**Computational Materials Python Cheat Sheet**

This cheat sheet was created in python’s interactive mode. In the interactive mode you type commands into the interpreter and directly find out the results. The “>>>” is the prompt used to let you know that python is ready to accept a command. To start python in interactive mode type **python** on command line. It will look as follows:

```
[wcs5879@fri ~]$ python
Python 2.7.1 (r271:86832, May 29 2012, 13:10:45)
[GCC 4.4.6 20110731 (Red Hat 4.4.6-3)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
```

Below are examples of how to use the python programming language.

### Variable Assignments

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;&gt;&gt; x=5</td>
<td>Declaring an integer variable x with value 5 or assign variable x to the integer 5</td>
</tr>
</tbody>
</table>

### Types of Variables

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;&gt;&gt; name = 'Bob'</td>
<td>String</td>
</tr>
<tr>
<td>&gt;&gt;&gt; I=5</td>
<td>Integer</td>
</tr>
<tr>
<td>&gt;&gt;&gt; F=5.0</td>
<td>Float</td>
</tr>
<tr>
<td>&gt;&gt;&gt; happy='True'</td>
<td>Bool</td>
</tr>
</tbody>
</table>

### Calculations with variables or arithmetic operators

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;&gt;&gt; y=7</td>
<td>Sum of two variables</td>
</tr>
<tr>
<td>&gt;&gt;&gt; y+5</td>
<td>Subtraction of two variables</td>
</tr>
<tr>
<td>&gt;&gt;&gt; y-5</td>
<td>Multiplication of two variables</td>
</tr>
<tr>
<td>&gt;&gt;&gt; y*2</td>
<td>Exponential of two variables</td>
</tr>
<tr>
<td>&gt;&gt;&gt; y**2</td>
<td>Remainder of a variable</td>
</tr>
<tr>
<td>&gt;&gt;&gt; y%2</td>
<td>Division of two <strong>integers</strong></td>
</tr>
<tr>
<td>&gt;&gt;&gt; y/14</td>
<td>Division with at least one float</td>
</tr>
<tr>
<td>&gt;&gt;&gt; y/14.0</td>
<td></td>
</tr>
</tbody>
</table>

### Type Conversion or Type Casting

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;&gt;&gt; str('5.1')</td>
<td>Variables to strings</td>
</tr>
<tr>
<td>&gt;&gt;&gt; int(5.0)</td>
<td>Variables to integers</td>
</tr>
<tr>
<td>&gt;&gt;&gt; float(5)</td>
<td>Variables to float</td>
</tr>
<tr>
<td>&gt;&gt;&gt; type(5.4)</td>
<td>Returns variable type</td>
</tr>
</tbody>
</table>

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
</table>
**Lists**

```python
>>> Names = ['bob', 'finn', 'ray']
>>> Names
['bob', 'finn', 'ray']
>>> Names[0]
'bob'
>>> Names[-1]
'ray'
>>> Names[2]
'ray'
>>> len(list)
3
```

Note python starts its indexing/numbering at 0 not 1
Select item at index 0
Select last item in the list
Select third item in list
Returns length of list

**Boolean Operators**

```python
>>> x=5
>>> x==5
True
>>> x!=5
False
>>> x>5
False
>>> x<=5
True
>>> True and True
True
>>> True and False
False
>>> True or True
True
>>> True or False
True
```

Equality: are the two values equal?
Inequality: are the two values not equal?
Greater than
Less than or equal to
Are both values True?
Are either values True?

**If Statements (Recall indentation rules in python!)**

```python
>>> x=5
>>> if x==5:
...     print True
...     print True
True
>>> if x==5:
...     print True
...     else:
...     print False
...     print False
True
>>> if x==2:
...     print 'value = 2'
...     elif x<2:
...     print 'value < 2'
...     else:
...     print 'value > 2'
...     value > 2
```
### Loops

```python
>>> x=0
>>> while x < 5:
...     print x
...     x += 1
...  0
  1
  2
  3
  4
```

While the variable `x` is less than 5, Print the value of `x` And add one to the value of `x`

Output from while loop

```python
>>> for i in range(0,5):
...     print i
... 
...  0
  1
  2
  3
  4
```

For `i` ranging from 0 until 5 Print the value of `i`

Output from for loop

### Defining functions

```python
>>> def print_number(x):
...     print x
...
>>> print_number(4)
4
```

Define the function which prints the input value, `x`.

Call the function with input value `x=4`.

```python
>>> def return_number_add_one(x):
...     return x+1
...
>>> new_number=return_number_add_one(5)
>>> print new_number
6
```

Define a function which returns the input value, `x`, plus one

Call the function and set the output equal to `new_number`

### Importing modules - Numpy

```python
>>> import numpy
>>> numpyarray = numpy.array([1,2,3],float)
>>> numpyarray
array([ 1.,  2.,  3.])
```

Import module (numpy)

Create a numpy array of floats called `numpyarray`

```python
>>> numpy.mean(numpyarray)
2.0
```

Use numpy to calculate the mean of the values in `numpyarray`

If you want to find more numpy functions, try googling numpy + (insert function your need!).