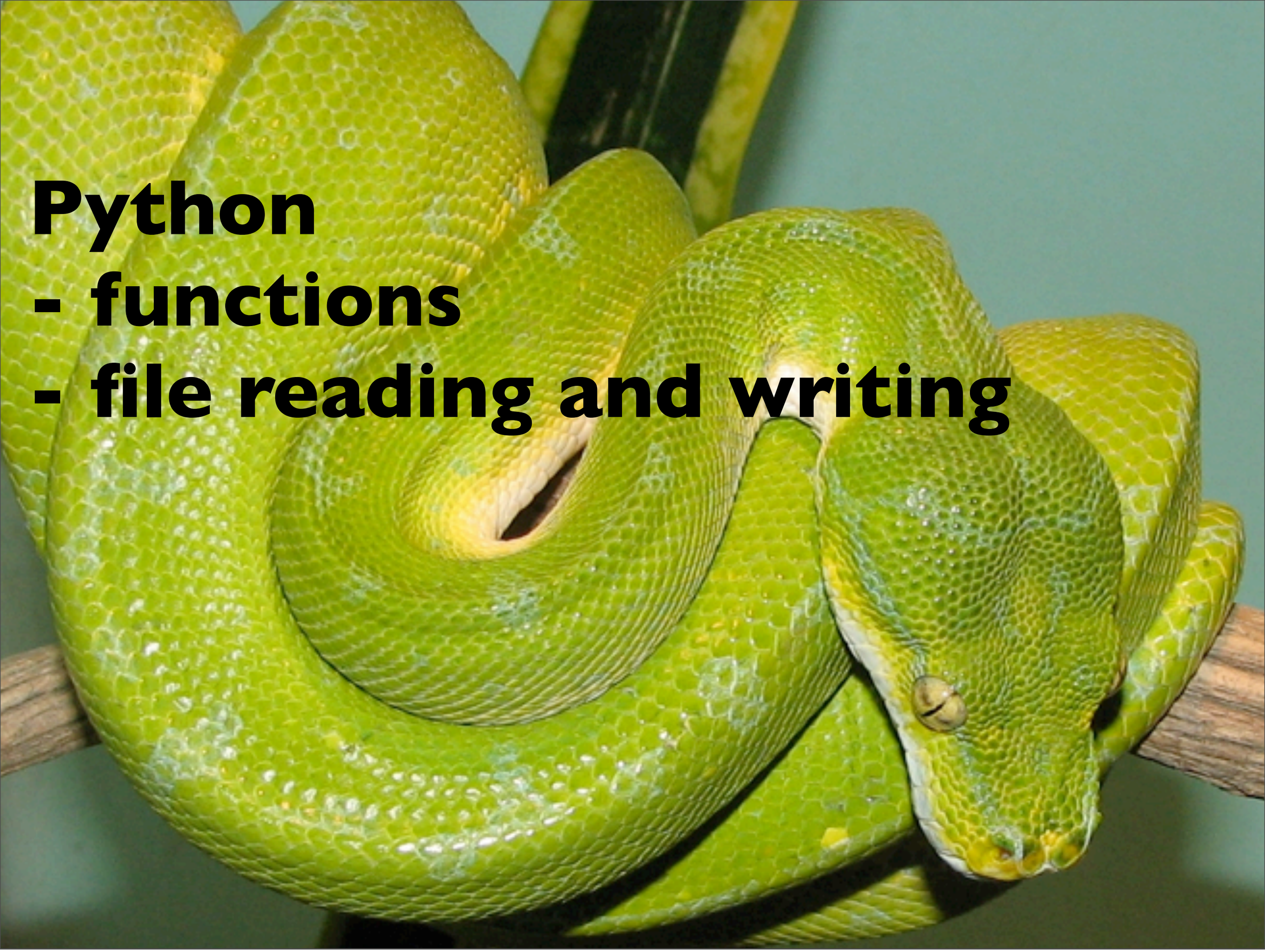

Python
- functions
- file reading and writing

```
for i in dir():
    print i,"=", eval(i)
```

Programming example

```
def f1():
    """ this function is doing nothing """
    pass

def f2():
    return "I did nothing"

def f3(a):
    return a

def f4(a):
    x = [a**i for i in range(0,5)]
    return x

def f4(a=0.5):
    x = [a**i for i in range(0,5)]
    return x

# calling these functions like this
a = f2()
b = f3(5)
c = f4()
d = f4(1.4)
print f1.__doc__
```

