Lab 03 report

Le Viet Cuong- 20235586

2. Working with method overloading.

```
//add a list of disc
public void addDigitalVideodisc(DigitalVideoDisc[] dvdList){
    for(DigitalVideoDisc disc: dvdList){
        if(qtyOrdered<MAX_NUMBER_ORDERED){
            itemsOrdered[qtyOrdered]= disc;
            System.out.printf("The disc: \"%s\" has been added. ", itemsOrdered[qtyOrdered].getTitle());
            System.out.printf("Cost: %.2f\n", itemsOrdered[qtyOrdered].getCost());
            qtyOrdered+=1;
        }
        else{
            System.out.println("The cart is almost full, please remove some disc.");
            break;
        }
    }
}

//add two disc
public void addDigitalVideodisc(DigitalVideoDisc dvd1,DigitalVideoDisc dvd2){
    //add dvd1;
    if(qtyOrdered<MAX_NUMBER_ORDERED){
        itemsOrdered[qtyOrdered]= dvd1;
        System.out.printf("The disc: \"%s\" has been added. ", itemsOrdered[qtyOrdered].getTitle());
        System.out.printf("Cost: %.2f\n", itemsOrdered[qtyOrdered].getCost());
        qtyOrdered+=1;
    }
    else{
        System.out.println("The cart is almost full, please remove some disc.");
    }

    //add dvd2;
    if(qtyOrdered<MAX_NUMBER_ORDERED){
        itemsOrdered[qtyOrdered]= dvd2;
        System.out.printf("The disc: \"%s\" has been added. ", itemsOrdered[qtyOrdered].getTitle());
        System.out.printf("Cost: %.2f\n", itemsOrdered[qtyOrdered].getCost());
        qtyOrdered+=1;
    }
    else{
        System.out.println("The cart is almost full, please remove some disc.");
    }
}
//add an arbitrary number of DVDs;
public void addDigitalVideodisc(int numberOfdvds, DigitalVideoDisc... dvdList){
    for(DigitalVideoDisc disc: dvdList){
        if(qtyOrdered<MAX_NUMBER_ORDERED){
            itemsOrdered[qtyOrdered]= disc;
            System.out.printf("The disc: \"%s\" has been added. ", itemsOrdered[qtyOrdered].getTitle());
            System.out.printf("Cost: %.2f\n", itemsOrdered[qtyOrdered].getCost());
            qtyOrdered+=1;
        }
        else{
            System.out.println("The cart is almost full, please remove some disc.");
            break;
        }
    }
}
```

Question: 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?

Answer: From my perspective, using the method which allows to pass an arbitrary number of arguments is better. When passing the array of arguments, we need to modify the size of that array, but for the second choice, we don’t need to do that, we can add as many as we want.

3. Passing parameter.

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

        //we try to swap name;
        swap(jungleDVD, cinderellaDVD);

        //the original title doesn't change;
        System.out.println("jungle dvd title: "+ jungleDVD.getTitle());
        System.out.println("cinderella dvd title: "+ cinderellaDVD.getTitle());

        //but this change the title; Why?
        changeTitle(jungleDVD, cinderellaDVD.getTitle());
        System.out.println("jungle dvd title: "+ jungleDVD.getTitle());
    }

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

    /*correct method to swap;
    public static void swap(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) {
        String temp= dvd1.getTitle();
        dvd1.setTitle(dvd2.getTitle());
        dvd2.setTitle(temp);
    }*/

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

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

First, we recall what is meant **pass by value** or **pass by reference**.

- **Pass by value**: The method parameter values are **copied** to another variable and then the copied object is passed to the method. That's why it's called pass by value.
- **Pass by reference**: An alias or reference to the actual parameter is passed to the method. That's why it's called pass by reference.

Answer: Java is a Pass by Value programming language.

Question 2:

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

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

Answer:

- Java is a Pass by Value programming language, which means that the actual parameter will be copied to a new parameter, and that parameter will be passed to the method. Hence, the method **swap** will only change the copied one and preserve the actual one.
- In the **changeTitle** method, the method **setTitle** of the object DVD is used, this method will change the private parameter of the class to a new value. This is why the title of the J**ungleDVD** changed.

