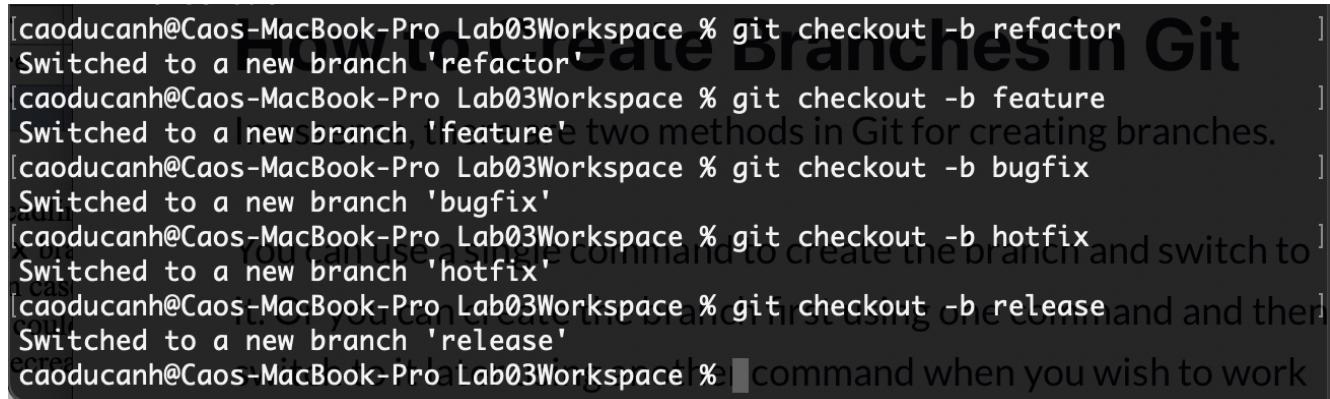


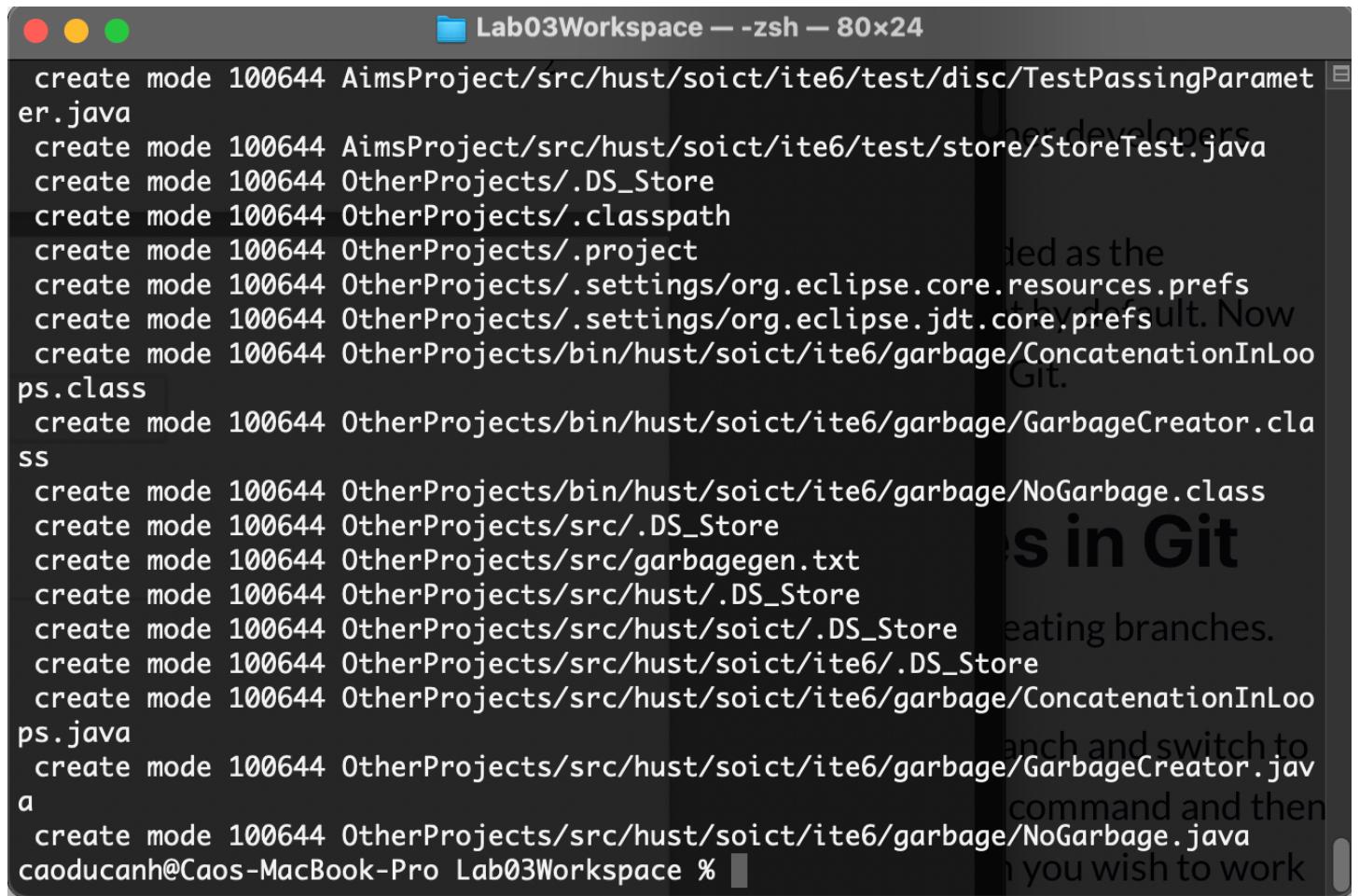
Lab03: Basic Object-Oriented Techniques

Section 1: Branch your repository



```
[caoducanh@Caos-MacBook-Pro Lab03Workspace % git checkout -b refactor
Switched to a new branch 'refactor'
[caoducanh@Caos-MacBook-Pro Lab03Workspace % git checkout -b feature
Switched to a new branch 'feature'
[caoducanh@Caos-MacBook-Pro Lab03Workspace % git checkout -b bugfix
Switched to a new branch 'bugfix'
[caoducanh@Caos-MacBook-Pro Lab03Workspace % git checkout -b hotfix
Switched to a new branch 'hotfix'
[caoducanh@Caos-MacBook-Pro Lab03Workspace % git checkout -b release
Switched to a new branch 'release'
caoducanh@Caos-MacBook-Pro Lab03Workspace %
```

Figure 1. Creating common branches to local repo

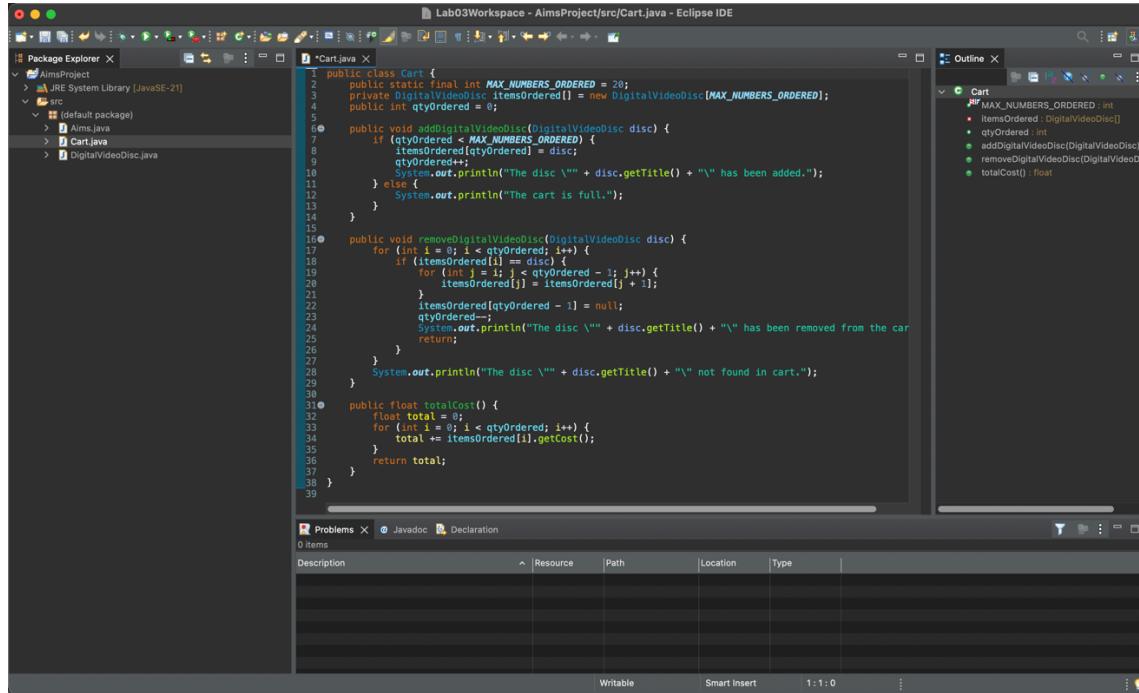


```
create mode 100644 AimsProject/src/hust/soict/ite6/test/disc/TestPassingParamet
er.java
create mode 100644 AimsProject/src/hust/soict/ite6/test/store/StoreTest.java
create mode 100644 OtherProjects/.DS_Store
create mode 100644 OtherProjects/.classpath
create mode 100644 OtherProjects/.project
create mode 100644 OtherProjects/.settings/org.eclipse.core.resources.prefs
create mode 100644 OtherProjects/.settings/org.eclipse.jdt.core.prefs
create mode 100644 OtherProjects/bin/hust/soict/ite6/garbage/ConcatenationInLoo
ps.class
create mode 100644 OtherProjects/bin/hust/soict/ite6/garbage/GarbageCreator.cl
ss
create mode 100644 OtherProjects/bin/hust/soict/ite6/garbage/NoGarbage.class
create mode 100644 OtherProjects/src/.DS_Store
create mode 100644 OtherProjects/src/garbagegen.txt
create mode 100644 OtherProjects/src/hust/.DS_Store
create mode 100644 OtherProjects/src/hust/soict/.DS_Store
create mode 100644 OtherProjects/src/hust/soict/ite6/.DS_Store
create mode 100644 OtherProjects/src/hust/soict/ite6/garbage/ConcatenationInLoo
ps.java
create mode 100644 OtherProjects/src/hust/soict/ite6/garbage/GarbageCreator.jav
a
create mode 100644 OtherProjects/src/hust/soict/ite6/garbage/NoGarbage.java
caoducanh@Caos-MacBook-Pro Lab03Workspace %
```

Figure 2. Initial commit to the 'master' branch on local repo

Section 2: Working with method overloading

2.1 Overloading by differing types of parameter

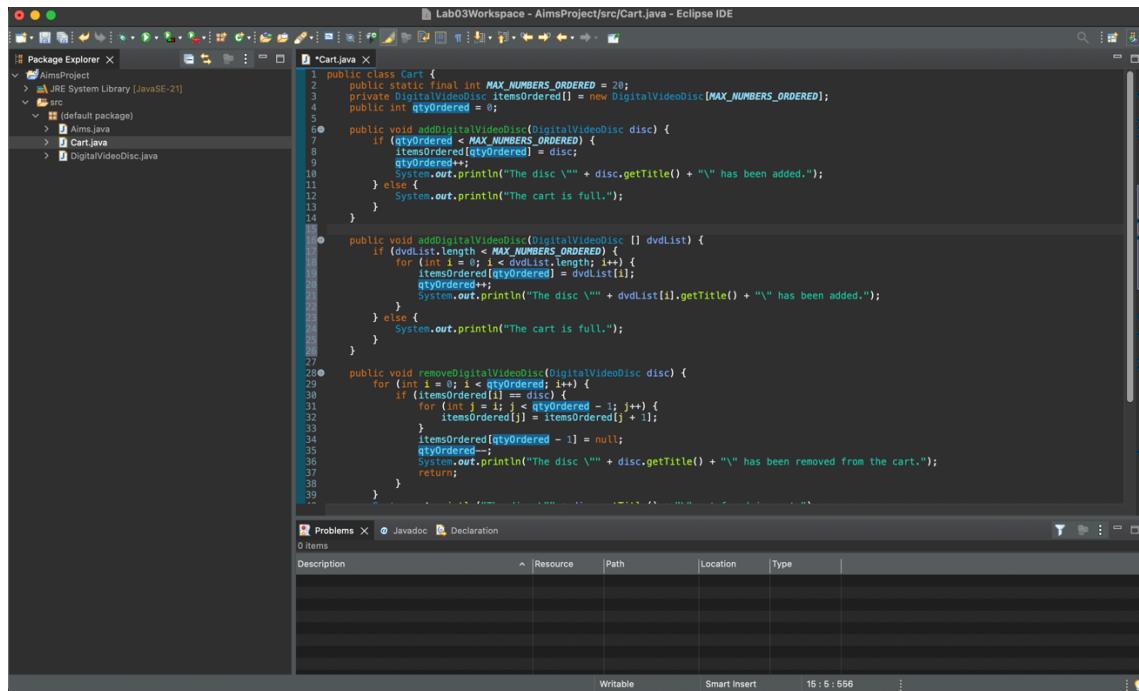


```

1  public class Cart {
2      public static final int MAX_NUMBERS_ORDERED = 20;
3      private DigitalVideoDisc[] itemsOrdered = new DigitalVideoDisc[MAX_NUMBERS_ORDERED];
4      public int qtyOrdered = 0;
5
6      public void addDigitalVideoDisc(DigitalVideoDisc disc) {
7          if (qtyOrdered < MAX_NUMBERS_ORDERED) {
8              itemsOrdered[qtyOrdered] = disc;
9              qtyOrdered++;
10             System.out.println("The disc '" + disc.getTitle() + "' has been added.");
11         } else {
12             System.out.println("The cart is full.");
13         }
14     }
15
16     public void removeDigitalVideoDisc(DigitalVideoDisc disc) {
17         for (int i = 0; i < qtyOrdered; i++) {
18             if (itemsOrdered[i] == disc) {
19                 for (int j = i; j < qtyOrdered - 1; j++) {
20                     itemsOrdered[j] = itemsOrdered[j + 1];
21                 }
22                 itemsOrdered[qtyOrdered - 1] = null;
23                 qtyOrdered--;
24                 System.out.println("The disc '" + disc.getTitle() + "' has been removed from the cart.");
25                 return;
26             }
27         }
28         System.out.println("The disc '" + disc.getTitle() + "' not found in cart.");
29     }
30
31     public float totalCost() {
32         float total = 0;
33         for (int i = 0; i < qtyOrdered; i++) {
34             total += itemsOrdered[i].getCost();
35         }
36         return total;
37     }
38 }
39

