Introduction to Numerics – Workshop 1

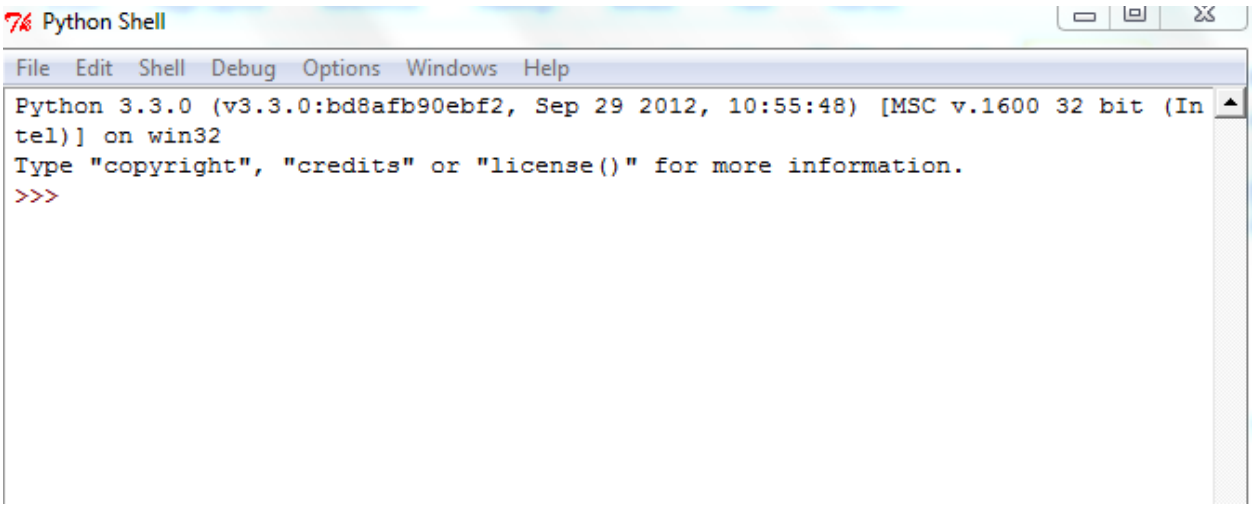
The numerics studied at A-Level are quite frankly rubbish, and deadly dull. To experience the joy and excitement of numerics you need to code. We are going to learn to do some basic coding in the Python language.

Coding is a very important skill, watch this video if you don't believe me.

Python is interpreted, and so is suitable mainly for prototyping work. Heavy, production numerics code is generally written in either Fortran, C or C++ which are compiled languages. The code you write in a compiled language is first transformed (i.e. compiled) into the machine level code a computer chip can understand, before running. This means will run faster (especially if you invoke the magic that is compiler optimisations) than interpreted code, such as Python of Javascript. We may look at compiled languages in the future.

Loading IDLE

On the school computers go to All Programs > ICT > Python. You will get a screen like the following



The Basics in Python

At its most basic, Python can work like an interactive calculator. Use python to work out the following (to raise something by a power in python you use the ** operator, for example to work out 2^3 you write 2**3)

Question:

$$2 + 2 =$$

$$2^{68} =$$

$$256 \times 678 =$$

$$89765 \div 14 =$$

$$14 \times 6411 =$$

What do you notice about the last two? This is because python has interpreted your 89765 and 14 as integers (whole numbers).

Question: In python, what do the following evaluate to?

$$89765.0 \div 14 =$$

$$89765.0 \div 14.0 =$$

Question: what can you say about what you need to do to have divisions treated correctly for real numbers?

Question: What does the % operator do? Try typing 76%4 and 76%7 to help you figure it out.

If I want to be able to refer back to a value then you need to assign it to a variable. To do this we use the '=' operator. For example, this code snippet assigns integers to the variables x and y, and then computes their product.

```
x = 4
y = 6
z = x+y
print('x + y = ', z)
```

The last line in the code above prints the string 'x + y = ', and then the value of z to the screen.

Note that in python the equals sign (=) is used for assignment. If you want to test for equality you have to use a double equals (i.e. ==) as shown in the next snippet of code.

This code can be modified to take in an input from a user, using the input command.

```
operand1 = int(input("Enter your first number: "))
operand2 = int(input("Enter your second number: "))
operator = input("Do you want to +,-,* or / ? ")

if operator == "+":
    result = operand1 + operand2
elif operator == "-":
    result = operand1 - operand2

print("The answer = " + str(result))
```

Notice that in the first two lines I have used the command int. This is because python reads in a string (a sequence of characters) from the user; to be able to perform arithmetic on the quantities entered they need to be converted to a number type. In this case, I have converted them to integers. Conversely, when I print out the result of the computation at the end of the program, I use the str command to convert my result back to a string.

