School of Computing and Mathematics
CSC-10024
Practical 1 (Week 2)

Practical 1: Spirograph - Accessing module content and submitting work via the Keele Learning Environment (KLE)

In order to complete this practical YOU MUST bring with you the USER NAME and PASSWORD allocated to you during registration. Remember to bring the ORIGINAL password allocated to you, even if you have since changed it!

Task 1

As a first task please logon to one of the terminals in the laboratory using the user name and password that was allocated to you during registration. If you have the opportunity, at some other time, check out your user name and password at a terminal in the other Computer Science labs or any other available terminal on campus (eg. in the Library).

Task 2

It is very important that you know how to send and receive e-mail using your university e-mail account. E-mail is the main method the university will be using to communicate with you in cases for example of cancelled lectures, library issues, timetable changes etc.

If you have not **sent and received** at least one e-mail message using your Keele account so far, or if you are still unsure whether you know how to do this please ask for help from one of the demonstrators.

Task 3

The University uses a Virtual Learning Environment (VLE) called Blackboard Learning System (also called KLE – Keele Learning System) to deliver support for module teaching. You will be able to access lecture handouts, practical sheets, tutorial sheets and other module supporting material via the KLE. You can access the KLE by logging in at the following web address:

```
    http://kle.keele.ac.uk
```

If you have logged in successfully you should see (under the Courses tab) a list of the modules (Blackboard calls them Courses) that you are currently enrolled on. Click on the link for this module:

Programming I - Programming Fundamentals [CSC-10024-2022-SEM1-A] and explore the **Learning Content** page.

You should be able to see at three folders in the Course Content area: **Lectures**, **Practicals** and **Tutorials**. You can examine the contents of each folder by clicking on it in order to look inside. You can exit the folder by clicking on ← (go back symbol) at the left top of the browser/screen.

The Lectures folder

In this folder you will find the lecture slides and any other material used by your lecturer during lectures (for example Java program code). Content will be added to this folder shortly before or right after it has been delivered. You should therefore be able to see the content of Lecture 1 in this folder.

The Practicals folder

In here you will find the practical sheets/handouts and any associated material. These will be released just before each practical session. You are expected to have revised this module up to and including the last lecture before each practical before coming to the practical session.

The Tutorials folder

This folder will contain tutorial handouts and any associated material. Tutorials will be released approximately one week before the session in which they will be discussed. You will be expected to work on your practicals during that week (prior to the session) and come to the tutorial session with a solution to discuss with your appointed tutor.

Task 4

As mentioned in the first lecture, your mark for this module will be deduced from two multiple-choice lab tasks and the work that you will submit for the last **two** practicals. This task is meant help you understand the mechanism by which you will be submitting your work for each of the assessed practicals.

A practical that requires you to submit work on the KLE is called an **Assignment** and it can be distinguished from other items by its icon that looks like this:

📄

Assignments will always have the practical sheet attached to them and may also include some start-up code in terms of a *compressed/zipped* NetBeans[1] project folder.

[1] You will be briefly introduced to the Apache NetBeans IDE below and more properly very soon in your lectures. For now just remember that it is an application (written in Java) that provides a convenient environment for developing Java applications.

Your first action should always be to read the practical sheet. This will explain what you need to do and what you will eventually have to submit for assessment.

An assignment called **Practical 1** has been setup for you. As explained above, the assignment contains this practical sheet (called `practical 1 - Spirograph.pdf`) and a zipped NetBeans project folder (called `spirograph.zip`).

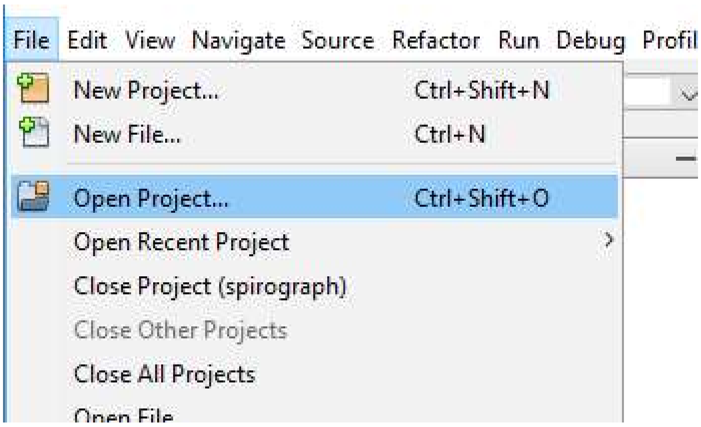
Please also note that this is not an assessed practical (i.e. it will not contribute in any way to your module mark.) even though it has been setup as an assignment. It is only meant to illustrate how you can submit your work on the KLE.

Please access `practical 1 - Spirograph.pdf` and continue reading from that if you are not doing that right now.

