IN-CLASS WORK: PYTHON — FIRST PROGRAM, VARIABLES, IN-OUT

1. Linux Environment: Repeat steps 1. & 2. of the linux exercise to get ready to work in the linux environment. I suggest that you do all your Phys338 course work within your directory ~/Phys338_s2025. Go to this directory with
`cd ~/Phys338_s2025`
You may want to make subdirectories, e.g. one for our python intro unit.

2. First Python Program

This step shows you how to write and run your first (or not so first) Python program. (Info for advanced: we will not use jupyter but will work with python scripts.) A python program or "python script" or "source file", is a file which has a set of commands in it. Write a file with the editor `gedit`, as you did in the linux exercise (or any other editor). So type in a terminal window the command

```
gedit first_program.py &
```

The file extension `.py` tells the `gedit` to go into Python mode, which will give you some helpful color coding. The ampersand allows you to get the command line back for more commands while `gedit` is still running.

Now, in the editor your first line has to be an instruction to fire up the Python interpreter, which we do with

```
#!/usr/bin/env python
```

Now add the following four lines

```
x=1
print(x)
y=2
print(x,y,x+y)
```

Save this, and then back at the command-line prompt type

```
chmod u+x first_program.py
```

This is a one-time step for this file that allows you to treat it as an executable file (read it as "change mode: user adds executable"). Now type at the command-line prompt (in the black terminal window)

```
./first_program.py
```

and see what results.

Hurray, you just wrote and ran your (first) program. So the commands are in `first_program.py`, the `chmod` command makes the file executable, and when you execute (or "run") the code, the so called python "interpreter" converts the human readable commands to commands being executed by the computer. The first line in your python script tells the interpreter that it is a python script.

3. Play with your program. Swap the lines of x=1 and `print(x)`. What happens? Go back to the unchanged `first_program.py` and change it such that you determine and print $5 + 3$ and also do it for $5.0 + 3.0$.

4. Read §2.2.1. & §2.2.2 of Newman's book.

5. Input & Output Read §2.2.3. in Newman's book. While you read the text, test the described commands. You can do this by changing successively your `first_program.py` or, I recommend, that you copy the `first_program.py` to another file with a different name like `inouttest.py` and then change the lines accordingly and run (excecute) them, confirming what they print on the screen (and read from the screen).

6. Write a program that reads in from screen two integers, two floats, and two strings and prints the sum of integers, the sum of floats and the sum of strings.

7. In/Out via redirecting (if time) It was kind of tedious to retype the numbers when you tested your program in the previous step. There is a way how you could have avoided having to type in all the values each time you run the program. You can write the values instead into a file, for example `in7` with the first line of the value of your first integer, the second line of your second integer, etc. If your program from step 6. is `classpython6.py` then run the program with
`./classpython6.py < in7`
And similarly you can print the output not on the screen but instead into a fileby using
`./inclass_python6.py < in7 > out7`
Look at the just created file `out7`

8. Copy, Read, Run Sample Python Script (if time;for advanced)

Get into your `~/phys338\_s2025` directory. Next make a directory for python sample files, so
`mkdir pythonsamples.dir`
Get into this directory (using `cd`) and copy the following sample file into your directory, so type (Note: The ./ at the end is part of the command.)
`cp ~kvollmay/share.dir/pythonsamples.dir/sample_inout.py ./`
When you run the script it reads in two files, so also do following copy commands
`cp ~kvollmay/share.dir/pythonsamples.dir/in1.dat ./`
`cp ~kvollmay/share.dir/pythonsamples.dir/in2.dat ./`
Next make the `sample_inout.py` executable (see step 2.) and look at this python script and run it and check if the resulting print out makes sense.