SciComp with Py

Course Overview; Py IDEs, Distribs, Resources; Raspberry Pi & Raspbian Jessie; 1st Handshake with Py Functions

Vladimir Kulyukin
Department of Computer Science
Utah State University

Outline

- Course Overview
- Py IDEs, Distributions, Resources
- Raspberry Pi & Raspbian Jessie
- 1st Handshake with Py Functions

Course Overview

Course Title Change

- As of Fall 2016, CS3430 no longer covers PERL
- In Spring 2016, the undergraduate committee of the USU CS Department has approved changing the course title to *Scientific Computing with Python*
- Everything in this class will be done on the Raspberry Pi with Py 2 & Py 3 (more on Py 2 vs. Py 3 later in the lecture) on Raspbian Jessie

Weekly Schedule

- Weeks 1- 4: Py Basics
- Week 5: Linear Algebra
- Week 6: Coding Exam 1; Artificial Neural Networks
- Weeks 7 – 11: CVIP (Computer Vision & Image Processing)
- Week 11: Coding Exam 2
- Weeks 12 – 16: ML (Machine Learning)
- Week 17: Final Coding Exam

Logistics

- We will use Canvas throughout the semester
- Lectures & reading materials will be posted on Canvas as announcements
- You will submit your assignments & exams via Canvas as well
- Your HW & exam grades will be posted on Canvas as well

Class Attendance for Registered Students

- Class attendance is optional for registered students: this has been the standard policy in all CS classes I have ever taught
- Do not feel stressed if you cannot make it to class (but, remember the borderline grading cases: if I know you, I am more likely to give you a higher grade in a borderline grading case)
- If you come to class, be a good citizen – do not talk in class & turn off your smartphone

Class Auditing

- You may register to audit the course if you want to sit in on the class
- USU gets upset with instructors who allow students to attend a class w/o registering
- Other instructors may have allowed you to audit courses w/o registering, but it is against university policy and is unfair to other students who pay the tuition

Contacting Me by Email

- I will answer every email from a registered student
- But, If you email me on Saturday or Sunday, your email will not be answered until the following Monday
- Saturday is my R&D day
- Sunday is for my family

Py IDEs, Distributions, Resources

Py 2 vs. Py 3

- If you are serious about Python, you need to learn both: Py 2 & Py 3 are two very close dialects of the same language
- Py 2 is the legacy and the present, and you need to master it
- Py 3 is the present and the intended future
- Transition from Py 2 to Py 3 has not been smooth, because the Py 2 ecosystem has amassed a significant amount of quality software and porting it to Py 3 has been somewhat slow; another issue is that Py 3 is not backward compatible
- Py 2 still ships with many Linux & Mac distributions; some third-party libraries are not available or not as stable in Py 3 but this situation is slowly changing
- More info at https://wiki.python.org/moin/Python2orPython3

Will Py 2 Ever End?

- 1st time I heard about the death of Py 2 in 2008 (it didn't die)
- 2nd time I heard about the death of Py 2 in 2016 (it didn't die)
- PEP 373 (http://legacy.python.org/dev/peps/pep-0373/) claims that the end of life data (EOL) for Py 2 is 2020 (PEP stands for Python Enhancement Proposal)
- Red Hat, a Linux distribution, announced that it plans to support Py 2 for another 13 years
- Will Py 2 Ever End? I will believe it when I see it

Py's Strengths

- Free
- Portable: distributions & interpreters exist on all major OS's
- Mixable

  C/C++ programs can call Py scripts

  Py can link to C/C++ libraries
- OOP Support
- Large 3rd party libraries for Image Processing, Machine Learning, Gaming, CGI, SQL, Networking, etc
- Py 2 is forward compatible

Py's Weaknesses

- Py programs can run more slowly than their C/C++ counterparts
- If and when this happens, you have to ask yourselves two questions:

    Do I really need this for my particular application? In many cases, no!

    Can I port the bottleneck to C/C++? In many cases, yes!

