

Java Refresher

Module 0 - Introduction to BlueJ and PLATE

After completing this module, you will have learnt the basics of BlueJ, an integrated development environment, and PLATE, the Peer Learning And Teaching Environment.

BlueJ is a development environment for developing Java applications and it is often recommended as a first development environment for new students due to its simplicity. If you are already competent in another development environment, you are free to use it.

PLATE is a teaching and learning environment developed by UTS and used by several Java programming subjects at UTS. PLATE hosts exercise material, and provides a submission link for students to submit their attempted solutions. PLATE will automatically run and test your solution against a pre-defined benchmark and produce a feedback report showing you what parts you got right and what parts you got wrong along with a mark. You are free to resubmit as many times as you wish until the due date of that assessment, and your mark will be recalculated based on your most recent submission. This approach is designed to allow you to receive immediate feedback on your work and to allow you to fix problems with your work before the due date.

The exercises below will walk you through the complete process of downloading exercise material, writing your solution, submitting it to PLATE, and interpreting PLATE's feedback report.


Exercises

Exercise 1.

Learning objective: Download and open the skeleton code for this module.

Since you reading this page, you will have already clicked on the link under EXERCISE DESCRIPTIONS:

Now, download the link to the Module0.jar file and save it in an an appropriate folder where you intend to do your work for this module.

 Module0.jar	SKELETON CODE Download the Module0.jar attachment to the left and open from within BlueJ.
	EXERCISE DESCRIPTIONS Click here to read through the exercise descriptions for Module 0.

A JAR file is a compressed ZIP file that contains Java code. A JAR file usually contains code that is compiled to binary and is ready to run, but a JAR file can also contain Java source code that is editable in a development environment such as BlueJ. This JAR file contains Java source code.

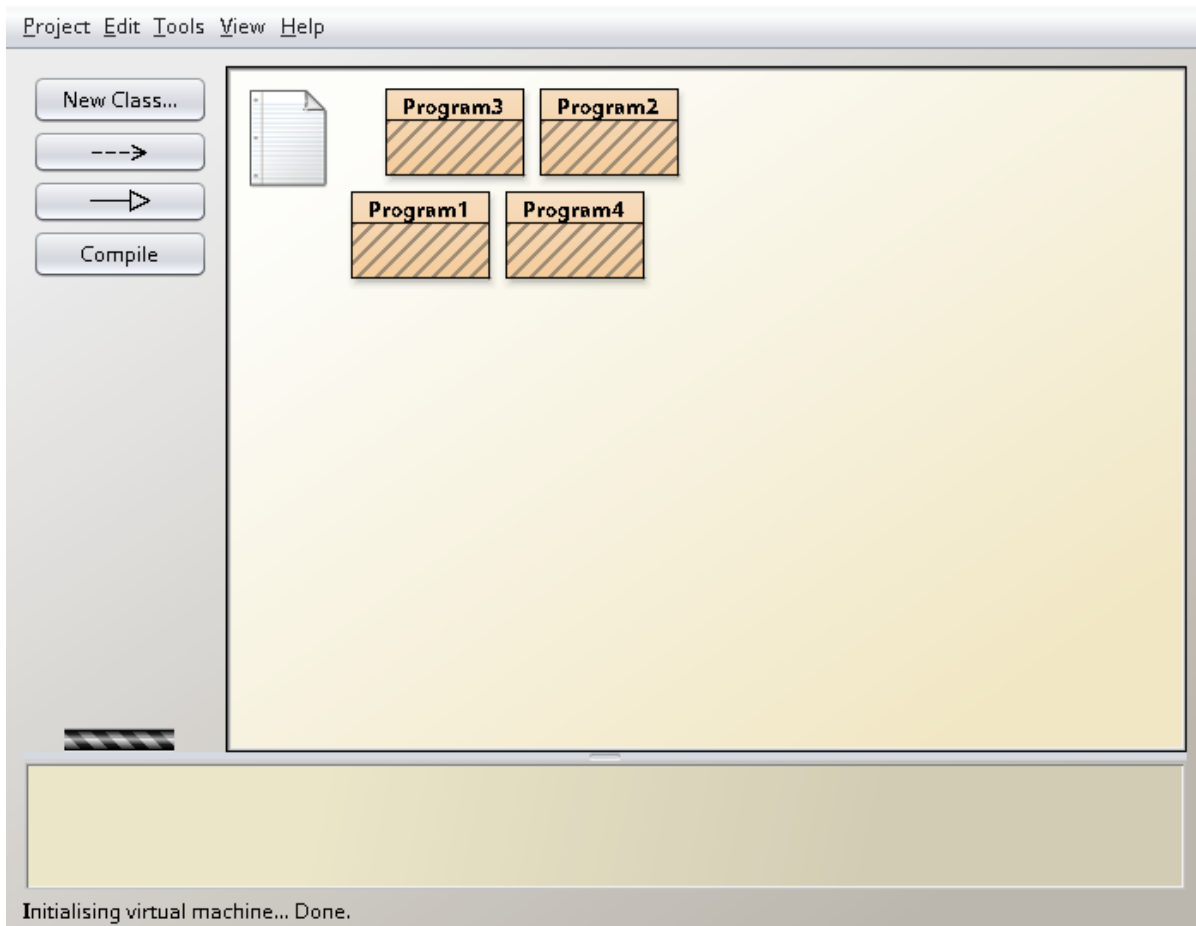
Exercise 2.

