TUTORIAL 1

Aim:

Introducing you to some basic concepts in R, within R itself. So you will learn R, while working in R. We will do this using a package called `swirl`.

Mode of study:

You can work through this tutorial in your own time, or in one of the assigned venues during the tutorial slot. Help is available during the tutorial slot in the venues.

Before you start:

If you have not worked through Practical 1 yet, you should do so before starting this tutorial.

Tip:

I strongly recommend making notes as you work through this tutorial. You will learn many R commands as you progress through the `swirl` lessons, and if at a later stage you can't remember how to do something, it will be easier to refer back to your notes than having to go through the `swirl` lessons again!

PRELIMINARIES

1. **Open RStudio.**

   In Practical 1 you typed R code directly in the R console. For instance, when typing `2 + 2` in the console, you got the answer, `4`.

   When you closed RStudio after working on Practical 1, you would have been asked if you wanted to save the workspace.

   If you chose No, none of the objects you created, or a history of the commands you entered, will be available now – which is obviously not ideal.

   If you chose Yes, you would see that the object `result` (which you created) is still available, and if you click on the History tab you will also see a list of all the code you typed during that session. However, if you now type new code – perhaps for a different practical – and create new objects and save again upon exiting RStudio, it will all be saved in the same place. This means that very soon, you will have a very cluttered RStudio workspace. It is therefore good practice to use *scripts* or *projects*.

USING SCRIPTS IN R

2. You can enter all of your R commands in a *script* and then save the script. This means that you will always have a complete record of what you did, and it also allows for reproducible workflow.

A script is simply a text file that holds R code. R scripts have the extension .r or .R.

You can create a new script file in RStudio by choosing

File → New File → R Script.

This will open a window on the upper left of RStudio, where you can type your code.

I have uploaded a script called Practical 1.R onto SUNLearn, which contains some of the code from Practical 1.

2.1 Open the script file Practical 1.R **in RStudio.**

You can run lines of code one at a time by pressing Ctrl + Enter while the cursor is positioned in the line, or by selecting more than one line of code and then pressing Ctrl + Enter. Instead of Ctrl + Enter you can also click the Run button at the top of the scripts pane. To run the entire script you click the drop-down next to the Source button and select Source with Echo, or alternatively press Ctrl + Shift + Enter.

2.2 Practice doing this by running a few lines of code from the Practical 1.R **script.**

USING PROJECTS IN R

3. Rather than using scripts, I prefer to use separate *projects* within RStudio. Managing paths and files are a lot easier within a project. You can also use scripts within projects. In fact, in a project you can keep all files associated with a project (such as input data, scripts, results and graphs) together.

You create a new project in RStudio by clicking File → New File and then deciding whether you want to create the project in a new directory or an existing directory. I strongly recommend starting a project in a new directory for every practical we do. You can then save data files (when required) in the same directory, as well as any output files you may create. When a new project is created in RStudio, a file with the extension .Rproj will be created in the specified directory. When you want to work on the project again, you use the Open Project command.

3.1 Create a new project called Tutorial 1 **in a new directory also called** Tutorial 1.

THE `swirl` PACKAGE

4. In Practical 1, you were introduced to packages in R.

You should now install (only if using your own computer) and load the package `swirl` (using any one of the two methods discussed in Practical 1).

Remember that installing the package does not automatically load it as well; you will have to do so separately.

---

5. The best way to learn R, is by doing! It therefore makes sense to learn R within R itself. The `swirl` package teaches you're the basics of R programming in an interactive way, within R.

5.1 **Launch `swirl` by typing `swirl()` at the command prompt in the console.** (No need to use a script for this tutorial.)

5.2 Once you've launched `swirl` (and presuming this is the first time you have launched it), you will be prompted to go through a few preliminaries. Just **follow the on-screen instructions. Take special note of the commands required to exit `swirl`, to return**

to the main menu, or to experiment in R on your own without feedback from `swirl`.

5.3 There are different courses available within `swirl`.

When prompted to choose a course**, you should select `1: R Programming`**.

---

6. There are different lessons within this course. You are welcome to go through all of the lessons; however, **at a minimum you should complete the following lessons this week**:

```
1: Basic Building Blocks
2: Workspace and Files
3: Sequences of Numbers
4: Vectors
5: Missing Values
6: Subsetting Vectors
7: Matrices and Data Frames
8: Logic
```

---

7. When completing a lesson, you will be asked whether you want to receive credit on Coursera.org for completing the lesson. **You can select No**.

After completing these lessons, you should be able to do the following in R:

* Understand how to perform basic arithmetic and logical operations

* Understand how working directories operate, and perform some basic operations on files directly

* Create number sequences

* Work with vectors

* Understand missing values

* Work with matrices and data frames

* Work with logical operators

You can now attempt Quiz 1 on SUNLearn.