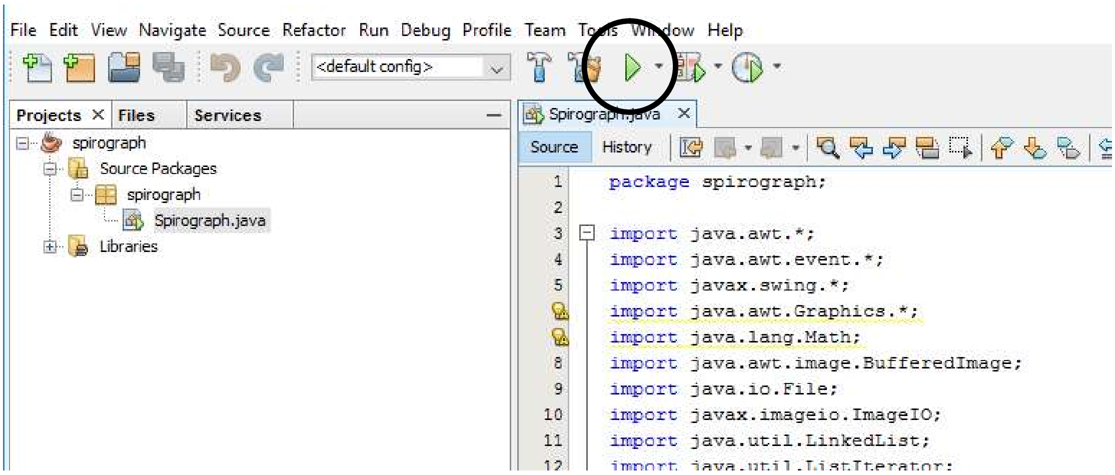
The Spirograph application

Download and unzip the `spirograph.zip` NetBeans project. Please ask a demonstrator to help you if you don't know how to do this.

Run the Apache NetBeans IDE application and open the spirograph project:


File Edit View Navigate Source Refactor Run Debug Profil
New Project... Ctrl+Shift+N
New File... Ctrl+N
Open Project... Ctrl+Shift+O
Open Recent Project
Close Project (spirograph)
Close Other Projects
Close All Projects
Open File

Run the spirograph program/application/project by pressing the green triangle on the toolbar at the top. You can achieve the same thing by pressing F6 (keyboard shortcut).



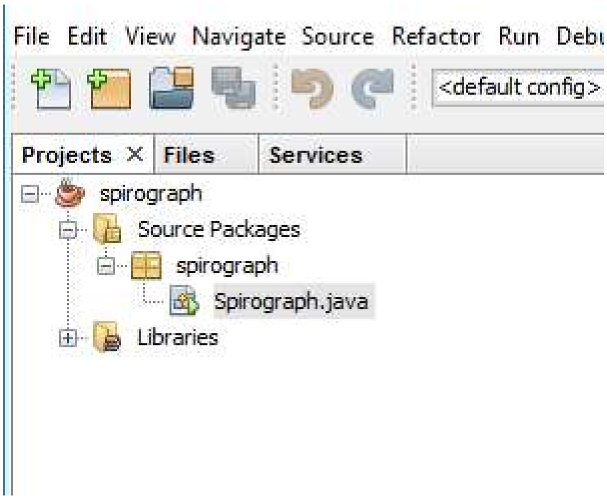
```
package spirograph;

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.awt.Graphics.*;
import java.lang.Math;
import java.awt.image.BufferedImage;
import java.io.File;
import javax.imageio.ImageIO;
import java.util.LinkedList;
import java.util.ListIterator;
```

The application simulates a type of *spirograph*[2]. Once the drawing is complete (i.e. the pen comes to rest in the middle of the drawing area) the result will be saved in an image file called `image.gif`. Verify that you can access this image file in the `spirograph` folder.

Your task is to produce and submit on the KLE your own spirograph trace/drawing. You can do this by changing (in the Java program) those *parameters*/values that govern how the spirograph pen moves.

The *program code* of the `Spirograph` *class* should be open for you in the editor panel in NetBeans. If not, then expand the folder structure of the spirograph project as shown in the figure below and double click on it.



There are approximately 200 lines of code in the `Spirograph` class file and **at this point you are <u>not</u> expected to be able to understand any of it**.

[2] A typical Spirograph is a drawing toy that produces interesting curves by, for example, constraining a pen at a point on a circle (or ellipsis) that moves in (or out of) another circle. This is usually achieved using gears of different sizes that move within each other and contain holes for placing the pen. You can find more about spirographs by looking the term up on the web.

Scroll to the point in the program that looks like:

```
////////////////////////////////////////////////////////////////////
// CHANGE VALUES IN THIS AREA
////////////////////////////////////////////////////////////////////

// Pen-oscillation frequencies in the x (horizontal direction)
// Use values between 0.0 and 3.0
double xf1 = 1.5;
double xf2 = 0.3;
// Pen-oscillation frequencies in the y (vertical direction)
// Use values between 0.0 and 3.0
double yf1 = 1.3;
double yf2 = 0.1;

// The rate of energy loss of the pen due to friction.
// Use values between 1 and 50.
// The smaller the value the longer it will take for the drawing to
// complete.
double dissipation = 10.0;  // Pixels per second.

// The starting colour.
// Use values between 0 and 255.
int red = 0;
int green = 127;
int blue = 255;

// Colour-channel increments. These determine how fast each colour channel
// changes as the pen moves.
// Use values between 0 and 100.
int red_inc = 3;
int green_inc = 2;
int blue_inc = 1;

////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
```

The first four *variables* called `xf1`, `xf2`, `yf1` and `yf2` principally determine the shape of the curve that results on the drawing surface. Even a small change to the values of these variables can result in a drastically different curve!