```

Figure 3. 'Cart' class from the previous lab



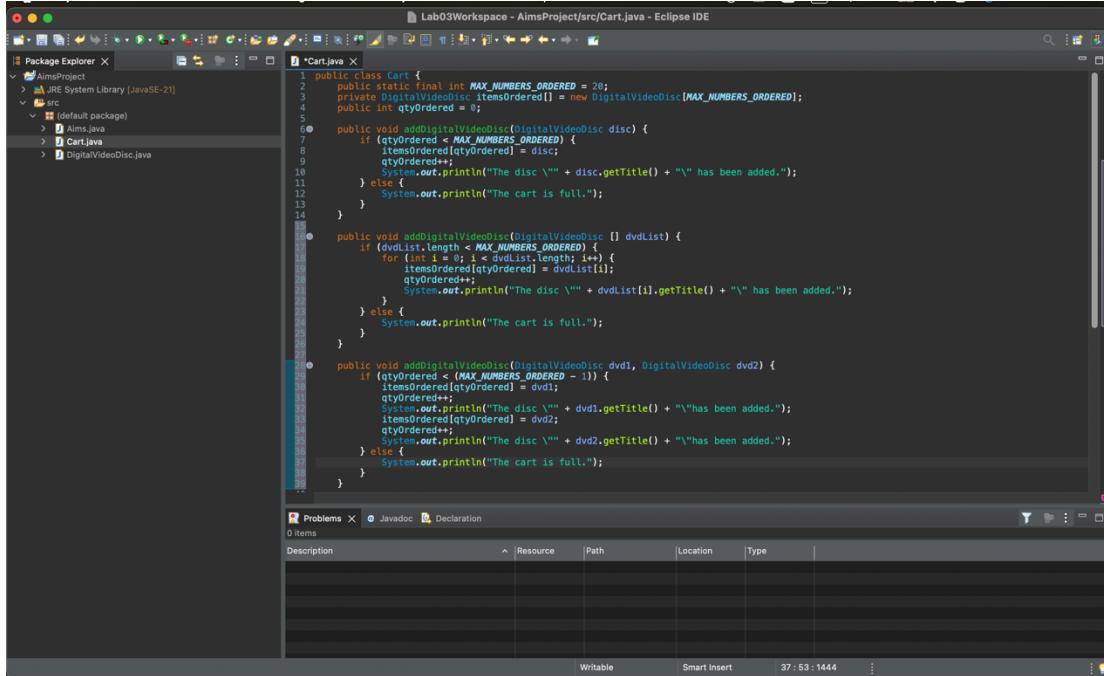
```

1  public class Cart {
2      public static final int MAX_NUMBERS_ORDERED = 20;
3      private DigitalVideoDisc[] itemsOrdered = new DigitalVideoDisc[MAX_NUMBERS_ORDERED];
4      public int qtyOrdered = 0;
5
6      public void addDigitalVideoDisc(DigitalVideoDisc disc) {
7          if (qtyOrdered < MAX_NUMBERS_ORDERED) {
8              itemsOrdered[qtyOrdered] = disc;
9              qtyOrdered++;
10             System.out.println("The disc '" + disc.getTitle() + "' has been added.");
11         } else {
12             System.out.println("The cart is full.");
13         }
14     }
15
16     public void addDigitalVideoDisc(DigitalVideoDisc[] dvdList) {
17         if (dvdList.length < MAX_NUMBERS_ORDERED) {
18             for (int i = 0; i < dvdList.length; i++) {
19                 addDigitalVideoDisc(dvdList[i]);
20                 qtyOrdered++;
21                 System.out.println("The disc '" + dvdList[i].getTitle() + "' has been added.");
22             }
23         } else {
24             System.out.println("The cart is full.");
25         }
26     }
27
28     public void removeDigitalVideoDisc(DigitalVideoDisc disc) {
29         for (int i = 0; i < qtyOrdered; i++) {
30             if (itemsOrdered[i] == disc) {
31                 for (int j = i; j < qtyOrdered - 1; j++) {
32                     itemsOrdered[j] = itemsOrdered[j + 1];
33                 }
34                 itemsOrdered[qtyOrdered - 1] = null;
35                 qtyOrdered--;
36                 System.out.println("The disc '" + disc.getTitle() + "' has been removed from the cart.");
37                 return;
38             }
39         }
40     }
41
42 }
43

```

Figure 4. Overload method 'addDigitalVideoDisc' to add a list of DVDs to cart

2.2. Overloading by differing the number of parameters



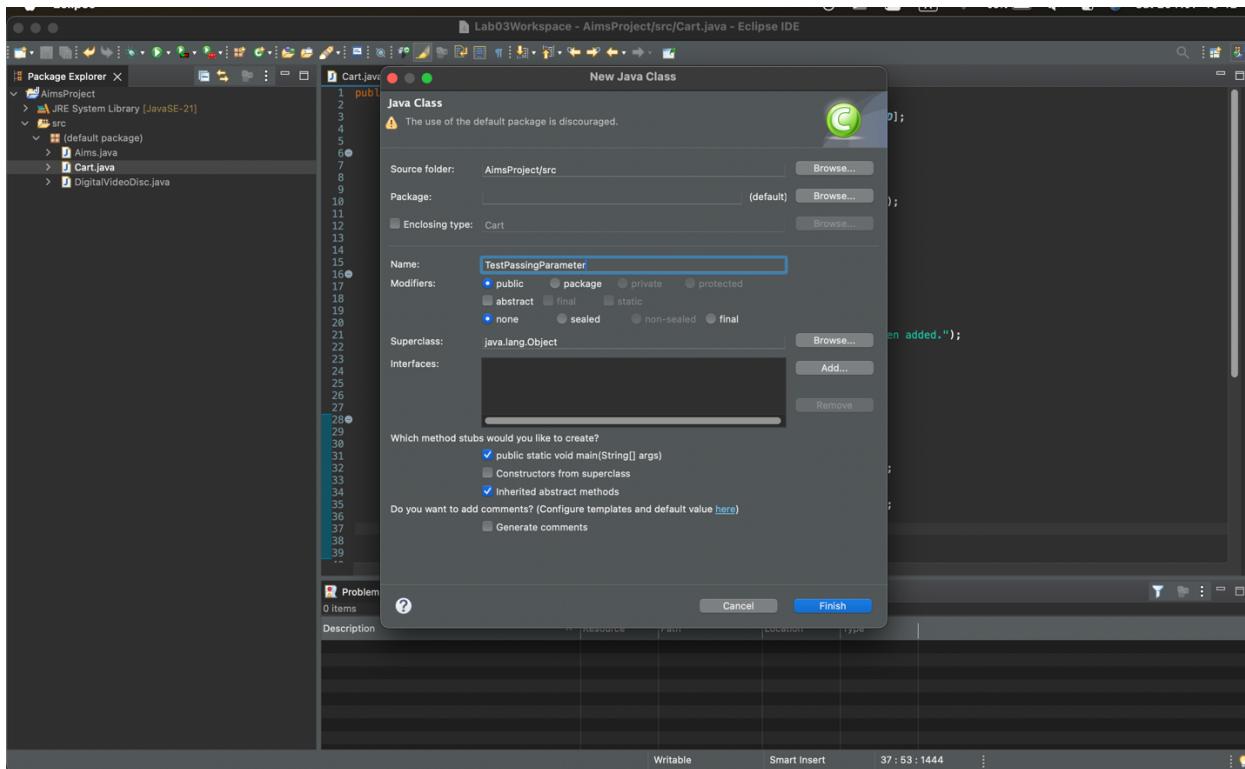
```

1  public class Cart {
2      public static final int MAX_NUMBERS_ORDERED = 20;
3      private DigitalVideoDisc itemsOrdered[] = new DigitalVideoDisc[MAX_NUMBERS_ORDERED];
4      public int qtyOrdered = 0;
5
6      public void addDigitalVideoDisc(DigitalVideoDisc disc) {
7          if (qtyOrdered < MAX_NUMBERS_ORDERED) {
8              itemsOrdered[qtyOrdered] = disc;
9              qtyOrdered++;
10             System.out.println("The disc '" + disc.getTitle() + "' has been added.");
11         } else {
12             System.out.println("The cart is full.");
13         }
14     }
15
16     public void addDigitalVideoDiscs(DigitalVideoDisc [] dvdList) {
17         if (qtyOrdered < MAX_NUMBERS_ORDERED) {
18             for (int i = 0; i < dvdList.length; i++) {
19                 itemsOrdered[qtyOrdered] = dvdList[i];
20                 qtyOrdered++;
21                 System.out.println("The disc '" + dvdList[i].getTitle() + "' has been added.");
22             }
23         } else {
24             System.out.println("The cart is full.");
25         }
26     }
27
28     public void addDigitalVideoDiscs(DigitalVideoDisc dvd1, DigitalVideoDisc dvd2) {
29         if (qtyOrdered < (MAX_NUMBERS_ORDERED - 1)) {
30             itemsOrdered[qtyOrdered] = dvd1;
31             qtyOrdered++;
32             System.out.println("The disc '" + dvd1.getTitle() + "' has been added.");
33             itemsOrdered[qtyOrdered] = dvd2;
34             qtyOrdered++;
35             System.out.println("The disc '" + dvd2.getTitle() + "' has been added.");
36         } else {
37             System.out.println("The cart is full.");
38         }
39     }
40 }

```

Figure 5. Another 'addDigitalVideoDisc' method to add 2 discs simultaneously to cart

Section 3: Passing parameter



New Java Class

Source folder: AimsProject/src

Package: (default)

Enclosing type: Cart

Name: **TestPassingParameter**

Modifiers: public package private protected
 abstract final static
 none sealed non-sealed final