Programming example

```
def fib(n):
    a, b = 0, 1
    for i in range(n):
        a, b = b, a + b
    return a

results=[]
n = 10
for ni in range(n+1):
    x = fib(ni)
    results.append(x)
print results
```

$$F(n) = \begin{cases} 0 & n = 0 \\ 1 & n = 1. \\ F(n-1) + F(n-2) & n > 1 \end{cases}$$

```
def f6(a,b):
    return b,a

a, b = f6(a,b)
```

```
def f7(a,b):
    return [a,b]

[a, b] = f7(a,b)
```

http://en.literateprograms.org/Category:Programming_language:Python

```
# converts temperature to fahrenheit or celsius

def print_options():
    print "Options:"
    print " 'p' print options"
    print " 'c' convert from celsius"
    print " 'f' convert from fahrenheit"
    print " 'q' quit the program"

def celsius_to_fahrenheit(c_temp):
    return 9.0 / 5.0 * c_temp + 32

def fahrenheit_to_celsius(f_temp):
    return (f_temp - 32.0) * 5.0 / 9.0

choice = "p"
while choice != "q":
    if choice == "c":
        temp = input("Celsius temperature: ")
        print "Fahrenheit:", celsius_to_fahrenheit(temp)
    elif choice == "f":
        temp = input("Fahrenheit temperature: ")
        print "Celsius:", fahrenheit_to_celsius(temp)
    elif choice != "q":
        print_options()
    choice = raw_input("option: ")
```

Collaborative effort: unit conversion module

use for the filename: units_Slunit_otherunit.py

include 3 functions:

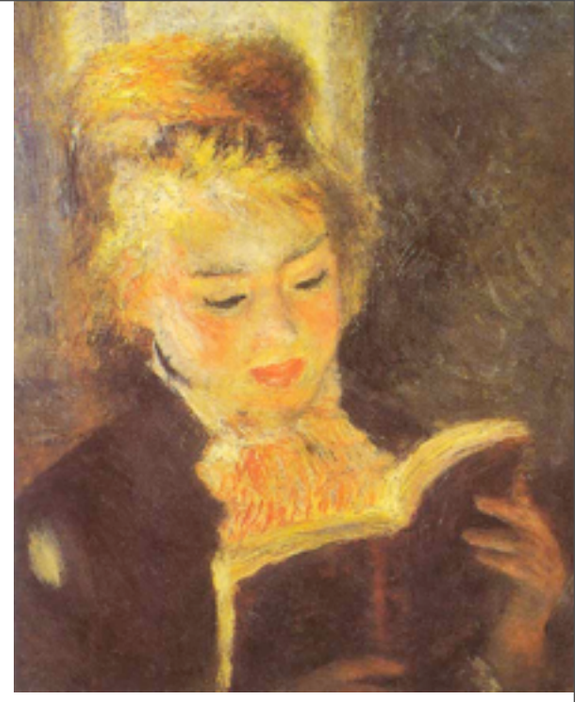
a_to_b(a), b_to_a(b), test_a_b()

the last function will be used to check whether your function is correct using some known values

- length: meter versus mile, yard,foot, inch
- area: km^2 versus square mile, acre, hectare
- volume: liter versus cup, pint, gallon
- weight: kilogram versus ounce, pound, short ton

http://en.wikipedia.org/wiki/U.S._customary_units

Reading a file



```
#!/usr/bin/env python
#
# reading frog data collected by Peter Beerli long time ago
# the datafile has a header with comments that start with #
# it also contains missing data that is marked with a .
#
import sys
import math

def read_data(file):
   data = []
   f = open(file,'r')
   for fi in f:
       if fi[0]=='#':
           continue
       line = fi.split()
       #print line
       data.append(line)
   f.close()
   return data

if __name__ == '__main__':
    if len(sys.argv)<2:
        print "Syntax: python frogs.py datafile"
        sys.exit(-1)

    data = read_data(sys.argv[1])
    print data
```

Manipulating data

```
#!/usr/bin/env python
#
# reading frog data collected by Peter Beerli long time ago
# the datafile has a header with comments that start with #
# it also contains missing data that is marked with a .
#
import sys
import math

def read_data(file):
   # code hidden see earlier slide

if __name__ == '__main__':
    if len(sys.argv)<2:
        print "Syntax: python frogs.py datafile"
        sys.exit(-1)

    data = read_data(sys.argv[1])
    # we want to extract the species name and 3 column values
    newdata =[]
    for di in data:
        species = di[0]
        length  = di[3]
        tibia   = di[4]
        cil     = di[5]
        if not(cil == '.' or species == '.' or length == '.' or tibia == '.' or cil == '.'):
            newdata.append([species, float(length),float(tibia),float(cil)])
    print newdata
```

partition the data

```
# see earlier slide for missing parts
def read_data(file):
   # see earlier slides for content

def unique(data, index):
    tmp = []
    for di in data:
        tmp.append(di[index])
    return list(set(tmp))

if __name__ == '__main__':
    # see earlier slide for content
    # read the data into data then transformed into newdata
    specieslist = unique(newdata,0)
    finaldata = []
    for si in specieslist:
        finaldata.append([])
    for di in newdata:
        for si,fi in zip(specieslist,finaldata):
            if di[0] in si:
                fi.append(di)
    # now we have partitioned the data into species
    # we can no present the means (std) of the values per group
    #
    print finaldata
```

