



Learn Python Programming

TCEF IT Club, 2018

Class Info

- Learn Python from beginning
- Class time: Saturday 7pm – 8:30pm EST
- Reference book: **Python for Kids: A Playful Introduction To Programming (Jason R. Briggs)**
- Instructors

Learning Plan

- ✓ Lectures in ZOOM Cloud meeting (45 minutes)
 - ✓ Hands-on Lab with TA (Teaching Assistant)
 - ✓ Home work
 - ✓ Study & Practice on your own
-

What is Python?

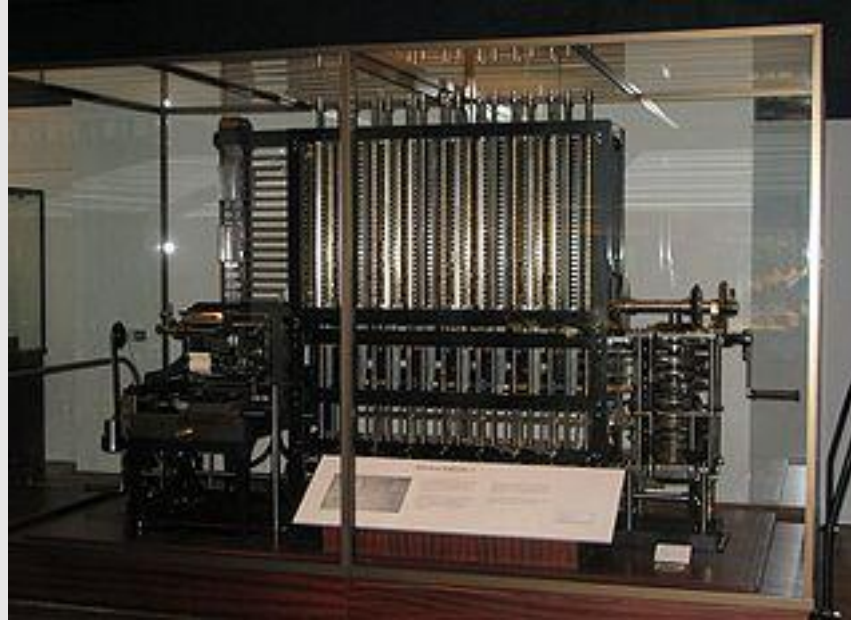
- Programming language
 - Then what is Programming Language?
 - A formal **language** that specifies a set of instructions that can be used to produce various kinds of output. **Programming languages** generally consist of instructions for a **computer**.
 - What is computer?
 - A **computer** is a device that can be instructed to carry out arbitrary sequences of arithmetic or logical operations automatically.
-

Computing History

Abacus: 2600 years ago



Modern Computer



- 1819 – 1822: Charles Babbage, proposed Difference Engine was a special-purpose digital computing machine for the automatic production of mathematical tables (such as logarithm tables, tide tables, and astronomical tables).
- Can calculate logarithmic and trigonometric functions, can be approximated by polynomials.

Modern Computer- first programming language

Ada Lovelace

- the first to recognize that the machine had applications beyond pure calculation
 - published the first algorithm for Charles Babbage's Analytical Engine: calculating *Bernoulli numbers*
 - Is regarded as the first to recognize the full potential of a "computing machine"
 - The first computer programmer
-



Modern Computer History

1890: **Herman Hollerith** designed a punch card system to calculate the 1880 census, accomplishing the task in just three years, saving the government \$5million. He establishes a company that later became part of IBM(International Business Machine) .



