
General Assembly

PYTHON PROGRAMMING 101

Ivan Hernandez, Ph.D

GOALS FOR THE SESSION

- Touch on fundamental Python programming techniques and tools
- Discover the key features of Python and how it compares to other programming languages
- Discuss its applications in data analysis and the types of problems it can solve
- Learn to code in Python
 - Variables
 - Lists, Dictionaries, Tuples
 - Functions
 - Program Control
 - Classes
- Apply your new skills to solve problems

WHAT IS PYTHON

WHAT IS PYTHON

- **Python is a programming language that emphasizes productivity and readability**
- **Primary Features of Python:**
 - **Interpreted** - Can execute code “on the fly”
 - **Object-oriented** - Can create “modules” of code
 - **High-level** - Can write code in terms of the concepts it represents
 - **Dynamic semantics** - Can have more flexibility with creating variables

PYTHON IS INTERPRETED

- **Interpreted:** Code implementations execute instructions without having to compile them into machine-language instruction.
- Some languages require you to transform the code first into a specific file. This file is in the machine's language (assembly).
 - You then run the compiled file to execute the program
 - You cannot just type a single line and get an instant result
- **Because it is interpreted, Python allows you to**
 - **run code line-by-line**
 - **see errors right away**
 - **interact with the variables you just created**

PYTHON IS OBJECT ORIENTED

- **Object-oriented (OO):** Instead of concentrating on isolated "actions", object-orientation enables us to focus on "objects" that contain data (attributes)
 - In Objected-oriented languages we create variables that hold a collection of functions
 - Example: make a “chat” object
 - One function: login()
 - Another function: send_message()
 - Another function: add_friend()
 - Another function: logout()
 - Now we could have just make many objects called “chat” for different users
 - **OO makes it easier to write and reuse code in other programs**

PYTHON IS HIGH-LEVEL

- **High-level programming:** Python does not require you to interact directly with the computer's architecture
 - In some languages, you have to explicitly deal with computer's hardware such as registers, memory addresses and call stacks
 - Python automatically handles those aspects, and allows you to deal more the conceptual parts of programming:
 - variables
 - arrays
 - arithmetic expressions
 - loops
- **Higher level languages like Python trade efficiency for ease of programming**

PYTHON IS DYNAMIC

- **Dynamic semantics:** Once data has been specified, the machine must be instructed to perform operations on the data.
- In some languages you have to explicitly write what type of information each variable hold
 - If you wanted to make a variable called “data” that would hold an integer, you would have to write:
 - `int data`
 - `data=2`
- Python automatically determines the type of variable you are using
- **Dynamic semantics makes Python code easier to write and allows more flexibility**

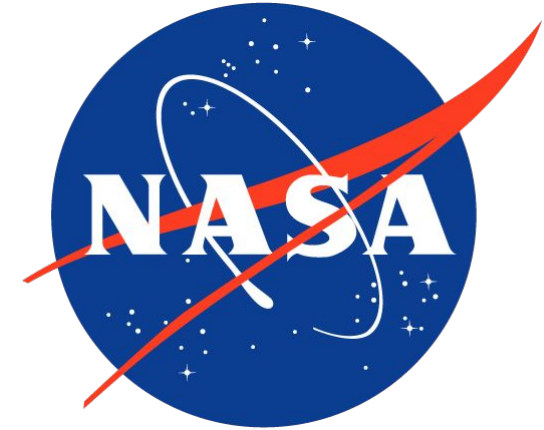
WHAT CAN PROGRAMMING LANGUAGES DO

- **What can programming languages do?**
 - **Analyze** (Text, Numbers)
 - **Retrieve** (Information from files, webpages, databases)
 - **Send** (Information through e-mails, databases)
 - **Create/Edit** (Text files, Images)

WHO USES PYTHON?

Google


Pinterest



Quora



You Tube



venmo



HOW IS PYTHON USED

‣Scripting Language

- Automating simple tasks (testing, building, deployment, monitoring)
- Acts as “glue” that holds other code together (can interact with C and Java code)

‣Website Design

- Django
- Flask

‣Analysis

- Machine Learning
- GIS
- Interfacing with Databases

‣Graphics

- Interfacing with Maya and Renderman
- Visualizing data analysis

EXAMPLES OF PYTHON IN INDUSTRY AND ACADEMIA

‣Industry

- [Drug discovery](#)
- [Financial services](#)
- [Films and special effects](#)

‣Academia

- [Gravitational waves](#)
- [Scientific visualisation](#)
- [Biomolecule simulation](#)

WHY PYTHON?

