An Introduction to Python

Getting Started

What You Need to Get Started?

- Python
  - An object-oriented language gaining popularity in the industry
- IPython
  - An interactive command-line terminal for Python
- ASCII text editor

Installing Python from ANACONDA

- Many distributors, but Anaconda seems the best.
- https://repo.continuum.io/archive/index.html

- Anaconda supports Windows, Mac OS, and Linux

Which Version?

- Python 3.x.x versus Python 2.x.x.
- Still, Python 2.x.x has fewer problems with add-on “tool boxes”.

Anaconda2-5.2.0-Linux-ppc64le.sh	269.6M	2018-05-30 13:04
Anaconda2-5.2.0-Linux-x86.sh	488.7M	2018-05-30 13:05
Anaconda2-5.2.0-Linux-x86_64.sh	603.4M	2018-05-30 13:04
Anaconda2-5.2.0-MacOSX-x86_64.pkg	616.8M	2018-05-30 13:05
Anaconda2-5.2.0-MacOSX-x86_64.sh	527.1M	2018-05-30 13:05
Anaconda2-5.2.0-Windows-x86.exe	443.4M	2018-05-30 13:04
Anaconda2-5.2.0-Windows-x86_64.exe	564.0M	2018-05-30 13:04

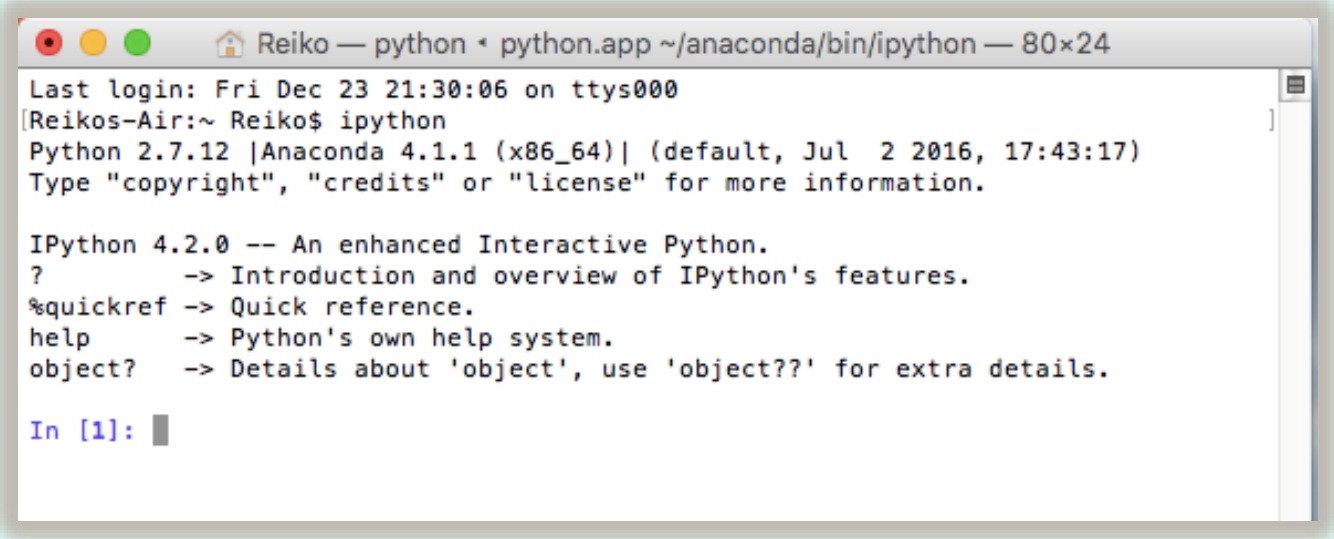
Starting IPython

- Upon successful installation, start programming!
- IPython allows you to use Python interactively.