Modern Computer History

The Universal Turing Machine

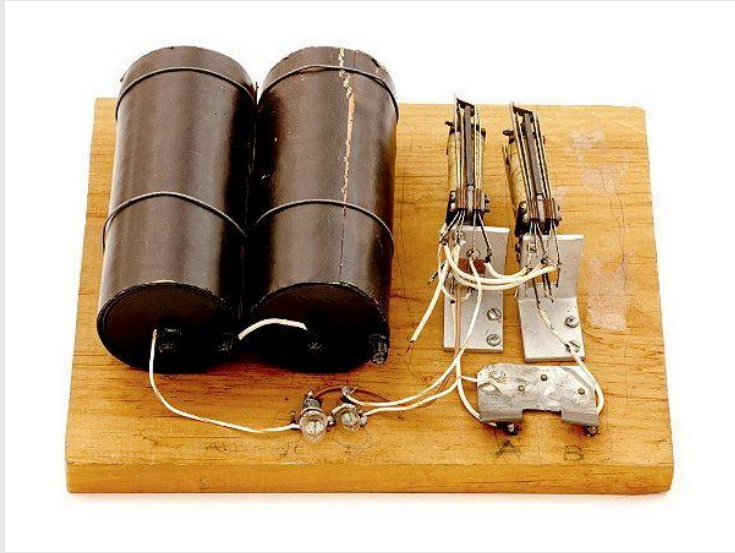
Alan Turing

- 1936, Turing invented the principle of the modern computer.
- The father of theoretical computer science and artificial intelligence.
- Turing Award: highest award in computer science



Modern Computer History

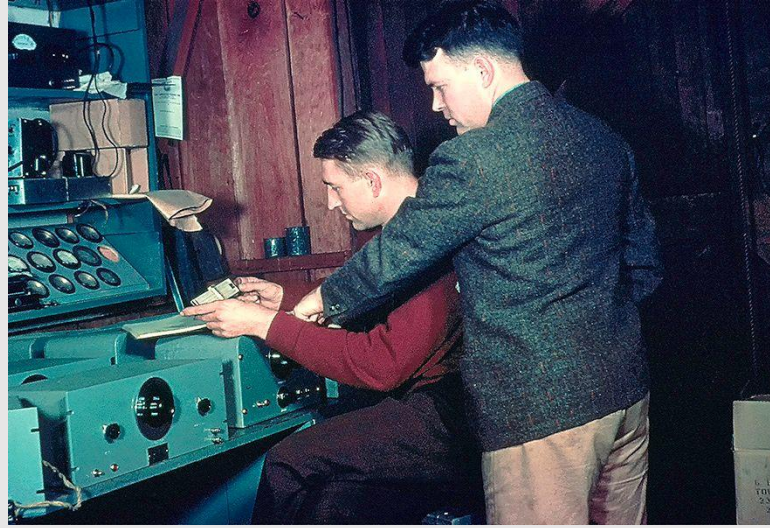
1937: Bell Laboratories



"Model K" Adder

provides proof of concept for
applying Boolean logic to the
design of computers

1939: HP (Hewlett-Packard is founded)

