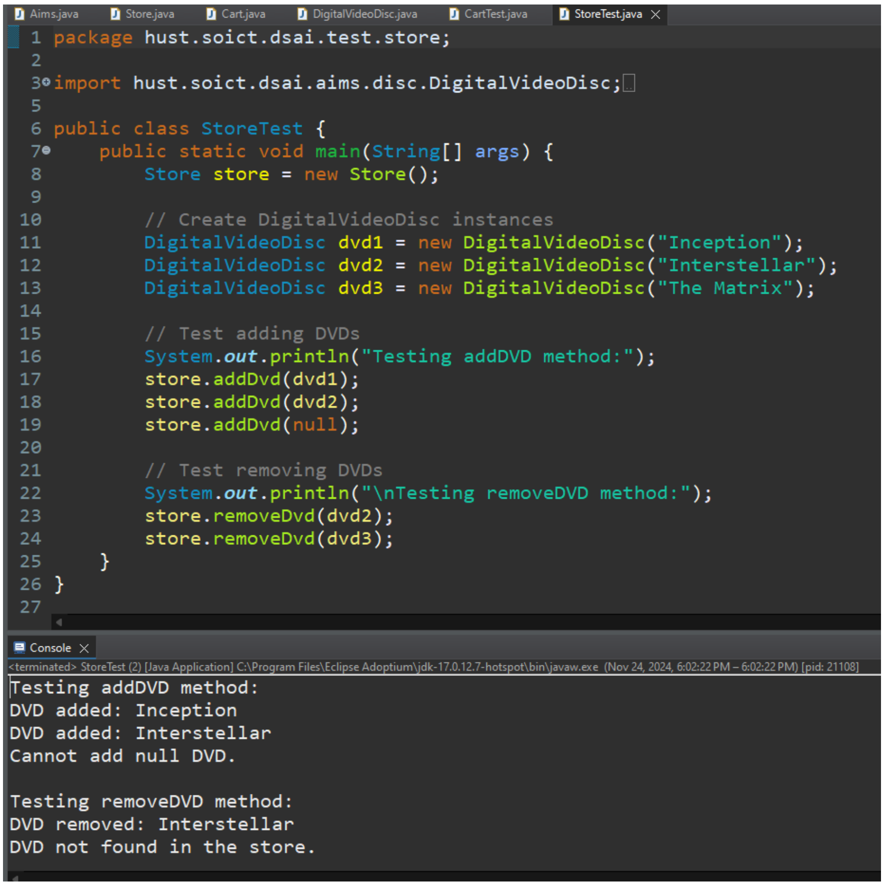
```java
package hust.soict.dsai.test.cart;
import hust.soict.dsai.aims.cart.Cart;

public class CartTest {
    public static void main(String[] args) {
        //Create a new cart
        Cart cart = new Cart();

        //Create new dvd objects and add them to the cart
        DigitalVideoDisc dvd1 = new DigitalVideoDisc("The Lion King",
            "Animation", "Roger Allers", 87, 19.95f);
        cart.addDigitalVideoDisc(dvd1);

        DigitalVideoDisc dvd2 = new DigitalVideoDisc("Star Wars",
            "Science Fiction", "George Lucas", 87, 24.95f);
        cart.addDigitalVideoDisc(dvd2);

        DigitalVideoDisc dvd3 = new DigitalVideoDisc("Aladin",
            "Animation", 18.99f);
        cart.addDigitalVideoDisc(dvd3);

        //Test the print method
        cart.printOrders();

        //Test the search method
        cart.searchByTitle("Aladin");
        cart.searchByTitle("Ball");

        cart.searchById(1);
        cart.searchById(10);
    }
}
```

```
<terminated> CartTest [Java Application] C:\Program Files\Eclipse Adoptium\jdk-17.0.12.7-hotspot\bin\javaw.ex
Added
Added
Added
DVD-The Lion King-Animation-Roger Allers-87: 19.95$
DVD-Star Wars-Science Fiction-George Lucas-87: 24.95$
DVD-Aladin-Animation-null-0: 18.99$
DVD found: DVD-Aladin-Animation-null-0: 18.99$
No DVD found with title: Ball
DVD found: DVD-The Lion King-Animation-Roger Allers-87: 19.95$
No DVD found with ID: 10
```

7. Implement the **Store** class



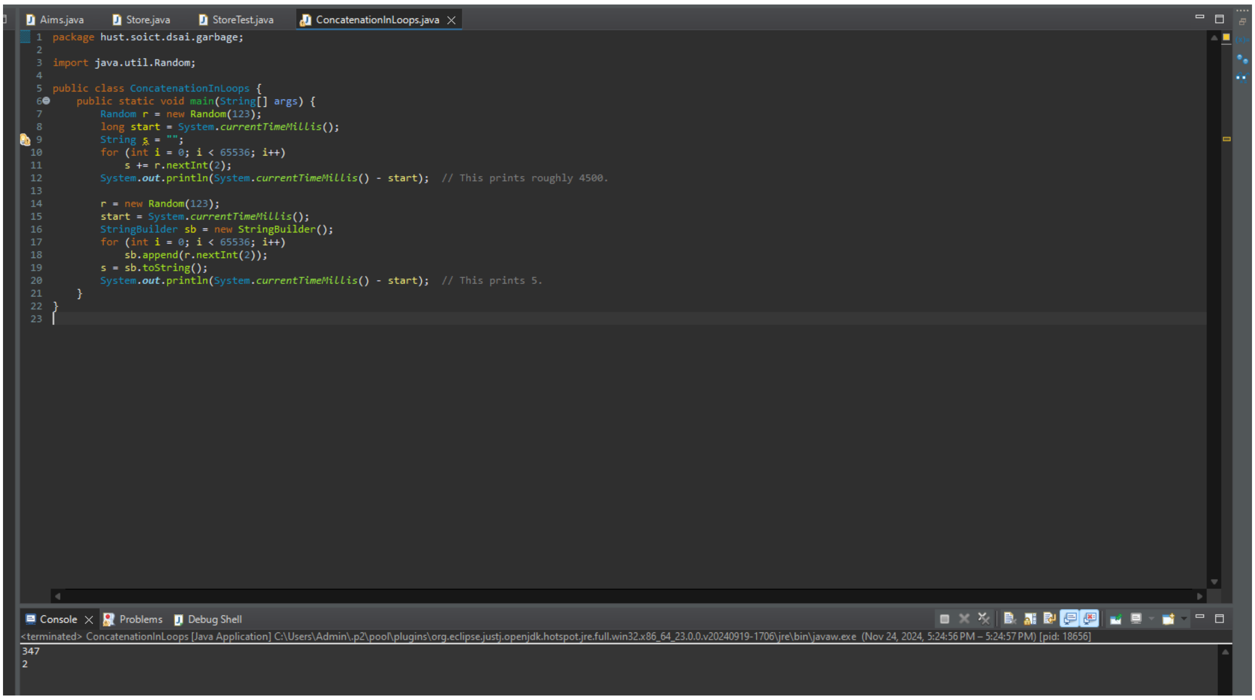
```java
package hust.soict.dsai.aims.store;

import hust.soict.dsai.aims.disc.DigitalVideoDisc;
import java.util.ArrayList;

public class Store {