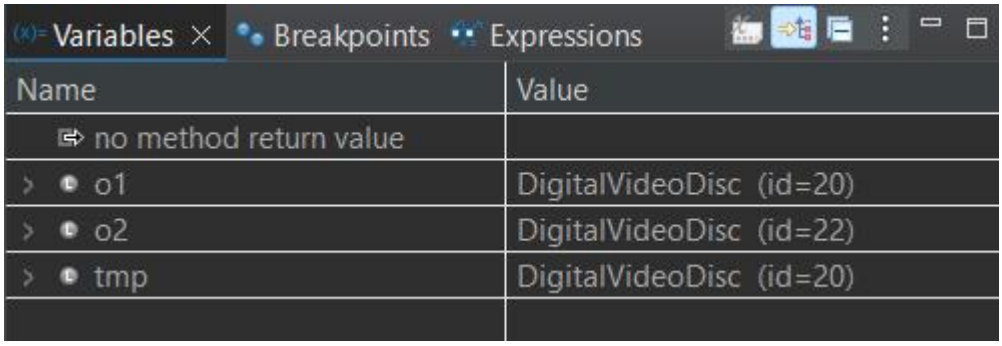
We apply this setter technique to write the true **swap** method:

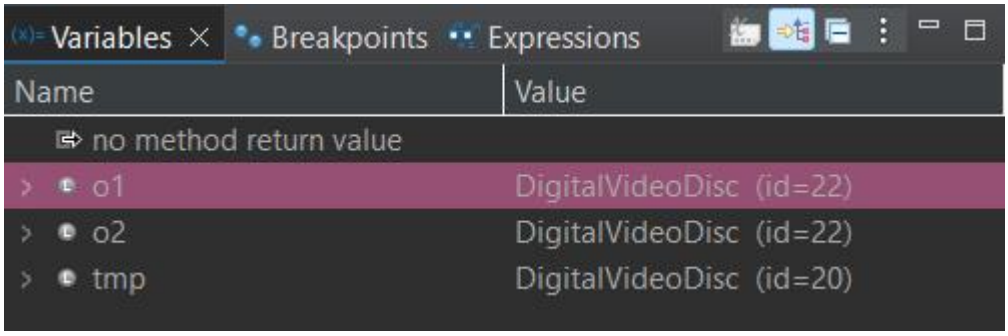
```
//correct method to swap;
public static void swap(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) {
    String temp= dvd1.getTitle();
    dvd1.setTitle(dvd2.getTitle());
    dvd2.setTitle(temp);
}
```

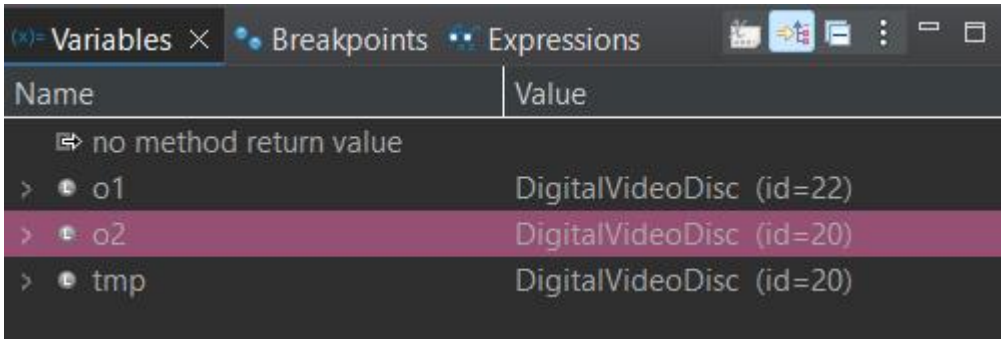
Result:

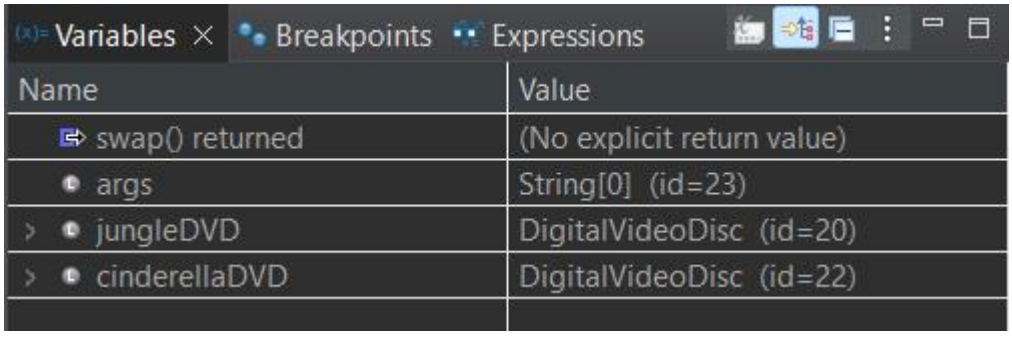
```
jungle dvd title: Cinderella
cinderella dvd title: Jungle
```