BENEFITS OF PYTHON

▸Benefits of using Python

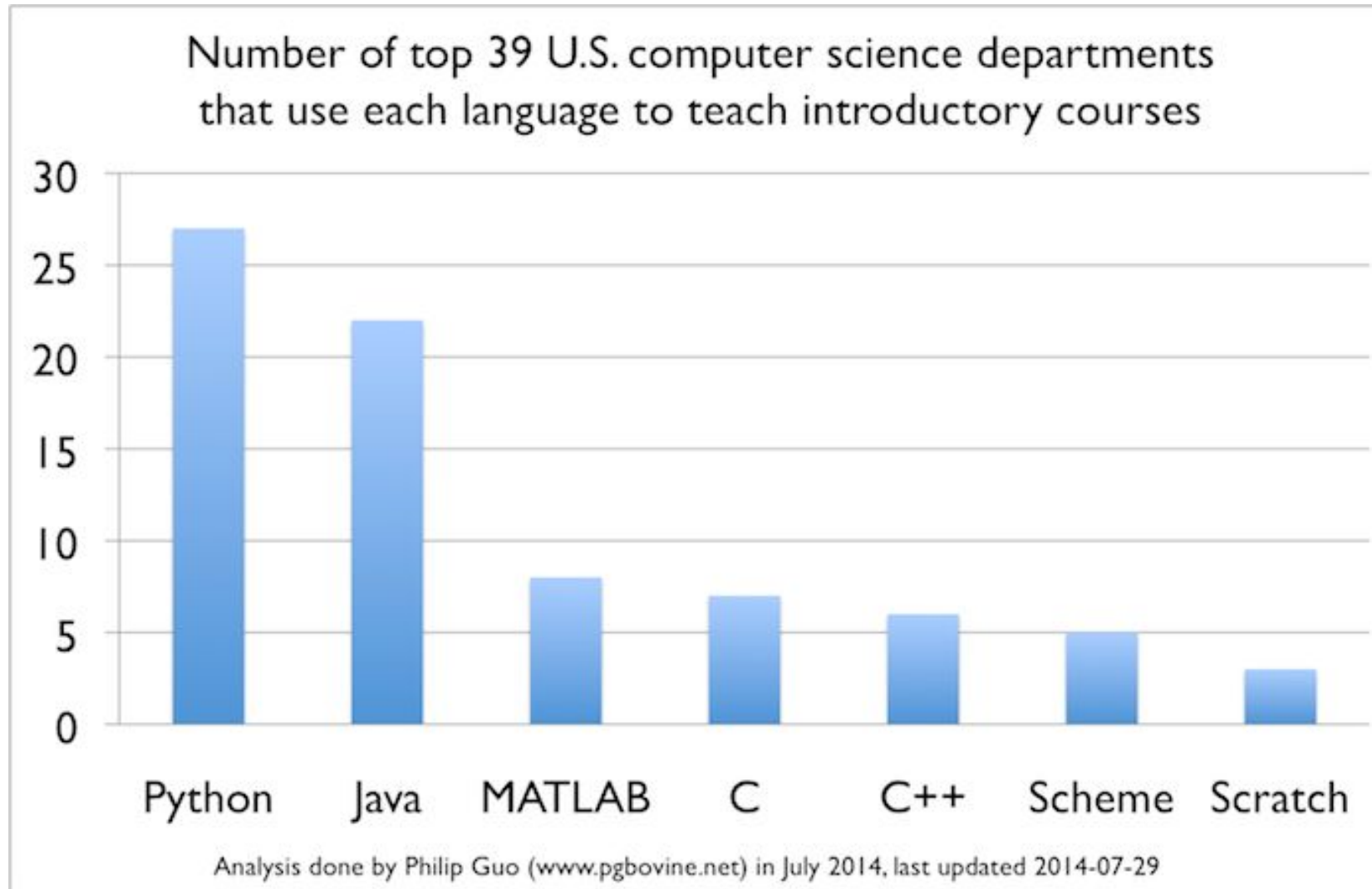
▸Easy to learn

▸Fast Coding

▸Readable

▸Popularity

PYTHON IS EASY TO LEARN



PYTHON IS READABLE

```
x = 12 - 2
```

```
y = "Hello"
```

```
z = 3.45
```

```
if z == 3.45:
```

```
    x = x + 1
```

```
    y = y + " World"
```

```
print x
```

```
print y
```

PYTHON IS FAST TO CODE: C VS. JAVA VS. PYTHON

Display the text, “Hello World!”

```
#include <stdlib.h>
#include <stdio.h>

int main(void)
{
    printf("Hello world!\n");
    return EXIT_SUCCESS;
}
```

C

```
public class HelloWorld
{
    public static void main(String[] args)
    {
        System.out.println("Hello world!");
    }
}
```

