CSC8001
Programming and Data Structures

Practical session 1:
Familiarisation Exercises.

The aim of these exercises is to familiarise you with the **BlueJ** environment for Java programming. You will be able to run for yourself some of the examples used in lectures.

The learning outcomes for the session are:

- learn how to run, edit and compile Java programs using **BlueJ**
- recognise some compile-time errors
- create objects on the **BlueJ** object bench, and apply methods to those object instances, including passing parameters to methods

If you have any questions during the following exercises, please ask one of the demonstrators for help!

0. Preliminaries

These familiarisation exercises are intended as an introduction to the environment that you will be using to complete coursework for this module. Read through each exercise carefully (even if you believe that you know how to undertake the activity being described) and follow the instructions. If you need help, ask for it.

You should be able to complete all the exercises for this week during the practical. If necessary, you can use the lab computers at other times when they are not in use for another class, or use one of the Windows PC clusters which can be found in various parts of the campus.

Please note that this document cannot give complete instructions on every tool that you may need to use. There are many things you will need to find out by yourself, by trying things, using on-line help facilities, and by reading on-line and printed documentation. We assume you are familiar with:

- Logging in to a Windows PC with your university username and password
- Using a WWW browser such as Google Chrome or Internet Explorer.
- Basic Windows file system management

1. Getting Started

The first thing you should do is organise your file space. If you create everything in one place (e.g. in your Home Folder) it will be very hard for you to find things. One sensible approach would be for you to create a folder for each of the modules you are taking, and to keep all files associated with that module in that folder.

So, create a new subfolder in your Home Folder and rename it with the name of the module (**CSC8001**). All of the folders and files associated with this module should be placed in this new folder.

Since there will be a number of files you will be creating during these familiarisation exercises, create a second folder inside CSC8001 called **GettingStarted**.

1.1 CSC8001 on Blackboard

Blackboard is the learning environment where all the learning resources for your modules will be placed. Login to: Blackboard.ncl.ac.uk

Navigate to CSC8001 → Practical Material.

Click the Welcome Project and unzip into your **GettingStarted** folder. Your new Welcome folder should contain four files

- Welcome.java
- README.txt
- package.bluej
- bluej.pkg
- bluej.pkh

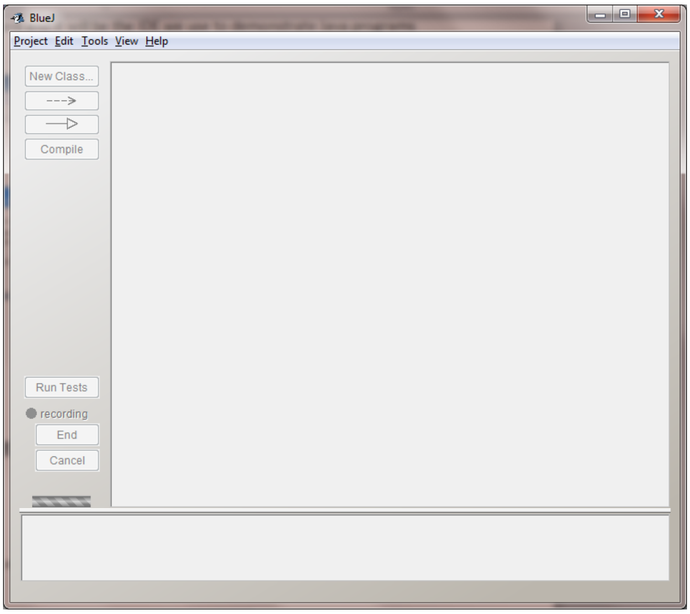
Welcome.java contains the Java source program, README.txt contains the documentation, while the other files are used internally by the Integrated Development Environment **BlueJ**. The presence of these two files within a folder identifies that folder as containing a **BlueJ** project.

1.2 Launching the Integrated Development Environment (IDE) and JAVA

The language we will use to learn the principles of object oriented programming is **Java**. The two main IDEs available to run Java are **BlueJ**, and **Eclipse**. We will concentrate on BlueJ in CSC8001, but you will move on to use Eclipse in other modules.