Learning objective: Open the Module0.jar file in BlueJ.

If you are using a lab machine in Building 11, BlueJ is already installed either in the Applications menu (on Linux) or in the Start menu (on Windows). It is also possible to [install BlueJ on your own personal computer](#).

Launch the BlueJ environment, and from the Project menu, select "Open Project...". Find and open the Module0.jar file. At the moment this file is opened, BlueJ will extract it into a new subfolder containing a set of Java source files that are relevant for this module. From now on, if you want to re-open your work, you should open this extracted folder from BlueJ and not the JAR file. Re-opening the JAR file will re-extract the original files and may wipe over any changes you had made.

You should now see the contents of your new project in the BlueJ project window:



Each of the 4 classes in this project are represented by labelled boxes. Currently each class is depicted with diagonally, striped lines which indicate that the classes have not yet been compiled.

Exercise 3.

Learning objective: Compile and run a program.

Before you can run a program, you must compile it into binary code. Press the compile button. If your classes compiled successfully, the diagonally, striped lines should disappear.

Now right-click on the `Program1` class and select `void main(String[] args)` from the menu that pops up. This will execute the `main` method of the program. In any program, the `main` method is always the entry point into the program and is what triggers the start of the program.

A standard main method is always defined with one parameter called `args`. In some programs, this parameter can be used to control or customise the behaviour of the program. In the next window that pops up, you have a chance to specify the `args` value. Since none of the programs we will be writing actually use this `args` parameter, you can simply ignore it and press the OK button. This will run the program, and you should see the program's output appear in a window called the "Terminal":

```
Hello world!
```

BlueJ now has two windows open: the window displaying the program's output (the "Terminal") and also the main project window from which you can open and close projects, and view all of the classes in a project.

Exercise 4.


Learning objective: Submit your project to PLATE.

From this point onwards, you will need to submit your work to PLATE. PLATE will either give you a PASS or FAIL, and in the latter case will produce a report indicating what parts of your program are correct and what parts are incorrect.

In this exercise, PLATE will assess your `Program1` class. Since this program already works as required, PLATE will give you a pass for Exercise 4.

Follow the instructions below to submit to PLATE:

1. From BlueJ's Project menu, select "Create Jar File..."
2. Select the "Include Source" checkbox. If you don't do this, PLATE will complain that the Java source files were not included.
3. Click continue
4. Select the Module0.jar file, click "Create" and accept to overwrite the previous JAR file. (If you prefer to keep the original JAR file, you can save this one under a different name, so long as you remember the name you chose)
5. Now back on the PLATE website, click on the "Your submission" link:

 Module0.jar	SKELETON CODE Download the Module0.jar attachment to the left and open from within BlueJ.
	EXERCISE DESCRIPTIONS Click here to read through the exercise descriptions for Module 0.