Java

```
print "Hello world!"
```

Python

PYTHON IS FAST TO CODE: C VS. JAVA VS. PYTHON

Determine if a string is a palindrome

```
#include <string.h>

int is_palindrome(const char *s)
{
    int i,l;
    l = strlen(s);
    for(i=0; i<l/2; i++)
    {
        if ( s[i] != s[l-i-1] ) return 0;
    }
    return 1;
}
```

C

```
public static boolean is_palindrome(String testMe){
    StringBuilder sb = new StringBuilder(testMe);
    return testMe.equals(sb.reverse().toString());
}
```

Java

```
def is_palindrome(s):
    return s == s[::-1]
```

Python

PYTHON IS POPULAR

PYPL	Tiobe	CodingDojo	IEEE Jobs	IEEE Open	IEEE Trending
Java	Java	SQL	C	C++	C
Python	C	Java	Java	Python	C++
PHP	C++	JavaScript	Python	C	Python
C#	C#	C#	C++	Java	Java
JavaScript	Python	Python	JavaScript	Swift	Swift
C++	JavaScript	C++	C#	JavaScript	R
C	PHP	PHP	PHP	C#	JavaScript
Objective-C	Assembly	iOS	Ruby	Ruby	Ruby
R	VB.NET	Ruby/Rails	HTML	PHP	Go
Swift	Perl		Swift	Ruby	C#
Matlab	Delphi		Assembly	HTML	PHP
Ruby	Ruby		Ruby	Go	Scala
VBA	Swift		Scala	Scala	Arduino
Visual Basic	Objective-C		Shell	Objective-C	Assembly
Scala	Matlab		Perl	Shell	Shell
Perl	Groovy		SQL	Arduino	Objective-C
lua	Visual Basic		Objective-C	Assembly	HTML
Delphi	Ruby		Matlab	Matlab	Rust
Go	Go		Visual Basic	Lua	Haskell
Haskell	PL/SQL		Go	Perl	Visual Basic

Combined chart of language popularity metrics.

USING PYTHON

PYTHON VERSIONS

▸ Versions of Python

▸ Python 2 (Usually Python 2.7)

- More widely used (Pre-installed in Macs and Linux)
- More support from community
- More compatible libraries

▸ Python 3

- More support officially (under active development)
- More consistent language
- More memory efficient

▸ 2.7 is recommended for people beginning Python because of the wider support and most libraries can be run as is

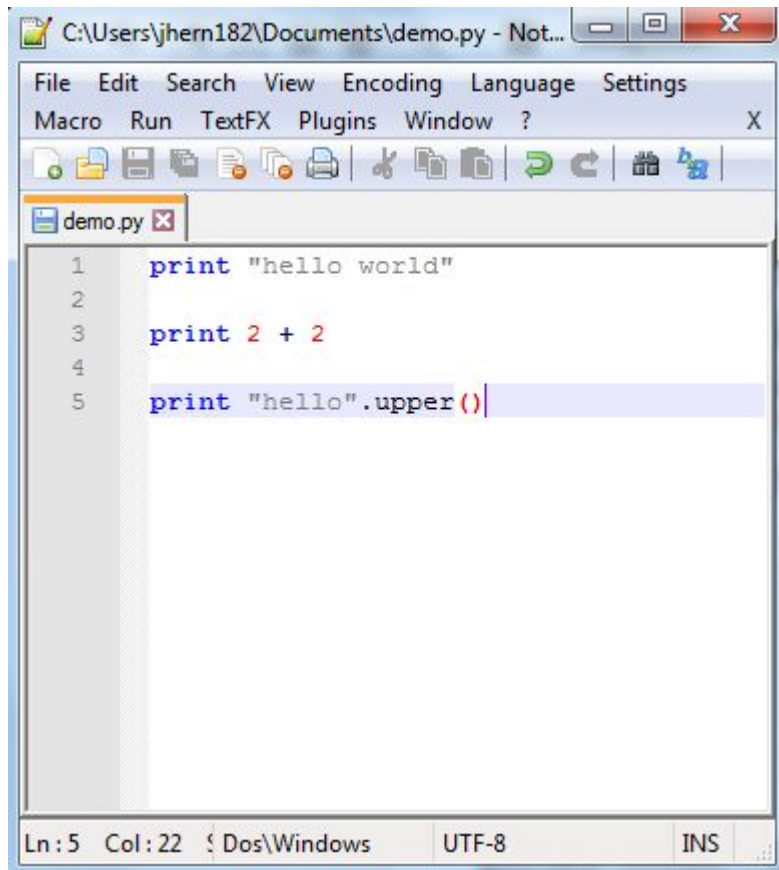
DOWNLOADING PYTHON

- You can download Python from the official website:
<https://www.python.org/downloads>
- You can also download a Python distribution:
 - All-in-One
 - Contains commonly used libraries pre-configured
- Recommended Distribution: Anaconda
 - <https://www.continuum.io/downloads>
 - Easiest to install new libraries
 - Contains a massive number of libraries
 - Contains different graphical interfaces to Python