Superclass: java.lang.Object

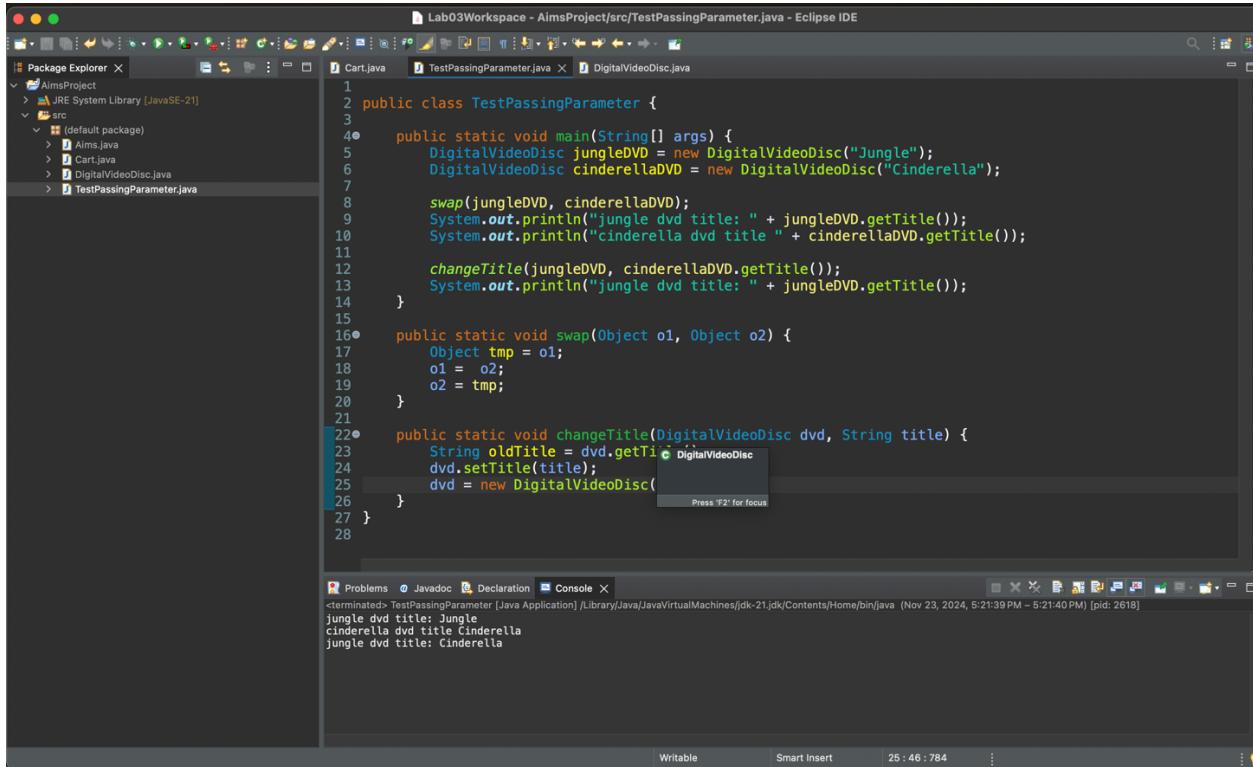
Interfaces:

Which method stubs would you like to create?
 public static void main(String[] args)
 Constructors from superclass
 Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))
 Generate comments

Cancel **Finish**

Figure 6. Create new class named 'TestPassingParameter' to test how Java passes parameters



The screenshot shows the Eclipse IDE interface with the 'TestPassingParameter' class open in the editor. The code demonstrates passing objects by value. The 'swap' method takes two Object references and swaps them. The 'changeTitle' method takes a DigitalVideoDisc object and a String title, and changes the title of the object. The run results in the console show that the titles of the two DVD objects remain unchanged after the swap, and the changeTitle method correctly updates the 'jungleDVD' object's title.

```

1  public class TestPassingParameter {
2
3●   public static void main(String[] args) {
4     DigitalVideoDisc jungleDVD = new DigitalVideoDisc("Jungle");
5     DigitalVideoDisc cinderellaDVD = new DigitalVideoDisc("Cinderella");
6
7     swap(jungleDVD, cinderellaDVD);
8     System.out.println("jungle dvd title: " + jungleDVD.getTitle());
9     System.out.println("cinderella dvd title " + cinderellaDVD.getTitle());
10
11    changeTitle(jungleDVD, cinderellaDVD.getTitle());
12    System.out.println("jungle dvd title: " + jungleDVD.getTitle());
13  }
14
15  public static void swap(Object o1, Object o2) {
16    Object tmp = o1;
17    o1 = o2;
18    o2 = tmp;
19  }
20
21  public static void changeTitle(DigitalVideoDisc dvd, String title) {
22    String oldTitle = dvd.getTitle();
23    dvd.setTitle(title);
24    dvd = new DigitalVideoDisc();
25  }
26
27 }
28

```

Console output:

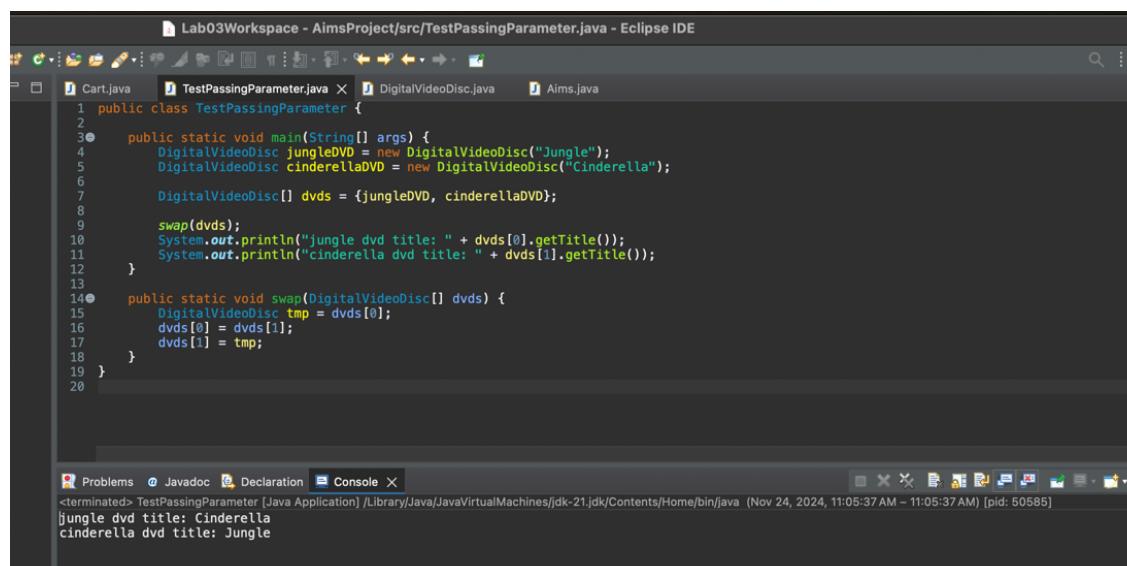
```

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

```

Figure 7. 'TestPassingParameter' class contents and run results

- The title of the two Objects remain unchanged after calling 'swap(jungleDVD, cinderellaDVD)' because Java is a pass by value language, including Object references. When passing the 2 DVDs Objects to the 'swap' method, the method receives copies of their references, thus only modifying these references, not the originals.
- When the 'changeTitle' method is called, a reference to the 'jungleDVD' object is passed to the method by value. This reference copy still points to the same object in memory. Thus, when 'dvd.setTitle(title)' is called, it directly modifies the title field of the 'jungleDVD' object.
- To rewrite 'swap' to make it works, we can use an array to hold the two objects then swap them:



The screenshot shows the Eclipse IDE interface with the rewritten 'TestPassingParameter' class open in the editor. The code uses an array to hold the two DVD objects and swap them by swapping their indices in the array. The run results in the console show that the titles of the two DVD objects are correctly swapped.

```

1  public class TestPassingParameter {
2
3●   public static void main(String[] args) {
4     DigitalVideoDisc jungleDVD = new DigitalVideoDisc("Jungle");
5     DigitalVideoDisc cinderellaDVD = new DigitalVideoDisc("Cinderella");
6
7     DigitalVideoDisc[] dvds = {jungleDVD, cinderellaDVD};
8
9     swap(dvds);
10    System.out.println("jungle dvd title: " + dvds[0].getTitle());
11    System.out.println("cinderella dvd title: " + dvds[1].getTitle());
12  }
13
14  public static void swap(DigitalVideoDisc[] dvds) {
15    DigitalVideoDisc tmp = dvds[0];
16    dvds[0] = dvds[1];
17    dvds[1] = tmp;
18  }
19
20

```

Console output:

```

jungle dvd title: Cinderella
cinderella dvd title: Jungle

```

Figure 8. Rewritten 'swap' method to swap titles of 2 DVDs

Section 4: Use debug run

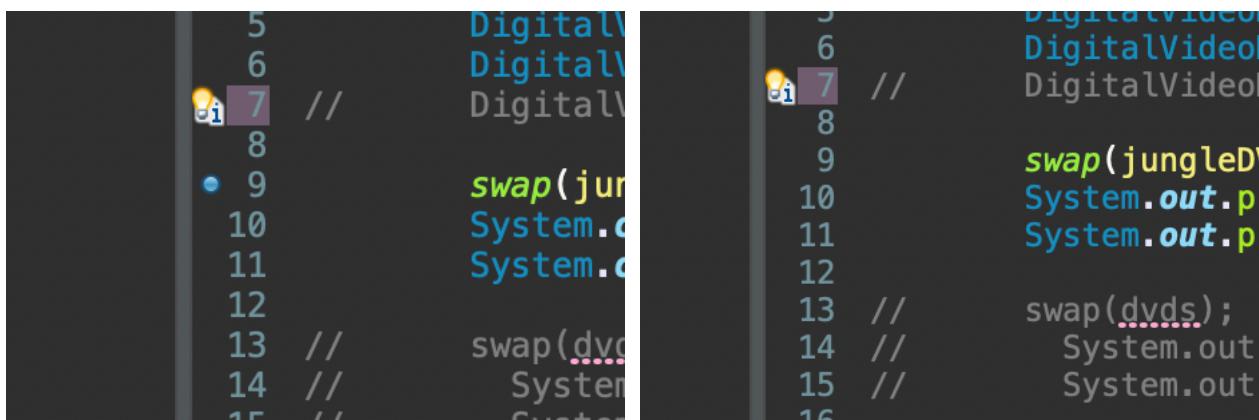


Figure 9. Toggling a breakpoint

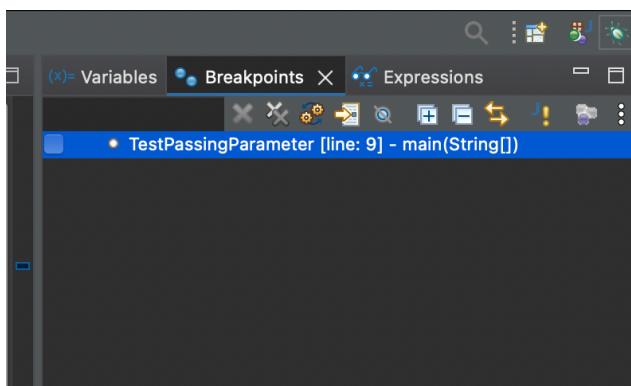


Figure 10. Enabling/Disabling breakpoints in the 'Breakpoints' window

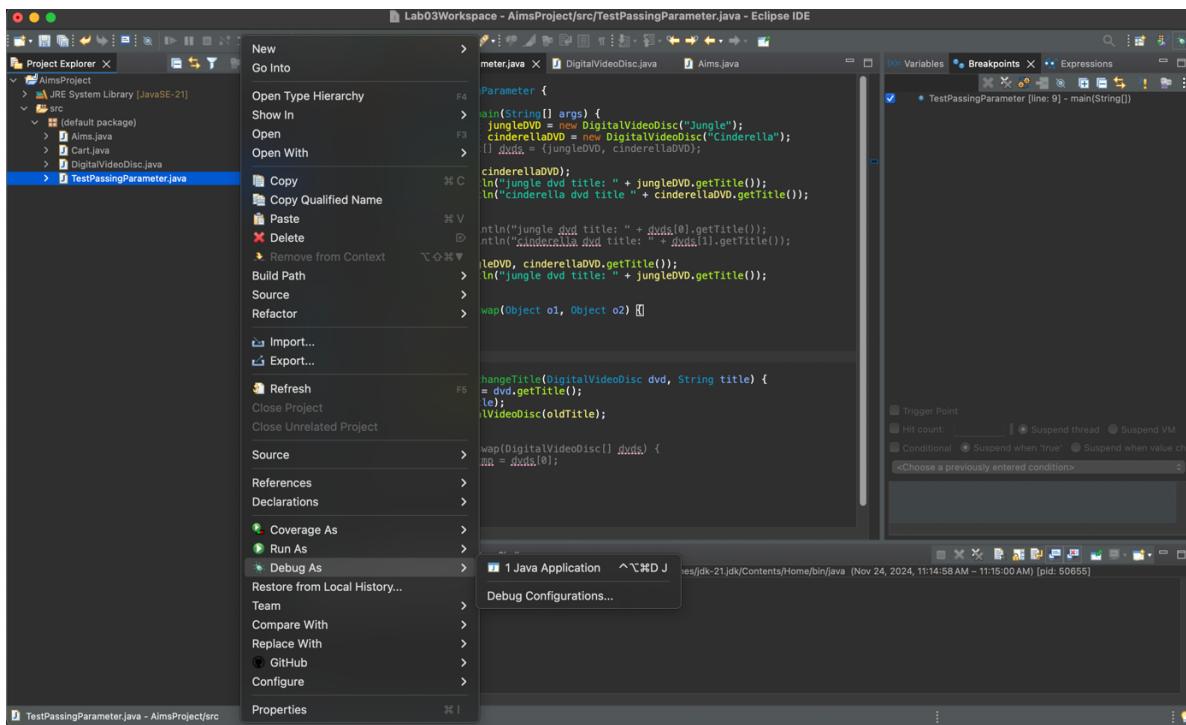


Figure 11. Steps to debug the class 'TestPassingParameter'

