823G5 Programming in Python

Lab 1: Basic coding and using PyCharm

Aim

To familiarise yourself with PyCharm and to get you used to:

- The code / execution workflow.
- Using variables.
- Performing basic mathematical operations.
- Using simple input and output functions.
- Creating a simple `while` loop.

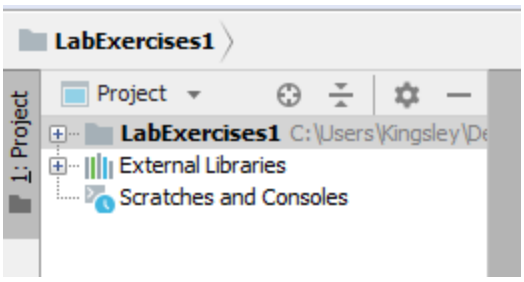
Exercises

Download the `Lab1Exercises` ZIP file. It contains some basic code for you to develop. The code is a set of Python source code files (`.py`). You will need to open a new PyCharm project and import the `.py` files into it. To do this:

- Start PyCharm.
    - Open the "Software Hub" (found on the desktop or searching the start menu).
    - Type "PyCharm" in the Software Hub search bar.
    - When PyCharm is found as an option, click "Launch".
- Select New Project (select Pure Python option) and accept the default choices in the dialogue box that comes up.
- Unzip the file of exercises downloaded from Canvas: `Lab1Exercises.zip`. Drop the unzipped `.py` files on the right-hand pane (where it says "Drop Files Here To Open") if you want the Python files to be retained where they were stored originally.
- Copy the unzipped `.py` files to the Project root folder. The files are now retained as part of the PyCharm project folder structure.

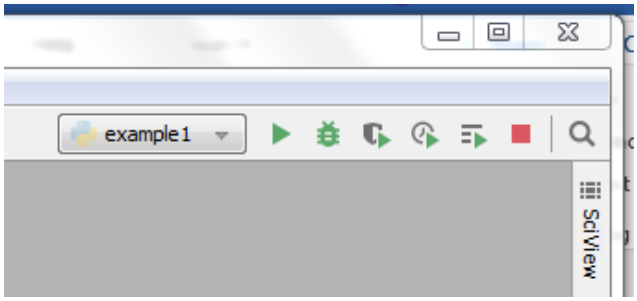
If you want to create a new `.py` file:

- Right Click the Root project name under the Project tab on the left-hand side. For example, if the project was called `Lab1Exercises`, then it would look like this:

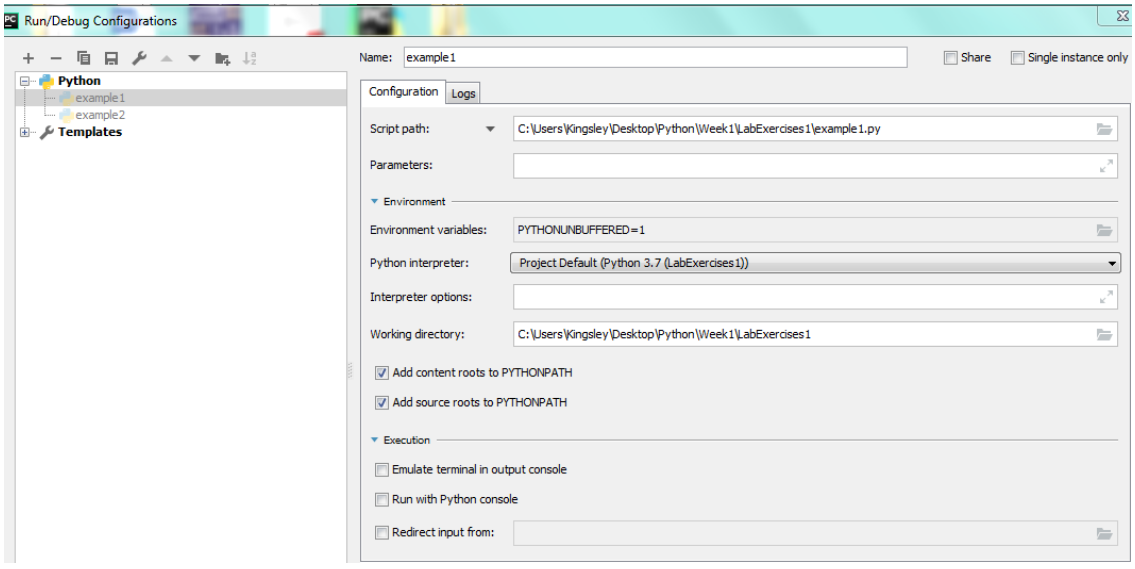


- Select New Python File.