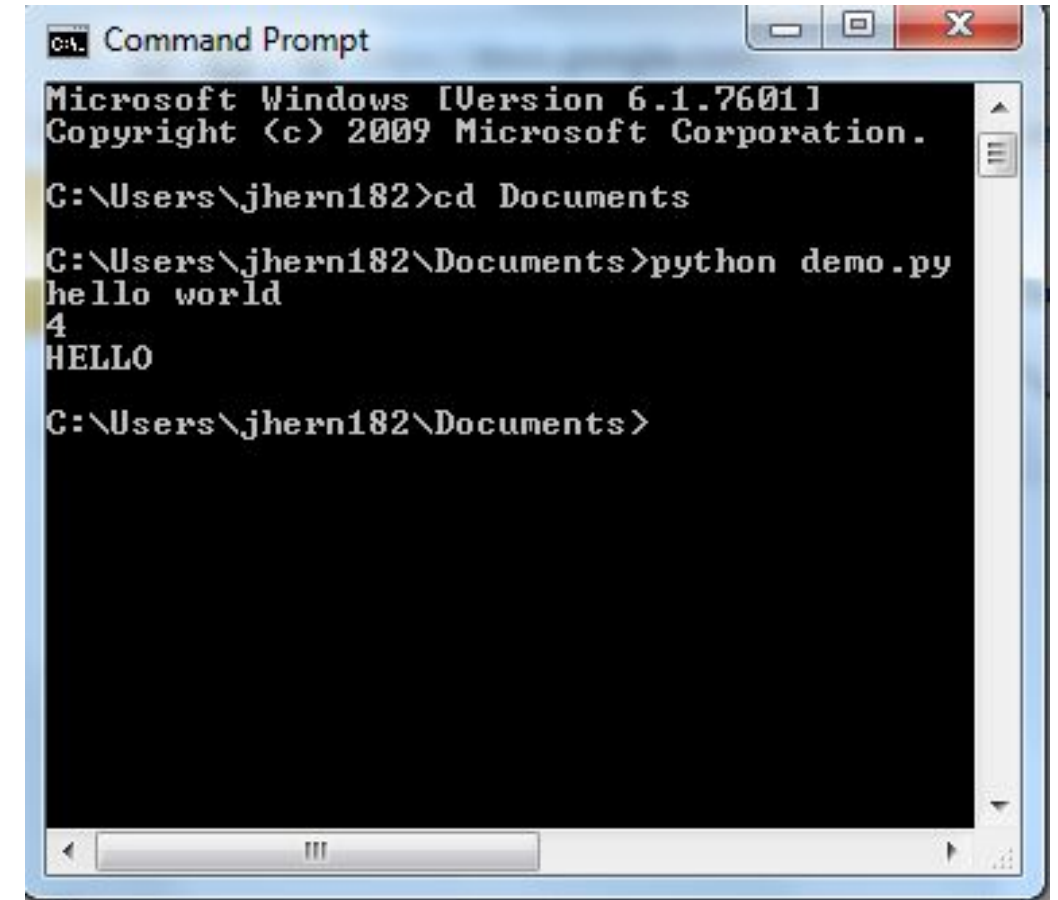
USING PYTHON - SIMPLEST METHOD

- **Load** text-editor (Notepad, Notepad++, Sublime, Etc)
- **Write** the full code (also called a “script”)
- **Save** code as a text file and give it “.py” extension
- **Open** command prompt/terminal and navigate to folder where file is saved
- **Type and run:** *python nameofprogram.py*

USING PYTHON - SIMPLEST METHOD



```
1 print "hello world"
2
3 print 2 + 2
4
5 print "hello".upper()
```



```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.

C:\Users\jhern182>cd Documents

C:\Users\jhern182\Documents>python demo.py
hello world
4
HELLO

C:\Users\jhern182\Documents>
```

USING PYTHON - ADVANCED METHODS

- There are other ways of running Python code
- We might want to **type/run code line-by-line** and get an instant result
 - Use **Ipython**
- We might want to **run the code in the same window as our text editor**
 - Use **Spyder**
- We might want to **display the results below specific segments of the code**
 - Use **Jupyter**

PYTHON GRAPHICAL INTERFACES

- **Included in Anaconda are three commonly used graphical interfaces with Python**
 - **Ipython:** For executing Python commands line by line and interacting with the results
 - **Spyder:** For creating larger scripts and executing them in the same window
 - **Jupyter:** For displaying code with its results, all in the same document