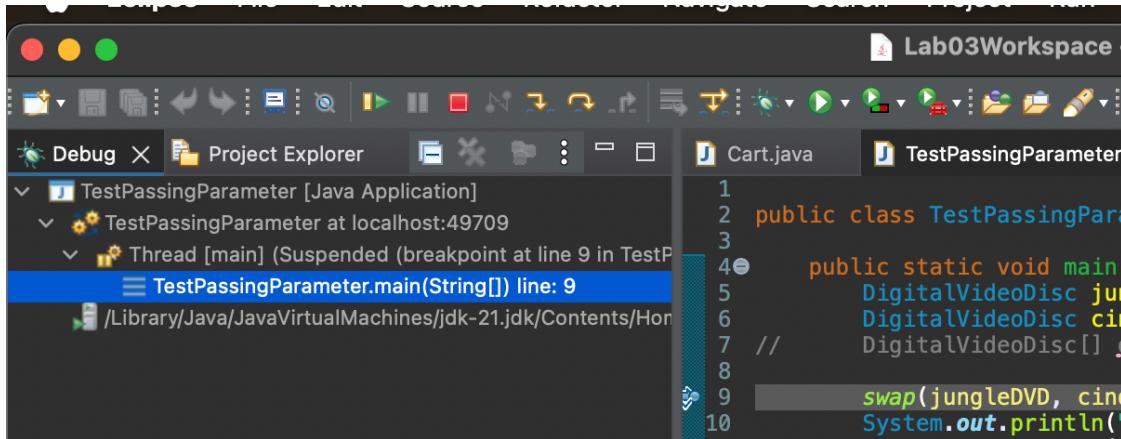


Figure 12. The toolbar in Debug perspective

```

17         changeTitle(jungleDVD, cinderellaDVD.getTitle());
18         System.out.println("jungle dvd title: " + jung
19     }
20
21     public static void swap(Object o1, Object o2) {
22         Object tmp = o1;
23         o1 = o2;
24         o2 = tmp;
25     }
26
27     public static void changeTitle(DigitalVideoDisc dvd,
28         String oldTitle = dvd.getTitle();
29         dvd.setTitle(title);
30         dvd = new DigitalVideoDisc(oldTitle);

```

Figure 13. Stepping into the 'swap' method

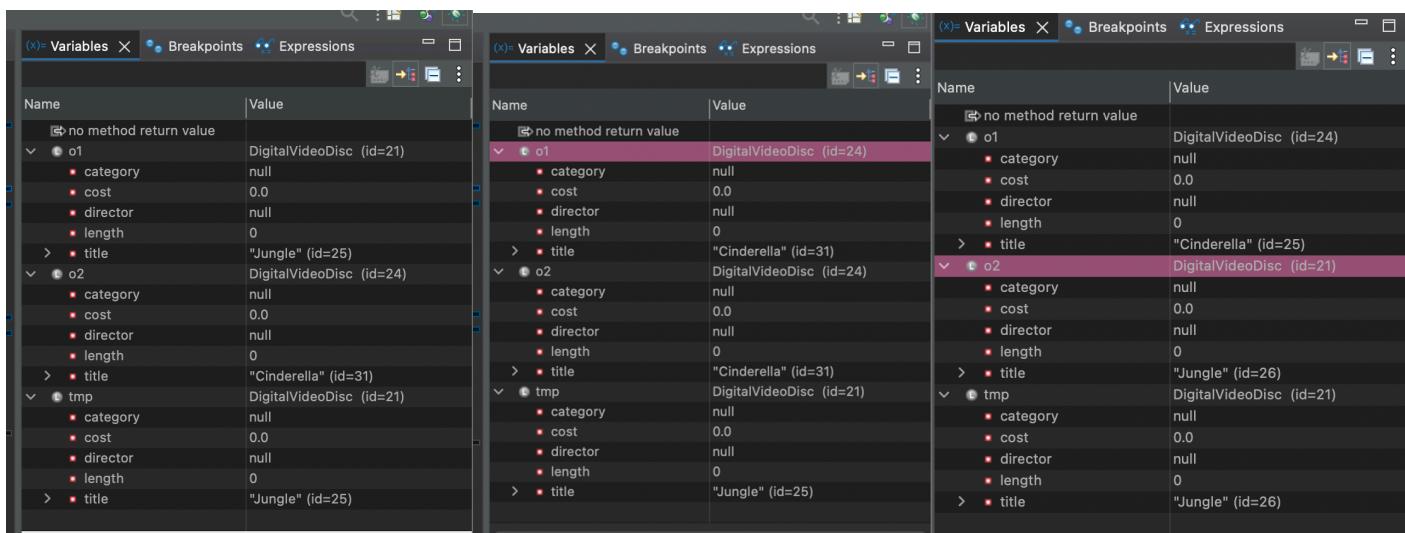


Figure 14. Observing variable 'o1', 'o2', 'tmp' change in 'swap' method

Lab03Workspace - AimsProject/src/TestPassingParameter.java - Eclipse IDE

```

1  public class TestPassingParameter {
2      public static void main(String[] args) {
3          DigitalVideoDisc jungleDVD = new DigitalVideoDisc("Jungle");
4          DigitalVideoDisc cinderellaDVD = new DigitalVideoDisc("Cinderella");
5          DigitalVideoDisc[] dvds = {jungleDVD, cinderellaDVD};
6
7          swap(dvds);
8          System.out.println("jungle dvd title: " + jungleDVD.getTitle());
9          System.out.println("cinderella dvd title: " + cinderellaDVD.getTitle());
10
11         // swap(dvds);
12         // System.out.println("jungle dvd title: " + dvds[0].getTitle());
13         // System.out.println("cinderella dvd title: " + dvds[1].getTitle());
14
15         changeTitle(jungleDVD, cinderellaDVD.getTitle());
16         System.out.println("jungle dvd title: " + jungleDVD.getTitle());
17     }
18
19     public static void swap(Object o1, Object o2) {
20         Object tmp = o1;
21         o1 = o2;
22         o2 = tmp;
23     }
24
25     public static void changeTitle(DigitalVideoDisc dvd, String title) {
26         String oldTitle = dvd.getTitle();
27         dvd.setTitle(title);
28         dvd = new DigitalVideoDisc(oldTitle);
29     }
30
31     // public static void swap(DigitalVideoDisc[] dvds) {
32     //     DigitalVideoDisc tmp = dvds[0];
33     //     dvds[0] = dvds[1];
34     //     dvds[1] = tmp;
35     // }
36     // ...
37 }
38
39

```

Variables View (Screenshot):

Name	Value
swap() returned	(No explicit return value)
args	String[0] (id=19)
jungleDVD	DigitalVideoDisc (id=21)
category	null
cost	0.0
director	null
length	0
title	"Jungle" (id=26)
cinderellaDVD	DigitalVideoDisc (id=24)
category	null
cost	0.0
director	null
length	0
title	"Cinderella" (id=25)

Figure 15. Using Step Return to return to the line after calling 'swap'

Variables View (Screenshot):

Name	Value
swap() returned	(No explicit return value)
args	String[0] (id=19)
jungleDVD	DigitalVideoDisc (id=21)
category	null
cost	0.0
director	null
length	0
title	abc
coder	0
hash	0
hashIsZero	false
value	(id=32)
cinderellaDVD	DigitalVideoDisc (id=24)
category	null
cost	0.0
director	null
length	0
title	"Cinderella" (id=25)

Figure 16. Changing the title attribute of 'jungleDVD'

```

1  public class TestPassingParameter {
2
3     public static void main(String[] args) {
4         DigitalVideoDisc jungleDVD = new DigitalVideoDisc("Jungle");
5         DigitalVideoDisc cinderellaDVD = new DigitalVideoDisc("Cinderella");
6         DigitalVideoDisc[] dvds = {jungleDVD, cinderellaDVD};
7
8         swap(jungleDVD, cinderellaDVD);
9         System.out.println("jungle dvd title: " + jungleDVD.getTitle());
10        System.out.println("cinderella dvd title " + cinderellaDVD.getTitle());
11
12        swap(dvds);
13        System.out.println("jungle dvd title: " + dvds[0].getTitle());
14        System.out.println("cinderella dvd title: " + dvds[1].getTitle());
15
16        changeTitle(jungleDVD, cinderellaDVD.getTitle());
17        System.out.println("jungle dvd title: " + jungleDVD.getTitle());
18
19    }
20
21    public static void swap(Object o1, Object o2) {
22        Object tmp = o1;
23        o1 = o2;
24        o2 = tmp;
25    }
26
27    public static void changeTitle(DigitalVideoDisc dvd, String title) {
28        String oldTitle = dvd.getTitle();
29        dvd.setTitle(title);
30        dvd = new DigitalVideoDisc(oldTitle);
31    }
32
33    public static void swap(DigitalVideoDisc[] dvds) {
34        DigitalVideoDisc tmp = dvds[0];
35        dvds[0] = dvds[1];
36        dvds[1] = tmp;
37    }
38
39

```

Console X Problems Debug Shell
TestPassingParameter [Java Application] /Library/Java/JavaVirtualMachines/jdk-21.jdk/Contents/Home/bin/java (Nov 24, 2024, 11:36:14 AM) [pid: 50885]
jungle dvd title: abc

Figure 17. Run result after modifying the variable

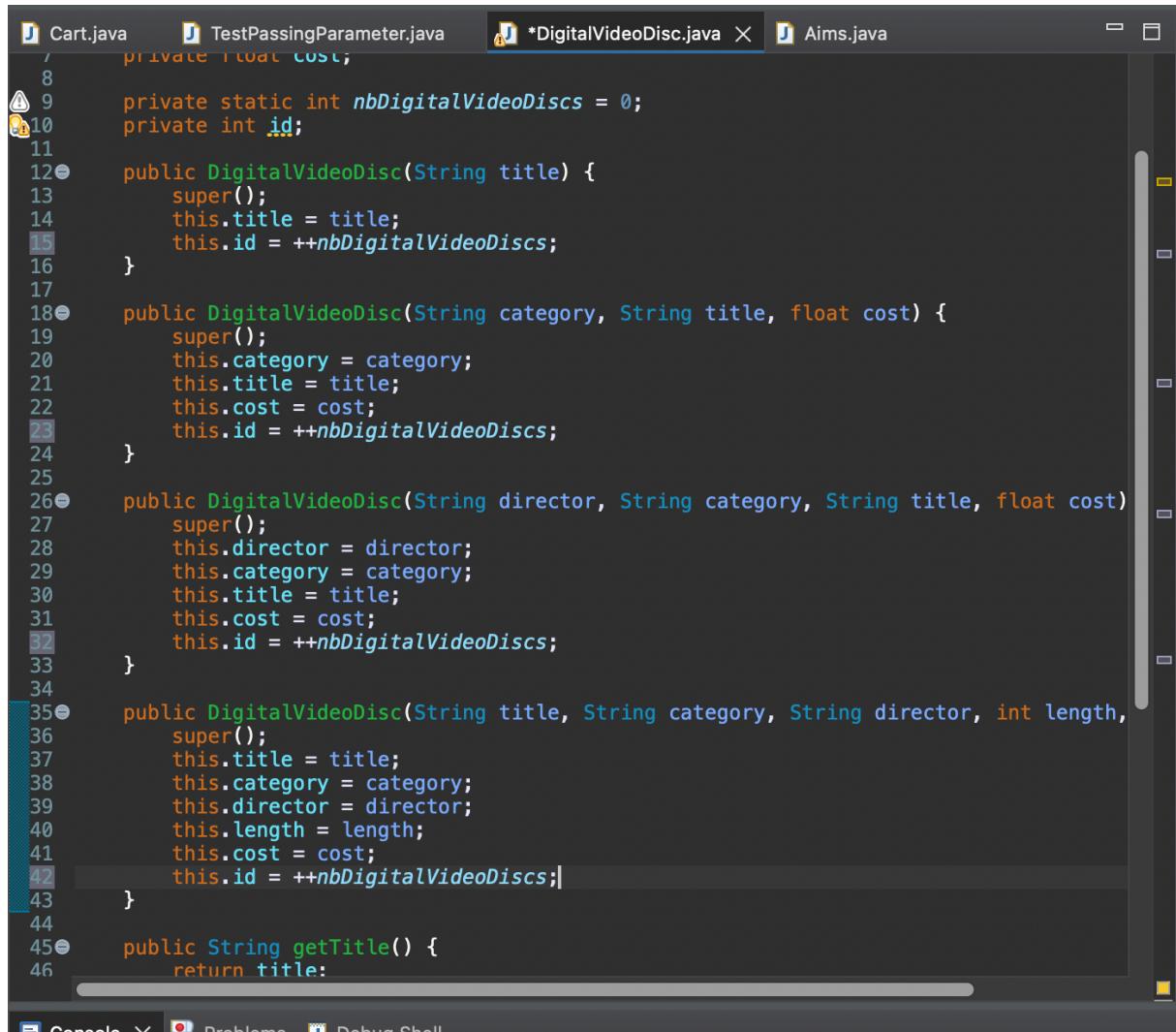
Section 5: Classifier Member and Instance Member

```

1  public class DigitalVideoDisc {
2
3     private String title;
4     private String category;
5     private String director;
6     private int length;
7     private float cost;
8
9     private static int nbDigitalVideoDiscs = 0;
10    private int id;
11
12    public DigitalVideoDisc(String title) {
13        super();
14        this.title = title;
15    }
16
17    public DigitalVideoDisc(String category, String title, float cost) {
18        super();
19        this.category = category;
20        this.title = title;
21        this.cost = cost;
22    }
23
24    public DigitalVideoDisc(String director, String category, String title, float cost)