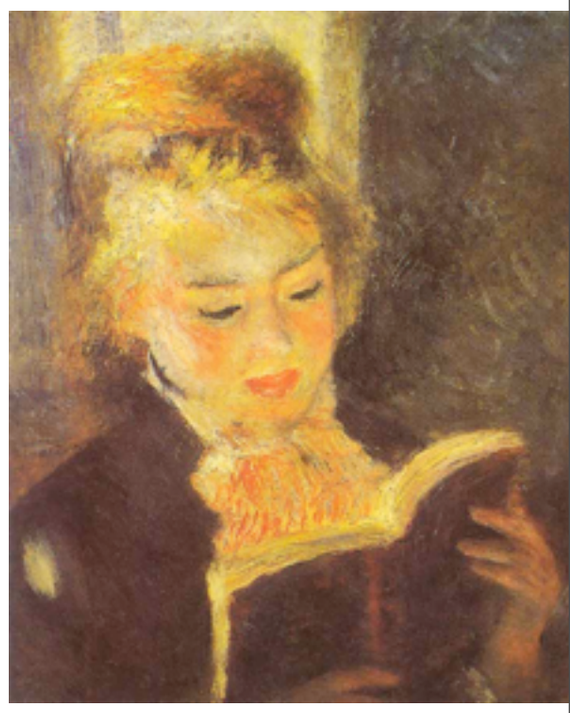
Analyzing data

```
# see earlier slides, this comes after the creation of finaldata

# now we have partitioned the data into species
# we can no present the means (std) of the values per group
#
print "species    n mean_length std_length mean_tibia std_tibia mean_cil std_cil"
for fi in finaldata:
    print "%-7.7s %4li " % (fi[0][0],len(fi)),
    means = averages(fi,[1,2,3])
    std  = stds(fi,[1,2,3])
    for i,j in zip(means,std):
        print "%9.3f (%7.4f)" % (i,j),
    print
# now we have partitioned the data into species
# we can no present the means (std) of the values per group
#
print
print "species    n scaled_tibia  scaled_cil"
for fi in finaldata:
    print "%-7.7s %4li " % (fi[0][0],len(fi)),
    newfi = [[x[2]/x[1],x[3]/x[1]] for x in fi]
    means = averages(newfi,[0,1])
    for i in means:
        print "%9.3f" % i,
    print
```

Writing a file

```
# see earlier slides, this comes after the creation of finaldata

# now we have partitioned the data into species
# we can no present the means (std) of the values per group
# and write these results into a file named “results”
 r = open(‘results’,’w’)
print "species    n mean_length std_length mean_tibia std_tibia mean_cil std_cil"
for fi in finaldata:
    x = "%-7.7s %4li " % (fi[0][0],len(fi))
    r.write(x)
    means = averages(fi,[1,2,3])
    std  = stds(fi,[1,2,3])
    for i,j in zip(means,std):
        x = "%9.3f (%7.4f) " % (i,j)
        r.write(x)
    r.write(‘\n’)
r.write(‘\n’)
r.write("species    n scaled_tibia  scaled_cil\n")
for fi in finaldata:
    x =  "%-7.7s %4li " % (fi[0][0],len(fi))
    r.write(x)
    newfi = [[x[2]/x[1],x[3]/x[1]] for x in fi]
    means = averages(newfi,[0,1])
    for i in means:
        x = "%9.3f" % i,
        r.write(x)
    r.write(‘\n’)
r.close()
```

Reading from stdin OR file



```
#!/usr/bin/env python
# reading frog data collected by Peter Beerli long time ago
# the datafile has a header with comments that start with #
# it also contains missing data that is marked with a .
import sys
import math

def read_data(file=None):
   data = []
   if file == None:
      f = sys.stdin
   else:
     f = open(file,'r')
   for fi in f:
       if fi[0]=='#':
           continue
       line = fi.split()
       data.append(line)
   f.close()
   return data

def help(args):
    for arg in args:
        if “-h” in arg:
            print "Syntax: python frogs.py datafile"
            print "or:     python frogs.py  < datafile"
            sys.exit(-1)

if __name__ == '__main__':
   help(sys.argv)
   la = len(sys.argv)
   if la==2:
        data = read_data(sys.argv[1])
    elif la>2:
        print “Too many arguments”
        help([‘-h’])
    else:
        data = read_data()
    print data
```