- Py has syntactic features that take some getting used to: whitespace is a special character in Py and no braces
- Py 3 is not backward compatible: there is no **import from __past__**

Py Application Domains

- Systems Administration
- Component Integration
- Database Programming
- Rapid Prototyping
- Data Mining
- GUI/CGI (this is now being replaced by Javascript)
- Image Processing & Computer Vision
- Machine Learning
- Gaming
- Scientific Computing

What We Will Use

- We'll use Py 2.7 (2.7.6 or higher will do) and Py 3.4
- Linear Algebra, CVIP & ML will be done in Py 2 because it is more stable & boasts greater 3rd party libraries
- Which OS? We'll use Raspbian Jessie (OS system for Raspberry Pi)
- Raspbian Jessie ships with Py 2 and Py 3 installed

Py IDEs

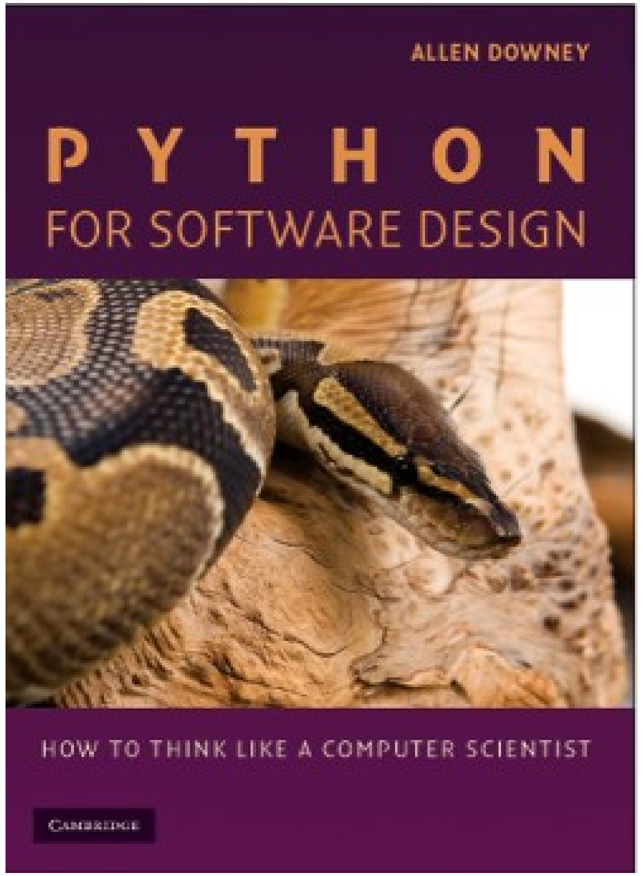
- On Linux (Ubuntu 14.04 LTS), I use IDLE for Py 2.7 or Py 3.4 and command line for execution and debugging and Emacs or XEmacs for editing; actually, any editor will do so long as it has Python syntax support
- Various versions of Python IDLE (default Py IDE for multiple platforms) are available from www.python.org/downloads
- Do I have recommendations for Mac OS or Windows?

  I have heard a lot of nice things from developer friends and students about PyCharm (https://www.jetbrains.com/pycharm/)

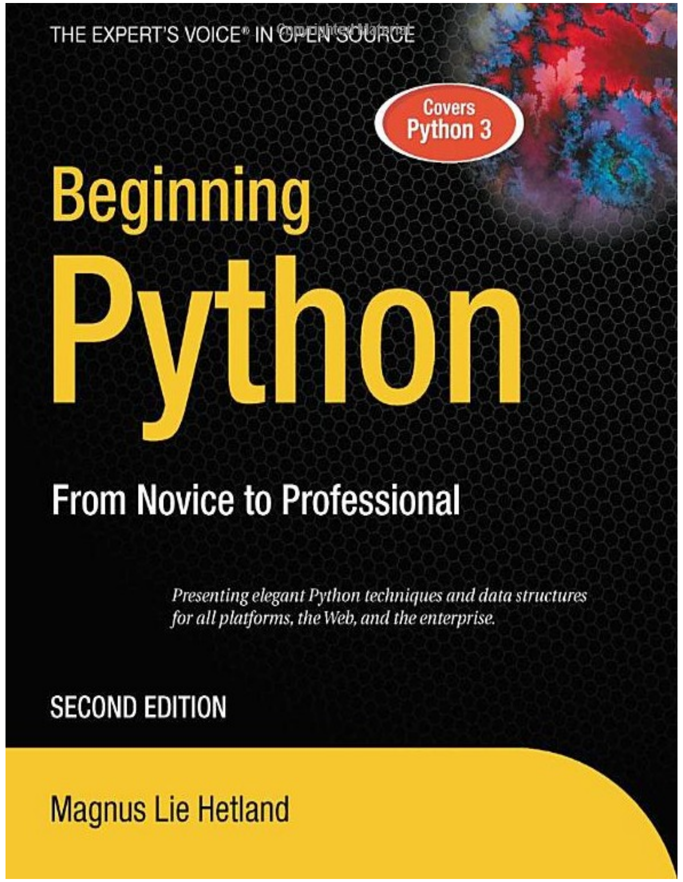
  I have also heard lots of nice things about JUPYTER (https://jupyter.org/) from researchers working on big data projects

Py Texts

No required texts for this class but you want want to take a look at these two books; Downey's book is free; Hetland's text used to be required for this course.