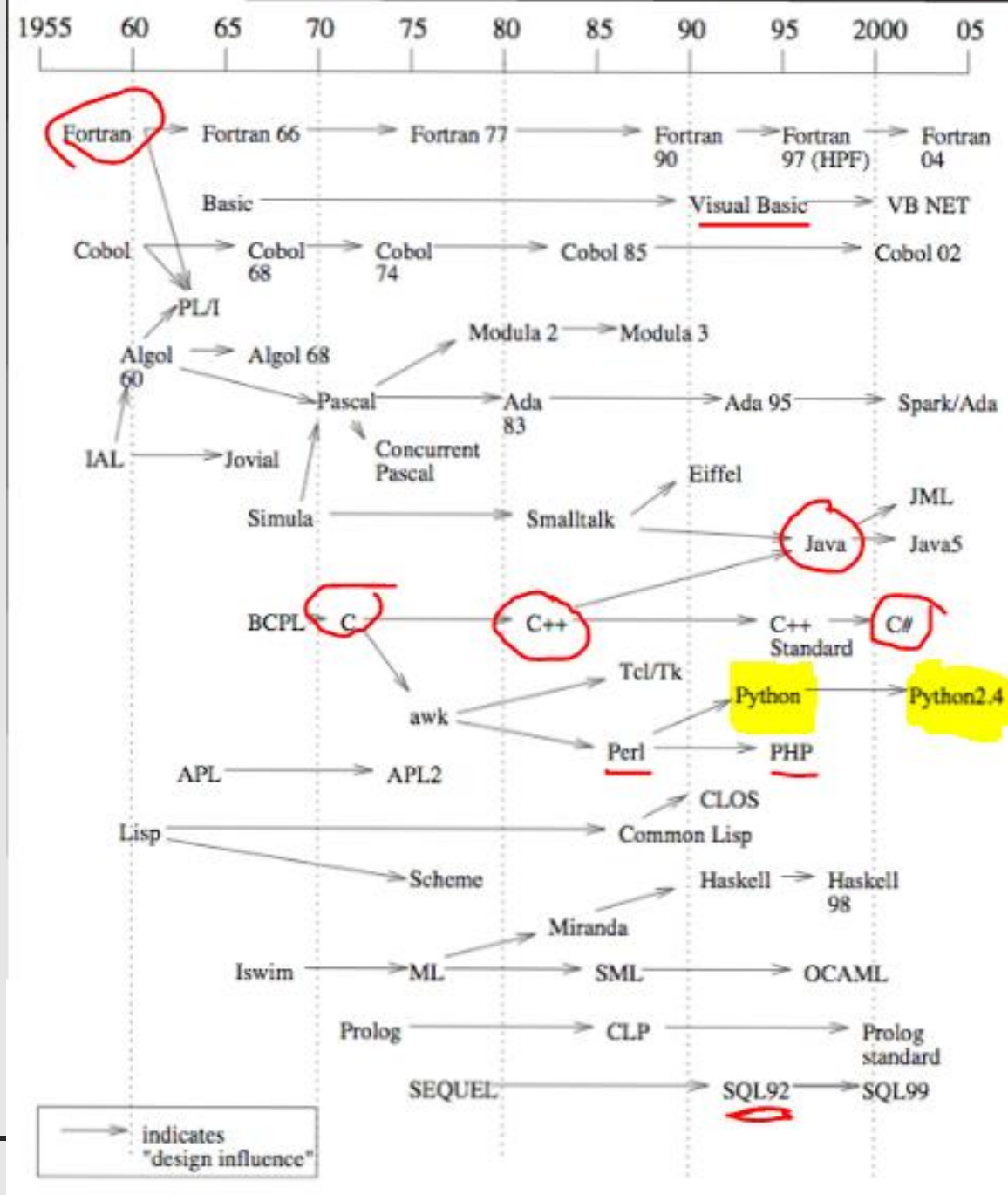


David Packard and Bill Hewlett found their company in a
Palo Alto, California

More computer history:

<http://www.computerhistory.org/timeline/computers/#16gebbe2ad45559efbc6eb35720d57b7>

Programming Language History



Programming Language History

History of Programming Language

1943 - ENIAC coding system	1970 - Pascal	1993 - Ruby
1951 - Regional Assembly Language	1970 - Forth	1993 - Lua
1952 - Autocode	1972 - C	1994 - CLOS (part of ANSI Common Lisp)
1954 - IPL (forerunner to LISP)	1972 - Smalltalk	1995 - Java
1955 - FLOW-MATIC (forerunner to COBOL)	1972 - Prolog	1995 - Delphi (Object Pascal)
1957 - FORTRAN (First compiler)	1973 - ML	1995 - JavaScript
1957 - COMTRAN	1975 - Scheme	1995 - PHP
1958 - LISP	1978 - SQL	1996 - WebDNA
1958 - ALGOL 58	1980 - C++	1997 - Rebol
1959 - FACT	1983 - Ada	1999 - D
1959 - COBOL	1984 - Common Lisp	2000 - ActionScript
1959 - RPG	1984 - MATLAB	2001 - C#
1962 - APL	1985 - Eiffel	2001 - Visual Basic .NET
1962 - Simula	1986 - Objective-C	2002 - F#
1962 - SNOBOL	1986 - Erlang	2003 - Groovy
1963 - CPL (forerunner to C)	1987 - Perl	2003 - Scala
1964 - BASIC	1988 - Tcl	2003 - Factor
1964 - PL/I	1988 - Mathematica	2007 - Clojure
1967 - BCPL (forerunner to C)	1989 - FL (Backus);	2009 - Go
1968 - Logo	1990 - Haskell	2011 - Dart
1969 - B (forerunner to C)	1991 - Python	
	1991 - Visual Basic	
	1991 - HTML	

Before 80s:

Establishing fundamental paradigms

1980s: consolidation, modules, performance. Object oriented programming

1990s: the Internet age, functional programming

Current trends: concurrent and distributed programming, security, component-oriented software development, open source , integration with database...

