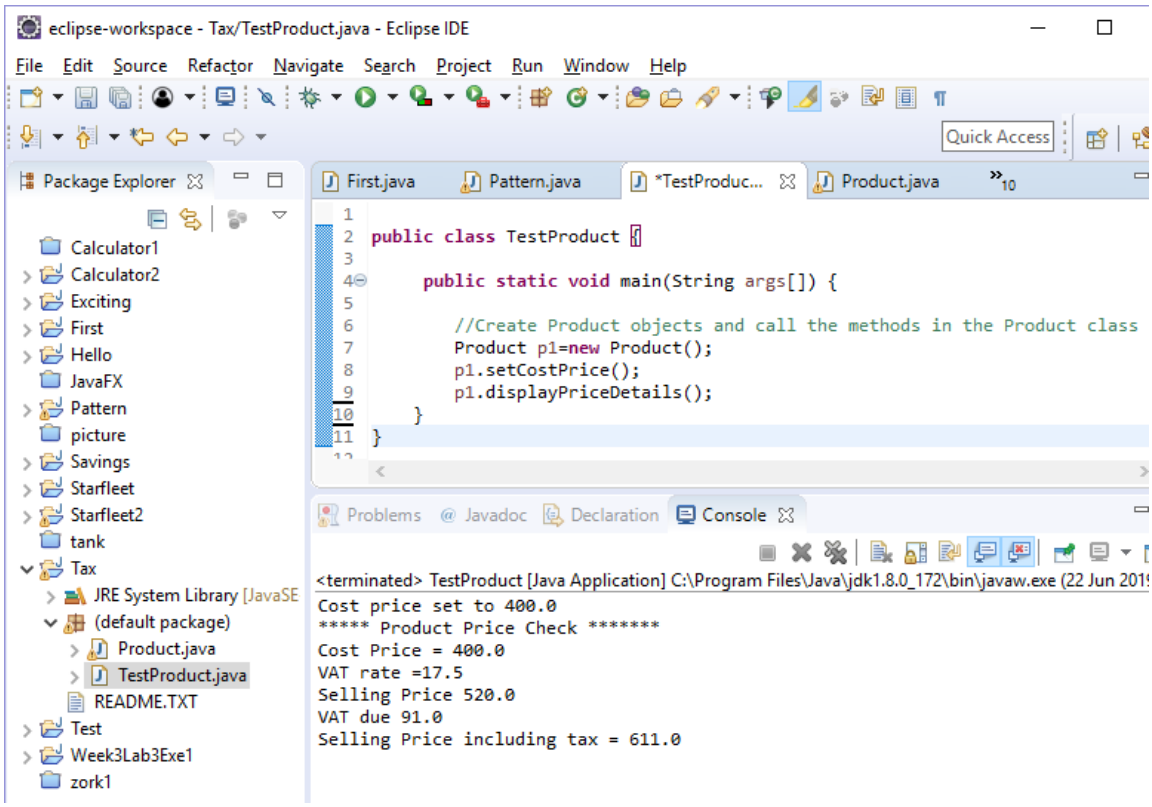
Week 2 In-Lesson Tasks Feedback

2.1.1 - Writing Methods

Task: Open the Tax project, in your Week 2, Lab 1 folder. Within the TestProduct class, create an object and test the **setCostPrice() and displayPriceDetails()** methods.

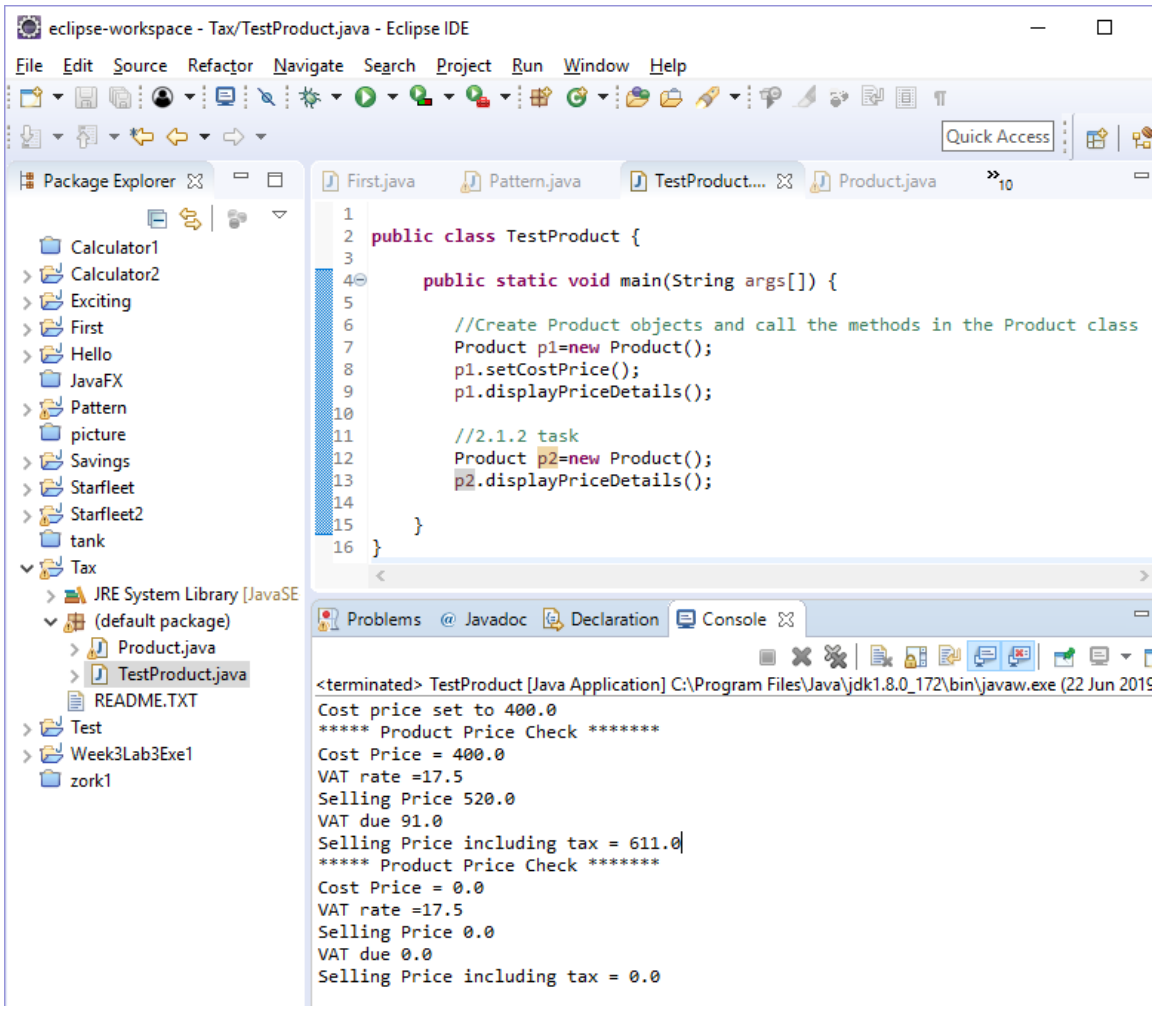
Here it is in the snapshot below:



2.1.2 - Methods Can Return Values

Task: Open the Tax project and test the above method within the TestProduct class.

Here it is as in the snapshot below.