Once you have imported the Python files, you need to “add a configuration”. This means telling Python what to do with the files when you do something with them like ask for the program file to run. The easiest way to do this is just to right click the source file name e.g., `example1.py` and ask for it to be Run. As soon as you do this, a suitable configuration file will be created automatically. You can see the configuration file by select “edit configuration”. You can find this by clicking on the panel that looks like this:



- Click the dropdown where the source file name is and you will find the “edit configuration” choice. It will look something like this:



- You will need a configuration file for each source file you want to Run directly. The configuration file determines what happens when you ask PyCharm to do something.
- To run a `.py` source code file, ensure the correct configuration is selected and then press the small green arrow button or Run from the main Run menu option. If you need to force a Python program to stop, use the square red stop button.

Now that you understand the basic workflow, we can try some coding challenges. If you look at the example code from `Lab1Exercises.zip`, you will see that there are some comment lines that tell you what needs to be done to complete each program. In the next section, some useful hints and facts are provided to help you. This will be very helpful if you have not done any coding previously

Hints

Using variables

- Variables should Have a clear purpose and use appropriate names that describe what they do.
- Python is case sensitive, so `myX` is different to `MyX`. Adopt a naming convention for variables and use it **consistently**. The creator's conventions are detailed in the PEP8 guide but the short version is to use `snake_case` for variable and function names, i.e. all lower case with underscores as necessary (with constants in `BLOCK_CAPITALS`) and `CamelCase` for `Classes` and `Exceptions`.
- Variables need to be initialized (given some initial value) before use. Otherwise, they are unknown to the Python interpreter and a run time error will result.
- Python will keep track of a variable's type (it is a dynamically typed language)

Mathematical operations

Operator	Description
`+, -`	Add and subtract
`*, /`	Multiply and divide
`**`	Exponent e.g. 2^3 written as 2 ** 3
`%`	Remainder after division e.g. 4 % 2 = 0 as 2 goes into 4 twice leaving no remainder. 4 % 3 = 1 as 3 goes into 4 once, leaving a remainder of 1
`math.sqrt(x)`	Returns the positive square root of `x`. You need to add the line `import math` before using the `sqrt()` function.

Useful coding tips

You can assign multiple variables on one line of code:

```
a, b = 0,1
c, d = 3.2, 'red'
```

You can force Python to convert a variable of one type to another (called "casting"). The most common use of this is to force a string to be regarded as a number type:

```
# Ask the user to display a number
x = input('Enter a positive integer number:')
# x is a string
y = int(x)
# y is an int
```

You can split long lines of code over multiple lines to improve readability using \:

```
print('hjkjhkjhkjhkj\
dfdfdfdf')
```

Coding challenges 1

1) Generate a sequence of int numbers in the pattern 0, 2, 4, …, up to 20 inclusive and display the numbers on the screen.
2) Allow the user to enter a string, and then print out each letter in the string one at a time on a separate line.
3) As above but print out the letters in the string in reverse order.
4) Starting with the initial value 0 and 1, generate a Fibonacci sequence. Each element in a Fibonacci sequence is the sum of the two previous elements e.g., 0, 1, 1, 2, 3, 5, 8, … Allow the user to specify how many elements should be generated.

Some suggested answers are provided in Canvas. If you are happy with these exercises, try out this more challenging one…

Coding challenges 2

Let's start by adding some more coding basics. These will be covered in more detail in the next lecture session. An if statement allows a program to make a choice. An example of an `if` statement looks like this:

```
x = int(input('Enter a value for x:  '))
if x < 10:
    print ('small value')
elif x < 20:
    print('middle sized value')
else:
    print('larger value')
```

If you need to break out of a `while` statement, you can use the `break` statement:

```
x = 1
while x < 10:
    if x > 5:
        break
    else:
        print(x)
        x += 1
```

Using this knowledge:

1) Allow the user to enter an int number. Determine if the number is prime or not. A number is prime if it is only divisible exactly by 1 and itself, e.g., 3, 7, 11 and 17. Two is the only even prime number. 1 is not generally regarded as prime. Display the answer "prime" or not prime as appropriate.

A suggested answer is provided on Canvas.

Dr. Benjamin Evans
B.D.Evans@sussex.ac.uk