IPYTHON

```
IPython 3.2.0 -- An enhanced Interactive Python.
Anaconda is brought to you by Continuum Analytics.
Please check out: http://continuum.io/thanks and https://anaconda.org
?          -> Introduction and overview of IPython's features.
%quickref  -> Quick reference.
help       -> Python's own help system.
object?    -> Details about 'object', use 'object??' for extra details.

In [1]: print("Hello world!")
Hello world!

In [2]: 2 * 3
Out[2]: 6

In [3]:
```

IPython lets you run short segments of code and instantly see their the results

IPython is ideal when you have a simple task, want to debug, or want to learn Python's commands

SPYDER

The screenshot displays the Spyder Python IDE interface. The main window is divided into three panes:

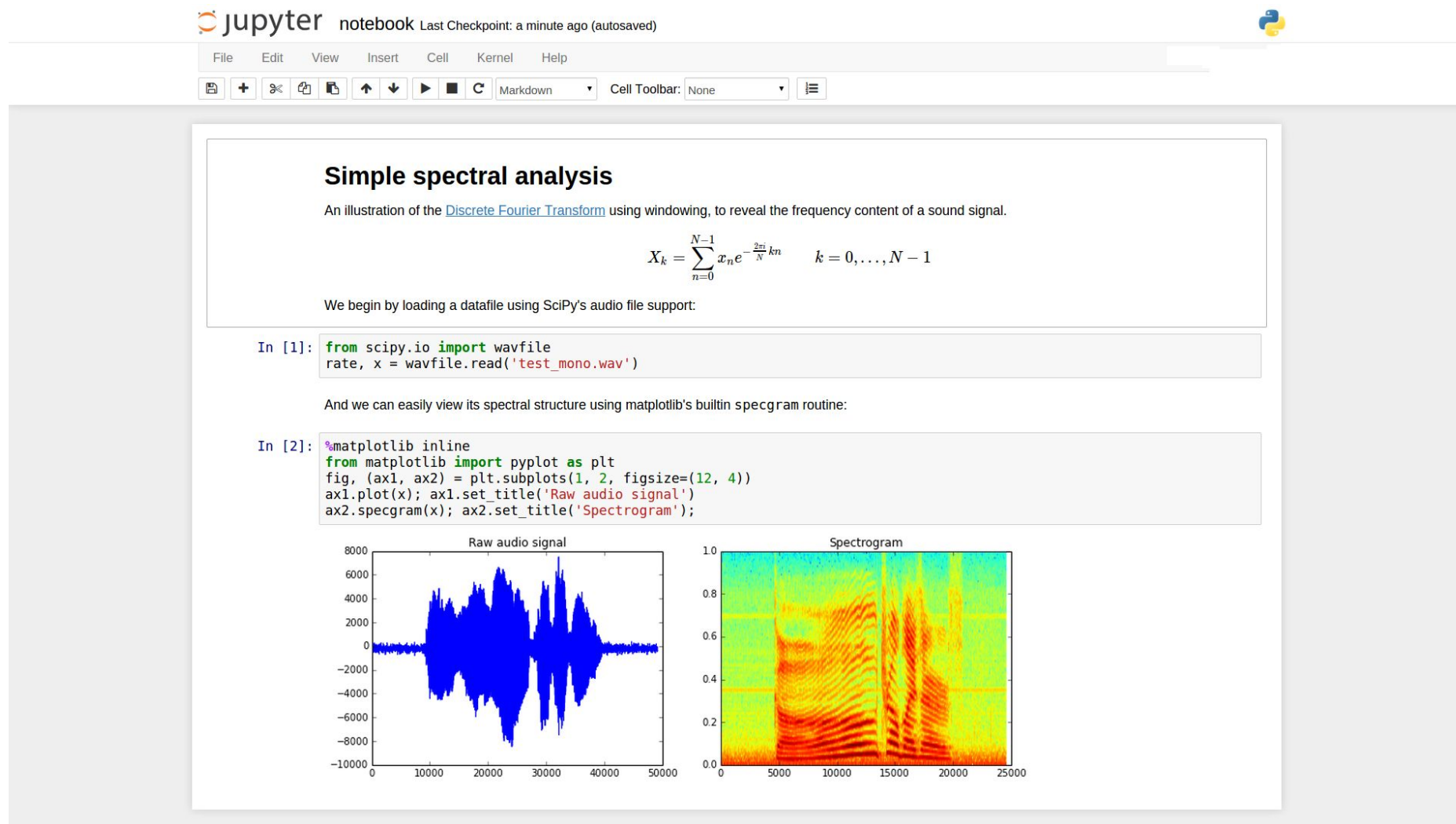
- Editor (Left):** Contains a Python script named `questionnaire.py`. The script uses `pandas` to read a CSV file, process data, and save it to a new CSV file. The script includes comments in Swedish and English.
- Variable explorer (Top Right):** A table showing the variables defined in the script. The variables are: `firstnames` (list, size 4), `frame` (DataFrame, size (234, 26)), `i` (int, size 1), `idx` (int, size 1), `iv` (list, size 20), `lastnames` (list, size 2), `names` (list, size 26), and `row` (unicode, size 1). The values are displayed in the right column.
- IPython console (Bottom Right):** Shows the execution of the script. It displays the output of the `print` statements and the error message: `UnicodeEncodeError: 'ascii' codec can't encode characters in position 11-12: ordinal not in range(128)`. The console also shows the warning: `SettingWithCopyWarning: A value is trying to be set on a copy of a slice from a DataFrame`.

