Introduction to Programming for Scientists

Prof. Steven Ludtke
N420, sludtke@bcm.edu

Lecture 2:
Conditions, Loops & Data

Reminder

- Class material at:

http://blake.bcm.edu/IP18

- (unfinished) intro to programming book posted on class wiki above. Please don't redistribute.

- If you missed the first lecture, it is archived on the site above.

Python

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

Type conversion

Type	Creation	Conversion
integer	number, no decimal	int()
floating point	use decimal point	float()
boolean	True or False	bool()
complex	x+yj	complex()
string	"X", 'X', """X""", '''X'''	str()
list	[a,b,…]	list()
tuple	(a,b,…)	tuple()
set	{a,b,…}	set()
dict	{a:b,c:d,…}	dict()

Python

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

Operators

- + - add (number), concatenate (list,str)
- - - subtract (number), difference (set)
- * - multiply (number), replicate copies (list,str)
- / - division (number)
- // - integer division (number)
- ** - raise to power (number)
- % - modulus (number), string format (str)
- & - logical and (number), intersection (set)
- | - logical or (number), union (set)
- ^ - logical exclusive or (number), symmetric difference (set)

Operators, Functions and Methods

- Operators : +, -, *, /, %, …
- Functions : sin(x), cos(y), len(s)
    - normally return a value
    - Not type-specific
- Methods : st.upper(), lst.append(5), lst.sort()
    - functions applied to a specific “object”
    - don't always return anything
    - methods are type-specific

Methods of Strings

- Remember strings are immutable !
- count, find
- replace
- split
- join
- in (statement, not a method)

Python

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

Simple Program Flow

- if condition :
  - Boolean operators
    - >, <, <=, >=, ==, !=, and, or, not, in
- for i in list:

for Loops

- Execute 'code' for each item in list, assigning the element to 'var' in each cycle:

  for var in list:

  code

Example:

```
a=[1,2,3,4,5]
for i in a:
    print(i,i*2)
```

Comments

- Anything after ‘#’ on a line is a comment

Homework #2

For all homework assignments, you are free to consult others on concepts, but the final code you turn in should be your own. If you are just learning programming for the first time, I would suggest that you try to spend at least 30 min thinking about each problem before seeking assistance. Even then, the first few assignments may be frustrating and time consuming, but if you don't practice the fundamentals now, you may be in real trouble later in the class. The only way to learn programming is by doing it. There are many possible solutions to each of these problems. If you need help, you can contact the TA or email me at any time, or find me any time my office door is open (mornings are usually better). While it may be possible to Google answers to some of these homework assignments, you won't learn much if you solve them this way. We will go over the solutions at the beginning of each class, so the homework must be emailed my midnight the previous night, as lecture notes will be posted the following morning !

To hand in your homework: At this point in the term, you will use Jupyter Notebook to create your homework solution(s) as blocks of code, with "Markdown" blocks for any descriptive text and comments (use #) to document the program inline. Download the notebook as a .ipynb file and email that as an attachment to:

James.Bell@bcm.edu, muyuan.Chen@bcm.edu, sludtke@bcm.edu

Homework #2

1. From grade school, you will recall that a factor is any integer that another integer is divisible by. For example, the factors of 12 are 1,2,3,4,6,12. There is the extended concept of prime factors which include only the factors which are prime numbers, but that is not our concern in this problem. Write a program to ask the user for an integer, and display all of the number's factors (exclude 1 and the number itself). No cheating and using a library function to do this. You should write the code using for loops, if statements and basic math.

2. Write a program to identify the winner of a rock,paper,scissors game. Ask the user what player 1 picked (rock, paper or scissors), then ask what player 2 picked. Finally, print the winner (player 1, 2 or tie)