Introduction to Python

L555

Dept. of Linguistics, Indiana University
Fall 2012

A Tour of Python

Today we're going to take a quick tour of Python.

- You'll see many different types of things, but nothing in-depth
- By the end, you should be able to:
  - Have some sense of what Python has to offer
  - Run some basic Python commands interactively
  - Write short Python programs and run them

Algorithms (reminder)

Definition

An algorithm is a set of instructions or a recipe for a computer to carry out.

Hello World

```
print 'Hello world.'
```

```
print 'Hello world.'
print 4 + 5
```


Data Types

Data **types** are the building blocks from which everything else is built

- Simple Types: numbers and strings
  - numbers: 3, 12.443, 89, ...
  - strings: "hello", 'manny', "34", ...
- Complex Types: lists and dictionaries (& sets & tuples)
  - lists: [1,2,3], [1,2,"a"], ["john", "george", "paul", "ringo"], ...
  - dictionaries: {"a":1, "b":16}, ...

Python is **dynamically typed**: you do not have to declare what type each variable is

Numbers

```
>>> 2+2
4
>>> 3/2
1
>>> 3/2.
1.5
```

Python has integers and floating point numbers (& complex numbers), and operations to convert between them:

```
>>> float(3)
3.0
>>> int(4.123)
4
```

Variables

What is a variable?

Definition

A variable is a name that refers to some value (could be a number, a string, a list etc.)

Example

Variables

What is a variable?

Definition

A variable is a name that refers to some value (could be a number, a string, a list etc.)

Example

1. Store the value 42 in a variable named *foo*
   ```
   foo = 42
   ```


Variables

What is a variable?

Definition

A variable is a name that refers to some value (could be a number, a string, a list etc.)

Example

1. Store the value 42 in a variable named *foo*
   ```
   foo = 42
   ```
2. Store the value of foo+10 in a variable named bar
   ```
   bar = foo + 10
   ```


Statements

What is the difference between an expression and a statement?

Definition

An expression *is* something, and a statement *does* something.

User Input

Example

1. Ask the user to input a name, and store it in the variable *name*
   ```
   name = raw_input('enter a number:  ')
   ```


User Input

Example

1. Ask the user to input a name, and store it in the variable *name*
   ```
   name = raw_input('enter a number:  ')
   ```
2. create a new string with a greeting
   ```
   greet = 'hello ' + name
   ```

User input

Example

1. Ask the user to input a number, and store it in the variable *foo*
   `foo = int(raw_input('enter a number:  '))`

User input

Example

1. Ask the user to input a number, and store it in the variable *foo*
   `foo = int(raw_input('enter a number:  '))`
2. Add foo and bar together
   `foo + bar`

User input

Example

1. Ask the user to input a number, and store it in the variable *foo*
   `foo = int(raw_input('enter a number:  '))`
2. Add foo and bar together
   `foo + bar`
3. Calculate the average of *foo* and *bar*, and save it in a variable named *avg*
   `avg = (foo + bar)/2`

Functions

What is a function?

Definition

A function is a mini-program. It can take several *arguments*, and *returns* a value.

Modules

What is a module?

Definition

Python is easily *extensible*. Users can easily write programs that extend the basic functionality, and these programs can be used by other programs, by loading them as a *module*

Example

1. load the math module
   `import math`

Modules

What is a module?

Definition

Python is easily *extensible*. Users can easily write programs that extend the basic functionality, and these programs can be used by other programs, by loading them as a *module*

Example

1. load the math module
   `import math`
2. Round 35.4 to the nearest integer
   `math.round(35.4)`

Modules

What is a module?

Definition

Python is easily *extensible*. Users can easily write programs that extend the basic functionality, and these programs can be used by other programs, by loading them as a *module*

Example

1. load the math module
   `import math`
2. Round 35.4 to the nearest integer
   `math.round(35.4)`
3. Round the quotient of foo and bar down to the nearest integer
   `math.floor(foo/bar)`

Saving and executing programs

Example

Script File: hello.py

```
# this script prints 'hello, world', to stdout
print "hello, world"
```

Run the program:

```
python hello.py
```


String Basics

- Strings must be enclosed in quotes (double or single)
- Strings can be concatenated using the + operator

Strings

- Many ways to write a string:
  - single quotes: `'string'`
  - double quotes: `"string"`
  - can also use `"""` to write strings over multiple lines:

```
>>> """<html>
... <body>
... something
... </body>
... </html>
... """
'<html>\n<body>\nsomething\n</body>\n</html>\n'
```

- There are string characters with special meaning: e.g., `\n` (newline) and `\t` (tab)
- Get the length of a string by the `len` function

String indices & slices

You can use slices to get a part of a string

```
>>> s = "happy"
>>> len(s)  # use the len function
5
>>> s[3]    # indexed from 0, so 4th character
'p'
>>> s[1:3]  # characters 1 and 2
'ap'
>>> s[:3]   # first 3 characters
'hap'
>>> s[3:]   # everything except first 3 characters
'py'
>>> s[-4]   # 4th character from the back
'a'
```


Creating/Editing Python files

Python files are simply text files, so we just need a text editor. Some options:

- Windows: Notepad or Wordpad → Save as plain text
  - Sometimes Windows is set up s.t. it forces you to add a `.txt` extension to your file.
  - This isn't a problem, but to get rid of it, (I think) you need to save as "All files" and also change your desktop settings so that they show file extensions
- Mac/Unix: pico, Emacs (or Aquamacs [which I use]), Vim, and probably others
  - I'll focus on emacs/aquamacs and IDLE (next slide) this semester, but use what you like ...

IDLE

Some text editors offer **syntax highlighting**, which shows variable names, indentation, etc.

Integrated Development Environments (IDEs) offer syntax highlighting, debugging features, streamlined code-running, etc.

- One IDE which comes with Python is IDLE (http://www.python.org/idle/doc/idlemain.html)
  - Windows: Once you've installed Python, this should be available from Start → Applications → Python27 → ...
  - Mac: Check the Applications folder (or use spotlight to find it)

Emacs

- emacs is a fairly basic text editor that can be run in a window or in the shell
- to start emacs:
  `emacs <filename>`
- to quit:
  `Ctrl-x Ctrl-c`
- save:
  `Ctrl-x Ctrl-s`
- search:
  `Ctrl-s`