    private ArrayList<DigitalVideoDisc> itemsInStore;

    public Store() {
        itemsInStore = new ArrayList<>();
    }

    public void addDvd(DigitalVideoDisc dvd) {
        if (dvd != null) {
            itemsInStore.add(dvd);
            System.out.println("DVD added: " + dvd.getTitle());
        } else {
            System.out.println("Cannot add null DVD.");
        }
    }

    public void removeDvd(DigitalVideoDisc dvd) {
        if (itemsInStore.contains(dvd)) {
            itemsInStore.remove(dvd);
            System.out.println("DVD removed: " + dvd.getTitle());
        } else {
            System.out.println("DVD not found in the store.");
        }
    }
}
```



```java
package hust.soict.dsai.test.store;

import hust.soict.dsai.aims.disc.DigitalVideoDisc;

public class StoreTest {
    public static void main(String[] args) {
        Store store = new Store();

        // Create DigitalVideoDisc instances
        DigitalVideoDisc dvd1 = new DigitalVideoDisc("Inception");
        DigitalVideoDisc dvd2 = new DigitalVideoDisc("Interstellar");
        DigitalVideoDisc dvd3 = new DigitalVideoDisc("The Matrix");

        // Test adding DVDs
        System.out.println("Testing addDVD method:");
        store.addDvd(dvd1);
        store.addDvd(dvd2);
        store.addDvd(null);

        // Test removing DVDs
        System.out.println("\nTesting removeDVD method:");
        store.removeDvd(dvd2);
        store.removeDvd(dvd3);
    }
}
```

```
<terminated> StoreTest (2) [Java Application] C:\Program Files\Eclipse Adoptium\jdk-17.0.12.7-hotspot\bin\javaw.exe  (Nov 24, 2024, 6:02:22 PM – 6:02:22 PM) [pid: 21108]
Testing addDVD method:
DVD added: Inception
DVD added: Interstellar
Cannot add null DVD.

