University of HUDDERSFIELD

python powered

CFS2160: Programming Stream

Tutorial/Practical 1

Variables, Values and Operations

Introduction

This week you will again meet PyCharm, explore the Python console (which you used briefly last week), and then write some simple programs.

Activities marked ↯ should be included in your logbook for assessment.

Activities in *italics* are for those who find the rest easy. Do not panic if you cannot do them - we will cover the ideas needed later on.

Purpose

This week we introduce the Python console, and use it to do some simple tasks. Then we will take the results of these experiments and turn them into potentially useful programs.

By the end of the week you should:

- Understand variables, and types.
- Understand operators, and how they can be applied to variables (and values).
- Be able to write programs that carry out useful calculations.

As of now, these practicals will tell you *what* to do and not *how* to do it. As you go you will become a more experienced PyCharm user, so do not worry if things seem a little mysterious at first. This really is quite normal.

Above all, remember:

If in doubt - Ask!

Activities

Login to the PC and then carry on as follows. The first sections below remind you how to start PyCharm. If you have done that, skip along the item 2.

Be very sure you know where you are saving your work, and that it is not on the C: drive (which is the PC at which you are sitting). It needs to be K:, or your own portable storage.

1. Locate PyCharm on the Start menu (it's in the JetBrains folder) and start it.

    1.1. You have no settings to import, so "OK" at the first window.
    Note: If you ever want to reset PyCharm to the default settings, simply locate a folder called `.PyCharm2017` (or similar) in the `JetBrains` folder on the K: drive and delete it. Similarly, if you want to copy your settings to a new computer, this is the folder to copy (or import). (You can also copy settings from within PyCharm.)

    1.2. In the next window, open the preview, and see what the settings do. Change the theme and editor colours if you want (maybe you prefer a darker background), but leave the Keymap alone! Click "OK".

    1.3. You should now have a window that invites you to open a project.

    1.4. Click "Create New Project".

    1.5. In the next window, pick a location for your project files. **Do not leave this on the C: drive.** This should be a handy location on your K: drive, or wherever else you plan to store your work. A portable drive or USB stick would be fine.
    If you are in doubt here - *ask*!

    1.6. In the "Interpreter" box, PyCharm is asking which version of Python this project will use. There is a copy of the correct Python interpreter on the J: drive, here: "`J:\Python\python.exe`", so enter that[1]. You need to click the little cog to be able to enter the path.

    1.7. Click OK and wait for PyCharm to start. Any IDE can seem a little overwhelming at first, and PyCharm is no exception. Remember that we have many months to get to grips with it, and we only need a few features to get started.

    1.8. Right-click on the name of your project (to the top-left) and pick "New Python File", and enter `hello.py` as the name of your first program. Click "OK".

    1.9. There should now be a blank editor window. Enter the one-line program from the first lecture.

    ```
    print ('Hello, World')
    ```

    See how PyCharm helpfully highlights parts of the code.

    1.10. Now to run it. There are many ways to do this in PyCharm, but the quickest with a small program is to right-click the file name and pick "Run". All being well, a new panel will appear at the bottom of the IDE, where you can see your output.

    1.11. Pat yourself on the back. Make some notes so that you will be able to get to this stage in the future.

2. Start the Python interpreter (also known as the Console) inside PyCharm[2] and type the following. Make sure you understand the responses (where there are any).

    ```
    >>> 2 + 2
    >>> x = 2
    >>> print x
    ```

[1] If you take your project home, you will need to change this setting to wherever you installed Python.
[2] Several ways to do it. There should be a button with the Python logo to the bottom left of the IDE, or it's about halfway down the Tools menu.

```
>>> print (2 + 2)
>>> print (2 + 2 * 3)
```

3. Try these, and make sure you understand:
```
>>> 'Spam' == 'Eggs'
>>> spam = 'Eggs'
>>> len (spam)
>>> x = 2
>>> x == 2
>>> type (x)
>>> y = 2.5
>>> type (y)
>>> x = y
>>> print (x)
```

4. Using the examples from the lecture and the results from the first two activities above, explore more about variables, operators, and expressions.

5. ~~3~~
A kindly teacher has a bag of sweets to be shared among a class of pupils. There are 35 sweets in the bag, and 15 pupils in the class. Write out expressions that will determine how many sweets each pupil will get, and how many sweets will be left over for the teacher. Test these at the Python console, and take a screenshot for your logbook.

6. ~~3~~
Write a program that creates two variables like so (as the first two lines):
```
number_of_pupils = 16
number_of_sweets = 47
```
and then prints out the number of sweets for each pupil, and the number of sweets left over. Include the program in your logbook (as a screenshot, showing the output).

7. Try your program with different values for the number of pupils and number of sweets. Are you happy that the results are correct? What happens if the number of pupils is somehow set to a negative value? Or set to zero? Or if the number of sweets is negative or zero? What should happen?

8. Check through the lecture slides and your own notes from the lecture. If anything is not clear, ask your tutor to explain. You may well not be the only one who doesn't follow everything.

9. *If you are feeling confident, go ahead and do some research and fix the problem of a negative value for the numbers of pupils and sweets. Handle zero for either too (the fix is probably different). At the same time, you could modify the program so that the user can enter the number of pupils and sweets at a suitable prompt or prompts.*

10. *If you manage the task above, investigate what happens if the user enters a floating-point value or string at your prompt that expects an integer. Can you work out how to fix this?*