Your submission

6. On the next page, click "Browse", find and select the JAR file you just created, and press OK/Open.
7. To the right of the "Browse" button, there is an "Upload" button. Click it.

PLATE will now compile and run your program. It will also compare the output of your program to what was expected. If all is correct, you will receive a PASS for Exercise 4.

Exercise 5.

Learning objective: Interpret PLATE's feedback report and add a missing line of output.

Class to edit: [Program2](#)

The goal of Exercise 5 is to write a program that generates the following output:

```
This is the first line.
This is the second line.
This is the third line.
This is the fourth line.
This is the fifth line.
```

This program should be contained within the [Program2](#) class. You will have noticed when submitting your previous exercise to PLATE that you received a PASS for Exercise 4, but a FAIL for Exercise 5. That is understandable since you have done nothing for Exercise 5, yet.

When you fail an exercise, PLATE will generate a report indicating what was wrong. You will see two columns, one showing your program's output, and one showing the expected output. The expected output is referred to by PLATE as the "Benchmark program's output".

<pre>== Benchmark program's output == This is the first line. This is the second line. This is the third line. This is the fourth line. This is the fifth line.</pre>	<pre> < </pre>	<pre>== Your program's output == This is the first line. This is the second line. This is the fourth line. This is the fifth line.</pre>
--	---------------------	--

Down the middle column between your program's output and the benchmark program's output are symbols marking the lines of output that are wrong. A less-than sign or a greater-than sign will always point to an extra line of output that appears in one column but does not appear in the other. Can you see what is wrong with your program?

PLATE's feedback report tells you that your program is missing a particular line of output. You will now edit your Exercise5 program and fix this problem.

In BlueJ's project window, double-click on the [Program2](#) class. This will open BlueJ's code editor window. Find where the missing line of code should go in the program and insert it. You may find it helpful to copy and paste one of the existing lines of code and just change part of it to match the expected output.

After making your change, run your program in BlueJ and make sure it produces the correct output. Once you have done that, follow again the instructions for submitting to PLATE. That is, export your project to a JAR file **including source** then submit it to the PLATE website.

Passed? Excellent! As you progress through each exercise below, be sure to continually submit to PLATE to confirm whether your solution is correct. If your solution is not correct, let PLATE's feedback report inform you on what you need to fix. You are allowed unlimited attempts to get it right, so do not feel you must get it right on the first submission.

Exercise 6.

Learning objective: Interpret PLATE's feedback report and remove an unwanted line of output.

Class to edit: [Program3](#)

The goal of this exercise is to write a program that produces the following output:

```
One
Two
Three
Four
Five
```

This program should be contained within the [Program3](#) class. Assuming you passed the previous exercise, PLATE has already gone onto this exercise and tried to assess it. Read the generated feedback report.

<pre>== Benchmark program's output == One Two Three Four Five</pre>	<pre> </pre>	<pre>== Your program's output == One Two Three > Yay! Four Five</pre>
---	--------------	---

This time the less-than or greater-than sign is pointing towards an extra line in your program's output that shouldn't be there. Remove this line from your program and resubmit to PLATE.

Exercise 7.

Learning objective: Check spacing, punctuation and case.

Class to edit: [Program4](#)

The goal of this exercise is to write a program called [Program4](#) that prints the following output:

```
My second grade homework
I have a dog named spoty
he is the perfect dog.
THE END
```

If you examine PLATE's feedback report, you will see that rather than indicate that a line of output is missing, or that you have an extra unwanted line of output, PLATE instead simply tells you that two of your lines of output are wrong. This is indicated in the middle column by a vertical bar | symbol:

<pre>== Benchmark program's output == My second grade homework I have a dog named spoty he is the perfect dog.</pre>	<pre> </pre>	<pre>== Your program's output == My second grade homework! I have a dog named spoty He is the perfect dog</pre>
---	--------------	--

THE END

THE END

Fix the mistakes in your program and resubmit to PLATE.

Hint: check the spaces between words, the punctuation (such as full stops and commas), and whether characters should be uppercase or lowercase.