Testing removeDVD method:
DVD removed: Interstellar
DVD not found in the store.
```

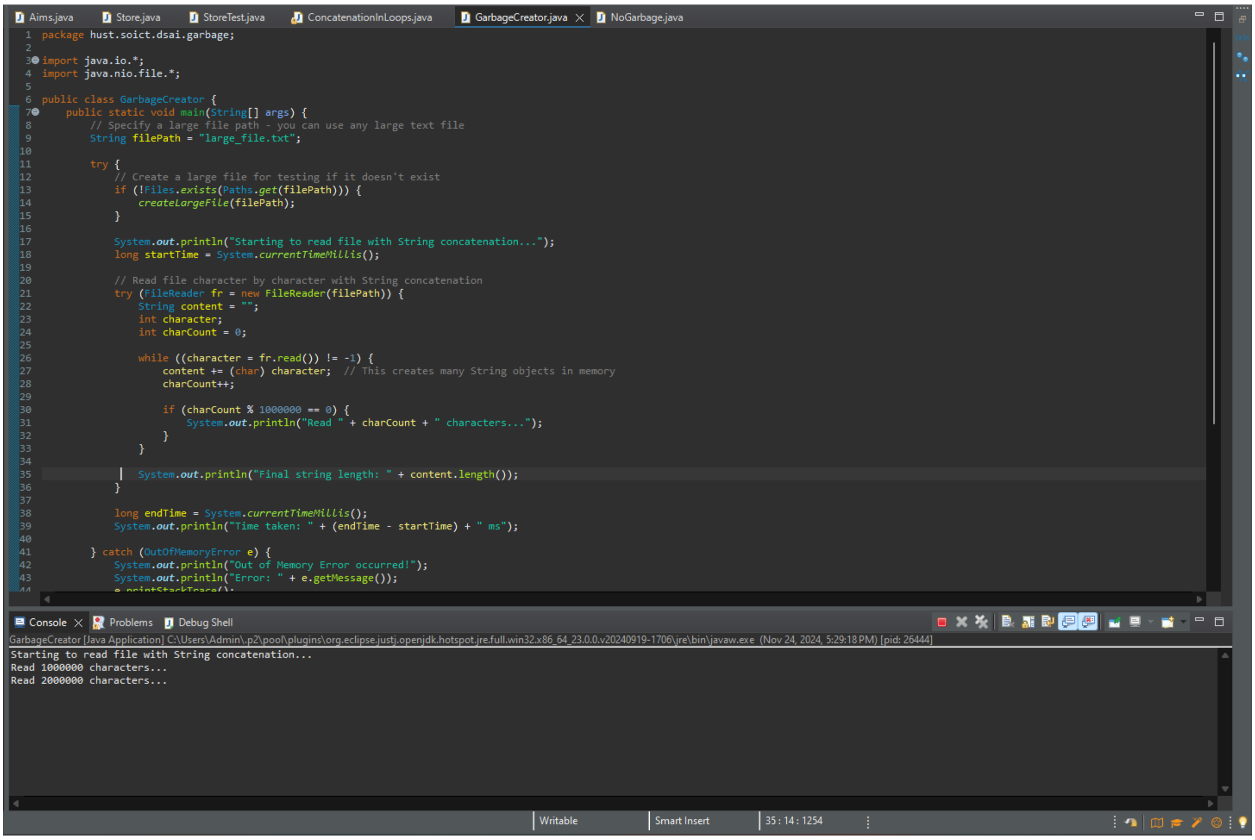
9. String, StringBuilder and StringBuffer



```java
package hust.soict.dsai.garbage;

import java.util.Random;

public class ConcatenationInLoops {
    public static void main(String[] args) {
        Random r = new Random(123);
        long start = System.currentTimeMillis();
        String s = "";
        for (int i = 0; i < 65536; i++)
            s += r.nextInt(2);
        System.out.println(System.currentTimeMillis() - start);  // This prints roughly 4500.