The status bar at the bottom indicates the file is named `Run file`, the permissions are `Permissions: RW`, the end-of-lines are `LF`, the encoding is `UTF-8`, the line number is `Line: 47`, the column number is `Column: 1`, and the memory usage is `Memory:`.

Spyder lets you create a script (left) and run the script (bottom-right) in the same window.

You can also see the variables you've created (top-right)

JUPYTER



Jupyter lets you create a notebook (document) that has code segments and the results from the code segments right below.

These documents can be shared easily and edited by the audience to get new results

ONLINE OPTIONS TO ACCESSING PYTHON

- Even if you do not have Python installed, you can still program in it on your computer using free cloud-based options
- Similar to IPython:
 - <https://repl.it/languages/python>
 - <https://www.pythonanywhere.com/try-ipython>
- Online alternative to Jupyter
 - <https://try.jupyter.org>
 - <https://tmpnb.org>

CODING IN PYTHON

CODING IN PYTHON

- **Case (Upper and lower) matter in Python**
 - If you say: `a=2`
 - Asking for “a” will return 2
 - Asking for “A” will give you nothing unless you’ve assigned a value to it
- **Each new line is a new command**
 - **If you want Python to display “hello” and then display “goodbye”, you have have to say:**
 - `print “hello”`
 - `print “goodbye”`
 - **Cannot say:** `print “hello” print “goodbye”`
 - You can use a semicolon to indicate a new line

CODING IN PYTHON

‣ **Indentation matters in Python**

- Typically used after a colon
- Used similar to how curly brackets {} are used in other languages
- Cannot indent a line unless needed for the code

‣ **Spacing (mostly) doesn't matter in Python**

- On the same line, you can have as many spaces as you want
- All three lines will run the same

‣ `print 2+2`

‣ `print 2 + 2`

‣ `print 2 + 2`

USING JUPYTER

INSTRUCTIONS ON USING JUPYTER

‣ **In Jupyter**

‣ **Open a notebook / Create a new notebook**

‣ **Write the code**

‣ **Run the code**

INSTRUCTIONS ON USING JUPYTER

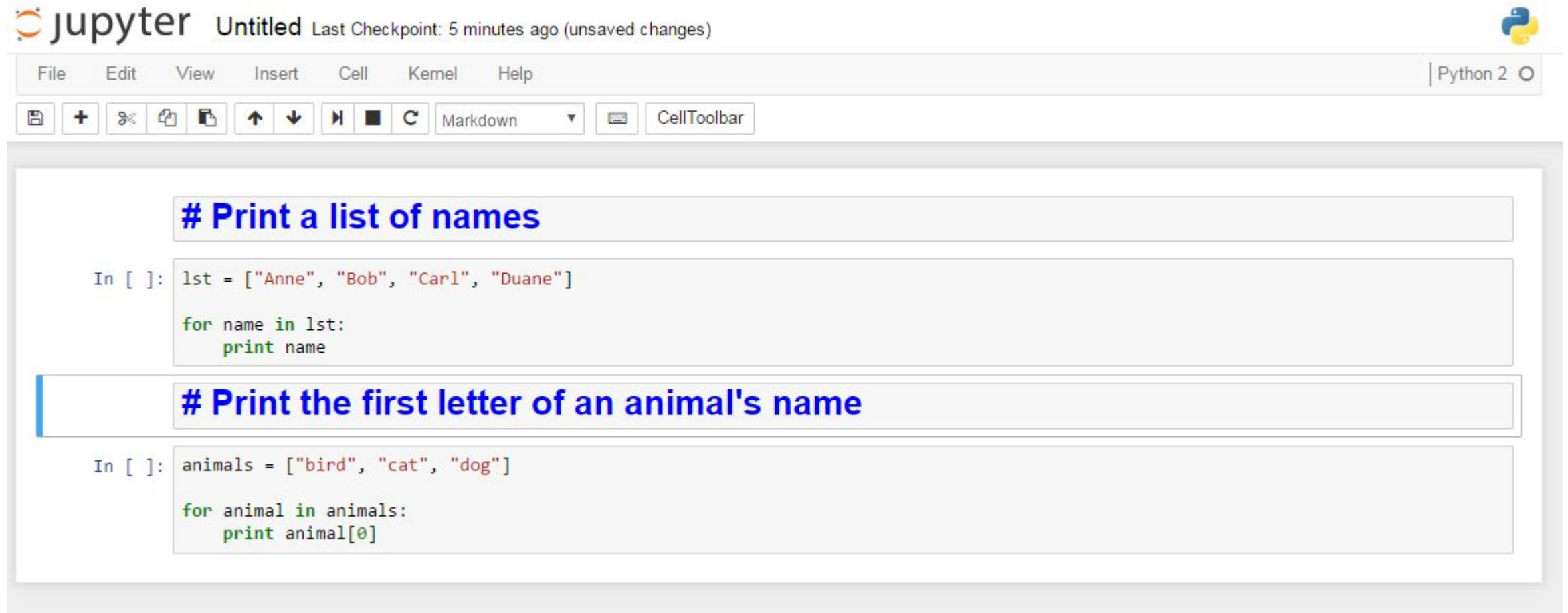
► Open a notebook / Create a new notebook