Why Python?

- Useful:
 - Data science
 - Mathematical computing
 - Web development
 - Finance and trading
 - System automation and administration
 - Computer graphics/game development
 - General and application specific scripting
 - Map and geography (GSI software)
- Easy to learn: very high level language
- Very Flexible: no hard rules, more forgiving of errors
- Python is the future of AI and Machine Learning: libraries such as scikit-learn






































Programming Language ranking

Source:
IEEE Spectrum

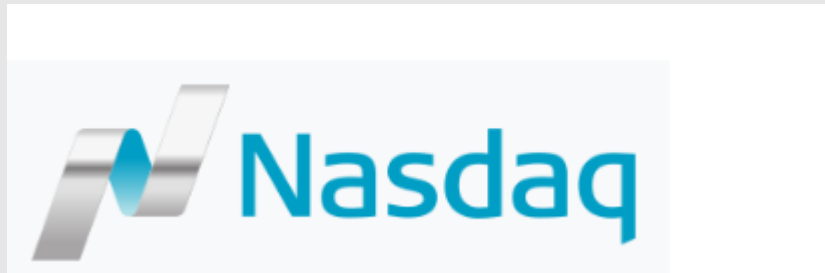
IEEE: Institute of Electrical and Electronics Engineers; world's largest technical professional organization.

<https://spectrum.ieee.org/computing/software/the-2017-top-programming-languages>

A comparison of Programming Languages:
<https://fusion809.github.io/comparison-of-programming-languages/>

Language Rank	Types	Spectrum Ranking
1. Python	 	100.0
2. C	  	99.7
3. Java	  	99.4
4. C++	  	97.2
5. C#	  	88.6
6. R		88.1
7. JavaScript	 	85.5
8. PHP		81.4
9. Go	 	76.1
10. Swift	 	75.3
11. Arduino		73.0
12. Ruby	 	72.4
13. Assembly		72.1
14. Scala	 	68.3
15. Matlab		68.0
16. HTML		67.0
17. Shell		66.3
18. Perl	 	57.6
19. Visual Basic		55.4
20. Cuda		53.9

Who is using Python?



History of Python

- Conceived in late 1980s
- Implementation was started in Dec. 1989
- Principal author: [Guido van Rossum](#)

Releases:

Python 1.0 - January 1994

Python 2.0 - October 16, 2000



Python 3.0 - December 3, 2008






Latest Python Version: 3.6.4 - December 19,
2017

Website: <https://www.python.org/>



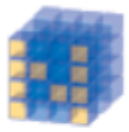
Python Based ecosystems - science






[Install](#)[Getting Started](#)[Documentation](#)[Report Bugs](#)[Blogs](#)


SciPy (pronounced "Sigh Pie") is a Python-based ecosystem of open-source software for mathematics, science, and engineering. In particular, these are some of the core packages:




NumPy
Base N-dimensional array package




SciPy library
Fundamental library for scientific computing




Matplotlib
Comprehensive 2D Plotting



IPython
Enhanced Interactive Console



Sympy
Symbolic mathematics



pandas
Data structures & analysis

Python Based ecosystems - Math



[RSS](#) · [Blog](#) · [Trac](#) · [Wiki](#) · [Questions?](#) · [Donate](#)
Online: [CoCalc](#) · [SageCell](#) or [Download](#), [Source Code](#)
v8.1 (2017-12-07) ·    · [Language](#) ▾

[Home](#) [Tour](#) [Help](#) [Library](#) [Download](#) [Development](#) [Links](#)

SageMath is a free [open-source](#) mathematics software system licensed under the GPL. It builds on top of many existing open-source packages: [NumPy](#), [SciPy](#), [matplotlib](#), [SymPy](#), [Maxima](#), [GAP](#), [FLINT](#), [R](#) and [many more](#). Access their combined power through a common, Python-based language or directly via interfaces or wrappers.