Allen Downey. “Think Python”


THE EXPERT'S VOICE® IN OPEN SOURCE
Covers Python 3
Beginning
Python
From Novice to Professional
Presenting elegant Python techniques and data structures for all platforms, the Web, and the enterprise.
SECOND EDITION
Magnus Lie Hetland

Magnus Hetland. “Beginning Python”

Py Online Resources

Go to www.python.org for everything (or almost everything) there is to know about Python. Documentation tab is very helpful. There are plenty of tutorials from basic to advanced. If you download your Py from www.python.org, it will automatically download IDLE (the vanilla integrated development environment for Python 2/3).

Py 3rd Party Libraries

We will use the following libraries in this course; we have created custom Raspbian Jessie distributions with these libraries already installed on the sdcards for RPi's.

NUMPY

SCIPY

MATPLOTLIB

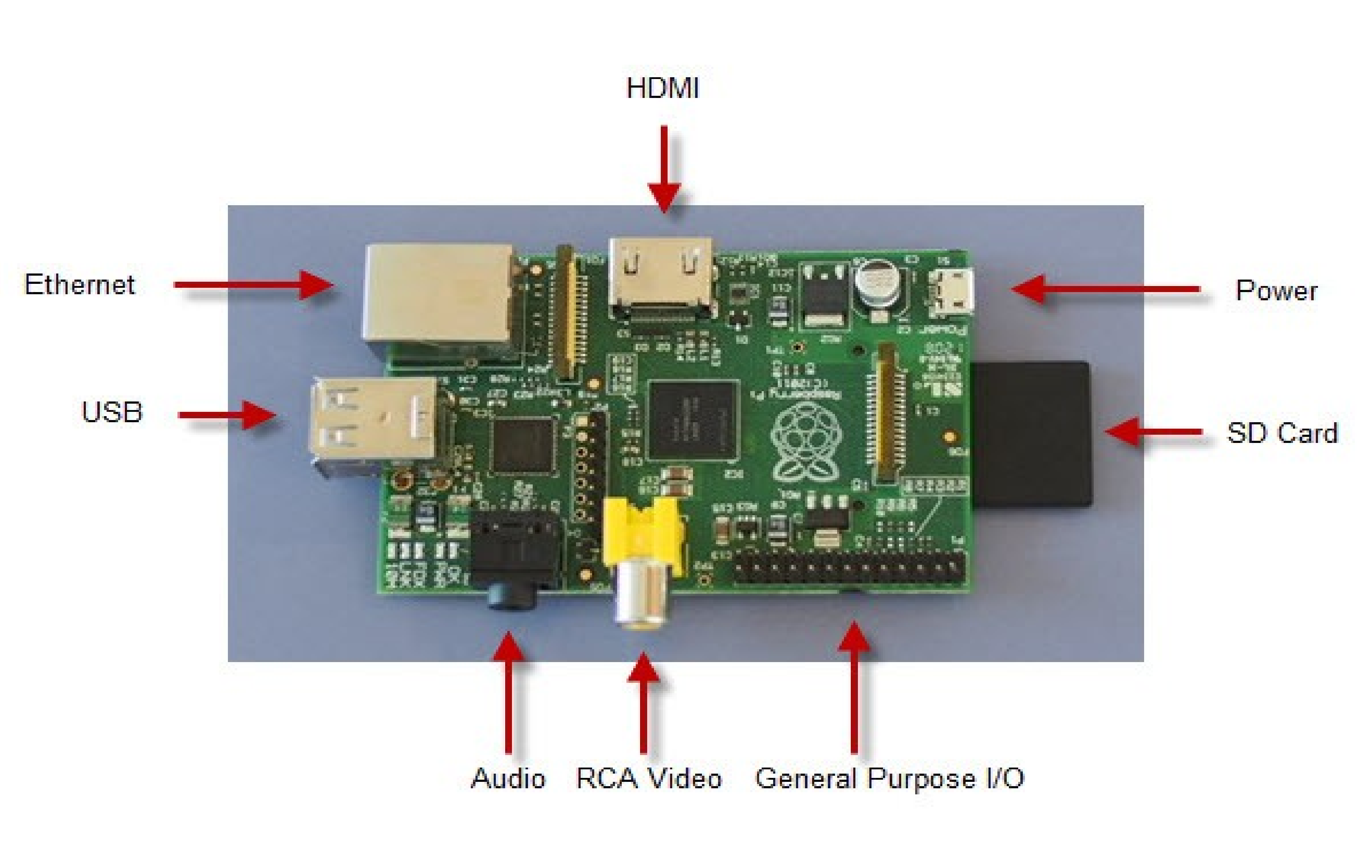
SCIKIT-LEARN

SCIKITS.TALKBOX

OpenCV 3.0

Raspberry Pi & Raspbian Jessie

Raspberry Pi



What Hardware You Will Need

- Power supply should be able to supply at least 700 mA but 1A or 2A are much better
- Displays: A low-cost 22-inch LCD makes an adequate display; a TV screen is fine; in general, any HDMI friendly monitor will do
- Storage: an SD card (32/64G or greater)
- Input: USB keyboard and mouse

Working w/ Raspberry Pi

There are several ways to turn your RPi into a regular computer:

1) Get an HDMI-friendly monitor – by far, the most common option

