Introduction to Programming for Scientists

LECTURE 2:
COMMANDS & METHODS

Prof. Steven Ludtke
N421, sludtke@bcm.edu

Last time ...

- integers
- floats
- strings
- lists
- import math,cmath; from math import *
- how to define functions
- help()

Programs you can Run

- Edit a text file, use a ‘.py’ extension
- for unix/mac, put:

  #!/usr/bin/env python

  on the first line of the file, and type:

  chmod a+x file.py
- On Windows, just use the editor in IDLE
- NOTE: on windows, as soon as the program exits, the window showing the output will close. If you put a raw_input() at the end of your program, it will wait until you press enter before closing the window so you can see the output.

What Can Computers Do ?

- Store data
- Rearrange data
- Decisions based on data
- Math
- Communicate

Python

- Data storage
  - ‘simple’ types - numbers, strings, ...
  - compound types - lists, dictionaries, sets, ...
- Operate on data
  - statements - a=b*10, print b*5+3, if a>5 : a/=2, ...
  - functions - sin(a), len(x), ...
  - methods (functions on an object) - “abc”.count(“b”)
- Interact with the outside world
  - User interactions - raw_input()
  - Disk and other device access - file i/o

Conditionals & Loops

- **if** (condition) :
  - Boolean operators
    - >, <, <=, >=, ==, !=, and, or, not, in
- **elif** (condition) :
- **else** :
- **while** (condition) :
- **for** i **in** list:
- **try**, **except**

- Nested loops - a loop inside a loop

```
for i in range (10):
    for j in range(10):
        print i,j
```

- Continue/break - interrupting the flow of a loop

```
for i in range(20):
    if i==5 : continue
    if i>17 : break
    print i
```

More on lists

- append, extend
- del, remove
- count
- index
- reverse, sort

List Generators

- [x... for x in y if z]

  example:

  ```
  a=[0,1,2,3,4,5,6,7]

  b=[i**2 for i in a if i>1]

  b -> [4,9,16,25,36,49]
  ```

Methods of Strings

- upper, lower, title, capitalize
- count, find, rfind, index
- replace
- split
- regular expressions later...

Dictionaries

- keys must be immutable, values are arbitrary
- { k1:v1, k2:v2, k3:v3, ... }

  Example:

  a={ 1:2,2:3,”a”:”b”,2.0:3.2,(1,2):”really?” }

  a[1] -> 2

  a[(1,2)] -> “really?”

  a[2] -> 3.2

Dictionary Methods

- has_key
- keys
- values
- items

Sets

- Sets have no order and are unique, but can be iterated over
- set([1,2,3,4,5])
- add, remove, discard, clear
- issubset, issuperset
- union, intersection, difference

Some Built-in functions

- int, float, str, list, tuple, set, dict
- range, xrange
- enumerate
- eval
- input & raw_input
- len
- max,min
- reversed, sorted
- type, isinstance

Writing Simple Programs

1: Consider how the data will be represented

2: Break the problem down into a sequence of simple steps

3: Write the code for each step

Simple problem ?

- Sort the letters in a string entered by the user and print out the sorted string.

Data Representation

- string from the user
- can’t rearrange strings, put it in a list
- back to a string for output

Simple Steps

- get string from user
- convert to list
- sort list
- convert list back to string
- print result

Homework #2

- Write a program to count how many letters (a-z) are in a string input by the user, and print out any non-zero counts.

  example : User enters: “AbDceegee”
  ```
  a: 1
  b: 1
  c: 1
  d: 1
  e: 4
  g: 1
  ```

  Turn in your program (by email, .py file) and the output from running your program on the string “How many Letters are in Alphabet ?”