Mission: *Creating a viable free open source alternative to Magma, Maple, Mathematica and Matlab.*

Do you want to learn how to use SageMath?
Read [Sage for Undergraduates](#) by Gregory Bard or
[Mathematical Computation with Sage](#) by Paul Zimmermann et. al.
translations: [Calcul mathématique avec Sage](#) (French), [Rechnen mit Sage](#) (German)

[CoCalc \(SageMathCloud\)](#)
or: SageMathCell



[Download 8.1](#)
[Changelogs](#) · [Source 8.1](#) · [Packages](#) · [Git](#)

[Help/Documentation](#)
[Video](#) · [Forums](#) · [Tutorial](#) · [FAQ](#) · [Questions?](#)



[Feature Tour](#)
[Quickstart](#) · [Research](#) · [Graphics](#)

[Library](#)
[Testimonials](#) · [Books](#) · [Publications](#) · [Press Kit](#)



[Search](#)

CHOP (SAGE)

[Sage Notebook](#)

admin | [Toggle](#) | [Home](#) | [Published](#) | [Log](#) | [Help](#) | [Sign out](#)

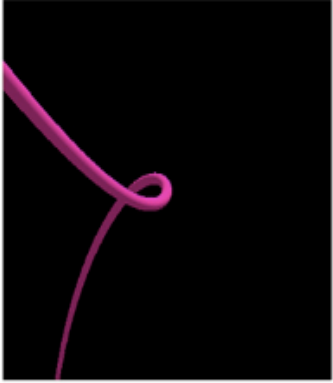
[Save](#) [Save & close](#) [Discard changes](#)


[Print](#) [Use](#) [Edit](#) [Text](#) [Revisions](#) [Share](#) [Publish](#)

GHOP
last edited on December 22, 2007 09:23 AM by admin
[File...](#) [Action](#) [Data...](#) [sage](#)


```
def f(t): #example taken from the SAGE reference manual
    return (t, t^2, t^3)

camera = Tachyon(camera_center=(5, 0, 4))
camera.texture('t')
camera.light((-20, -20, 40), 0.2, (1, 1, 1))
camera.parametric_plot(f, -5, 5, 't', min_depth=6)
camera.show()
```






Python Based ecosystems – Physics



pymunk


 Star 170

build canceled

Table Of Contents

- [News](#)
- [Installation](#)
- [Overview](#)
- [API Reference](#)
- [Examples](#)
- [Showcase](#)
- [Tutorials](#)
- [Benchmarks](#)
- [Advanced](#)
- [Issue Tracker](#)
- [Source Repository](#)
- [Downloads](#)
- [License](#)

