**Hello Boots!**

Each of you should work through the following, but you'll be in breakout rooms in your groups, so do help each other if you get stuck.

You'll be typing and using some Python code in this exercise. Don't worry – at this stage, you don't need to worry too much about understanding each of the commands - we'll be covering that in the next session.

1. Create a new Python script file by clicking File > New File, and then selecting "Python File" from the dropdown list that appears at the top.

2. Click the dropdown arrow next to the play button at the top right, and select "Run Current File in Interactive Window". This will run the script file and bring up the iPython console. Currently we have no code in our script though, so this will just bring up the iPython console.

Note : if this is the first time you are running anything, you may be asked to select an Interpreter. You should see a highlighted piece of text at the bottom right of VSCode, which will begin with a number that represents your Python version number (e.g. 3.9.16). Click this, and a "Select Interpreter" window will appear at the top. For now, you only have a base environment, so just select the one that says 'base'. We'll come back to environments later. Also don't worry if this doesn't appear.

3. In the iPython Console command interface at the bottom of the interactive window on the right, type :

**print ("Hello world!")**

then hit SHIFT + ENTER (or click the little play button to the left). Brace yourself for excitement and observe what happens.

4. In the iPython Console, type :

**name = "Dan"**

then hit SHIFT + ENTER (you can replace "Dan" with whatever your name is). Unless your name is also Dan, in which case, you've just saved yourself several seconds of work.

Now look at the Jupyter tab (this may ben named Jupyter:Variables if you've already run some code). You should see that a variable called 'name' has been created in memory, and currently holds a value – a piece of text (string) that represents whatever you name you used.

5. In the iPython Console, type :

**print (f"Hello {name}")**

then hit enter. Suddenly things are getting more exciting, right? (Look, back in the 80s getting a machine to write your name on the TV screen was mind blowing!)

6. Variable values are so-called because the values they hold in memory can change. So let's change our one and only variable value. Overwrite the value stored against the variable 'name' with a new name. Note how the variable value changes in the Variable Explorer. Then print a greeting to the new name, as you did above. Tip : you can use the 'Up Arrow' and 'Down Arrow' cursor keys to

scroll back and forward through time!  Well, in terms of the commands you've input previously anyway…

7. Let's create another type of variable, called a *list*.  In the iPython console, type :

**my_list = [1,2,3,4,5]**

then hit SHIFT + ENTER.  Look at the Jupyter:Variables tab – we've got a new variable!  Double click on this new variable in the Variable Explorer – what do you see?

8. Let's change a bit of our list.  In the iPython console, type :

**my_list[2] = "HSMA"**

then hit enter.  Look back at the Jupyter:Variables tab, then double click on the the list variable again.  What do you see that's different?

9. We can also use the iPython console to tell us the current value stored against a variable, by simply typing the name of the variable and hitting enter. Try that with your two variables.  (Important – this will only work in the iPython console.  If you want a program that you're writing (in the Editor pane) to print out the values of variables, you need to use the print command.  You'll see examples of that in the Python training).

10. OK, let's get you to try writing a very simple program.  Remember – a program is just a series of instructions.  We place each instruction on a new line (by hitting enter). In your Editor pane, write a program that stores your name in an appropriately named variable, then prints a message that says hello to you (using your name), then changes the name stored in your variable to another name, and then says hello to that name.  Once you've written your program, save it as a .py file (this is the file format for Python programs / scripts) with whatever filename you choose (e.g. my_first_program.py) and then run it.  Observe what happens in the iPython Console and the Jupyter:Variables tab.

11. Once you've run the program, go into the iPython Console and type the name of the variable that you created in your program, and hit enter.  Observe what happens.

12. At the top of your iPython console, you'll see a little button that says "Restart".  Click that, and then confirm.  Watch what happens to the Jupyter:Variables tab.

A "Kernel" is a program that runs and inspects your code.  The iPython console includes a Kernel that's automatically started when the iPython console is booted.  If you reset the Kernel, then it effectively reboots and wipes all of the variable values etc that are stored in memory.  You don't normally need to do this, or worry very much about the Kernel at all, but sometimes you may need to reset the Kernel if your code won't stop executing and standard methods of interrupting the code execution (which we'll get onto) don't work.  Think of it as the "Turn it off and on again" button for Python.  Closing and reopening VSCode will also reset the Kernel.

13. You've been provided with some demo .py files for this session (remember – these can be downloaded from the GitHub repository for this session).  Make sure you've got the files saved locally somewhere on your computer.  Now open the file called "1d_demo1.py".  You should see a new tab open in your Editor pane (if you find it opens in the pane to the right, it's because the focus was on

that window – to avoid that happening, just click in the Editor window first. But don't worry, you can just left click and drag the new file tab over to the left hand pane). This file contains a very simple program. Run the program. Watch what happens.

The program you're running contains something known as an "Infinite Loop". This means that the computer has been told to keep repeating a set of instructions *forever*. Those of you who grew up with home computers in the 80s will likely have written your own similar infinite loops in BASIC (Beginner's All Purpose Symbolic Instruction Code), that likely went something like this :

```
10 PRINT "I AM AMAZING!"
20 GOTO 10
```

For additional effect, you may have done this in a computer store in your local town (or Boots / WH Smith – yes, they did used to sell computers, bizarrely), where they used to allow people to come in and try out these magical new home computer machines.

What you're observing in the code that you've got running is a modern day equivalent in Python. Sometimes you want to deliberately create an infinite loop (and we'll talk about situations like that later in the course). But most of the time, you've probably mucked up somewhere in your code, and your program is now stuck in an infinite loop from which it can never escape!!!!!!!!!!!!!

Unless… you click in the iPython Console and click the Interrupt button, which will basically tell Python to shut up and break out of your program. If it doesn't work, then give it a moment. If it still doesn't work, you should resort to the "Turn it off and on again" button (resetting the Kernel) mentioned above.

Or you could just have your computer forever print out the words "Dan is the best!". That's equally valid.

14. Open the file "1d_demo2.py". Have a look at the code as a group (so this is a good opportunity to make sure you're all up to the same point, and to help each other if you're not). I don't expect you to be able to fully understand this at the moment, because I haven't yet taught you most of the code that's in here. But see if you can work out what this program is doing. Have a look at it line by line. Run it. Play around with it.