Python is a *interpreted* computer language.

- use like calculator
- create scripts
- add-on modules/libraries

To obtain the base Python software, download the software (www.python.org) that is suitable for your computer (apple, PC, or unix). Pre-compiled binaries are available for the common computer platforms.

Needed for this class:

- Python (www.python.org)
- numpy (numpy.scipy.org)
- matplotlib (matplotlib.sourceforge.net)

Strongly recommended:

- ipython
- *good* text editor

When downloading these, be sure to get the version that corresponds to the version of python installed on your computer. Read and follow the installation instructions.

Getting Started

- Open a terminal window
- Type 'python'

built-in operations:

- +,-,/,*
- %,**
- ==,!=,<,...
- not, and, or, in
- float, int, round, abs...

data types:

- lists[], tuples(), dictionaries{}
- methods associated with objects
- integers and floats (long and complex)
- strings
- other functions through libraries
- cntr-D or 'exit()' to leave interpreter

Getting Started-ipython

- tab completion
- history (up,down arrows, %hist)
- %run
- %logstart, %logoff, %logon
- %timeit

Getting Started-Writing a Script

- A short example: Calculate the value of $\frac{1}{x}$ for different values of x
- Open a terminal window
- Type 'ipython' or 'ipython -pylab'

a simple script, type in editor, save, and run in ipython

```
## this is a comment
## use them to describe lines in your program
for i in range(1,11): ## loop from 1 to 10
    y=1./i
    print i,y
```

- 'Ctrl-C' to break

Getting Started

Note the following elements to this scripts:

- colon-indent structure to define blocks (for looping and logic)
- variable assignment (i and y)
- keywords include 'print', 'range', 'for', 'in'
- color high-lites in text editor
- the impact of integer and floating point numbers on the calculation
- the range command produces a list, a basic data structure in python

Misc. basic operations

Compound Operations

- +=,-=,*=, etc.
- List (and tuple) indexing and slicing
- Lists changeable, tuples fixed
- stings are lists
- methods and dot notation

Half of a π

π can be calculated in a variety of ways. One method is to calculate the infinite product:

$$\frac{\pi}{2} = \frac{2}{1} \cdot \frac{2}{3} \cdot \frac{4}{3} \cdot \frac{4}{5} \cdot \frac{6}{5} \cdots \tag{1}$$

How can this be programed using python?

Half of a π

```
num1=2.; den1=1.
num2=2.; den2=3.
pi2=2*(num1*num2)/(den1*den2)
err=1.e-20 ## chng=1.
while 1: ##or while chng>err:
    pi1=pi2
    num1=num1+2.
    num2=num2+2.
    den1=den1+2.
    den2=den2+2.
    pi2=pi1*(num1*num2)/(den1*den2)
    print pi2
    if abs(pi1-pi2)<err: ##or chng=abs(pi1-pi2)
        break
```

If only there was something else

In this script two new commands are introduced that allow conditional testing. Conditions include equality (==), greater than(>), not equal (!=), and less than or equal (<=). If blocks have the following structure:

```
if **condition**:
    true?, then execute some commands
elif **condition**:
    otherwise if this is true, execute these commands
elif **condition**:
    more statements
else:
    none of the above true, do these statements
```

Conditional loopiness

- In contrast, the 'while' block loops until a condition is no longer true. 'break' and 'continue' statements are used to leave a loop. The 'break' statement completely exits the loop, and the 'continue' statement moves to the top of the loop.
- An 'else' statement can be used with a 'while' loop. Statements in the 'else' block are executed if the loop does not execute a 'break' command.

To be e or not to be e

The constant e can be calculated using the infinite series:

$$e = \sum_{n=0}^{\infty} \frac{1}{n!} \tag{2}$$

How can e be calculated?

To be e...

```
def fact(n):
    if n==0:
        x=1
    else:
        x=1
        for i in range(1,n+1):
            x=x*i
    return x
e2=0
```

```
for i in range(20):
    e2=e2+1./fact(i)
    print i,e2
```

Functions

In the previous example, a function is created for the factorial operation. Two keywords are used to create functions: 'def' and 'return'. A function starts with 'def', followed by the name of the function.

```
def function_name(arguments):
    statements...
    return values
```

Functions

- long comments can be added by enclosing text in triple quotes

```
'''
this is a comment
it can span multiple lines
'''
```

- if this type of comment is put at start of function, used as 'docstring'
- `if __name__==__main__:`
- Can be used for testing. refer to unit testing and doc testing modules.

Getting more functions

- Python includes many built-in functions. Additional functionality is available through modules, that are collections of functions saved in a file.
- Basic math operations (preciously introduced) include exponents (x**y or pow(x,y)), modulus (10%3), absolute value (abs(-1)), maximum and minimum (max(x),min(x)).
- Additional math functions are available through the math module including trig. functions(sin,cos,tan,asin,atan,hypot, conversion from degrees to radians (radians), common and natural logs (log10,log), and many more

Getting more functions

1st Way

```
import math
a=math.log10(10.)
print '%e5.2'%(a)
```

2nd Way

```
import math as M
    a=M.log10(10.)
    print '%e5.2'%(a)
```

3rd Way

```
from math import log10
a=log10(10.)
print '%e5.2'%(a)
```

load everything

```
from math import *
a=log10(10.)
print '%e5.2'%(a)
```

os: Operating System Module
Basic commands for manipulating files:

os.getcwd returns current working directory

os.mkdir make new directory

os.chdir change directory

os.listdir('directory') returns a list of items in 'directory'

os.remove('file') remove file

os.rmdir('directory') remove empty directory

os.rename(oldfile,newfile) rename file or directory

shutil.rmtree('directory') remove directory and its contents

shutil.copy()

shutil.copy2() keeps file access and modification times

shutil.copytree() copy entire directory

Existence and file parameters

os.path.isfile()

os.path.isdir()

os.path.islink()

os.path.exists()

os.path.getatime() last time accessed

os.path.getmtime() last time modified

os.path.getsize() file size

'walking' a directory

os.walk('dir') descends through all files and directories, returning current path, directories in path, and files in path (as three lists in a tuple).

os.path.walk(dir, function,args) executes function(args,dirname,files) descending through directory

filename glob'ing

- uses unix wildcard specs.
- glob.glob('string')
- coupled with os commands to work on specific files

Running shell commands

os.system Basic execution of command (eg. os.system('ls'))

subprocess module recent addition, supersedes several commands (eg os.popon)

subprocess.Popen('command') executes command, allows interaction through arguments (eg 'stdin=subprocess.PIPE' and 'stdout=subprocess.PIPE')

subprocess.call('command') execute command without interaction

Changing PythonPath

- List all directories searched by python scripts:
  - sys.path
- Add to PythonPath (temporarily):
  - sys.path.append('newDir')