```

Figure 18. Create a class attribute named 'nbDigitalVideoDiscs' and an instance attribute 'id'



```
Cart.java    TestPassingParameter.java    *DigitalVideoDisc.java X    Aims.java
8
9     private static int nbDigitalVideoDiscs = 0;
10    private int id;
11
12    public DigitalVideoDisc(String title) {
13        super();
14        this.title = title;
15        this.id = ++nbDigitalVideoDiscs;
16    }
17
18    public DigitalVideoDisc(String category, String title, float cost) {
19        super();
20        this.category = category;
21        this.title = title;
22        this.cost = cost;
23        this.id = ++nbDigitalVideoDiscs;
24    }
25
26    public DigitalVideoDisc(String director, String category, String title, float cost) {
27        super();
28        this.director = director;
29        this.category = category;
30        this.title = title;
31        this.cost = cost;
32        this.id = ++nbDigitalVideoDiscs;
33    }
34
35    public DigitalVideoDisc(String title, String category, String director, int length,
36        super();
37        this.title = title;
38        this.category = category;
39        this.director = director;
40        this.length = length;
41        this.cost = cost;
42        this.id = ++nbDigitalVideoDiscs;
43    }
44
45    public String getTitle() {
46        return title;

```

Figure 19. Update 'nbDigitalVideoDiscs' inside every constructor methods and assign the value to 'id'

Section 6: Update the 'Cart' class

```

40     this.length = length;
41     this.cost = cost;
42     this.id = ++nbDigitalVideoDiscs;
43 }
44
45     public String getTitle() {
46         return title;
47     }
48     public String getCategory() {
49         return category;
50     }
51     public String getDirector() {
52         return director;
53     }
54     public int getLength() {
55         return length;
56     }
57     public float getCost() {
58         return cost;
59     }
60     public void setTitle(String title) {
61         this.title = title;
62     }
63
64     @Override
65     public String toString() {
66         return "DVD: " + this.title +
67             " - Category: " + this.category +
68             " - Director: " + this.title +
69             " - DVD length: " + this.length +
70             " - Cost: $" + this.cost;
71     }
72
73     public boolean isMatch(String title) {
74         return this.title.toLowerCase().contains(title.toLowerCase());
75     }
76 }
77
78

```

Console X Problems Debug Shell

<terminated> TestPassingParameter [Java Application] /Library/Java/JavaVirtualMachines/jdk-21.jdk/Contents/Home/bin/java (Nov 2)

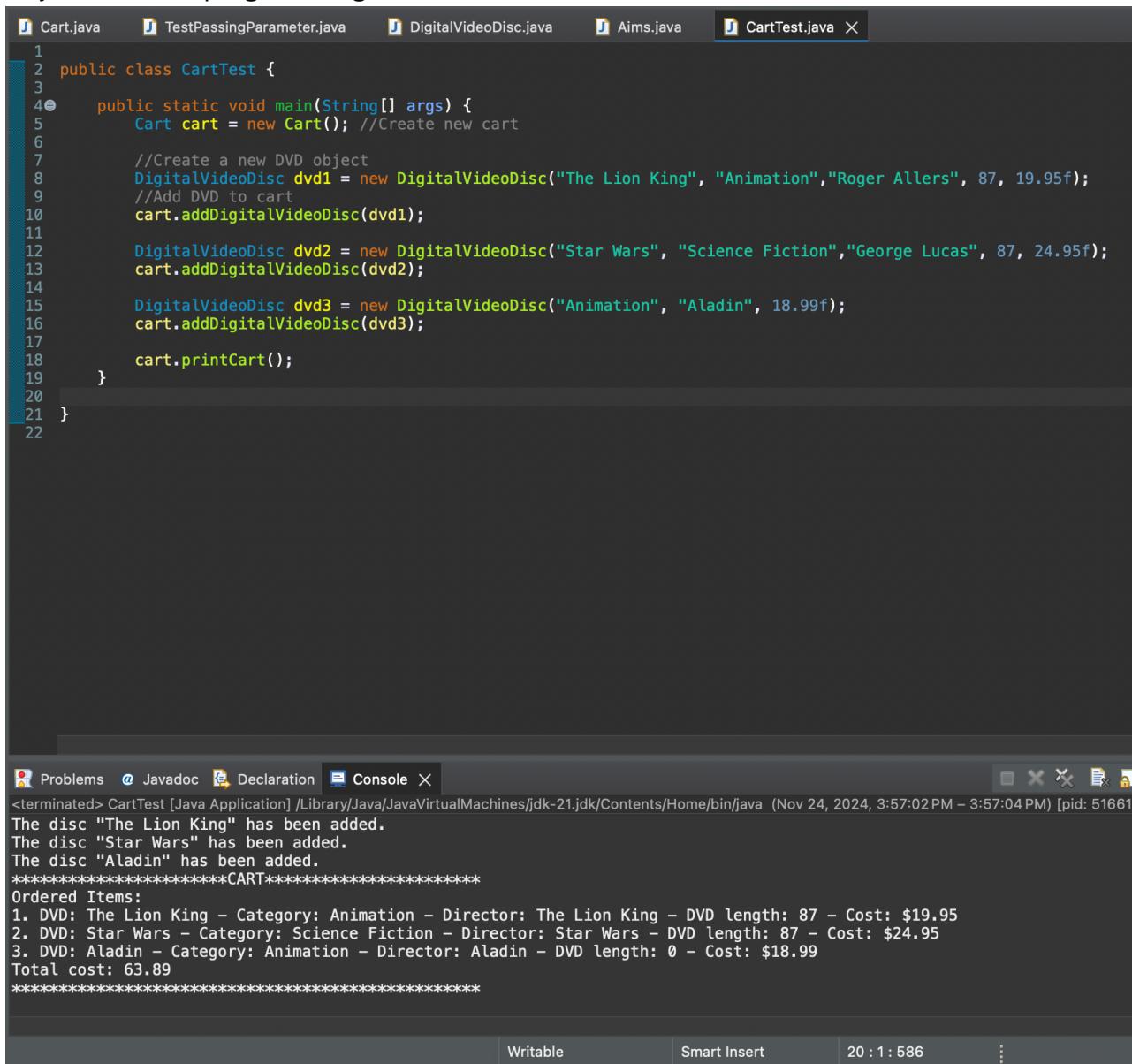
Figure 20. Write a new 'toString' method to return a DVD's info by a format and a 'isMatch' method to check for DVD's title match

```

64     public void searchByID(int id) {
65         boolean found = false;
66         for (int i = 0; i < qtyOrdered; i++) {
67             if (itemsOrdered[i].getId() == id) {
68                 System.out.println("Found" + itemsOrdered[i]);
69                 found = true;
70             }
71         }
72         if (!found) {
73             System.out.println("No DVDs were found match the ID provided");
74         }
75     }
76
77     public void searchByTitle(String keyword) {
78         boolean matchFound = false;
79         for (int i = 0; i < qtyOrdered; i++) {
80             if (itemsOrdered[i].isMatch(keyword)) {
81                 System.out.println("Found" + itemsOrdered[i]);
82                 matchFound = true;
83             }
84         }
85         if (!matchFound) {
86             System.out.println("NO DVDs were found with '" + keyword + "' in the title");
87         }
88     }
89
90     public void printCart() {
91         System.out.println("*****CART*****");
92         System.out.println("Ordered Items:");
93         for (int i = 0; i < qtyOrdered; i++) {
94             System.out.println(i+1 + ". " + itemsOrdered[i]);
95         }
96         System.out.println("Total cost: " + totalCost());
97         System.out.println("*****");
98     }
99
100 }
101

```

Figure 21. New methods in 'Cart' to print out cart contents, search DVDs by titles and IDs

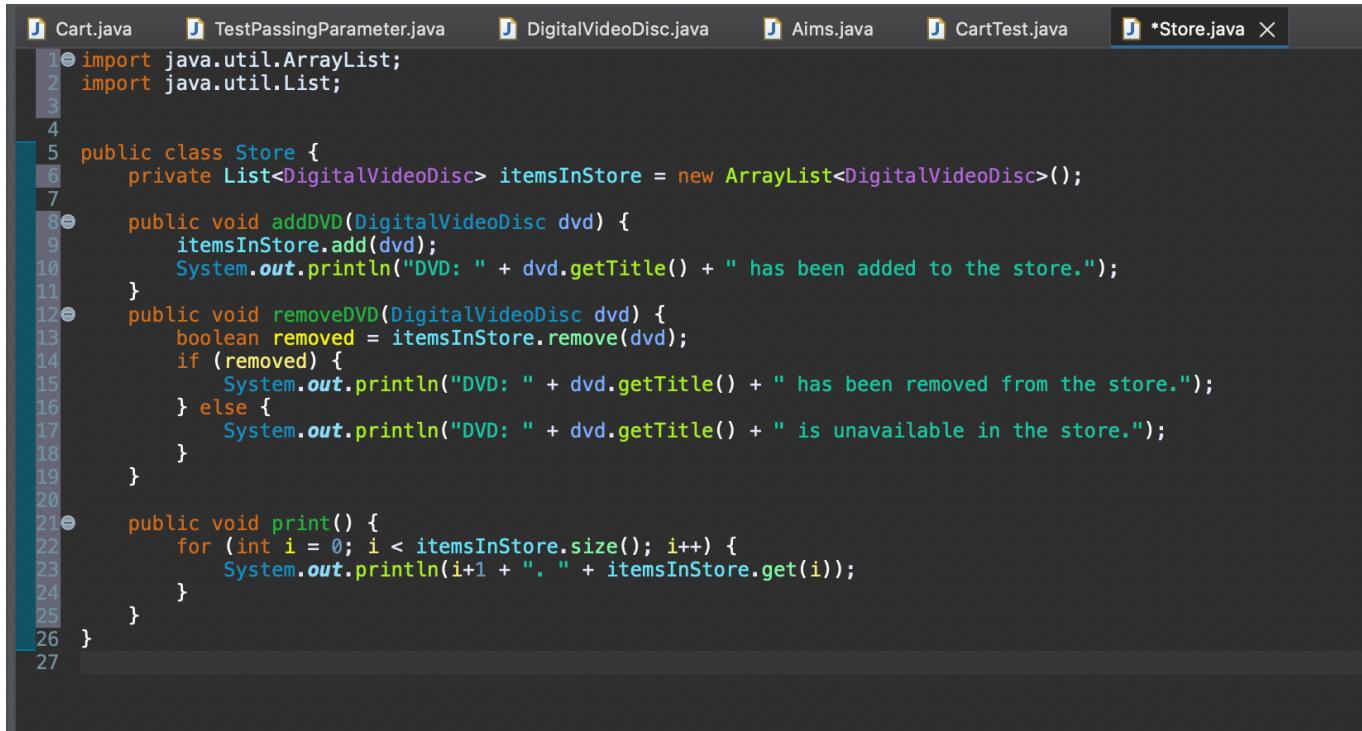


The screenshot shows an IDE interface with the following details:

- Top Bar:** Shows tabs for Cart.java, TestPassingParameter.java, DigitalVideoDisc.java, Aims.java, and CartTest.java (which is the active tab).
- Code Editor:** Displays the content of CartTest.java. The code creates a new Cart object, adds three DigitalVideoDisc objects to it, and then prints the cart's contents.
- Console Output:** Shows the execution results:
 - The disc "The Lion King" has been added.
 - The disc "Star Wars" has been added.
 - The disc "Aladin" has been added.
 - *****CART*****
 - Ordered Items:
 - 1. DVD: The Lion King – Category: Animation – Director: The Lion King – DVD length: 87 – Cost: \$19.95
 - 2. DVD: Star Wars – Category: Science Fiction – Director: Star Wars – DVD length: 87 – Cost: \$24.95
 - 3. DVD: Aladin – Category: Animation – Director: Aladin – DVD length: 0 – Cost: \$18.99
 - Total cost: 63.89
 - *****
- Bottom Status Bar:** Shows "Writable", "Smart Insert", and the current time "20 : 1 : 586".