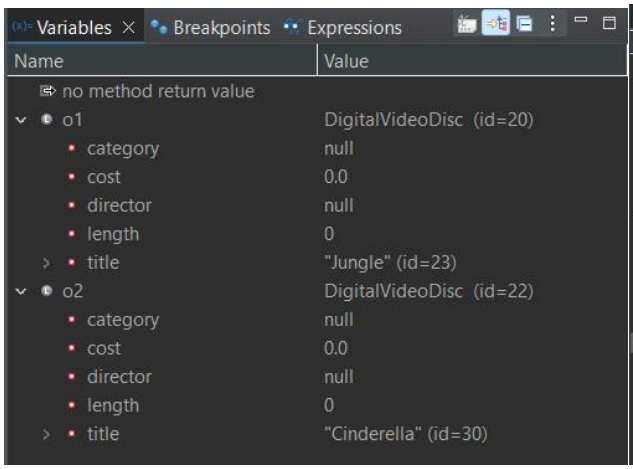
4. Use debug run

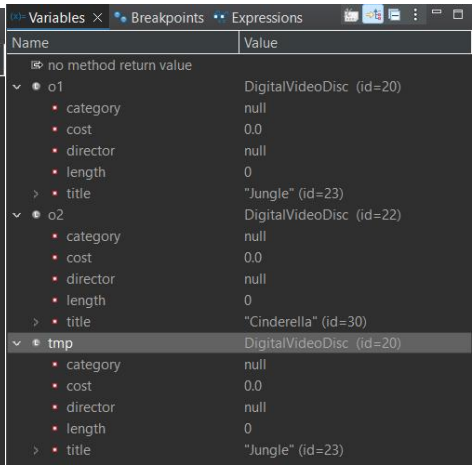


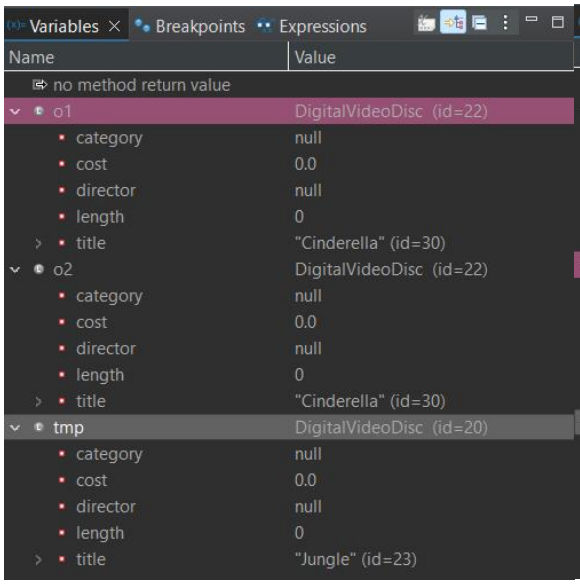


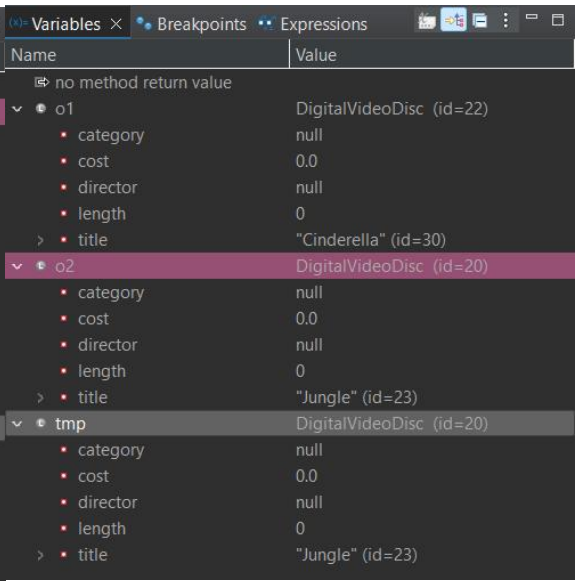


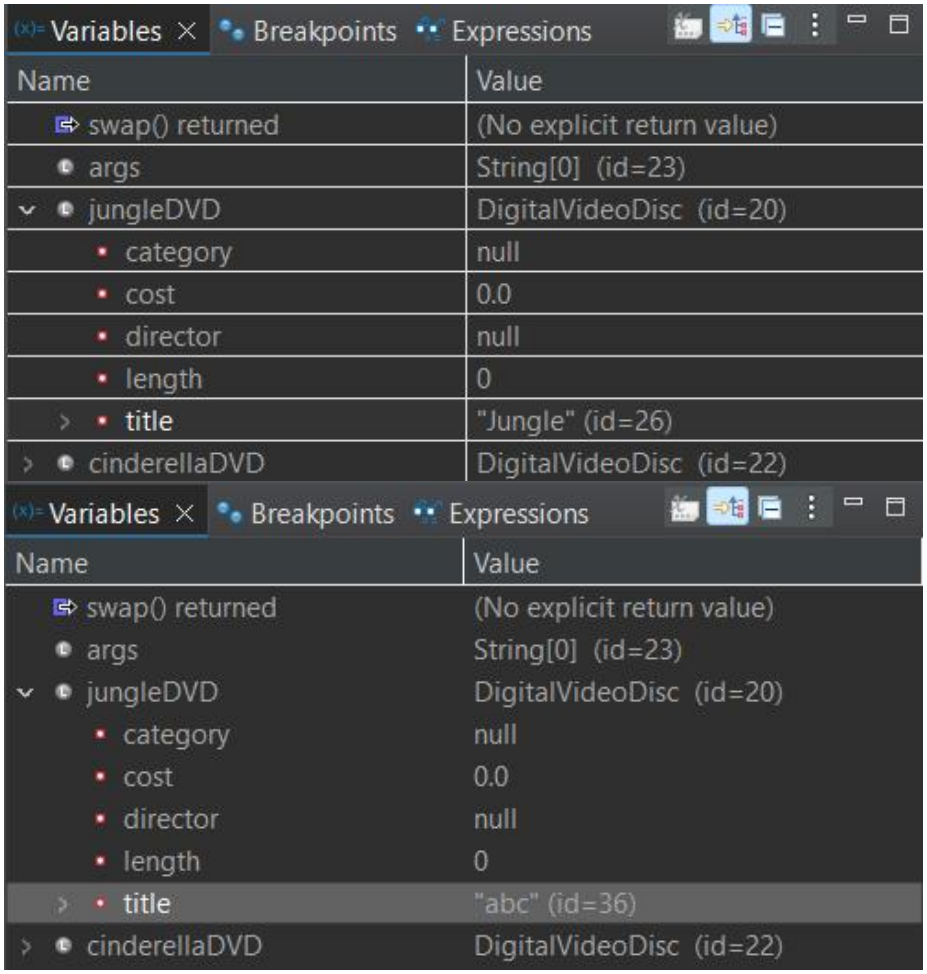



Variables
Breakpoints
Expressions
Name
Value
no method return value
o1
DigitalVideoDisc (id=20)
category
null
cost
0.0
director
null
length
0
title
"Jungle" (id=23)
o2
DigitalVideoDisc (id=22)
category
null
cost
0.0
director
null
length
0
title
"Cinderella" (id=30)


Variables
Breakpoints
Expressions
Name
Value
no method return value
o1
DigitalVideoDisc (id=20)
category
null
cost
0.0
director
null
length
0
title
"Jungle" (id=23)
o2
DigitalVideoDisc (id=22)
category
null
cost
0.0
director
null
length
0
title
"Cinderella" (id=30)
tmp
DigitalVideoDisc (id=20)
category
null
cost
0.0
director
null
length
0
title
"Jungle" (id=23)


Variables
Breakpoints
Expressions
Name
Value
no method return value
o1
DigitalVideoDisc (id=22)
category
null
cost
0.0
director
null
length
0
title
"Cinderella" (id=30)
o2
DigitalVideoDisc (id=22)
category
null
cost
0.0
director
null
length
0
title
"Cinderella" (id=30)
tmp
DigitalVideoDisc (id=20)
category
null
cost
0.0
director
null
length
0
title
"Jungle" (id=23)