        r = new Random(123);
        start = System.currentTimeMillis();
        StringBuilder sb = new StringBuilder();
        for (int i = 0; i < 65536; i++)
            sb.append(r.nextInt(2));
        s = sb.toString();
        System.out.println(System.currentTimeMillis() - start);  // This prints 5.
    }
}
```

```
<terminated> ConcatenationInLoops [Java Application] C:\Users\Admin\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_23.0.0.v20240919-1706\jre\bin\javaw.exe  (Nov 24, 2024, 5:24:56 PM – 5:24:57 PM) [pid: 18656]
347
2
```

String concatenation

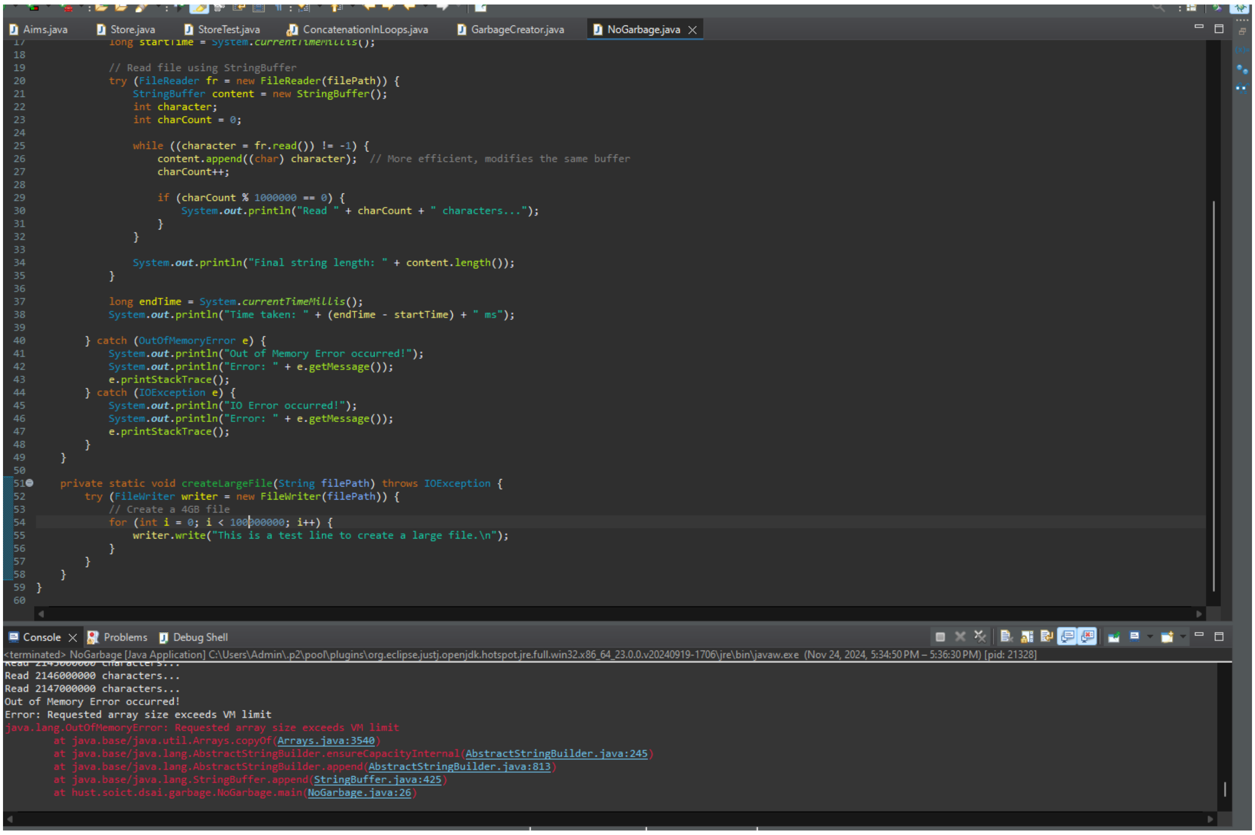


```java
package hust.soict.dsai.garbage;

import java.io.*;
import java.nio.file.*;

public class GarbageCreator {
    public static void main(String[] args) {
        // Specify a large file path - you can use any large text file
        String filePath = "large_file.txt";

        try {
            // Create a large file for testing if it doesn't exist
            if (!Files.exists(Paths.get(filePath))) {
                createLargeFile(filePath);
            }

            System.out.println("Starting to read file with String concatenation...");
            long startTime = System.currentTimeMillis();

            // Read file character by character with String concatenation
            try (FileReader fr = new FileReader(filePath)) {
                String content = "";
                int character;
                int charCount = 0;

                while ((character = fr.read()) != -1) {
                    content += (char) character;  // This creates many String objects in memory
                    charCount++;

                    if (charCount % 1000000 == 0) {
                        System.out.println("Read " + charCount + " characters...");
                    }
                }

                System.out.println("Final string length: " + content.length());
            }

            long endTime = System.currentTimeMillis();
            System.out.println("Time taken: " + (endTime - startTime) + " ms");

        } catch (OutOfMemoryError e) {
            System.out.println("Out of Memory Error occurred!");
            System.out.println("Error: " + e.getMessage());
            e.printStackTrace();
```

```
GarbageCreator [Java Application] C:\Users\Admin\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_23.0.0.v20240919-1706\jre\bin\javaw.exe  (Nov 24, 2024, 5:29:18 PM) [pid: 26444]
Starting to read file with String concatenation...