We will talk about the if and elif commands in a minute.

Task: In Idle, go to File > New Window to launch the editor. And copy in the above code. Save it somewhere in your workspace as calculator.py. To run the code, either press F5 or Click Run > Run Module.

Question: What happens if you enter a number that is not an integer, for example 5.768 ?

Task: Fix this bug. (You may need to Google how to convert a string to a floating point (i.e. a decimal number) value in Python)

Program Control

You will have noticed that in the calculator code above I have used the commands if and elif. These are used to control the flow of the program dependent on some test conditions. The general form of this kind of statement is shown below

```
if <test1>:
    <statements1>
elif <test2>:
    <statements2>
else:
    <statements3>
```

As an example of using this, here is a short program is_even.py to test whether a number entered by the user is even.

```
n = int(input("Please enter an integer: "))

if n%2==0:
    print("This number is even")
else:
    print("This number is odd")
```

Question: Does this program give the correct answer when the user enters 0. Explain your answer.

Task: Using elif statements extend your calculator.py code so that it can do multiplication and division.

Task: Write a short program that takes in an integer as input from the user and then tests to see if it is a multiple of 2 or 3. If it is a multiple of neither it should print out "multiple of neither". Remember that some numbers are a multiple of both 2 and 3. Think about how this affects the control of your program.

Different Data Types in Python

So far we have looked at two data types – integers and doubles. Doubles store real numbers – we shall skip over the details.

We will also need to use the list data type. Lists are a bit like vectors, in that we can access a component of a list. To set up the vector $(2,4,5)$ and call it x, we can type

x = [2,4,5]

Typing x and hitting enter will display this list to the screen. To access a particular entry of a list you reference the position in the list, for example the third entry of the vector is 5, to access this entry type x[2]. Note, that '2' gives the third entry, this is because all vectors in Python are indexed starting at 0.

```
>>> x = [2,4,5]
>>> x
```

Lists can be contained within lists, using the syntax [[list1],[list2]].

Question: What would you type to store the 3×3 matrix $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$ as a nested list?

To extract particular entries of a nested list you use a similar syntax, for example if X contained a 3×3 matrix, then X[1][1] would give the middle value, i.e the entry $X_{2,2}$.

Always remember that when indexing a list, you start at 0.

A summary of the basic commands and the in-built maths commands are below (we will look in more detail at some of them later)

Operators	Description
`x + y`	Addition
`x - y`	Subtraction
`x * y`	Multiplication
`x / y, x // y`	Division
`x ** y`	Exponentiation
`x % y`	Remainder
`x < y, x <= y,` `x > y, x >= y,` `x == y, x != y`	Comparison
`x if y else z`	Ternary selection
`x | y, x ^ y, x & y`	Bitwise OR, XOR, AND
`x << y, x >> y`	Shift x left or right by y bits

Function	Description
`pow(x, y)`	Exponentiation
`abs(x)`	Absolute value
`round(x,n)`	Rounding (normal)
`int(x)`	Integer part
`hex(x)`	Hexadecimal
`bin(x)`	Binary
`float(x)`	Convert to floating point number
`complex(x, y)`	Convert to complex number

Some common list operations are shown below

- Common list operations

`L=[]`	Empty list
`L=[0, 1, 2, 3]`	List with four items
`L=['abc', ['def', 'ghi']]`	Nested list
`L[i], L[i][j]`	Index
`L[i] = 1`	Index assignment
`L[i:j]`	Slice
`L1 + L2`	Concatenate
`L * 4`	Repeat
`for x in L: print x`	Iteration
`del L[1], del L[i:j]`	Delete

Question: How can you create an empty array to store a 3×3 matrix.

Task: Write a program determinant.py that reads in from the user, 1 entry at a time a 3×3 matrix (I suggest you ask for entries by going along the rows) and then computes the determinant of this matrix.

More Mathematical Operations

More mathematical operations are contained in the Python math module. To use this you must import it into the workspace first.

```
>>> import math
```

There are loads of functions in this module, go to https://docs.python.org/3/library/math.html and explore them. To use one of these functions you must prefix the function name with "math." See below for an example:

```
>>> import math
>>> x = 5
>>> math.factorial(x)
120
>>> math.cos(x)
0.28366218546322625
>>>
```

Task: Using the program outline shown below, write a program called solve_quadratic.py that will solve a quadratic equation of the form $ax^2 + bx + c = 0$.

```
print("This program solves the quadratic ax^2+bx+c = 0")
import math
a = float(input("Please enter the coefficient of x^2: "))
b =
c =

discriminant =
if discriminant < 0:
    print("no real solutions")
elif discriminant > 0:


else:
```

Task: Modify your calculator so that it can also compute the exponential of a number, and sin, cosine and tangent of a number too.