A 5-Minutes Python Primer

```
disipio@Riccardo-MacBook-Air.local: python
Python 2.7.5 (default, Mar  9 2014, 22:15:05)
[GCC 4.2.1 Compatible Apple LLVM 5.0 (clang-500.0.68)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Ready?

```
>>> # this is a comment
>>> import os, sys # additional functionalities: import external modules
>>> from math import sqrt # or maybe just a subset
```

```
>>> print ‘Hello, world’
Hello, world
>>> greeting = “Hello, world”
>>> print greeting
Hello, world
```

Now with numbers

```
>>> x = 42
>>> print x
42
>>> y = 3.14
>>> print y
3.14
```

Math

OPERATION	Operator	Example
Addition	+	5 + 5
Subtraction	-	5 - 5
Multiplication	*	5 * 5
Division	/	5 / 5

```
>>> print 5*5
25
>>> print “Hello” * 5
HelloHelloHelloHelloHello
```

More operations stored in additional libraries

```
>>> from math import *
>>> log(2)
0.6931471805599453
```

```
>>> cos(pi)
-1.0
>>> sqrt(2)
1.4142135623730951
```

Lists

Lists are ordered arrays.

```
>>> numbers = [ 1, 2, 3, 5 ]
>>> print numbers[2]
3
>>> print numbers[-1]
5
>>> print numbers[0:2]
[1, 2]
>>> print numbers[2:]
[3, 5]
>>> print numbers[:2]
[1, 2]
>>> numbers.append(8)
[1, 2, 3, 5, 8]
```

Same with strings

```
>>> crew = [ “Picard”, “Riker”, “Data”, “Worf” ]
>>> print crew[1]
Riker
```

Fancy ways to create lists (list comprehension):

```
>>> l1 = [ i*i for i in range(5) ]
[0, 1, 4, 9, 16]

>>> l2 = [ i*i for i in range(10) if i % 2 == 0 ]
[0, 4, 16, 36, 64]
```

Strings

Also strings are lists!

```
>>> name = “Riker”
>>> name[:2]
Ri
```

```
>>> formatted_string_1 = “my lucky number is %i” % 7
>>> formatted_string_2 = “My name is %s, pi = %3.2f” % ( “Riccardo”, pi )
```

Dictionaries

Similar to C++/STL maps, are unordered key-indexed arrays. Extremely useful in Python, but can affect performance.

```
>>> dic = { ‘a’ : 1, ‘b’ : 2, ‘c’ : 3 }
>>> dic[‘b’]
2
```

NB: you can add an entry or change the values

```
>>> dic[‘d’] = 4
>>> dic[‘a’] = -1
>>> print dic
{'a': -1, 'c': 2, 'b': 1, 'd': 4}
```

If…elif…else statement

Controls the flow of the program

```
>>> if 4 % 2 == 0: print “even”
...
even
```

```
>>> if i % 2 == 0: print “even”
... else: print “odd”
```

```
>>> if i == 0: print “zero”
... elif i == 1: print “one”
... else: print “other number”
```

```
>>> b = 0
>>> status = True if b == 0 else False
True
```

Loops

```
>>> for i in range( 3 ):
...   print i*i # nb: indentation is mandatory!
...
0
1
4
```

```
>>> for key, value in dic.iteritems():
...   print key, value
...
a -1
b 2
b 1
d 4
```

What if I wanted to print the items, sorted by key?

```
>>> for k in sorted( dic.keys() ): print dic[k]
...
-1
1
2
4
```

Functions

No need to define return values.

```
>>> def square( x ): return x*x
...
>>> square( 2 )
4
```

```
>>> def complex( a, b ): return (a, b)
...
>>> complex( 1.0, -0.3 ) # returns a “tuple”
(1.0, -0.3)
```

```
>>> def square_and_cube( x ): return x*x, x*x*x
...
>>> for i in range(5): square_and_cube(i)
...
(0, 0)
(1, 1)
(4, 8)
(9, 27)
(16, 64)
```

```
>>> def say_hi( person = “Riccardo” ):  print “Hi”, person
...
>>> say_hi()
Hi, Riccardo
>>> say_hi(“Joe”)
Hi, Joe
```

ROOT with Python

In brief, a physicist’s life is much easier with PyROOT!

```
disipio@Riccardo-MacBook-Air.local: source $ROOTDIR/bin/thisroot.sh
disipio@Riccardo-MacBook-Air.local: python
Python 2.7.5 (default, Mar  9 2014, 22:15:05)
[GCC 4.2.1 Compatible Apple LLVM 5.0 (clang-500.0.68)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> from ROOT import *
>>> h = TH1F( “histogram”, “My Fancy Histogram”, 5, 0, 4 )
>>> h.FillRandom( “gaus”, 1000 )
>>> h.Draw()
```

Official documentation: http://root.cern.ch/drupal/content/pyroot