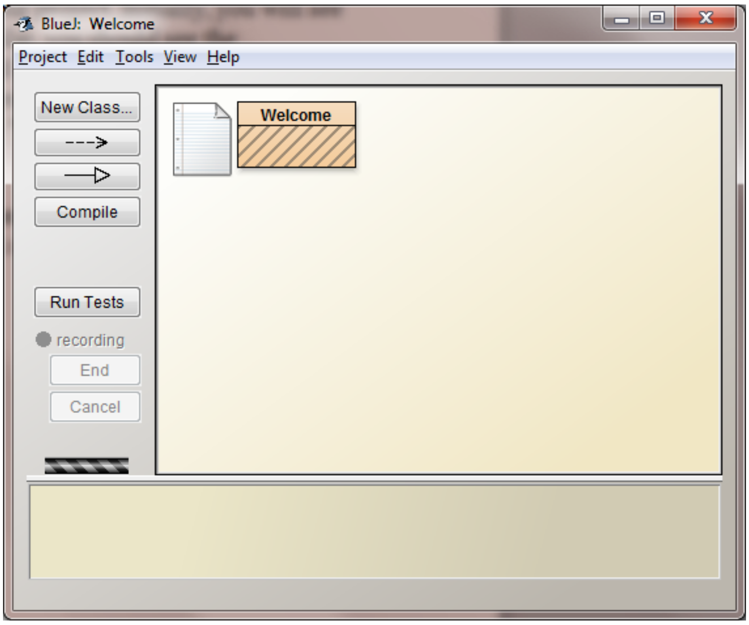
To launch BlueJ, open the Start menu by clicking on the Windows logo in the bottom left corner, and then select Editors and IDEs from the menu. You will find BlueJ listed here. Alternatively, after opening the Start menu simply type **BlueJ** and click on the **BlueJ App** which appears in the search results.

Once BlueJ has started, you should have this window in front of you:



Select the Project menu and then the Open Project item. This will open a File Dialog to allow you to navigate around to find your project. Initially, you will see all of the folders in your Home Folder. In particular, you should see the folder **CSC8001**. Double click on this folder to find the Welcome folder, which you downloaded from Blackboard in the previous exercise. Although Windows regards it as an ordinary folder, **BlueJ** recognises the two "bluej" files in the folder, and hence regards the folder as a **BlueJ** project.

Select the Welcome icon in the **BlueJ** window and click the Open button. The project window will now change, and it will look something like this:



1. The icon in the top left hand corner represents the document describing the project (README.txt in the folder). You can open it by double clicking on it. Do this now, and add a comment that the program was downloaded by you on this particular date.
2. The other icon represents the class file containing the Java source code for the Welcome program. Notice that at present it is coloured orange, and has black diagonal lines across it.
3. You now need to *compile* the class. Press the Compile button in the Project window. This will cause **BlueJ** to call the JDK compiler to compile all of the Java source files for the project (only one in this case). The compilation should be successful, and the class icon in the project window will change to remove the black diagonal bars.
4. To run the program, move the cursor over the class window and *right-click*. A menu will appear giving the opportunity to create a new Welcome **object**. Do this now, using the default name (something like welcome1) for the new object. When the object appears (i.e. the "red blob") on the *object bench*, right-click over the object, and select the **void greet()** method. The method will then run, and create a new output window with a text message printed in it.

Note that this window does not disappear when the program terminates - you have to close it explicitly (by clicking on the cross in the top right hand corner) when you no longer need it.

Congratulations! You've just compiled and run your first Java program.

2. Editing a Java Program

Here you are going to change the Java program to personalise it.

1. Double-click on the **Welcome** icon in the **BlueJ** project window (or right-click on it, and select Open Editor). This will bring up a window in which the Java source code of the program is displayed.

Find the point in the program where the text is printed out, and alter it to print out some other message, preferably containing some personal information such as your name. You might do this by editing one of the existing statements, or you might add another printing statement by copying the format of those already there. Notice that as soon as you make any alteration to the source text, the icon in the project window will show the black diagonal lines again. Be careful just to change the text within the double quote symbols (").