Figure 22. Testing the new methods in a new class 'CartTest'

Section 7: Implement the 'Store' class

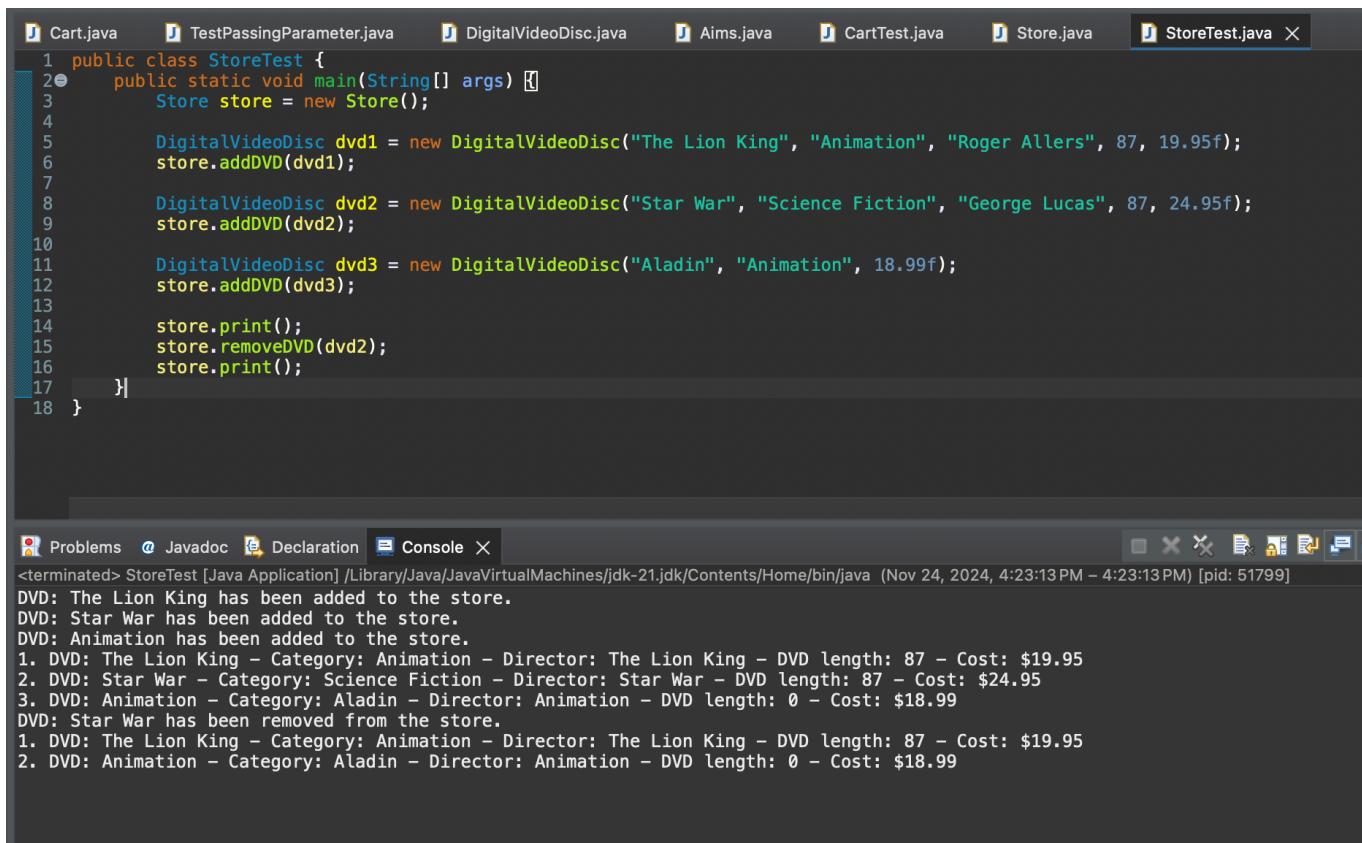


```

1 import java.util.ArrayList;
2 import java.util.List;
3
4
5 public class Store {
6     private List<DigitalVideoDisc> itemsInStore = new ArrayList<DigitalVideoDisc>();
7
8     public void addDVD(DigitalVideoDisc dvd) {
9         itemsInStore.add(dvd);
10        System.out.println("DVD: " + dvd.getTitle() + " has been added to the store.");
11    }
12
13    public void removeDVD(DigitalVideoDisc dvd) {
14        boolean removed = itemsInStore.remove(dvd);
15        if (removed) {
16            System.out.println("DVD: " + dvd.getTitle() + " has been removed from the store.");
17        } else {
18            System.out.println("DVD: " + dvd.getTitle() + " is unavailable in the store.");
19        }
20    }
21
22    public void print() {
23        for (int i = 0; i < itemsInStore.size(); i++) {
24            System.out.println(i+1 + ". " + itemsInStore.get(i));
25        }
26    }
27 }

```

Figure 23. New 'Store' class with single attribute 'itemsInStore[]' and 3 methods 'addDVD', 'removeDVD', and 'print'



```

1 public class StoreTest {
2     public static void main(String[] args) {
3         Store store = new Store();
4
5         DigitalVideoDisc dvd1 = new DigitalVideoDisc("The Lion King", "Animation", "Roger Allers", 87, 19.95f);
6         store.addDVD(dvd1);
7
8         DigitalVideoDisc dvd2 = new DigitalVideoDisc("Star War", "Science Fiction", "George Lucas", 87, 24.95f);
9         store.addDVD(dvd2);
10
11         DigitalVideoDisc dvd3 = new DigitalVideoDisc("Aladin", "Animation", 18.99f);
12         store.addDVD(dvd3);
13
14         store.print();
15         store.removeDVD(dvd2);
16         store.print();
17     }
18 }

```

Console Output:

```

<terminated> StoreTest [Java Application] /Library/Java/JavaVirtualMachines/jdk-21.jdk/Contents/Home/bin/java (Nov 24, 2024, 4:23:13 PM – 4:23:13 PM) [pid: 51799]
DVD: The Lion King has been added to the store.
DVD: Star War has been added to the store.
DVD: Animation has been added to the store.
1. DVD: The Lion King – Category: Animation – Director: The Lion King – DVD length: 87 – Cost: $19.95
2. DVD: Star War – Category: Science Fiction – Director: Star War – DVD length: 87 – Cost: $24.95
3. DVD: Animation – Category: Aladin – Director: Animation – DVD length: 0 – Cost: $18.99
DVD: Star War has been removed from the store.
1. DVD: The Lion King – Category: Animation – Director: The Lion King – DVD length: 87 – Cost: $19.95
2. DVD: Animation – Category: Aladin – Director: Animation – DVD length: 0 – Cost: $18.99

```

Figure 24. New 'StoreTest' to test methods in 'Store' clas

Section 8: Re-organize your projects

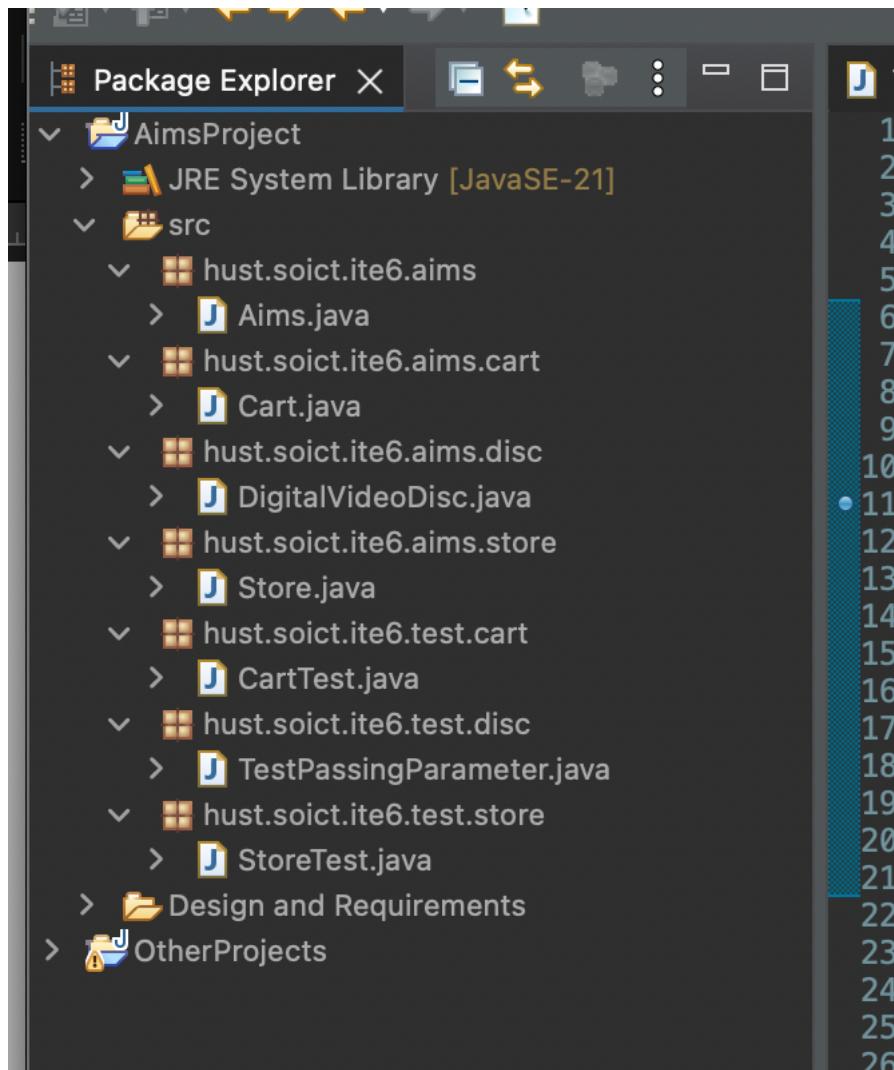
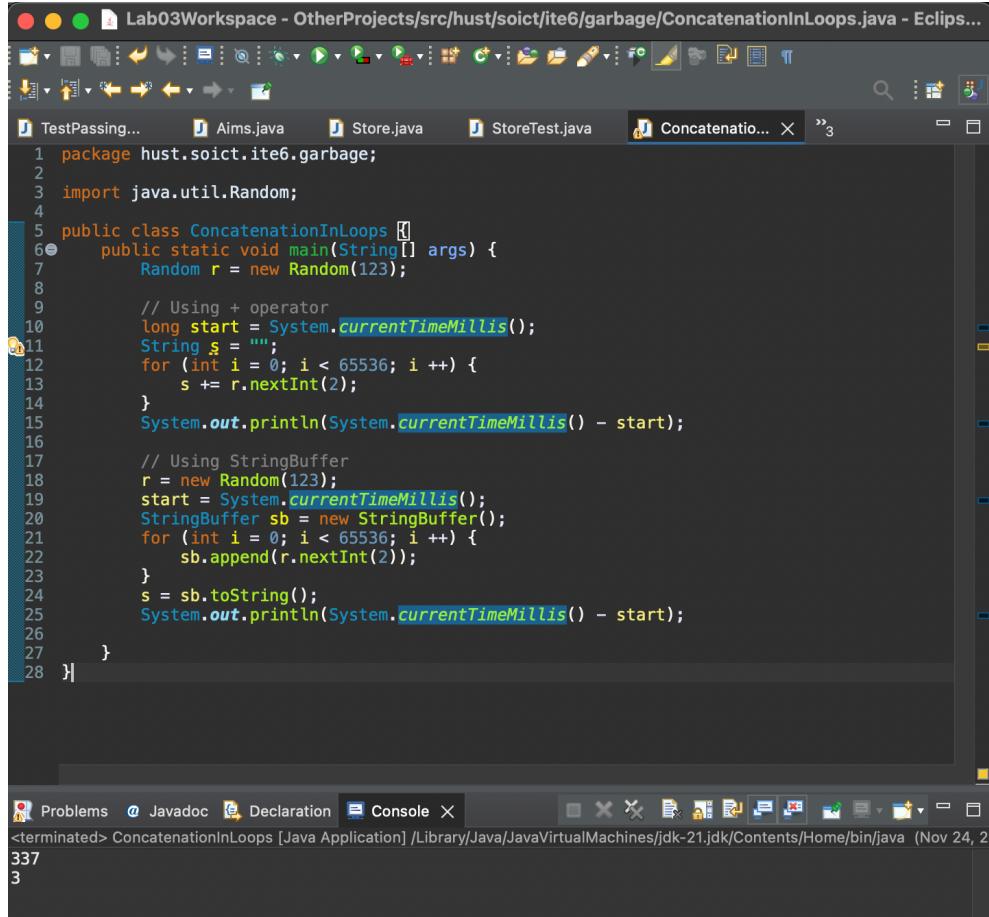


Figure 25. Reorganized classes inside packages

Section 9: String, StringBuilder and StringBuffer



The screenshot shows the Eclipse IDE interface with the following details:

- Project:** Lab03Workspace - OtherProjects/src/hust/soict/ite6/garbage/ConcatenationInLoops.java
- Code:** The code compares two methods of string concatenation within loops. The first uses the '+' operator, and the second uses a StringBuffer. Both methods print the time taken for the loop execution.

```

1 package hust.soict.ite6.garbage;
2
3 import java.util.Random;
4
5 public class ConcatenationInLoops {
6     public static void main(String[] args) {
7         Random r = new Random(123);
8
9         // Using + operator
10        long start = System.currentTimeMillis();
11        String s = "";
12        for (int i = 0; i < 65536; i++) {
13            s += r.nextInt(2);
14        }
15        System.out.println(System.currentTimeMillis() - start);
16
17        // Using StringBuffer
18        r = new Random(123);
19        start = System.currentTimeMillis();
20        StringBuffer sb = new StringBuffer();
21        for (int i = 0; i < 65536; i++) {
22            sb.append(r.nextInt(2));
23        }
24        s = sb.toString();
25        System.out.println(System.currentTimeMillis() - start);
26
27    }
28 }