2) Get a custom Raspberry Pi monitor at adafruit.com – more expensive but your Pi will become like a laptop

3) Use your laptop to remotely connect to your RPi: on Linux and Mac OS, you can use ssh; on Windows, you will have to use putty

Basic Linux Commands

- **ctrl+alt+t** starts a command terminal
- If you have not worked on Linux, get comfortable with the following Linux commands (there is abundant online documentation on each of them):

  ls – list the contents of the current directory

  cd – change directory (/home/pi/ is the default home folder on RPi)

  pwd – present working directory

  mkdir – make directory

  rm – removes a file

  rmdir – removes a directory

Internet Connectivity

- Wi-Fi on the pi can be pretty slow: Bluezone tends to be faster than Verizon Wireless
- If you may want to download iceweasel, a more efficient web browser, with the following three commands in the command line window:

```
$ sudo apt-get update
$ sudo apt-get upgrade
$ sudo apt-get install iceweasel
```

- If you get frustrated with the slowness of Wi-Fi at home, I recommend just developing on the Pi, saving your solution on a USB drive, and using your laptop/PC to upload it to Canvas from the USB drive

Running Py Code

- Py interpreter is an interactive program that allows you to work with Py source w/o having to create, edit, save, and compile source files
- As you read Py docs, I suggest that you keep the Py interpreter window running and try code snippets right away
- Most Py IDEs make interpreters (aka shells) easily available