2. Re-compile the program, and run it as before. The output window will open up again and should display a message containing your alterations. (The output window also has a **Clear** menu item to delete its current contents without closing it.) If you made a mistake in editing the program, the compilation will fail. If the compilation fails, the source window will be opened (if it wasn't already opened), an error message will appear at the bottom of the source window, and the offending line in the source code itself will be highlighted.

3. Compile Errors

A programming language such as Java has strict syntax rules that must be followed for a program to be successfully compiled. If you make a slip, and introduce a syntax error into your program, the JDK compiler will produce a compile-time error message. The purpose of this exercise is to introduce you to this process by you carefully introducing syntax errors into an otherwise valid program and seeing what compile-time errors result.

In the steps that follow you will be changing the source code. Sometimes you may change more than you anticipated, and you can use the editor's undo function to recover.

After each of the statement changes discussed below you should return the Java code to its original (valid) state.

1. If you need to, obtain a new copy of the Welcome project and open the project using **BlueJ**.
2. Open the class file **Welcome** showing in the project window, and if necessary scroll down to find the greet method:

```
public void greet()
```

3. Go to one of the lines that include a string of text enclosed in double quote characters (") - delete ONE of the double quote characters, and note what happens visually in the window. The editor itself knows a bit about Java, because if you delete the final double quote character ("), some of the characters in the text will

change colour. However, try compiling the source code (click the **Compile** button in the editor window) with one double quote missing (this is a standard typing mistake to make). Note how the syntax error is reported to you. Put the line back to its original state.

4. Try deleting the semicolon at the end of a line and compiling.
5. Try deleting one of the braces ({ or }) and compiling.
6. Try changing the spelling of some of the other "words" in the program.
7. Try making two syntax errors in the code and compiling. How many of those errors does the compiler report? It is important to note that the compiler may only report the first error and then stop.

4. The Shapes Project

You have seen the *shapes* project in lectures, and we have also seen how the various shapes can be re-sized, moved and had their colours changed to create a picture. Now is your chance to experiment with it yourself.

The shapes code can be retrieved by downloading from Blackboard. You should create a folder with a suitable name to extract the files into.

If you open this project in **BlueJ**, you will find that there are four source files (i.e. four classes) in the project window; `Circle`, `Square`, `Triangle`, and `Canvas`. Each class may need to be compiled, so click the `Compile` button. You will see that the classes are compiled one by one.

Ignoring the `Canvas` class for the time being, create some instances of the other classes on the object bench, and manipulate those objects. In particular, see if you can create the house picture seen in the lectures.

You will notice that the menu for each object (of whatever shape) includes a `slowMoveVertical` method. Try this method out, and see if you can make the sun "set".

Can you create another picture simply using combinations of these three shapes?

5. The Picture Project

Again, the *picture* project has been seen in the lectures. You can retrieve the code of the project from blackboard.

Open the Editor window to see the source of the Picture class. Find where the sun is given the colour *yellow*, and change it so that the sun is some other colour (say *red*) when the Picture object is created.

Modify the program so that the picture has two suns, one yellow and one red at different positions in the sky.

Based on your experience with invoking the slowMoveVertical method in the previous exercise, insert some code into the (single sun version of the) Picture class to cause the sun to set after the picture has been drawn.

It would be nice if there were a method that made the sun set as a separate operation. Can you see any problems with trying to implement such a method? Can you see any way of modifying the class to achieve this behaviour? If so, implement these changes.

The README file associated with each project is a good place to record any observations and changes you may make, just to keep track of what you did in each project, and your reactions when you did it.

A permanent record of what you have done should be kept, so you should make sure that everything you do is recorded in the README file for each project. It is also a good idea to make a copy of each class before you change it, so that if things go wrong with your modifications, you can always go back to a previous, working, version and start again.

Finally, when you have finished with **BlueJ**, you should end either by selecting the Quit item from the Project menu, or close down the **BlueJ** project window by clicking on the cross at the right-hand end of the window's title bar.

For any questions, ask a demonstrator during the practical sessions, email steve.riddle@ncl.ac.uk or use the Padlet page referred to in lectures.