```

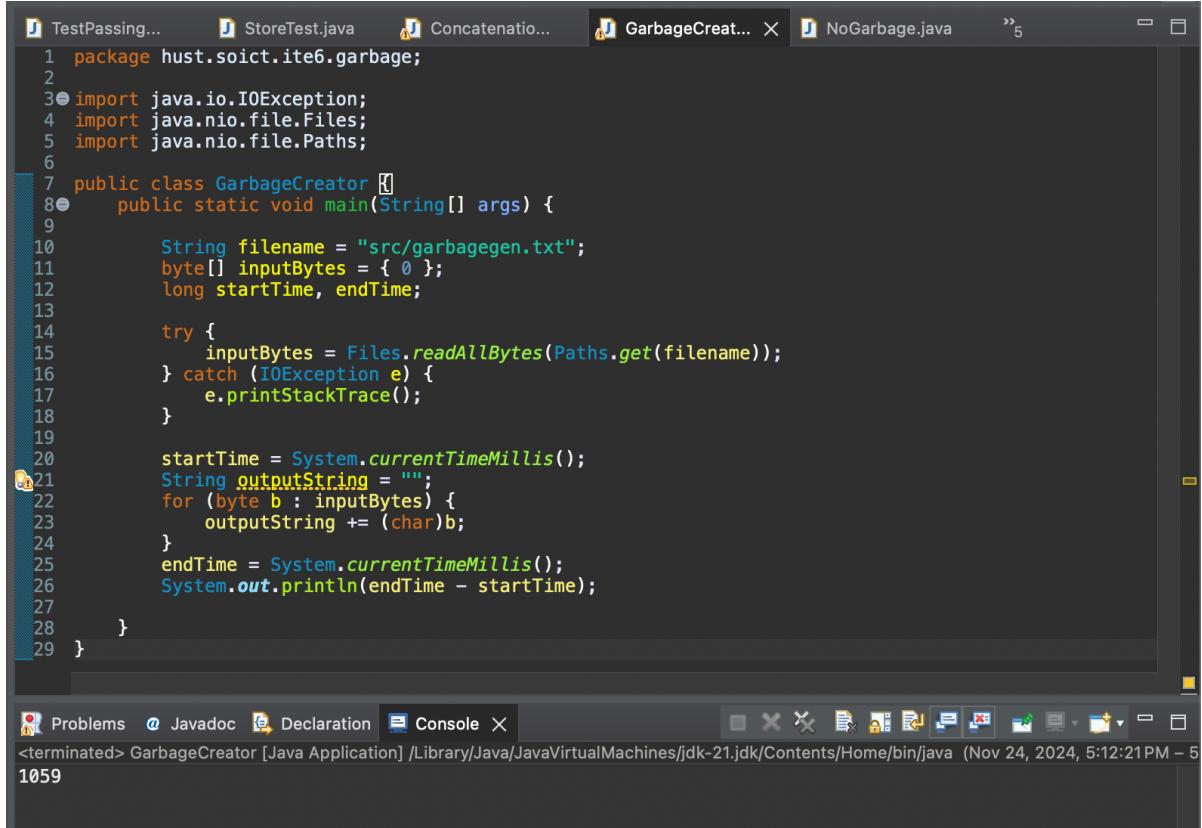
Figure 26. This shows string concatenation using `StringBuffer` yields better performance than using the `+` operator

Another demonstration:



The terminal window displays a large amount of text, likely a log or a file content, which is extremely long and contains many characters. The text is mostly illegible due to its length but appears to be a sequence of characters and numbers.

Figure 27. A text file with extreme long string



The screenshot shows an IDE interface with several tabs at the top: TestPassing..., StoreTest.java, Concatenatio..., GarbageCreat..., NoGarbage.java, and a file with a double question mark icon. The GarbageCreator.java tab is active, displaying the following code:

```

1 package hust.soict.ite6.garbage;
2
3 import java.io.IOException;
4 import java.nio.file.Files;
5 import java.nio.file.Paths;
6
7 public class GarbageCreator {
8     public static void main(String[] args) {
9
10        String filename = "src/garbagegen.txt";
11        byte[] inputBytes = { 0 };
12        long startTime, endTime;
13
14        try {
15            inputBytes = Files.readAllBytes(Paths.get(filename));
16        } catch (IOException e) {
17            e.printStackTrace();
18        }
19
20        startTime = System.currentTimeMillis();
21        String outputString = "";
22        for (byte b : inputBytes) {
23            outputString += (char)b;
24        }
25        endTime = System.currentTimeMillis();
26        System.out.println(endTime - startTime);
27    }
28 }
29

```

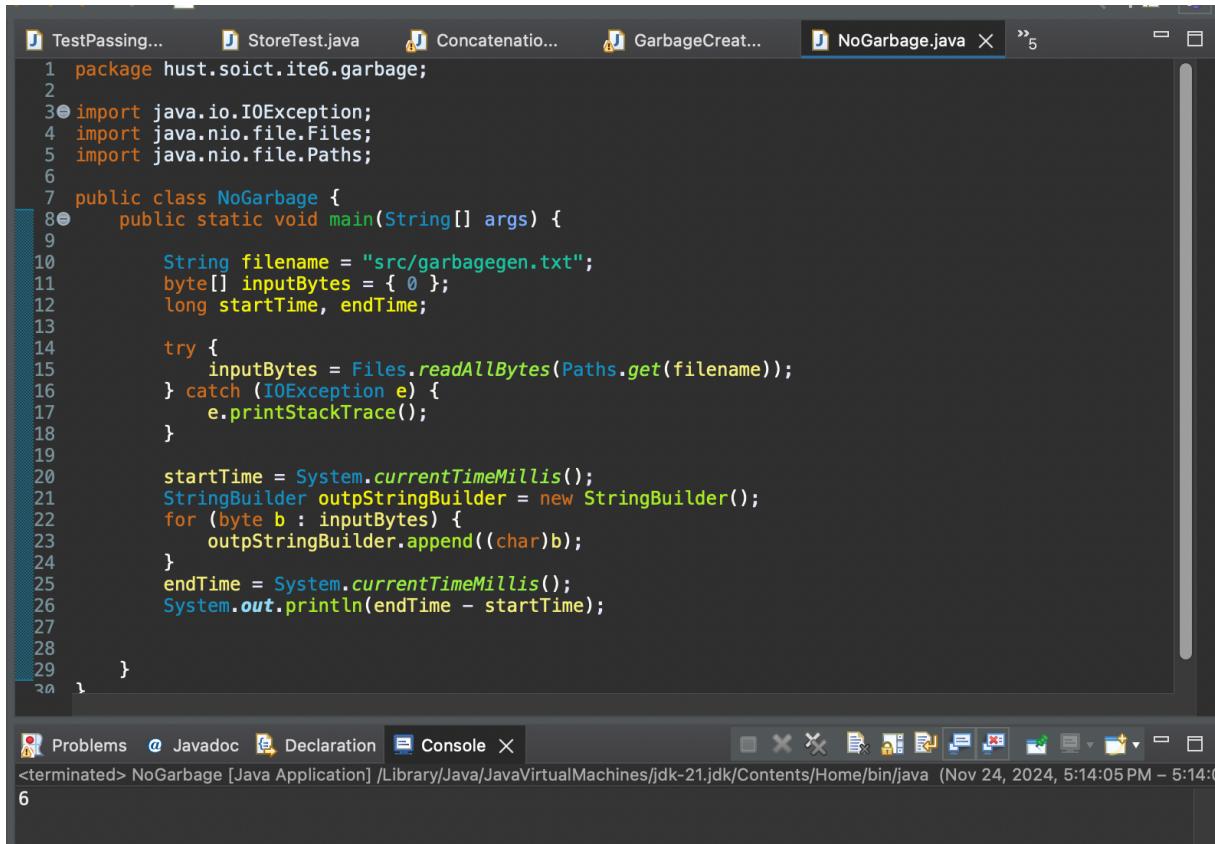
Below the code editor is a toolbar with icons for Problems, Javadoc, Declaration, and Console. The Console tab is selected, showing the output of the program's execution:

```

<terminated> GarbageCreator [Java Application] /Library/Java/JavaVirtualMachines/jdk-21.jdk/Contents/Home/bin/java (Nov 24, 2024, 5:12:21PM - 5
1059

```

Figure 28. The 'GarbageCreator' reads the above file and process it using the + operator take about 1sec of runtime



The screenshot shows an IDE interface with several tabs at the top: TestPassing..., StoreTest.java, Concatenatio..., GarbageCreat..., NoGarbage.java, and a file with a double question mark icon. The NoGarbage.java tab is active, displaying the following code:

```

1 package hust.soict.ite6.garbage;
2
3 import java.io.IOException;
4 import java.nio.file.Files;
5 import java.nio.file.Paths;
6
7 public class NoGarbage {
8     public static void main(String[] args) {
9
10        String filename = "src/garbagegen.txt";
11        byte[] inputBytes = { 0 };
12        long startTime, endTime;
13
14        try {
15            inputBytes = Files.readAllBytes(Paths.get(filename));
16        } catch (IOException e) {
17            e.printStackTrace();
18        }
19
20        startTime = System.currentTimeMillis();
21        StringBuilder outpStringBuilder = new StringBuilder();
22        for (byte b : inputBytes) {
23            outpStringBuilder.append((char)b);
24        }
25        endTime = System.currentTimeMillis();
26        System.out.println(endTime - startTime);
27
28    }
29

```

Below the code editor is a toolbar with icons for Problems, Javadoc, Declaration, and Console. The Console tab is selected, showing the output of the program's execution:

```

<terminated> NoGarbage [Java Application] /Library/Java/JavaVirtualMachines/jdk-21.jdk/Contents/Home/bin/java (Nov 24, 2024, 5:14:05 PM - 5:14:0
6

```

Figure 29. Another implementation using StringBuilder shortens the runtime to only 6ms

Section 10: Release flow demonstration

```
to read about a specific subcommand or concept.
See 'git help git' for an overview of the system.
caoducanh@Caos-MacBook-Pro Lab03Workspace % git checkout -b feature/demonstrate-release-flow
Switched to a new branch 'feature/demonstrate-release-flow'
caoducanh@Caos-MacBook-Pro Lab03Workspace % git commit -m "Add a feature for demonstration"
```

Figure 30. Create a new branch in the local repo

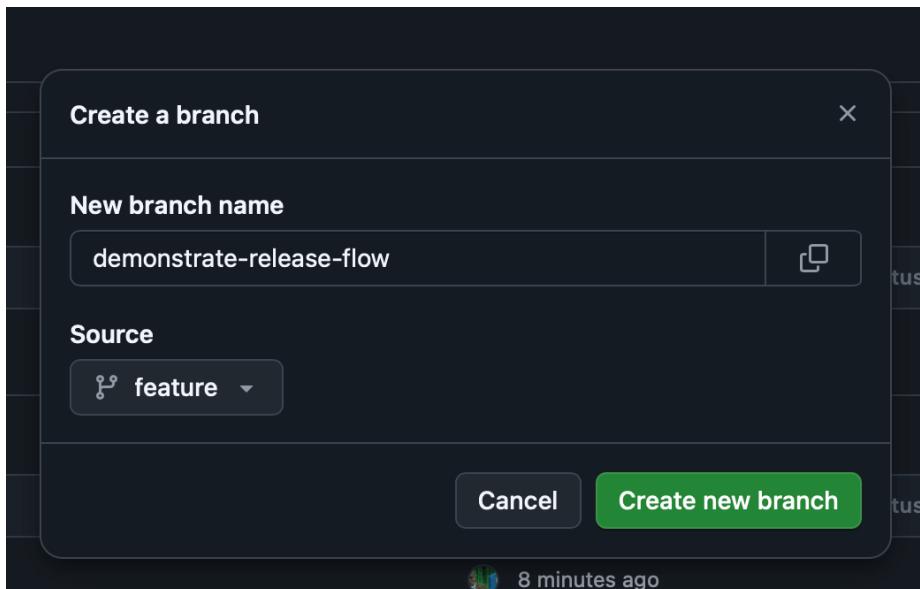


Figure 31. Creating remote branch

```
[caoducanh@Caos-MacBook-Pro Lab03Workspace % git commit -m "Add feature for demo"
On branch feature/demonstrate-release-flow
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    deleted:  AimsProject/Design and Requirements/AimsUML.asta.lock
new topic or a new feature to our application. The next section shows us how to apply Release
Ynthesis
Untracked files:
  (use "git add <file>..." to include in what will be committed)
  .DS_Store
  metadata/
  wing command and resolve conflicts if any.
  no changes added to commit (use "git add" and/or "git commit -a")
caoducanh@Caos-MacBook-Pro Lab03Workspace %
```