Read 1000000 characters...
Read 2000000 characters...
```

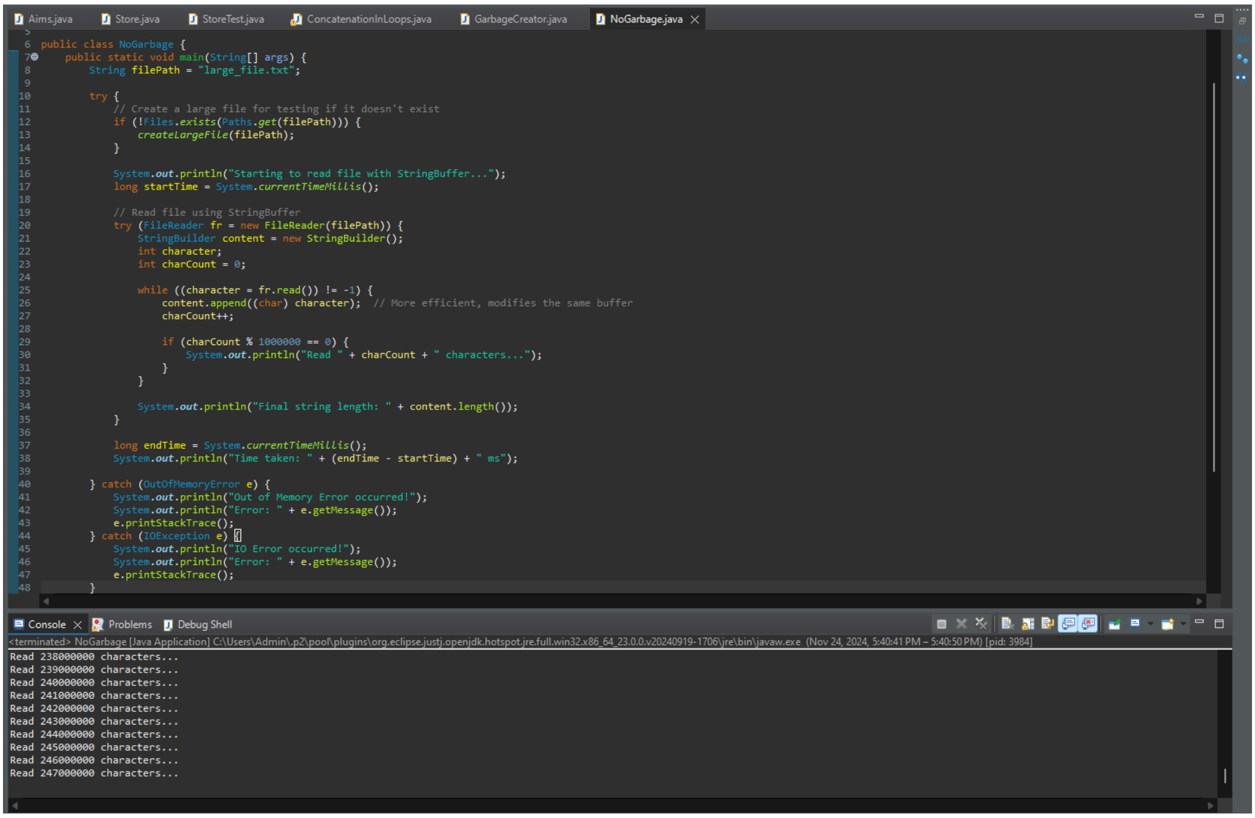
- ⇨ Very slow. Take so much time to read the file\

Using stringBuffer & string Builder:



```java
        long startTime = System.currentTimeMillis();

        // Read file using StringBuffer
        try (FileReader fr = new FileReader(filePath)) {
            StringBuffer content = new StringBuffer();
            int character;
            int charCount = 0;

            while ((character = fr.read()) != -1) {
                content.append((char) character);  // More efficient, modifies the same buffer
                charCount++;

                if (charCount % 1000000 == 0) {
                    System.out.println("Read " + charCount + " characters...");
                }
            }

            System.out.println("Final string length: " + content.length());
        }

        long endTime = System.currentTimeMillis();
        System.out.println("Time taken: " + (endTime - startTime) + " ms");

    } catch (OutOfMemoryError e) {
        System.out.println("Out of Memory Error occurred!");
        System.out.println("Error: " + e.getMessage());
        e.printStackTrace();
    } catch (IOException e) {
        System.out.println("IO Error occurred!");
        System.out.println("Error: " + e.getMessage());
        e.printStackTrace();
    }
}

private static void createLargeFile(String filePath) throws IOException {
    try (FileWriter writer = new FileWriter(filePath)) {
        // Create a 4GB file
        for (int i = 0; i < 100000000; i++) {
            writer.write("This is a test line to create a large file.\n");
        }
    }
}
}
```

```
<terminated> NoGarbage [Java Application] C:\Users\Admin\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_23.0.0.v20240919-1706\jre\bin\javaw.exe  (Nov 24, 2024, 5:34:50 PM – 5:36:30 PM) [pid: 21328]
Read 2146000000 characters...
Read 2147000000 characters...
Out of Memory Error occurred!
Error: Requested array size exceeds VM limit
java.lang.OutOfMemoryError: Requested array size exceeds VM limit
	at java.base/java.util.Arrays.copyOf(Arrays.java:3540)
	at java.base/java.lang.AbstractStringBuilder.ensureCapacityInternal(AbstractStringBuilder.java:245)
	at java.base/java.lang.AbstractStringBuilder.append(AbstractStringBuilder.java:813)
	at java.base/java.lang.StringBuffer.append(StringBuffer.java:425)
	at hust.soict.dsai.garbage.NoGarbage.main(NoGarbage.java:26)
```



```java
public class NoGarbage {