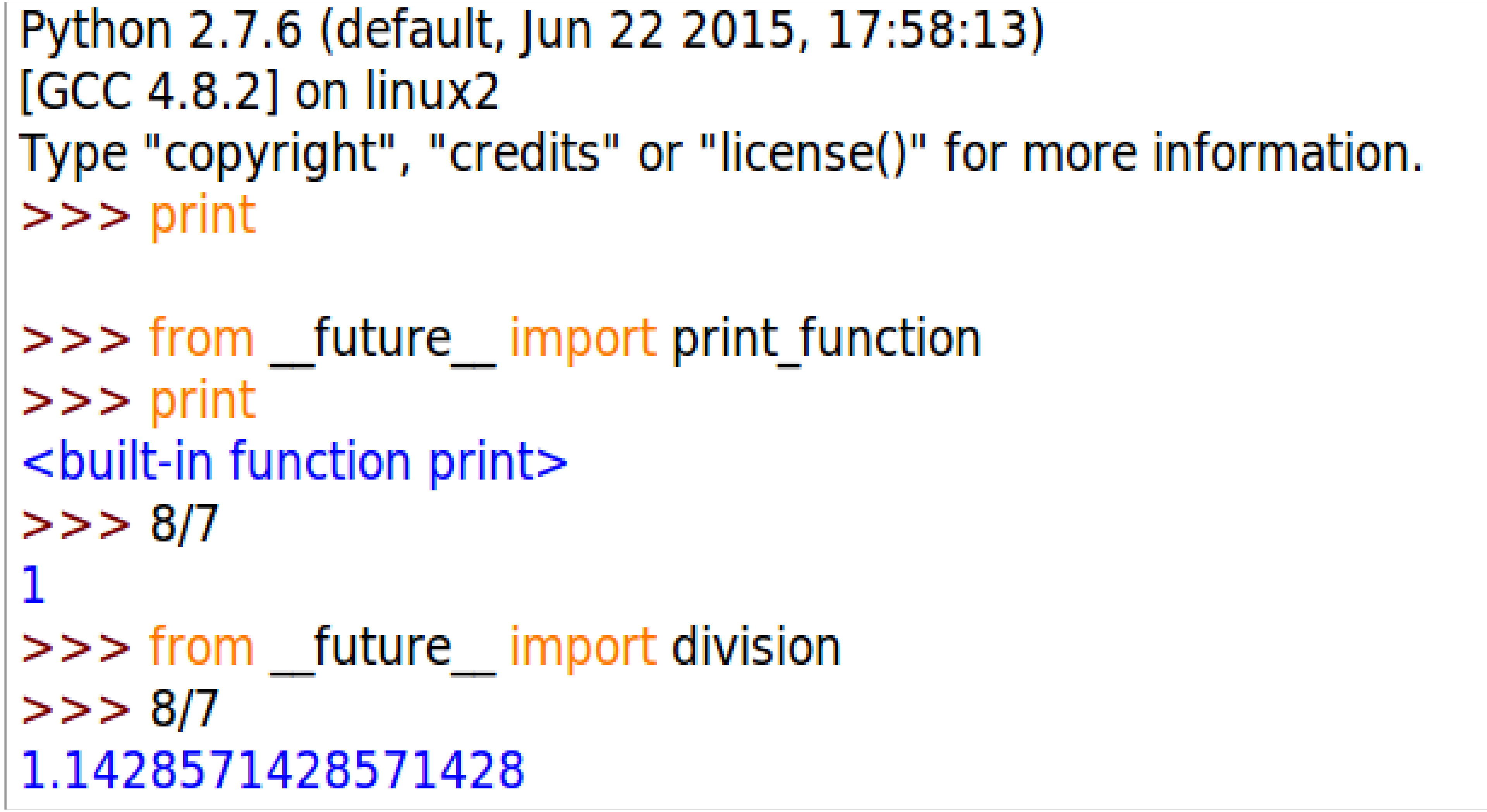
Py Shell

Below is a sample interaction in Py 2 IDLE Shell:

```
>>> print 'Hello, Python!'

Hello, Python!

>>> “Hello, Python!”

Hello, Python!

>>> 5

5

>>> 5 + 10

15
```

Forward Compatibility of Py 2

__**future**__ is a module that Py 2 programmers can use to import Py 3 features



```
Python 2.7.6 (default, Jun 22 2015, 17:58:13)
[GCC 4.8.2] on linux2
Type "copyright", "credits" or "license()" for more information.
>>> print

>>> from __future__ import print_function
>>> print
<built-in function print>
>>> 8/7
1
>>> from __future__ import division
>>> 8/7
1.1428571428571428
```

Running Python File Code in IDLE

Start IDLE, create a new file, type the code below, save it as hello_python.py:

```
#!/usr/bin/python

print 'Hello, Python!'
```

When ready, press F5 or Run | Run Module:

```
>>>=============== RESTART ==============

>>>

Hello, Python!
```

Running Py Interpreters in Command Terminals

- You can also start a Py 2/3 interpreter in a command terminal
- Here is how:

  - open a terminal (**ctrl+alt+t**)

  - at the prompt type **python** and hit enter if you want to work with Py2

  - at the prompt type **python3** and hit enter if you want to work Py3

1st Handshake with Py Functions

Adding Numbers in Py 2

This is the so called shebang line

add2.py

```
#!/usr/bin/python
def add2(x, y):
    return x + y
```

add3.py

```
#!/usr/bin/python
def add3(x, y, z):
    return x + y + z
```

Adding Numbers in Py 3

Note that the shebang line explicitly refers to Py 3

add2.py

```
#!/usr/bin/python3
def add2(x, y):
    return x + y
```

add3.py

```
#!/usr/bin/python3
def add3(x, y, z):
    return x + y + z
```

Comments on Py Code

- No types for variables: Py is a run-time typing (dynamic typing) programming language
- No curly braces like in C/C++/Java to mark code blocks: code blocks are marked with indentation
- No semicolons at the end of statements
- Py is case-sensitive: large-cap letters are different from small-cap letters in function names, file names, strings, arrays, etc.
- White space & indentation MATTERS in Py