The screenshot displays the JupyterLab web interface. At the top left is the Jupyter logo, and at the top right, it says "Hosted by Rackspace" with a Rackspace logo. Below the header are three tabs: "Files", "Running", and "Clusters". The "Files" tab is active, showing a file browser. A message above the file list says "Select items to perform actions on them." The file list contains several folders ("communities", "datasets", "featured") and notebooks ("Welcome Julia - Intro to Gadfly.ipynb", "Welcome R - demo.ipynb", "Welcome to Haskell.ipynb", "Welcome to Python.ipynb", "Welcome to Spark with Python.ipynb", "Welcome to Spark with Scala.ipynb"). On the right side of the file list, there are buttons for "Upload", "New", and a refresh icon. The "New" button is clicked, opening a dropdown menu. The menu is divided into two sections: "Text File", "Folder", and "Terminal" in the top section; and "Notebooks", "Apache Toree - Scala", "Bash", "Haskell", "Julia 0.3.2", "Python 2", "Python 3", "R", and "Ruby 2.1.5" in the bottom section. The "Python 2" option is highlighted, and a tooltip next to it says "Create a new notebook with Python 2".

INSTRUCTIONS ON USING JUPYTER

► Write the code



The screenshot displays the Jupyter Notebook web interface. At the top, the Jupyter logo is followed by the text "jupyter Untitled" and "Last Checkpoint: 5 minutes ago (unsaved changes)". A Python logo is in the top right corner. Below this is a menu bar with "File", "Edit", "View", "Insert", "Cell", "Kernel", and "Help". To the right of the menu bar is a status bar showing "Python 2" and a refresh icon. Below the menu bar is a toolbar with icons for saving, creating a new cell, deleting a cell, duplicating a cell, moving a cell up/down, running the cell, and other actions. A dropdown menu is set to "Markdown".

The main area contains two code cells. The first cell has a blue header "# Print a list of names" and contains the following Python code:

```
In [ ]: lst = ["Anne", "Bob", "Carl", "Duane"]  
  
for name in lst:  
    print name
```

The second cell has a blue header "# Print the first letter of an animal's name" and contains the following Python code:

```
In [ ]: animals = ["bird", "cat", "dog"]  
  
for animal in animals:  
    print animal[0]
```

INSTRUCTIONS ON USING JUPYTER

► Run the code

Tip: Press Ctrl and Enter at the same time to run a section of code in Jupyter

Print a list of names

```
In [1]: lst = ["Anne", "Bob", "Carl", "Duane"]  
  
for name in lst:  
    print name
```

```
Anne  
Bob  
Carl  
Duane
```

Print the first letter of an animal's name

```
In [2]: animals = ["bird", "cat", "dog"]  
  
for animal in animals:  
    print animal[0]
```

```
b  
c  
d
```

CODE-ALONG IN JUPYTER

- **Launch jupyter** on your computer (Three options)
 - Go to the start menu or where your programs are saved, and click on the jupyter notebook icon
 - Or, open the command line and type: jupyter-notebook
 - Or, Go to the following webpage: <https://try.jupyter.org>
- **Download** the following file to where your notebook folders are: <https://goo.gl/It7gy8>
- **Open** it in the Jupyter notebook file explorer (Or click “Upload” at top-right and select the file from its folder)