2.1.3 - Passing Information to Methods

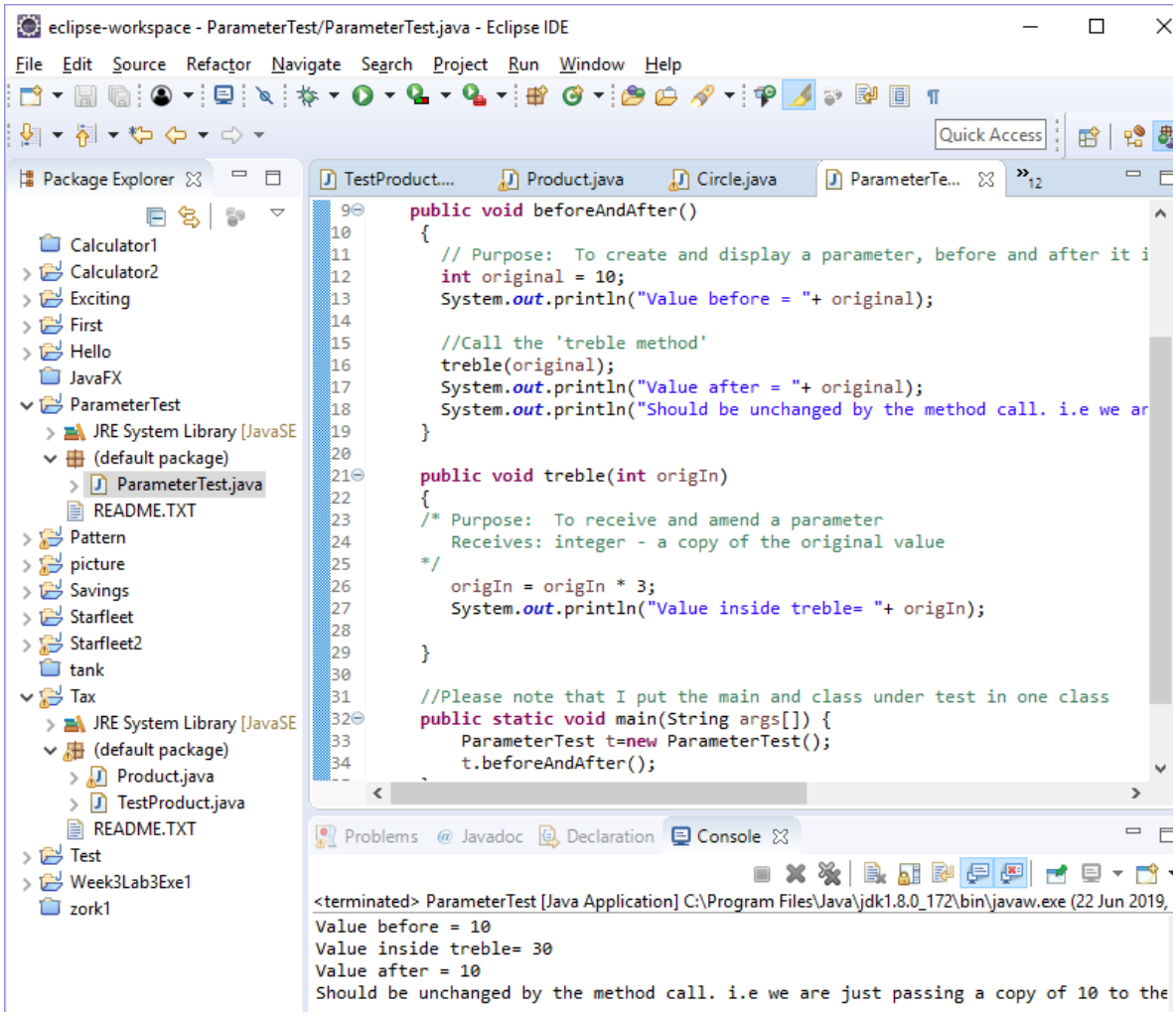
Task: Open the picture project and review the moveLeft, moveRight, moveVertical and moveHorizental method in the Circle class. Take a particular look at the methods with parameters.

Here are the code extracts below. Please note that the moveVertical and moveHorizental methods each takes a parameter. Within the moveRight and moveLeft method, the value 20 and -20 are passed to the moveHorizontal method.

```
 /**
 * Move the circle a few pixels to the right.
 */
public void moveRight()
{
		moveHorizontal(20);
}

/**
 * Move the circle a few pixels to the left.
 */
public void moveLeft()
{
		moveHorizontal(-20);
}

/**
 * Move the circle horizontally by 'distance' pixels.
 */
public void moveHorizontal(int distance)
{
		erase();
		xPosition += distance;
		draw();
}

/**
 * Move the circle vertically by 'distance' pixels.
 */
public void moveVertical(int distance)
{
		erase();
		yPosition += distance;
		draw();
}
```

Task: Watch the video below and then run the beforeAndAfter method yourself by going to your Week 2, Lab 1 folder and finding the ParameterTest project folder.

The snapshot below illustrates this.

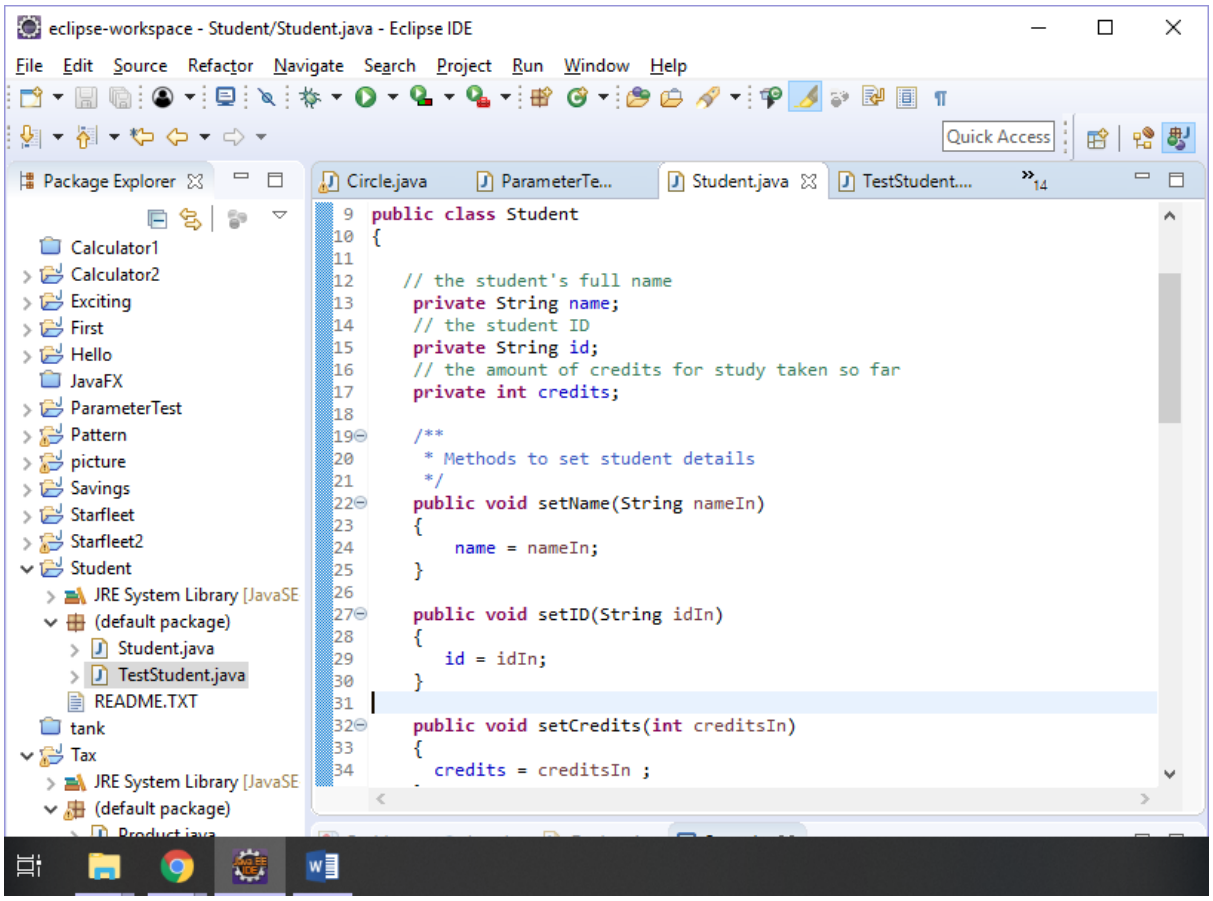


2.1.4 - Using Strings

Task: Study the student project.

