## Lab 1 : Hello, World! and the Terminal

**Topics covered** Compiling a Java Program, terminal

**Aims** Learn to use the terminal (a.k.a. command-line) for some simple commands, including compiling and running a program in Java

**Practice in** lab skills, using the terminal

**Exercise 1** (approx. 15 minutes): Before you write a Hello from the computer to the world, you should introduce yourself to the class. The tutor will start things off: when it comes to your turn, say who you are, and something *interesting* about yourself. If you think that being a ninja crime fighter in your spare time is interesting, then do please share: just don't expect *everyone* to believe you...

**Exercise 2** (approx. 5 minutes): Make yourself aquainted with the `Ed`. `Ed` is where we will provide your lecture and tutorial material, as well as your first port of call for any questions you may have about the course.

If you are stuck on any material, whether it be lecture, lab or assessment, have a look to see if anyone else is discussing it on `Ed`, and if not then ask a question! Just remember, do not post any code for assessable work on `Ed`.

Take a moment to reply to John's welcome post with a "hello" of your own, just to make sure you know how to use the system.

**Exercise 3** (approx. 10 minutes): Reboot your lab computer to Linux (Fedora). Log in using your unikey login details and start up a terminal window using Ctrl + Alt + T .

All of your work should be saved in your home directory (the default directory that opens when you open the terminal) to ensure that the work can be accessed from any of the School of IT computers. Use the below table of commands to create a new directory (a.k.a. folder) for your work in your home directory, and navigate into that directory.

| Command | Effect |
|---|---|
| `ls` | Display all directories and files in the current directory. |
| `cd` *directoryName* | "**c**hange **d**irectory" i.e. move into the directory *directoryName*. |
| `cd ..` | Go back up one directory level. |
| `cd ~` | Go back to your home directory. |
| `mkdir` *directoryName* | Make a new directory called *directoryName* (in the current location). |

For example, you may create a directory structure like this. Try running the command `ls` (or `ls -l` for more information) after each command to see the changes you are making:

```
~> mkdir info1103

~> cd info1103

~/info1103> mkdir lab01

~/info1103> cd lab01

~/info1103/lab01>
```

You should create a new folder at the beginning of each lab so that all your work is organised.

**Exercise 4** (approx. 15 minutes): Write out the "Hello, World!" program in a command-line editor `vi`(There are other choices, e.g. `emacs`).

In the directory you just created, run the command `vi` to open a blank file with the name "HelloWorld.java". Be aware that you *must* name your file "`HelloWorld.java`".

```
~\info1103\lab01\> vi HelloWorld.java
```

The editor uses two modes, Insert mode and Command mode. You'll do your actual text editing in Insert mode, while you'll use Command mode to copy and paste, navigate, save, quit, and execute other commands. Press "i" to enter Insert mode and start typing. Here's the code:

```
1 public class HelloWorld {
2    public static void main(String[] args) {
3       System.out.println("Hello, World!");
4    }
5 }
```

Notice how everything is nicely laid out? You should do that. Don't just copy and paste this – typing is something you might need to practice!

When you're finished editing your text file, press Esc to enter Command mode. To save your changes, type `:w`. To quit the editor and return to the Terminal, type `:q`. To save and quit in one command, type `:wq`. To quit without saving your changes, type `:q!`.

You should see the created java file by running the command `ls`. Compile your java file with the `javac` command. If there are any errors, see if you can work out what they mean and fix them.

```
~\info1103\lab01\> javac HelloWorld.java
```

Run the program by entering

```
~\info1103\lab01\> java HelloWorld
```

**Exercise 5** (approx. 2 minutes): Find out what version of Java is installed on your current machine. Do this by typing `java -version`.

Fill in the Java version here: _____

**Exercise 6** (approx. 15 minutes): Create a program called `Box.java` to print out characters to draw a box on the screen.You can use a graphical text editor from now if it's an easier way for you. `Pluma` (a fork of Gedit) is the default editor on lab's computers. But you can try other editors on your own computer, such as `Atom`, `Sublime`, `Notepad++` etc.

Run the command `pluma` to open the pluma editor, or alternatively run `pluma Box.java` to open pluma and create a blank file named "Box.java" in one command.

```
~\info1103\lab01\> pluma Box.java
```

```
> javac Box.java
> java Box
+--------+
|        |
|        |
|        |
+--------+
```

**Exercise 7** (approx. 25 minutes): Take a look at the sample Java program on page 49 of the textbook. Here it is in case you don't have the book yet:

**Part 1** Copy the program shown in Listing 1.1 from page 49.

```
import java.util.Scanner;

public class FirstProgram {

   public static void main(String[] args) {
      System.out.println("Hello");
      System.out.println("I will add two numbers for you");
      System.out.println("Enter two whole numbers on one line");
      int n1, n2; // <-- this is saying we'll have two integers
      Scanner keyboard = new Scanner(System.in);
         // The statement above lets us read input from the keyboard.
      n1 = keyboard.nextInt();   // reads the first number
      n2 = keyboard.nextInt();   // reads the next number
      System.out.println("The sum of your numbers is:");
      System.out.println(n1+n2);
      keyboard.close();
   }

}
```

Compile the program using the `javac` command. You might have to change some things to make it compile correctly without any errors.

**Part 2** Modify your program so it prints out the sum of three numbers. Compile and run it as before.

**Part 3** Write a complete program that works out in which year a person's $n$th birthday will occur, given the year of their birth.

---

**Extensions**

*These are optional!*

If you finish all work up until this point, you can try the extension exercises below.
Check with your tutor before you continue though.

---

**Extension 1:** Change your Box program to print out a whole load of boxes. And triangles. Maybe a circle. Use the circle to calculate the value of $\pi$ (actually this is harder, so unless you're feeling adventurous you could skip this). Go wild: print your name in outlined letters. For example:

```
> java PrintName
+--+    +--+ +--+ +--+  +---+ +--------+
|   \  /   | |  | |  | /   /  |        |
|    \/    | |  | |  |/   /   |  +-----+
|          | |  | |      /    |  |
|          | |  | |     \     |  +--+
|  |\  /|  | |  | |      \    |  +--+
|  | -- |  | |  | |  |\   \   |  |
|  |    |  | |  | |  | \   \  |  +-----+
|  |    |  | |  | |  |  \   \ |        |
+--+    +--+ +--+ +--+   +--+ +--------+
```

Actually you can draw anything you like.

**Note:** If you want to print a backslash: `\` you need to do put *two* backslashes in a row else you'll get strange results.

**Extension 2:** Get your program to use the input `String [] args` that are required in the `main` method, to print a useful message about the input. You'll have to

- learn how to *input* the `args` variable (pass them to the HelloWorld program)
- learn how to *access* the `args` variable

**Extension 3:** If you run out of work to do during the lab, each chapter of the textbook has a Graphics Supplement, which will teach you how to draw graphical displays for your programs. This content is not assessable, however it is interesting for anyone looking for more challenging work.

The first Graphics Supplement begins on page 64 of the textbook.