NEXT STEPS

PYTHON LIBRARIES

- **Explore different libraries**

- **Python Module of the Week:** <https://pymotw.com/2/contents.html>

- **Python Package library:** <https://pypi.python.org/pypi?%3Aaction=browse>

- **GitHub:**

- <https://github.com/vinta/awesome-python>

- <https://github.com/trending/python?since=monthly>

HOW TO INSTALL ADDITIONAL LIBRARIES

‣ **Option 1: Install via Anaconda command line Installer**

‣ Go to the command line\terminal and type: `conda install libraryname`

‣ **Option 2: Install via pip command line Installer**

‣ Go to the command line\terminal and type: `pip install libraryname`

‣ **Option 3: Install via Anaconda Package Manager**

‣ Open the Anaconda Navigator

‣ Click on “Environments”

‣ Select “Not Installed” from the drop-down menu

‣ Scroll to find library or use search box

PYTHON NEWSLETTERS

▸ Stay Informed

- PyCoders: <http://pycoders.com>
- Python Tips: <http://newsletter.pythontips.com>
- Python Weekly: <http://www.pythonweekly.com>

PYTHON JOBS

‣ Explore the Python Job Market

‣ <https://www.python.org/jobs>

‣ <http://pythonjobs.github.io>

‣ <http://jobs.pythonweekly.com>

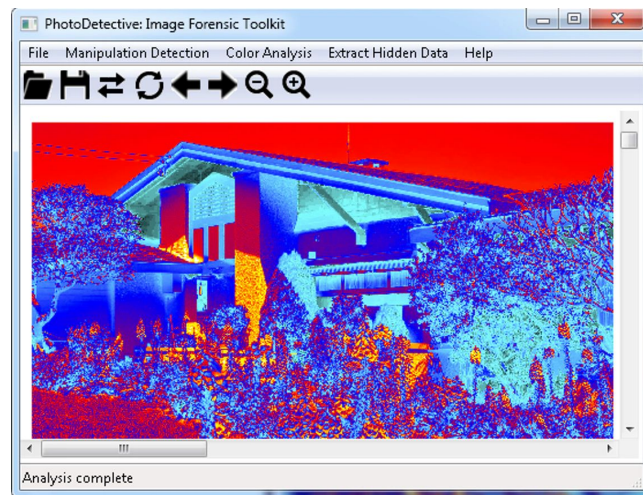
APPLY PYTHON: APPS

► Create Applications

► Graphical Interfaces: <https://wiki.wxpython.org/Getting%20Started>

► Desktop Apps (Mac, Win, Linux): <http://www.pyinstaller.org>

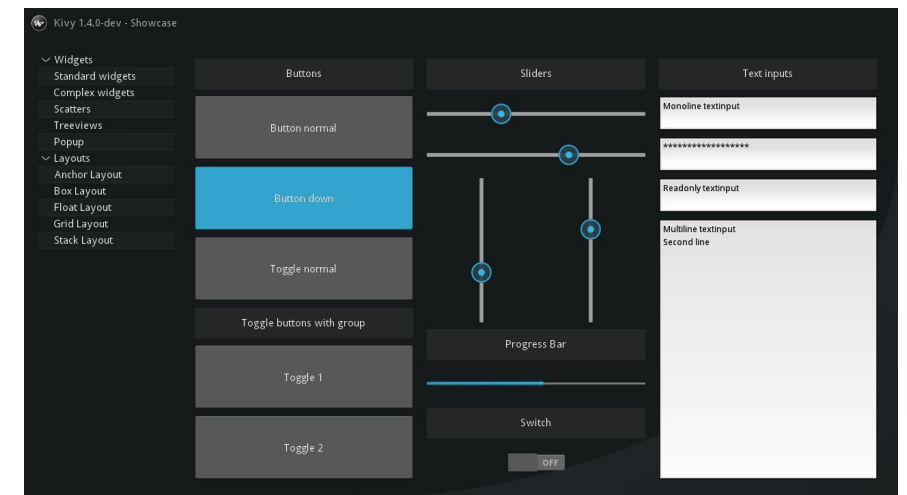
► Smartphone Apps: <https://kivy.org>



Forensic Image Analysis Program
GUI created with Wx Python



PyInstaller to convert Python
programs to executables



Interface of Kivy to create Smartphone Apps
from Python scripts

APPLY PYTHON: DATA SCIENCE

- **Data Science:** How can you use Python to extract meaning from Data?
 - Make predictions
 - Understand underlying segments



- **Workshop:** <https://generalassemb.ly/education/data-science-101/chicago/37465>
- **Part-time:** <https://generalassemb.ly/education/data-science>
- **Immersive Course:** <https://generalassemb.ly/education/data-science-immersive>

SUMMARY

SUMMARY

- Python provides a language that is easy to learn, fast to code, readable, popular
- As a programming language Python offers the ability to analyze, retrieve, send, create, and edit information
- Python is accessible on most operating systems for free and Anaconda provides an all-in-one distribution
- Python code can be created and executed via Spyder, Ipython, or Jupyter notebooks
- Knowing Python opens the door to many applications such as creating webpages, designing apps, and conducting data science, among others