Pymunk



pymunk

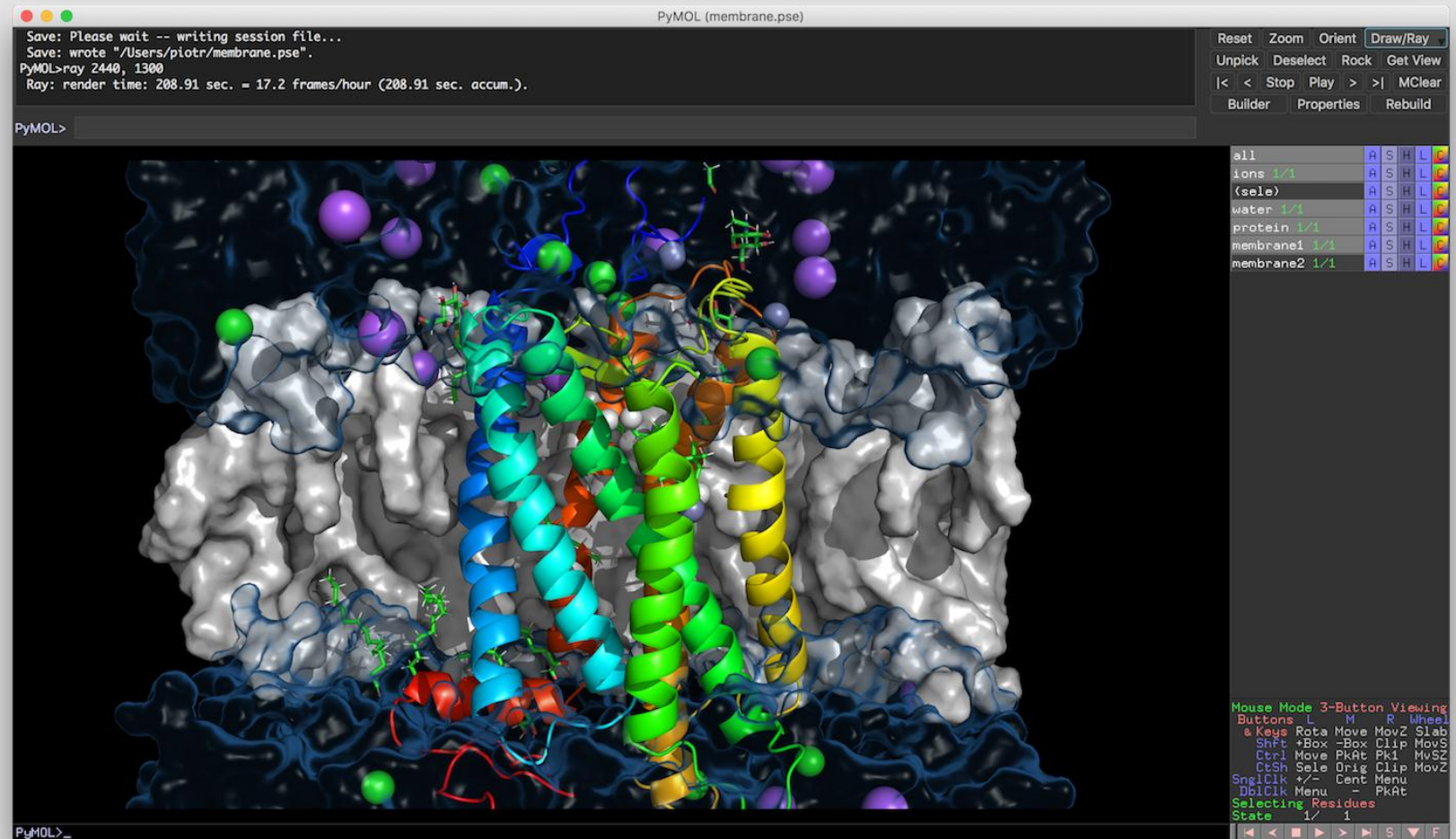
Pymunk is a easy-to-use pythonic 2d physics library that can be used whenever you need 2d rigid body physics from Python. Perfect when you need 2d physics in your game, demo or other application! It is built on top of the very capable 2d physics library [Chipmunk](#).

The first version was released in 2007 and Pymunk is still actively developed and maintained today.

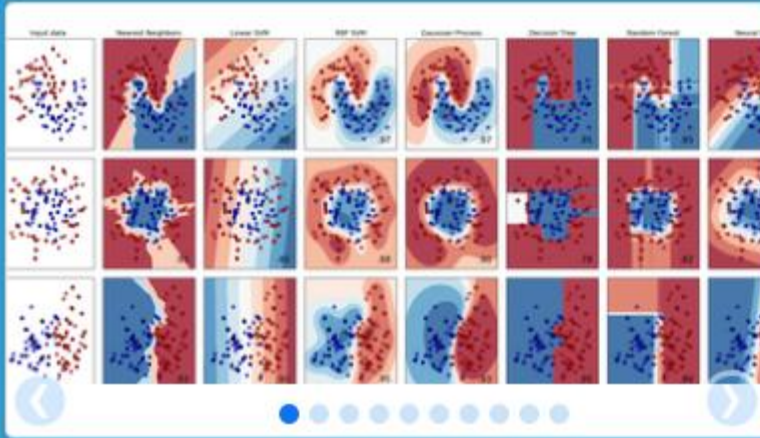
Python Based ecosystems – Biology

PyMOL:

a molecular visualization system created by [Warren Lyford DeLano](#). It is user-sponsored, [open-source software](#), released under the [Python License](#).