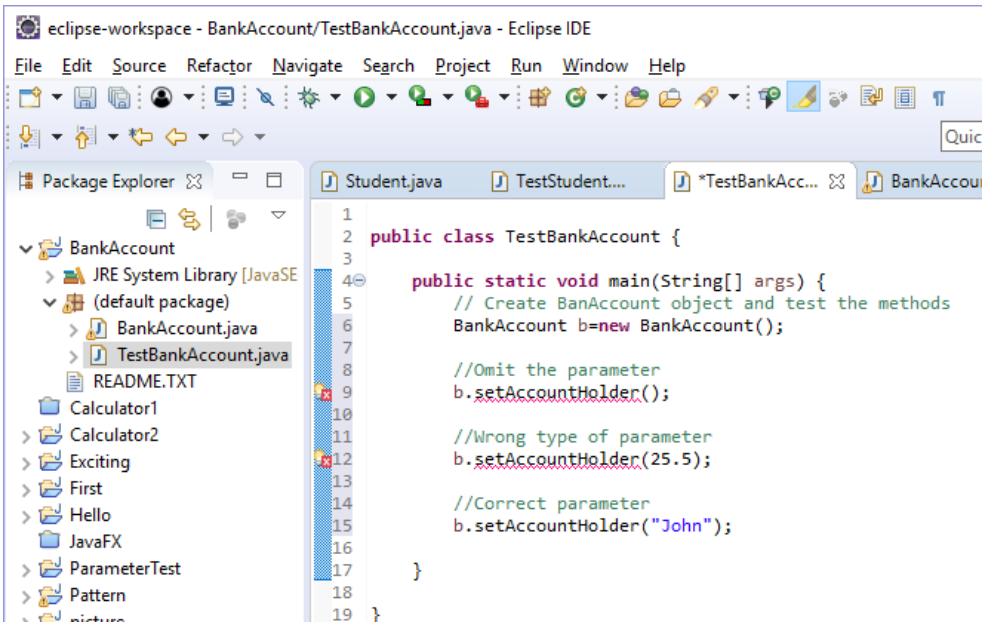
Notice how we pass **Strings as parameters** in the 'setName(String nameIn)' method, to set the students' name. As you can see the method specifies one formal parameter - nameIn.

The snapshot below demonstrates the use of String *nameIn* and *idIn* as parameter. Their values are passed onto *name* and *id* respectively.



Task: Open the BankAccount project, within the TestBankAccount class, pass incorrect parameters to methods and view errors.

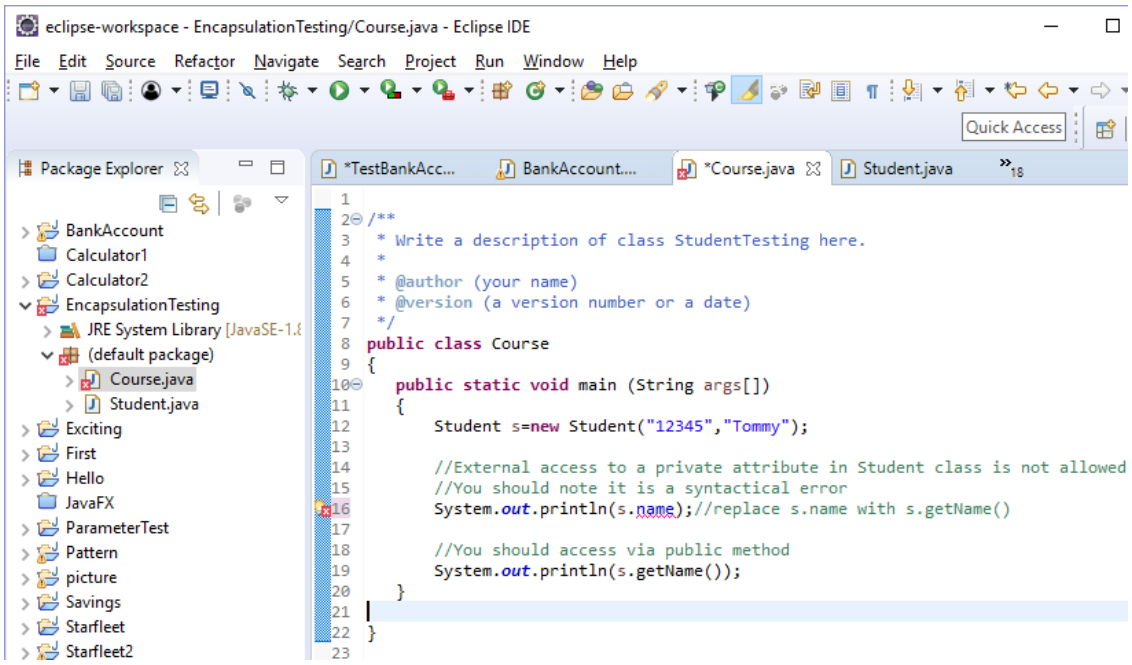
Here are some examples in the snapshot below. Incorrect ones are highlighted in red.



2.2.2 - Encapsulation

Task: Open the EncapsulationTesting project. Investigate and correct the syntax error in the Course class.

Here it is the snapshot that illustrates this.