Variables
Breakpoints
Expressions
Name
Value
no method return value
o1
DigitalVideoDisc (id=22)
category
null
cost
0.0
director
null
length
0
title
"Cinderella" (id=30)
o2
DigitalVideoDisc (id=20)
category
null
cost
0.0
director
null
length
0
title
"Jungle" (id=23)
tmp
DigitalVideoDisc (id=20)
category
null
cost
0.0
director
null
length
0
title
"Jungle" (id=23)




```
jungle dvd title: abc
cinderella dvd title: Cinderella
```

---

5. DigitalVideoDisc

```java
public class DigitalVideoDisc {
    private String title;
    private String category;
    private String director;
    private int id;
    private int length;
    private float cost;
    private static int nbDigitalVideoDisc= 0;

    //overloading method
    public DigitalVideoDisc(String title) {
        super();
        this.title = title;
        nbDigitalVideoDisc+=1;
        this.id = nbDigitalVideoDisc;
    }

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

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

```java
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;
    nbDigitalVideoDisc+=1;
    this.id = nbDigitalVideoDisc;
}
public int getId() {
    return id;
}
public void setId(int id) {
    this.id= id;
}
```

```java
//return String
public String toString() {
    return "DVD - [" + this.getTitle() + "] - [" + this.getCategory()
            + "] - [" + this.getDirector() + "] - [" + this.getLength() + "] : ["
            + this.getCost() + "]$";
}

public boolean isMatch(String title) {
    String discTitle= this.title.toLowerCase();
    if(discTitle.indexOf(title.toLowerCase())!=-1) {
        return true;
    }
    return false;
}
```

6. Cart Class

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

Answer: This method should return the String itself.

```
//return String
public String toString() {
    return "DVD - [" + this.getTitle() + "] - [" + this.getCategory()
            + "] - [" + this.getDirector() + "] - [" + this.getLength() + "] : ["
            + this.getCost() + "]$";
}
```

When a customer searches for DVDs by title, he or she provides a string of keywords. If any DVD has the title containing any word in the string of keywords, it is counted as a match. Note that the comparison of words here is case-insensitive.

When a customer searches for DVDs by category, he or she provides the category name. If any DVD has the matching category (case-insensitive), it is counted as a match.

When a customer searches for DVDs by price, he or she provides either the minimum and maximum cost, or just the maximum cost.

```
public void print() {
    System.out.println("**********CART********** ");
    System.out.println("Ordered Items:");
    for(int i=0; i<qtyOrdered; i++) {
        System.out.println(itemsOrdered[i].toString());
    }
    System.out.printf("Total cost: [%.2f]\n", totalCost());
    System.out.println("********************************************** ");
}

public void Find(int id) {
    boolean found= false;
    for(int i=0; i<qtyOrdered; i++) {
        if(itemsOrdered[i].getId()==id) {
            System.out.println(itemsOrdered[i].toString());
            found= true;
        }
    }
    if(!found) {
        System.out.println("No match is found.");
    }
}

public void Find(String title) {
    boolean found= false;
    for(int i=0; i< qtyOrdered; i++) {
        if(itemsOrdered[i].isMatch(title)) {
            System.out.println(itemsOrdered[i].toString());
            found= true;
        }
    }
    if(!found) {
        System.out.println("No match is found.");
    }
}
```

7. Store class

```java
public class Store {
    DigitalVideoDisc[] itemsInStore= new DigitalVideoDisc[200];
    int numberOfDiscs= 0;

    public void addDVD(DigitalVideoDisc disc) {
        itemsInStore[numberOfDiscs]= disc;
        numberOfDiscs+=1;
    }

    public void removeDVD(DigitalVideoDisc disc) {
        for(int i=0; i<numberOfDiscs ; i++){
            if(itemsInStore[i].equals(disc)){
                for(int j=i; j<numberOfDiscs-1; j++){
                    itemsInStore[j]= itemsInStore[j+1];
                }
                itemsInStore[numberOfDiscs-1]= null;
                numberOfDiscs-=1;
                break;
            }
        }
    }

    public void print() {
        for(int i=0; i<numberOfDiscs; i++) {
            System.out.println(itemsInStore[i].toString());
        }
    }
}
```