Python Based ecosystems – AI



scikit-learn

Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license



Natural
Language Toolkit
Computer program



Natural
Language
ToolKit

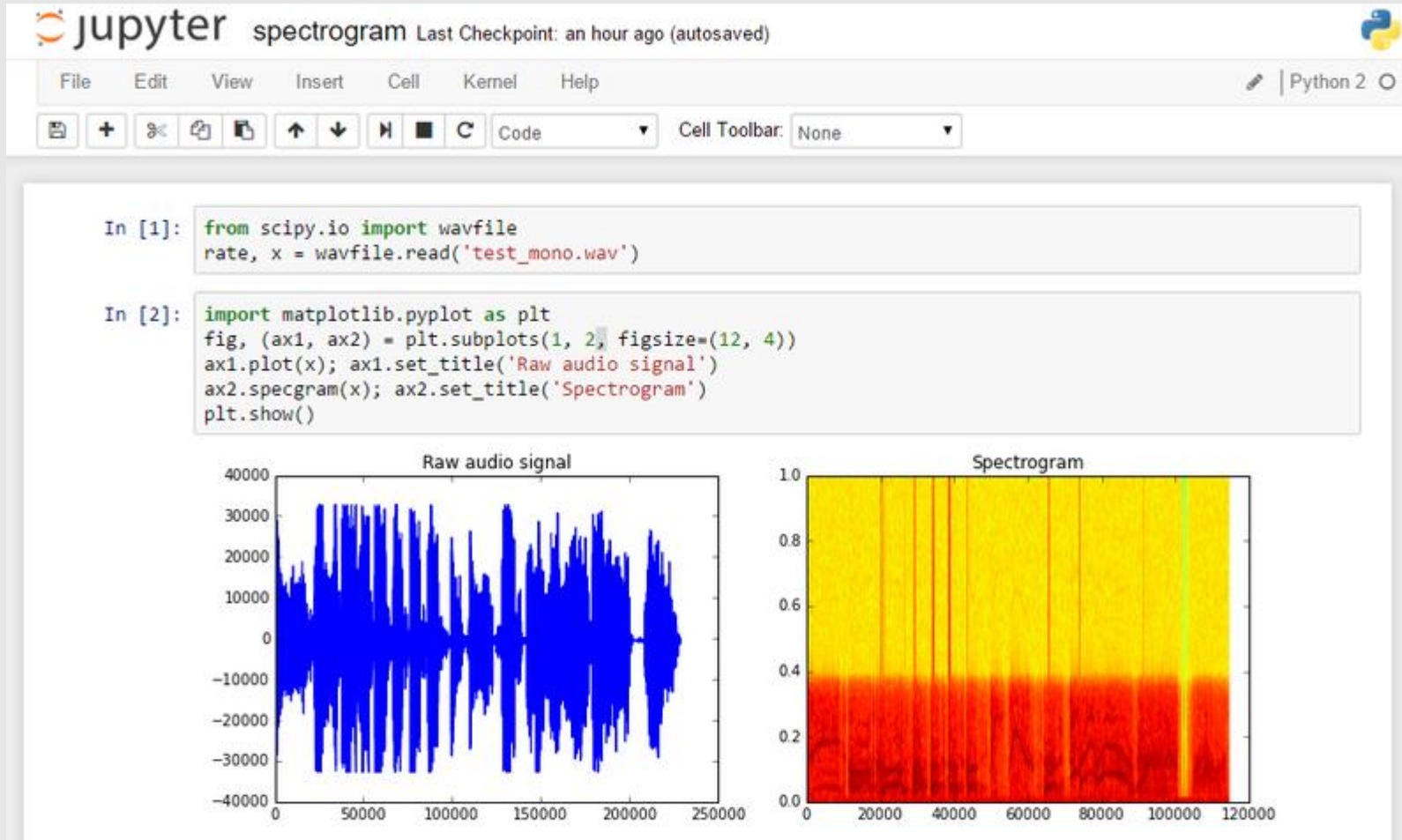


python™



Python Hands on - IDE

IDE stands for Integrated Development Environment.



```
In [5]: # Get Kaggle Titanic Datasets
t_train = pd.read_csv('/Users/williamliu/Dropbox/NYC-DAT-88/Homework_8/input/titanic_train.csv')
t_test = pd.read_csv('/Users/williamliu/Dropbox/NYC-DAT-88/Homework_8/input/titanic_test.csv')

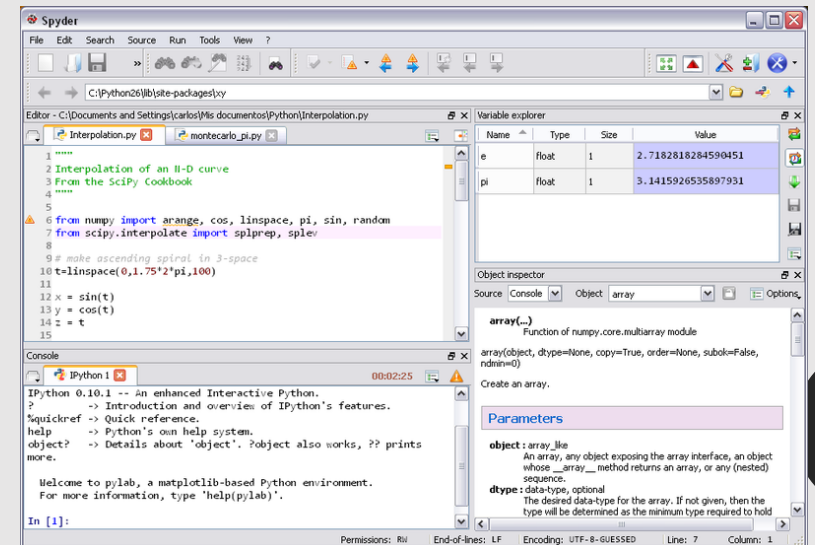
In [6]: print t_train.head()
```

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket
0	1	0	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171 7.2
500	NaN	S						
1	2	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38	1	0	PC 17
599	71.2833	C85	C					
2	3	1	Heikkinen, Miss. Laina	female	26	0	0	STON/O2. 3101282
7.9250	NaN	S						
3	4	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0	113803
53.1000	C123	S						
4	5	0	Allen, Mr. William Henry	male	35	0	0	373450 8.05
00	NaN	S						

```
In [691]: t_train['BoolSex'] = [1 if field=='male' else 0 for field in t_train.Sex]