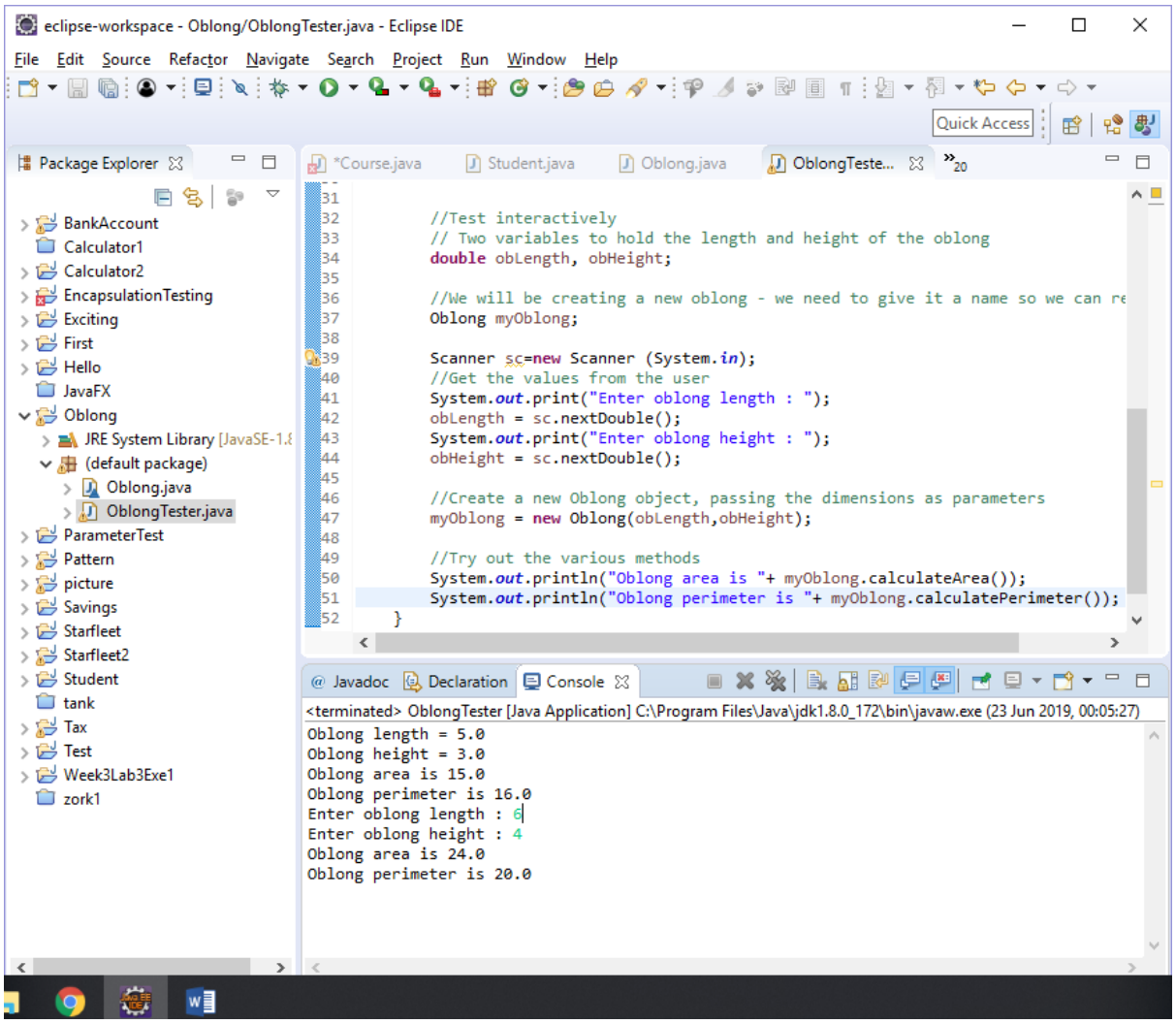


2.2.3 - Classes as data types

Task:

Open the Oblong project in Eclipse. Review the code in *Oblong* and *OblongTester* class. Run the project and interact with the program by entering the *length* and *height*, then observing the output. Try to match the code with the program behaviour.

Please see the snapshot below. Code line 26-29 produce the first four lines of the output. Code line 41 onwards produce the next four lines of the output.



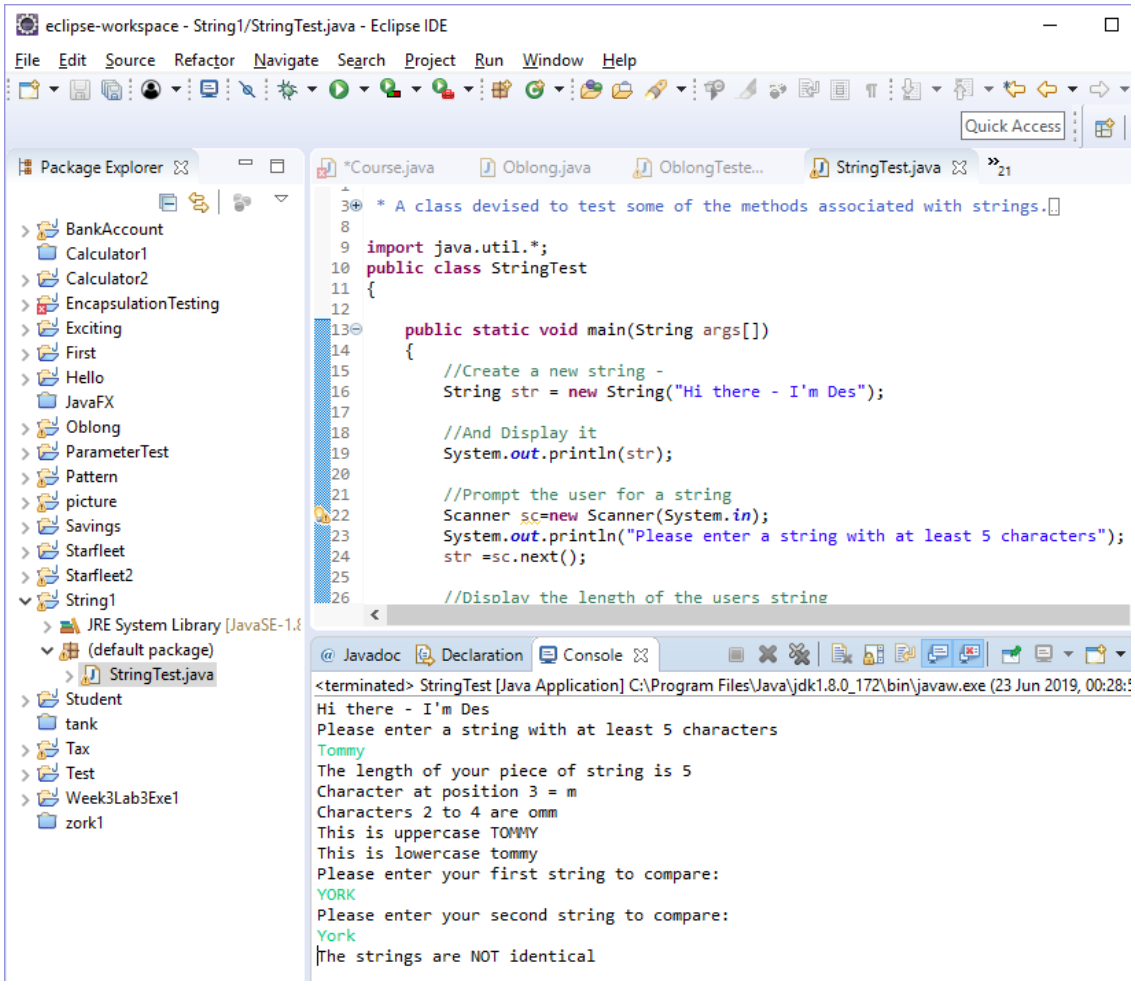
2.2.4 - The String Class

Task: Open the String1 project. Review the code in the *StringTest* class and make a note of the String methods that have been invoked. Run and interact with the program.

The following String methods have been invoked:

String constructor
length
charAt
substring
toUpperCase
toLowerCase

Here it is the snapshot with interactions:



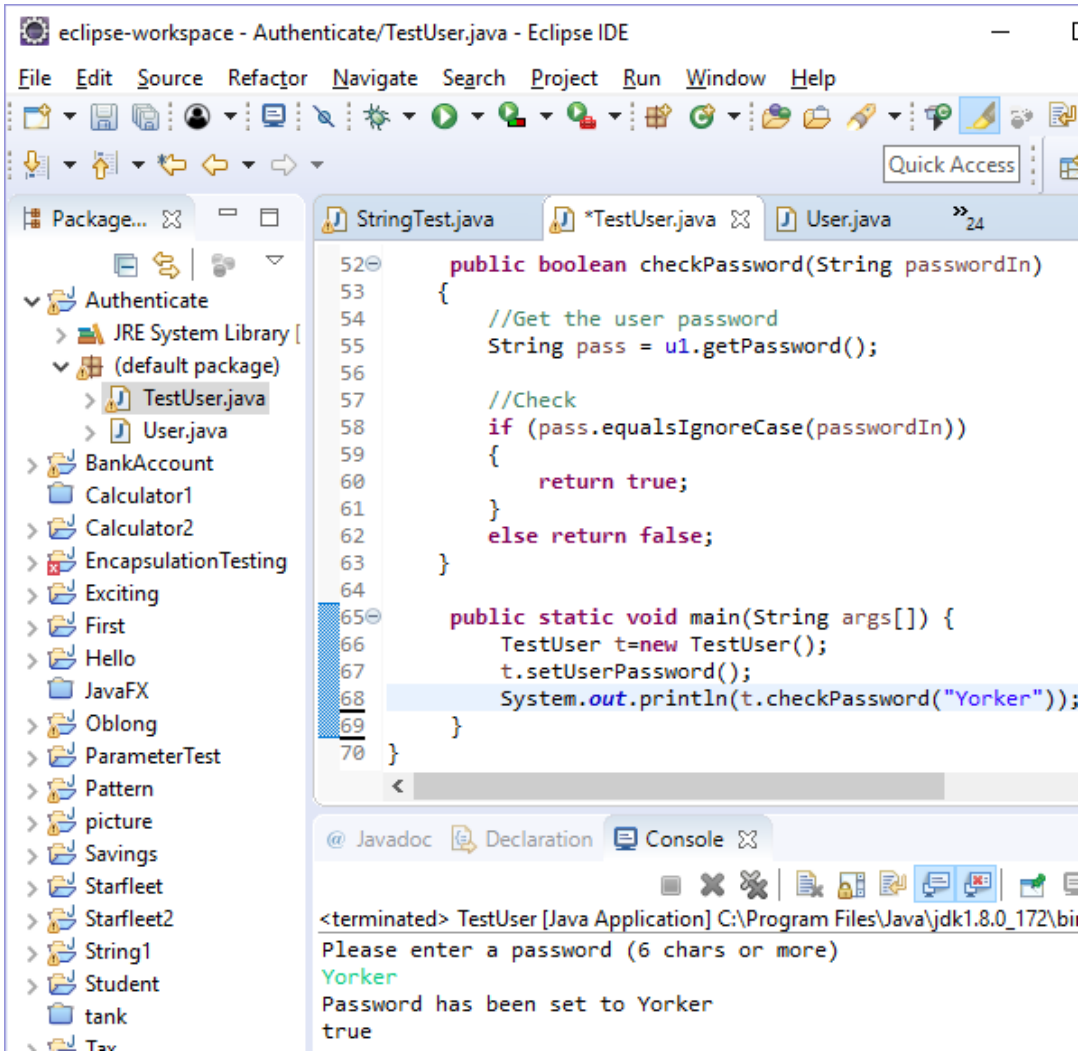
2.2.5 - Objects as Parameters

Task: Open the Authenticate project and review the code in the *User* and *TestUser* class. Make a note of the parameters for *setPassword* and *checkPassword* method. Run and interact with the program by entering at least a bad and a good password.

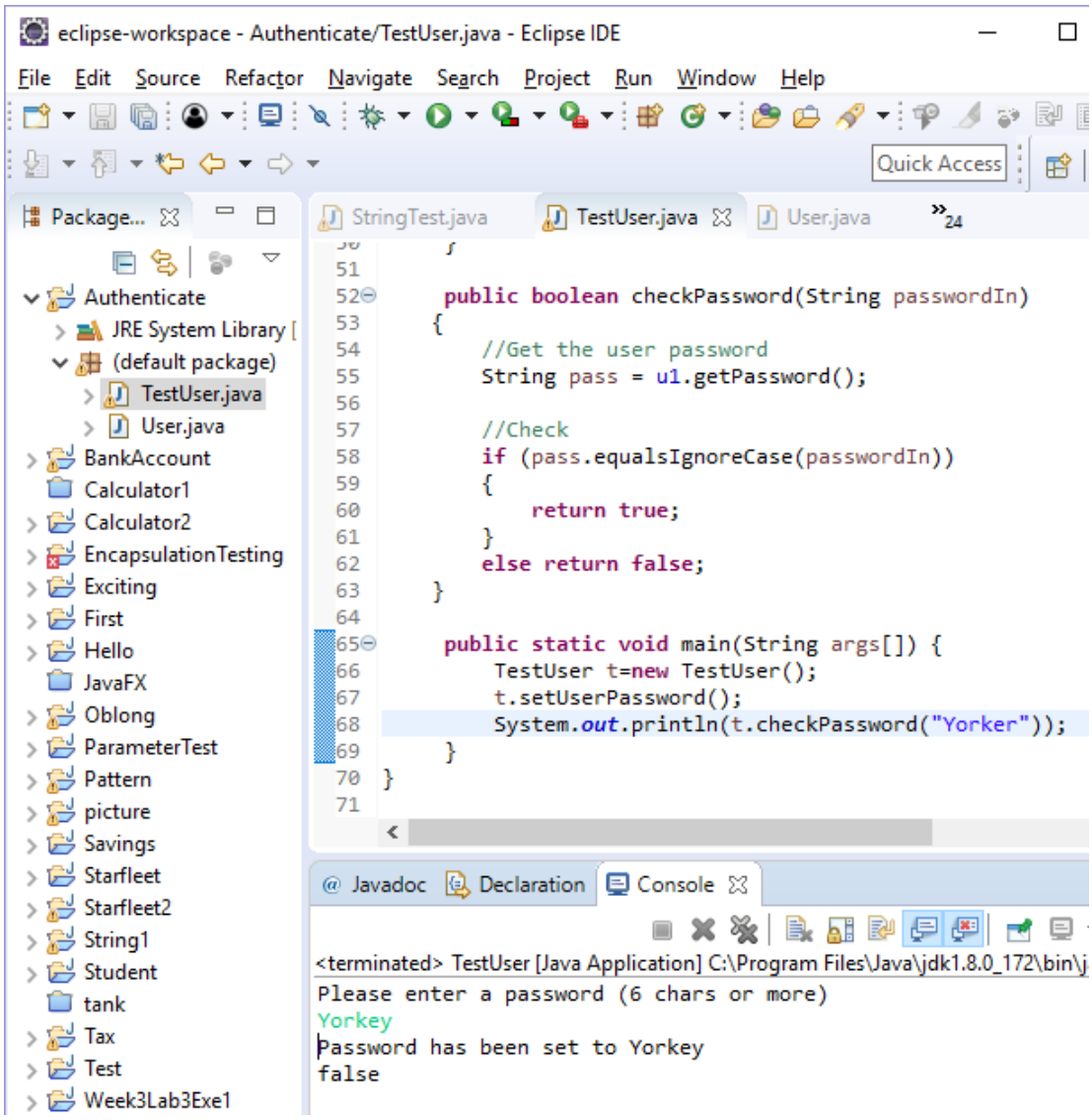
Parameters:

- setPassword(String passwordIn)
- checkPassword(String passwordIn)

Here is the snapshot for a good password.

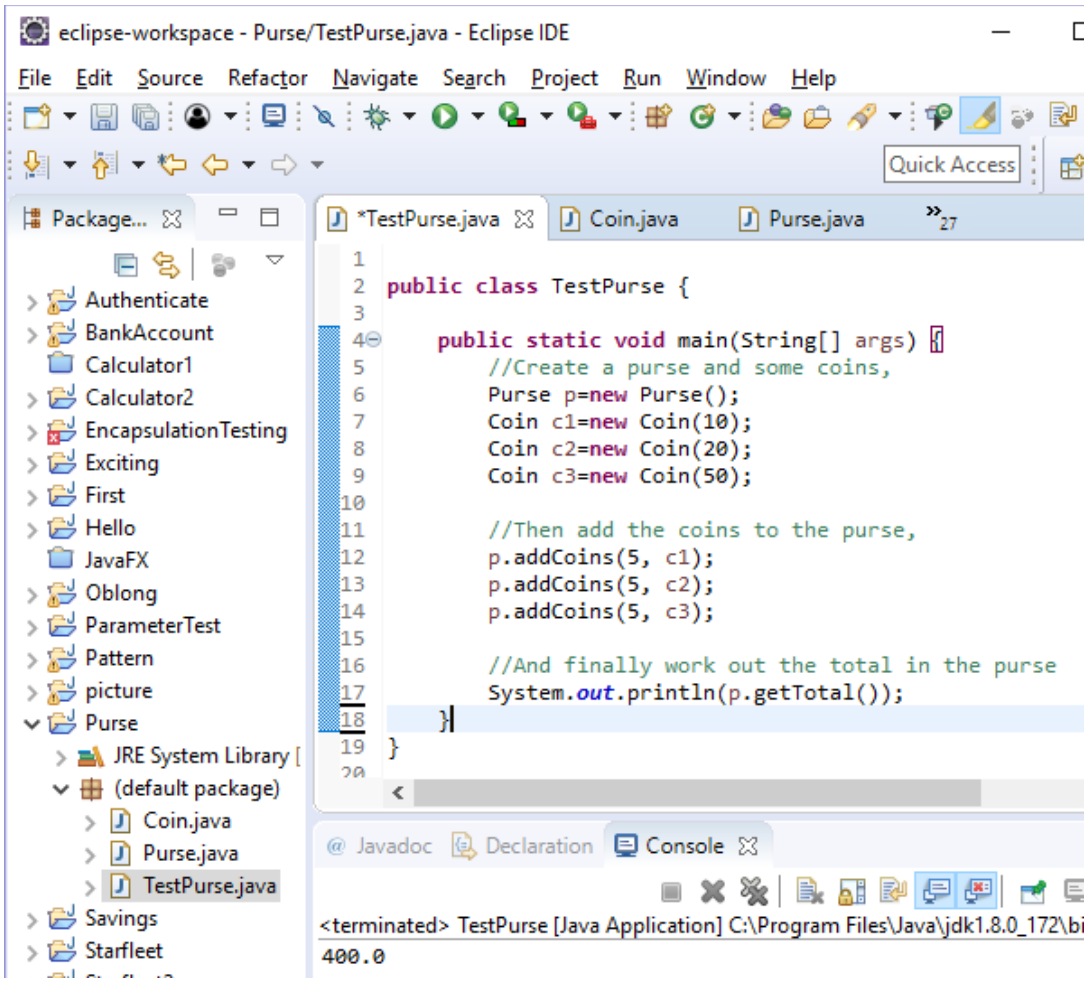


And here is the snapshot for a bad password.