    public static void main(String[] args) {
        String filePath = "large_file.txt";

        try {
            // Create a large file for testing if it doesn't exist
            if (!Files.exists(Paths.get(filePath))) {
                createLargeFile(filePath);
            }

            System.out.println("Starting to read file with StringBuffer...");
            long startTime = System.currentTimeMillis();

            // Read file using StringBuffer
            try (FileReader fr = new FileReader(filePath)) {
                StringBuilder content = new StringBuilder();
                int character;
                int charCount = 0;

                while ((character = fr.read()) != -1) {
                    content.append((char) character);  // More efficient, modifies the same buffer
                    charCount++;

                    if (charCount % 1000000 == 0) {
                        System.out.println("Read " + charCount + " characters...");
                    }
                }

                System.out.println("Final string length: " + content.length());
            }

            long endTime = System.currentTimeMillis();
            System.out.println("Time taken: " + (endTime - startTime) + " ms");

        } catch (OutOfMemoryError e) {
            System.out.println("Out of Memory Error occurred!");
            System.out.println("Error: " + e.getMessage());
            e.printStackTrace();
        } catch (IOException e) {
            System.out.println("IO Error occurred!");
            System.out.println("Error: " + e.getMessage());
            e.printStackTrace();
        }
```

```
<terminated> NoGarbage [Java Application] C:\Users\Admin\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_23.0.0.v20240919-1706\jre\bin\javaw.exe  (Nov 24, 2024, 5:40:41 PM – 5:40:50 PM) [pid: 3984]
Read 238000000 characters...
Read 239000000 characters...
Read 240000000 characters...
Read 241000000 characters...
Read 242000000 characters...
Read 243000000 characters...
Read 244000000 characters...
Read 245000000 characters...
Read 246000000 characters...
Read 247000000 characters...
```

- Much faster and more efficient
- The only thing stringBuffer differs from stringBuilder is about thread safety. Usually I think stringBuilder is the go to unless we need to care about thread safety.

- Although it still reach the limit but it is because the file I create is too big (~ 4GB)

⇨ Definitely better than string concatenation.