t_test['BoolSex'] = [1 if field=='male' else 0 for field in t_test.Sex]
t
```

The variable explorer shows the following variables:

- `t_test`
- `t_test`
- `t_train`
- `tree_model`
- `print_function`
- `__import__` (name, globals, locals, fromlist, level)
- `__builtin__`



Python Hands on - Anaconda



install anaconda:

Download Anaconda 4.2.0 for Python 3.5:

For Windows 64bit

https://repo.continuum.io/archive/Anaconda3-4.2.0-Windows-x86_64.exe

For MacOSx 64bit

https://repo.continuum.io/archive/Anaconda3-4.2.0-MacOSX-x86_64.pkg

For Linux 64bit

https://repo.continuum.io/archive/Anaconda3-4.2.0-Linux-x86_64.sh

install OpenCV:

conda install --channel <https://conda.anaconda.org/menpo/opencv3>

Python Face Detection Demo code:

<https://gtscnc.org/download/python-face-detection-demo>

Class 1 Homework

1. Reading:

Python for Kids: chapter 2 “Calculations and Variables”

History of Programming Languages:

https://en.wikipedia.org/wiki/History_of_programming_languages

History of Computer:

<http://www.computerhistory.org/timeline/computers/#169ebbe2ad45559efbc6eb35720d57b7>

2. Explore:

Python website: <https://www.python.org/>

3. Run Python code on Jupyter Notebook