Task: Investigate the Purse project. Use the TestPurse class to create some coins, and add them to the purse. Then find the total in the purse.

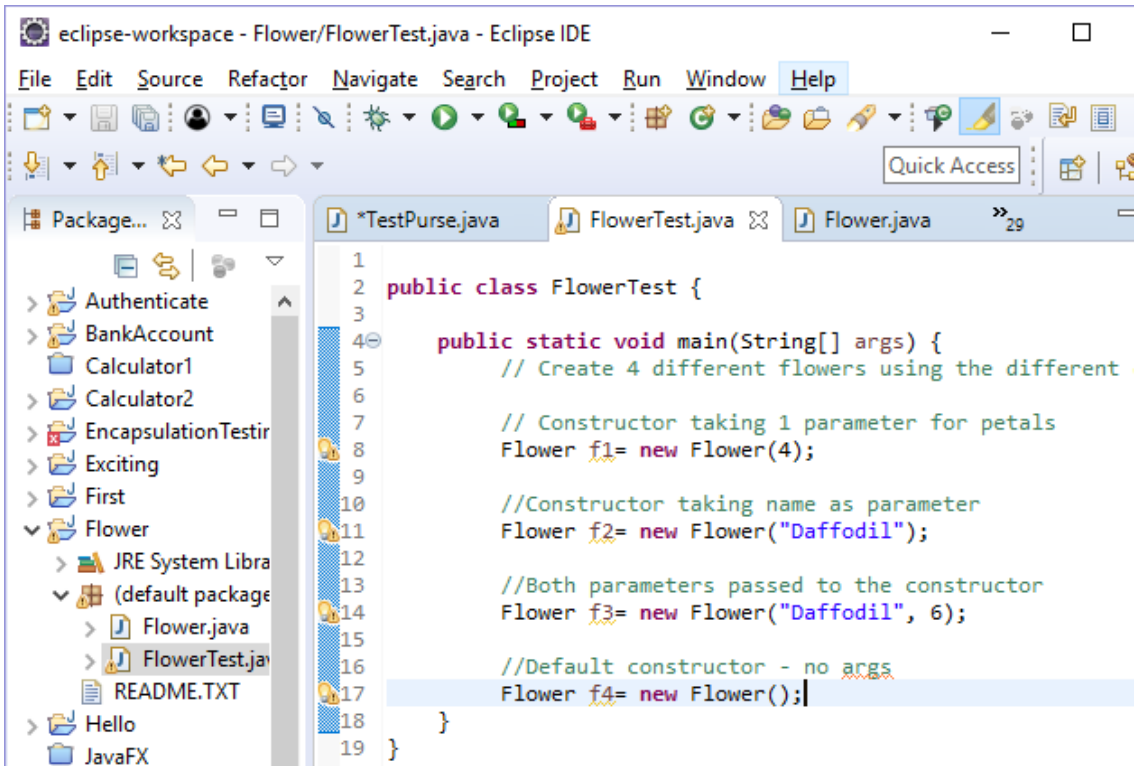
The snapshot below illustrates this.



2.3.1 - The Oblong Class

Task: Investigate *Flower* Project with its constructors. Within *FlowerTest* class, create 4 different flowers using the different constructors.

Here it is as in the snapshot below:



2.3.2 - More Class Diagram to Implementation Examples

Task: Inspect the code for the BankAccount class within BankAccount1 project, and identify where the class attributes are defined. Notice accountNumber and accountName are defined as Strings. As you know, Strings are objects in their own right. ***It is perfectly normal for the attributes of one class to be the objects of another class***.

```
// the class attributes
private String accountNumber;
private String accountName;
private double balance;
```

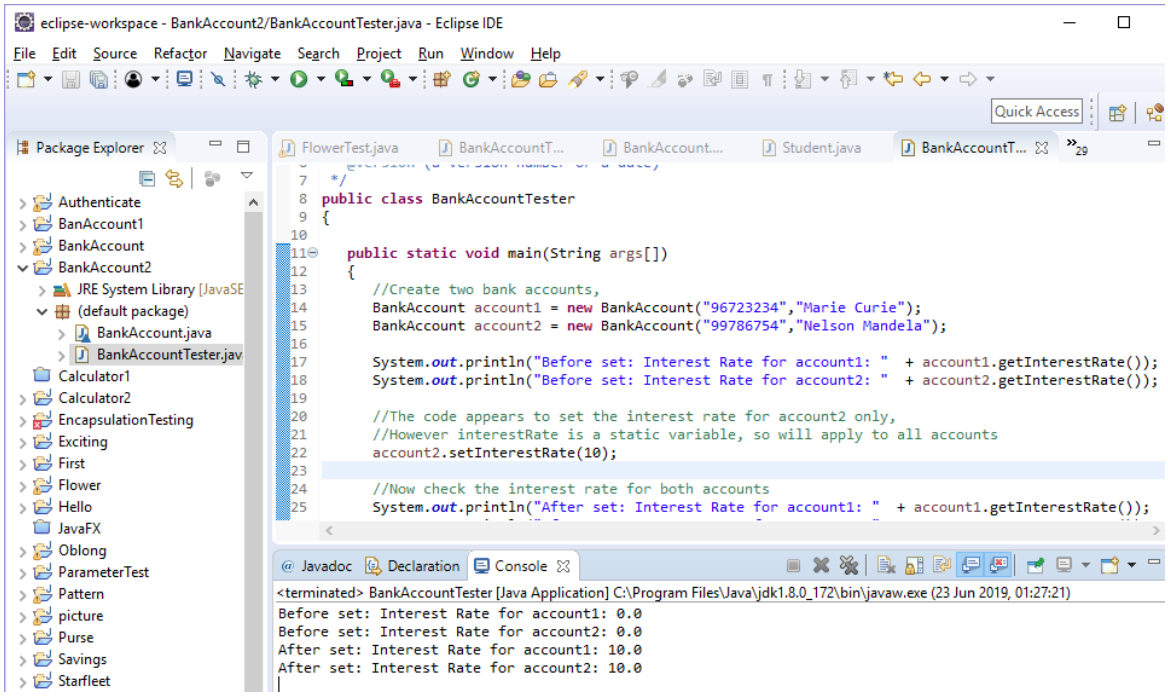
Task: View the Student Class and match the code to the class diagram.

Yes, the code exactly matches the class diagram with three attributes, two constructors, two mutator and four accessor methods.