Figure 32. Commit changes to the branch 'feature/demonstrate-release-flow' on local repo

```
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
[caoducanh@Caos-MacBook-Pro Lab03Workspace % git push origin feature/demonstrate-release-flow
Enumerating objects: 138, done.
Counting objects: 100% (138/138), done.
Delta compression using up to 8 threads
Compressing objects: 100% (104/104), done.
Writing objects: 100% (138/138), 194.67 KiB | 7.49 MiB/s, done.
Total 138 (delta 14), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (14/14), done.
remote:
remote: Create a pull request for 'feature/demonstrate-release-flow' on GitHub by visiting:
remote: https://github.com/KazooBoye/ObjectOrientedProgrammingLab744521/pull/new/feature/
remote: demonstrate-release-flow Create README.md yesterday
remote:
remote: To https://github.com/KazooBoye/ObjectOrientedProgrammingLab744521.git
 * [new branch]      feature/demonstrate-release-flow -> feature/demonstrate-release-flow
caoducanh@Caos-MacBook-Pro Lab03Workspace %
```

Figure 33. Adding remote repo as reference the local repo then push changes to the remote repo

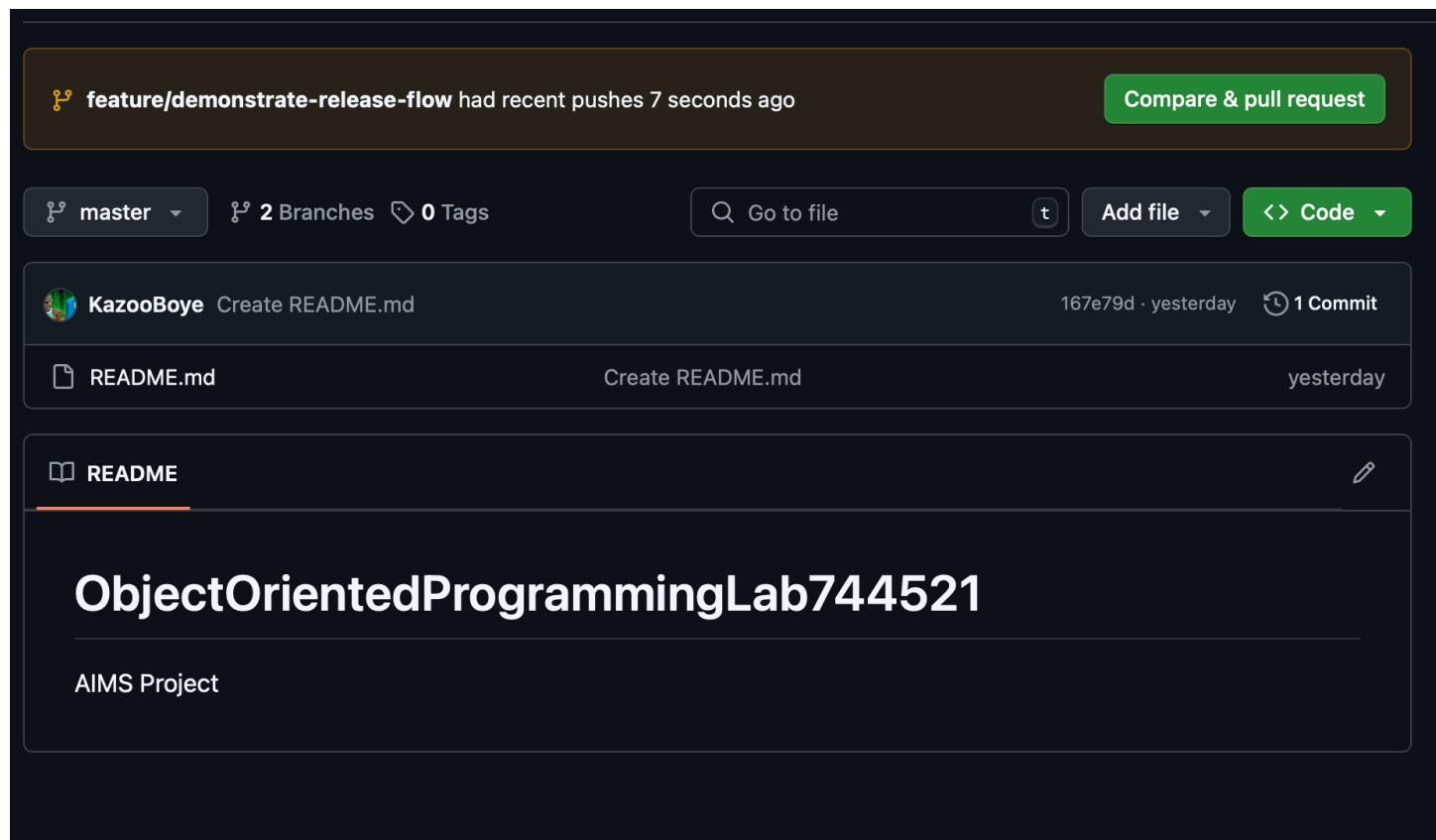


Figure 34. New pull request has been created on the remote repo

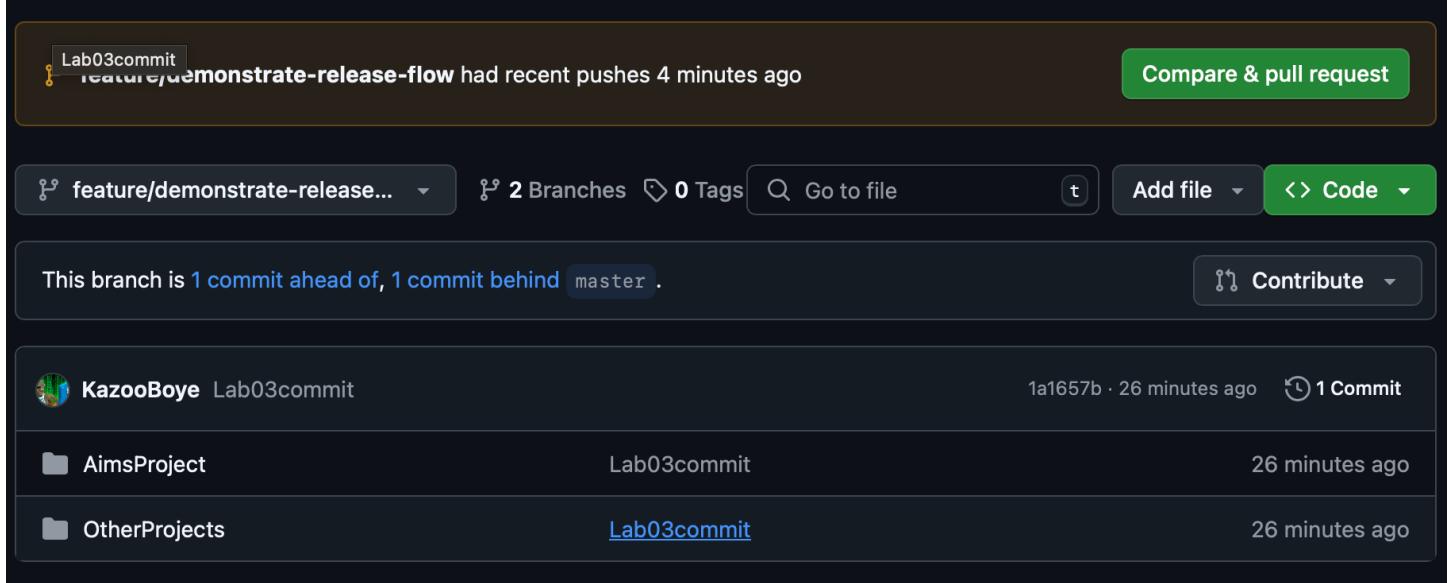


Figure 35. Switch to the new branch and compare changes

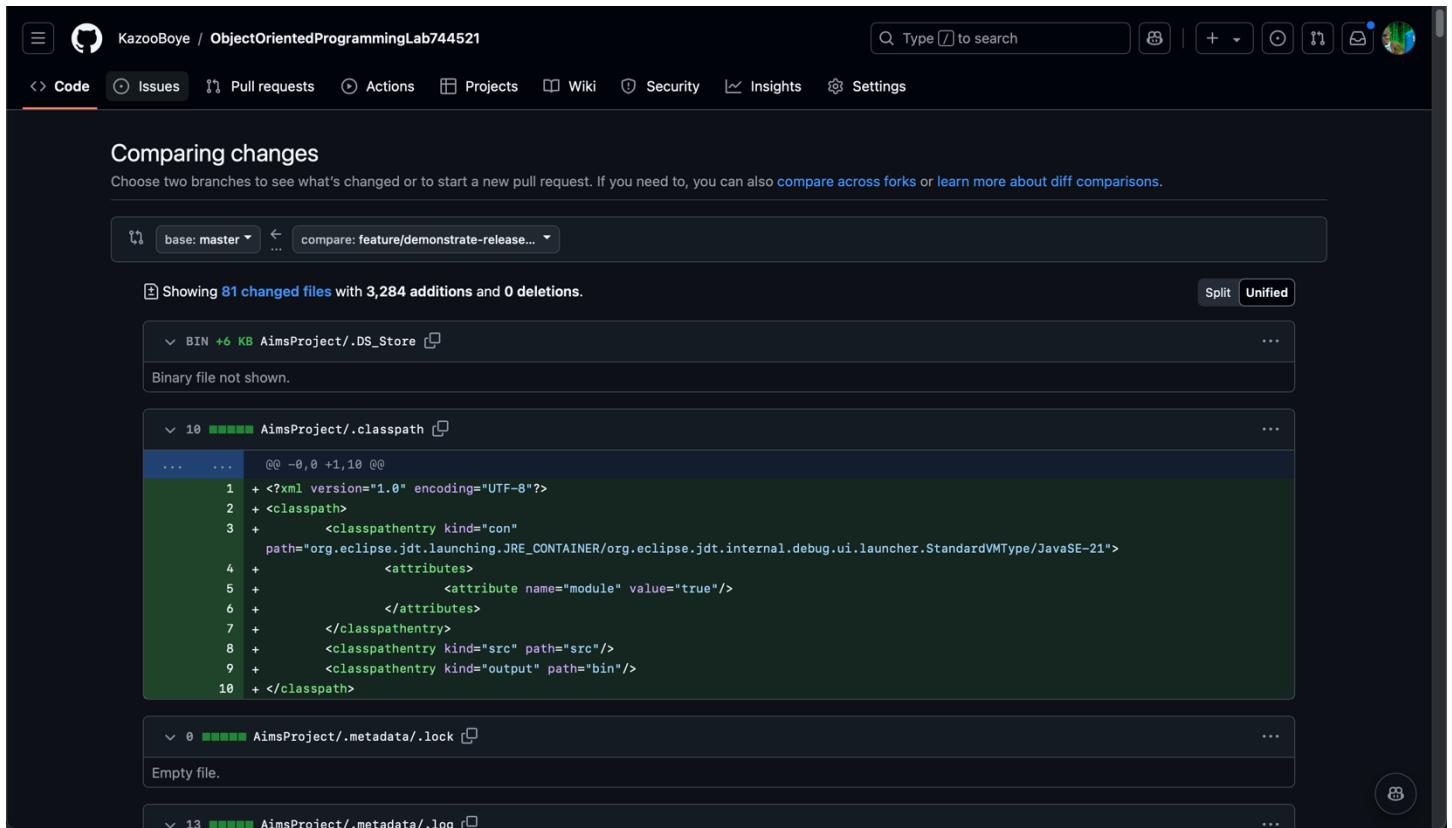


Figure 36. Creating a pull request to merge the current branch to 'master'

The screenshot shows a GitHub repository page. At the top, the repository name is 'ObjectOrientedProgrammingLab744521' with a 'Public' badge. To the right are 'Pin' and 'Unwatch' buttons. Below the repository name, there are buttons for 'master' (with a dropdown arrow), '2 Branches' (with a dropdown arrow), '0 Tags', 'Go to file' (with a search icon), 'Add file' (with a dropdown arrow), and 'Code' (with a dropdown arrow). A search bar is also present. The main content area shows a merge request from 'KazooBoye' to 'master'. The merge request details are: 'Merge https://github.com/KazooBoye/ObjectOrientedProgrammingLab...'. It was made by '2adbf27' 2 minutes ago and contains 4 commits. Below this, there are two commit entries: 'AimsProject' and 'OtherProjects', both labeled 'Lab03commit' and timestamped '45 minutes ago'. At the bottom, there is a section for the 'README' file, which is currently empty. A large 'Add a README' button with a book icon is centered, and a sub-instruction 'Help people interested in this repository understand your project by adding a README.' is displayed below it.

Figure 37. Confirm changes have been made to 'master'