The variable called `dissipation` controls how quickly the pen loses its energy and ends up in the middle (resting position). The larger this value is, the faster the energy loss of the pen.

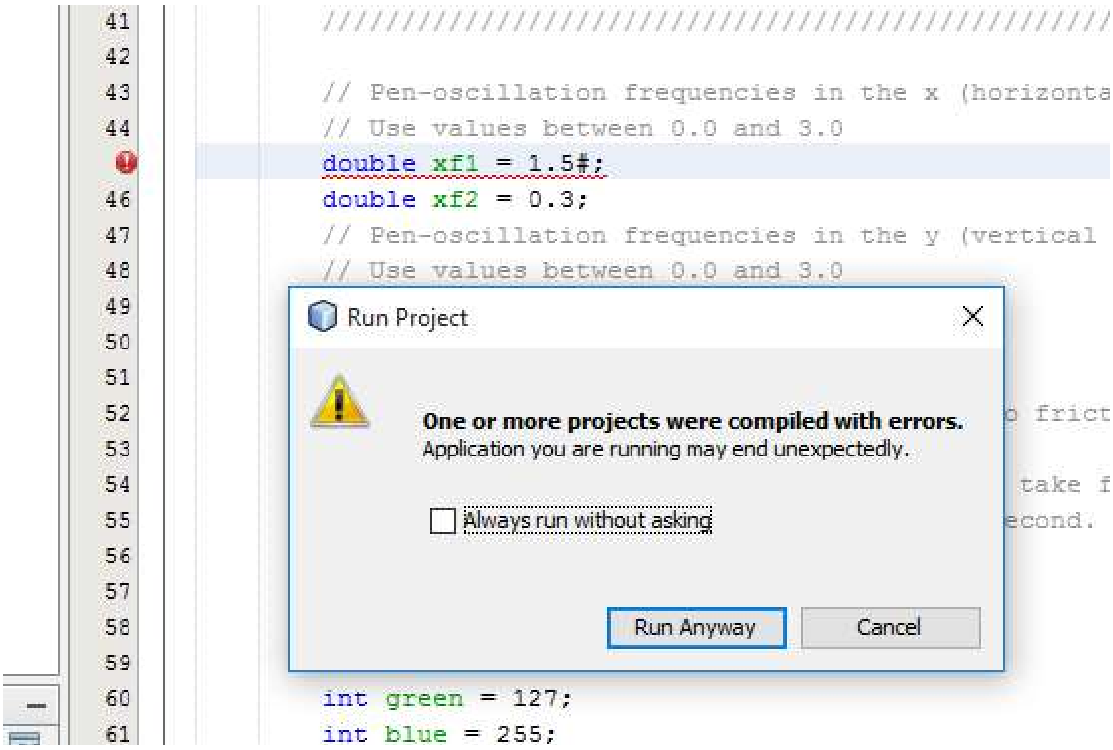
Finally, the *integers* `red`, `green` and `blue` define the starting colour of the curve and `red_inc`, `green_inc` and `blue_inc` (also integers/whole numbers) determine how quickly each *colour channel* changes with time as the curve is being drawn.

Experiment by making (small) changes to the above values in order to produce a curve of your own liking and then submit the resulting image to this assignment portal.

Start by changing one of the above variables and re-run the program to see what effect your change produces. Make sure that you only change the number between the equals sign (=) and the semicolon (`;`).

Run the application each time you make a change to see the effect of your change on the resulting trace. In fact, if you have made a change you program will be *compiled* before it is run. More on this later…

If there is a problem after you have made a change, for example if you have violated any Java syntax rules you will be notified when you attempt to run it. The offending syntax will also be underlined in red in the editor as you type. See example below:



```
//////////////////////////////////////////////////

// Pen-oscillation frequencies in the x (horizonta
// Use values between 0.0 and 3.0
double xf1 = 1.5#;
double xf2 = 0.3;
// Pen-oscillation frequencies in the y (vertical
// Use values between 0.0 and 3.0
```

Run Project

One or more projects were compiled with errors.
Application you are running may end unexpectedly.

□ Always run without asking

```
int green = 127;
int blue = 255;
```

In this case, click Cancel and try to fix the problem before running the application again. As a demonstrator to help you if you have lost track of what is happening or you are unsure of what you are meant to be doing.

Please note that at this point you are not expected to understand the Java program given to you. The purpose of this exercise is to give you a brief and early introduction to the NetBeans environment, to show you what a Java program looks like and to get you to submit something to the KLE.

The most important objective however is to enjoy the spirograph!