2.3.3 - The Static Keyword

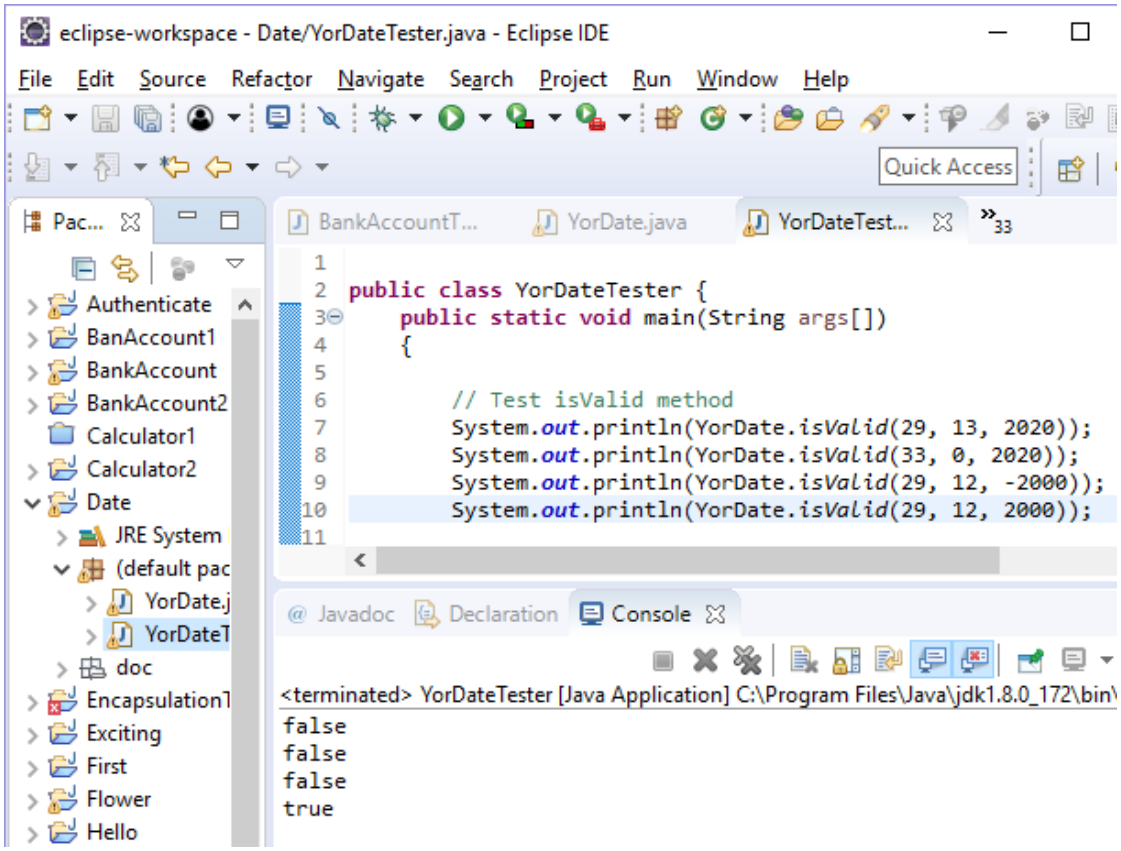
Task: Open the bankAccount2 project, view the BankaccountTester code, and run it. Notice the change of the interestRate of one object, applies to others as well.

Here it is in the snapshot below. Please note line 22 sets account2 interest to 10 and this applies to account1.



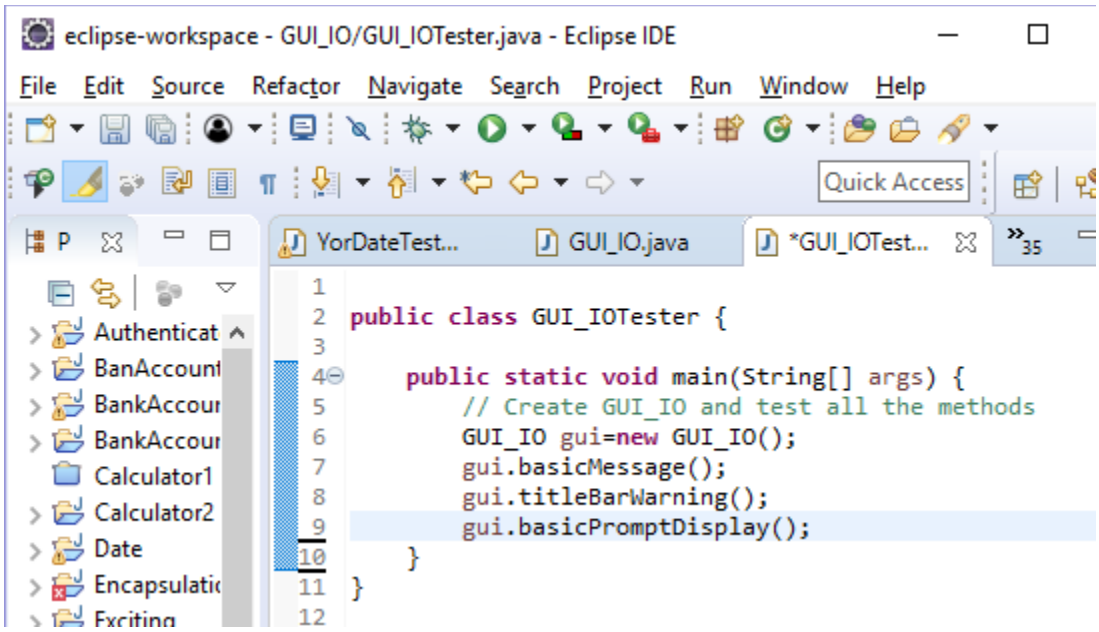
Task: Open the Date project and investigate the isValid static method in the YorDate class. Within the YorDateTester class, create and run a few more tests for the isValid method.

Here it is the snapshot below:



Task: Open the GUI_IO project, within the GUI_IOTester class and create a GUI_IO object. Test all the methods.

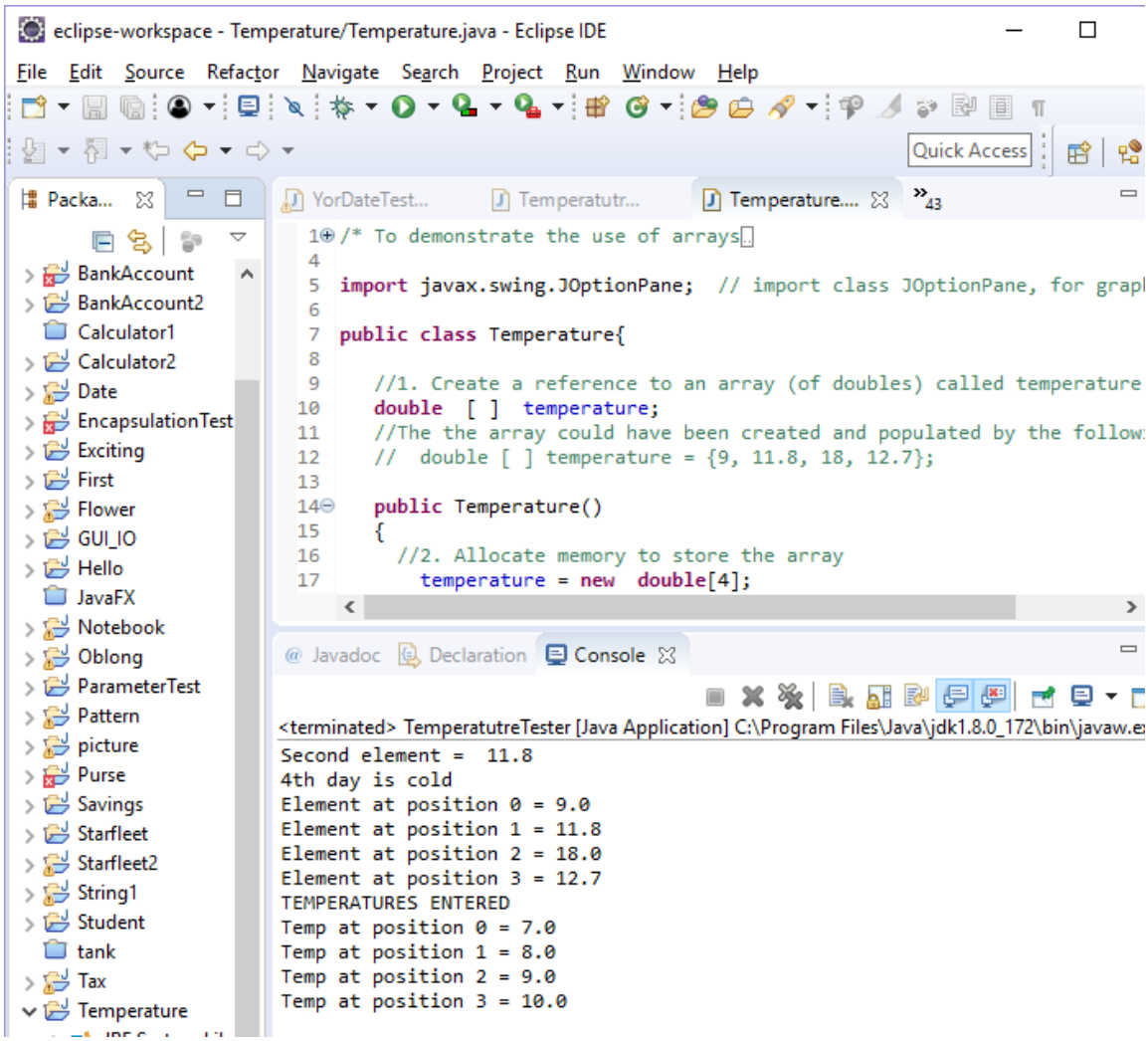
Here it is in the snapshot below:



2.4.1 - Fixed-Size Collections - Arrays

Task: Investigate the **Temperature** project, notice the various uses of arrays. Run and interact with the program, match the output with the program code.

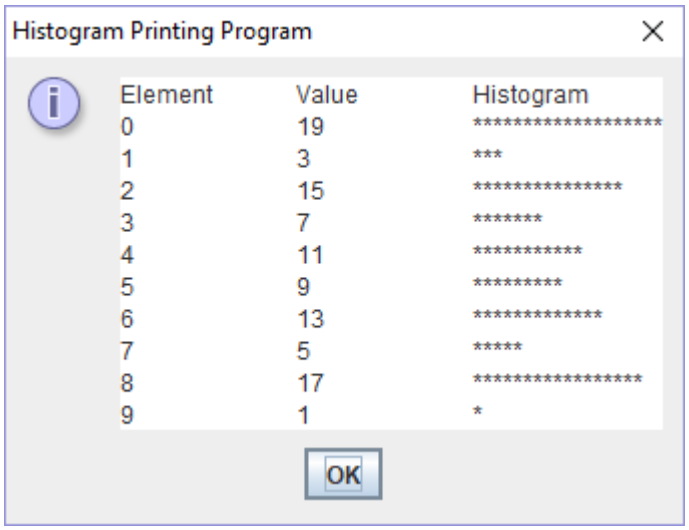
After entering four temperatures 7, 8, 9 and 10 via JOptionPane, I get this:



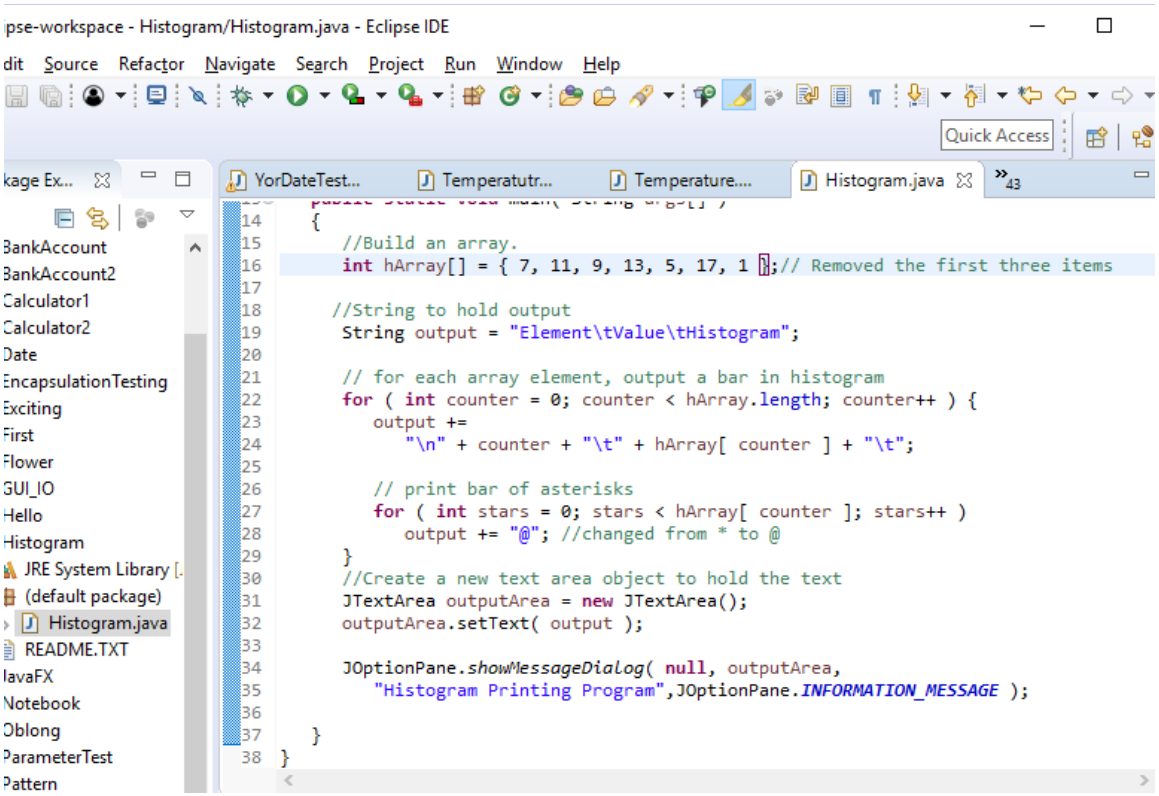
Task: Investigate the **Histogram** project.

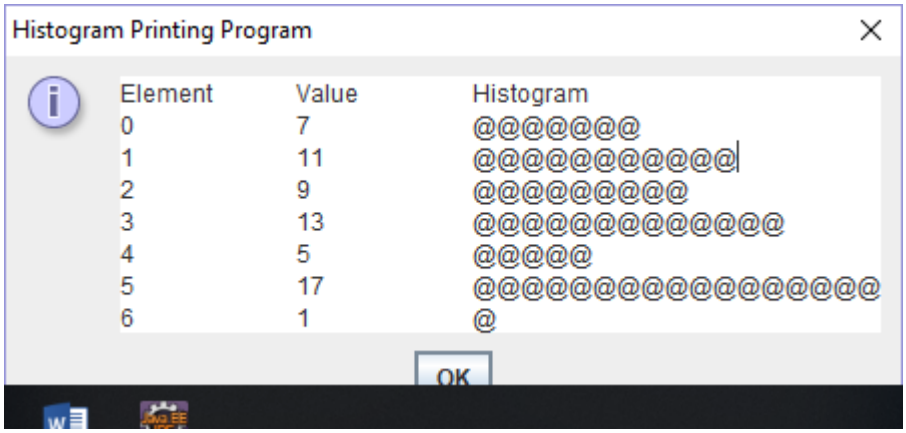
Try changing, adding or removing values in the array; for example changing the symbol from a '*' to another character.

I get this before changes:



I get the following snapshot after the changes (removed the first three element and use the character @ instead) made:




Histogram Printing Program
Element Value Histogram
0 7 @@@@@@@
1 11 @@@@@@@@@@@
2 9 @@@@@@@@@
3 13 @@@@@@@@@@@@@
4 5 @@@@@
5 17 @@@@@@@@@@@@@@@@@
6 1 @
OK

2.4.3 - Object Structures with Collections

Task: Investigate the Notebook project, store and display a few more notes.

Here it is the snapshot. Please note Wednesday is displayed first as a result of line 13.