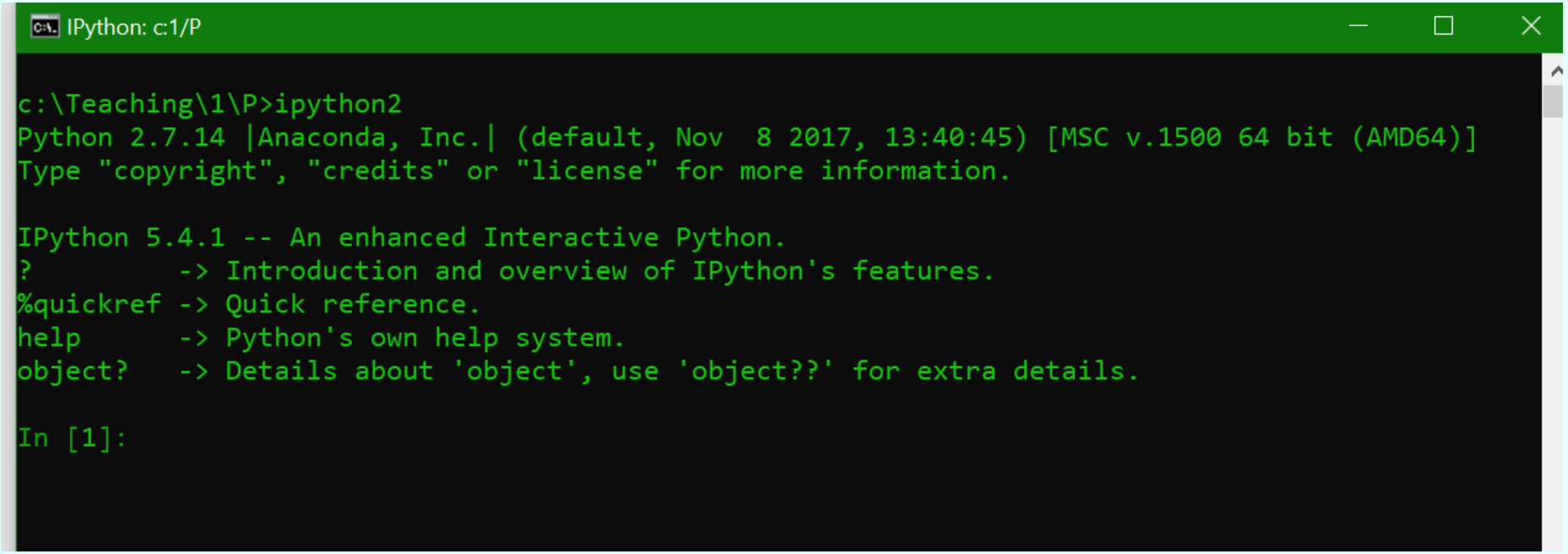
IP[y]: IPython
Interactive Computing

- The Anaconda distribution contains IPython.
- There are many other IDLEs such as

On MacBook, open a Terminal, and type ipython2



On Windows, open a command prompt, and type ipython2



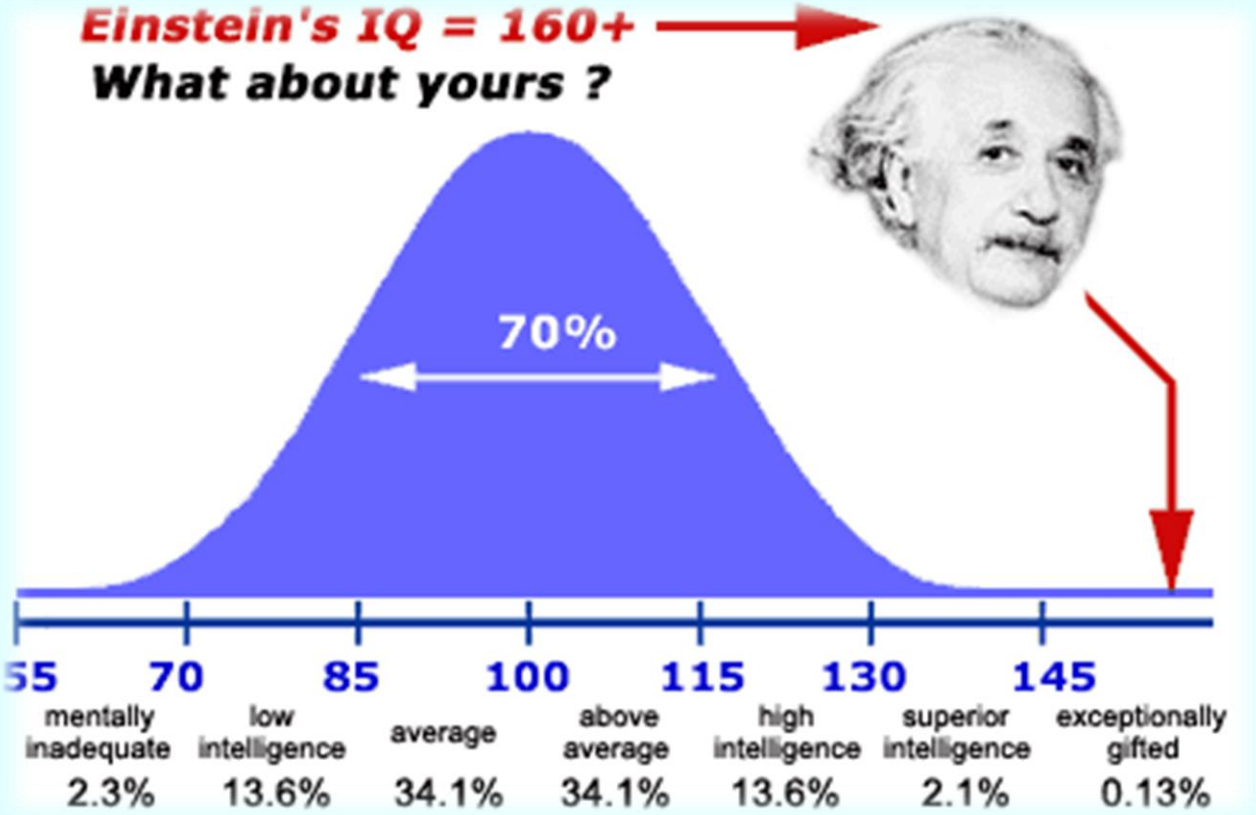
Editor for Writing Python Scripts

- You must use an editor for ASCII text, not something that has hidden control characters.
-  is free and really great.
- gedit works for Linux and Windows, but not on Mac OS X's latest versions
- Instead, use Mac Book's TextEdit or TextMate.
- To edit in plain text by default with TextEdit, go to `TextEdit > Preferences` in the menu bar. On the New Document tab, select `Plain Text` in the Format section.

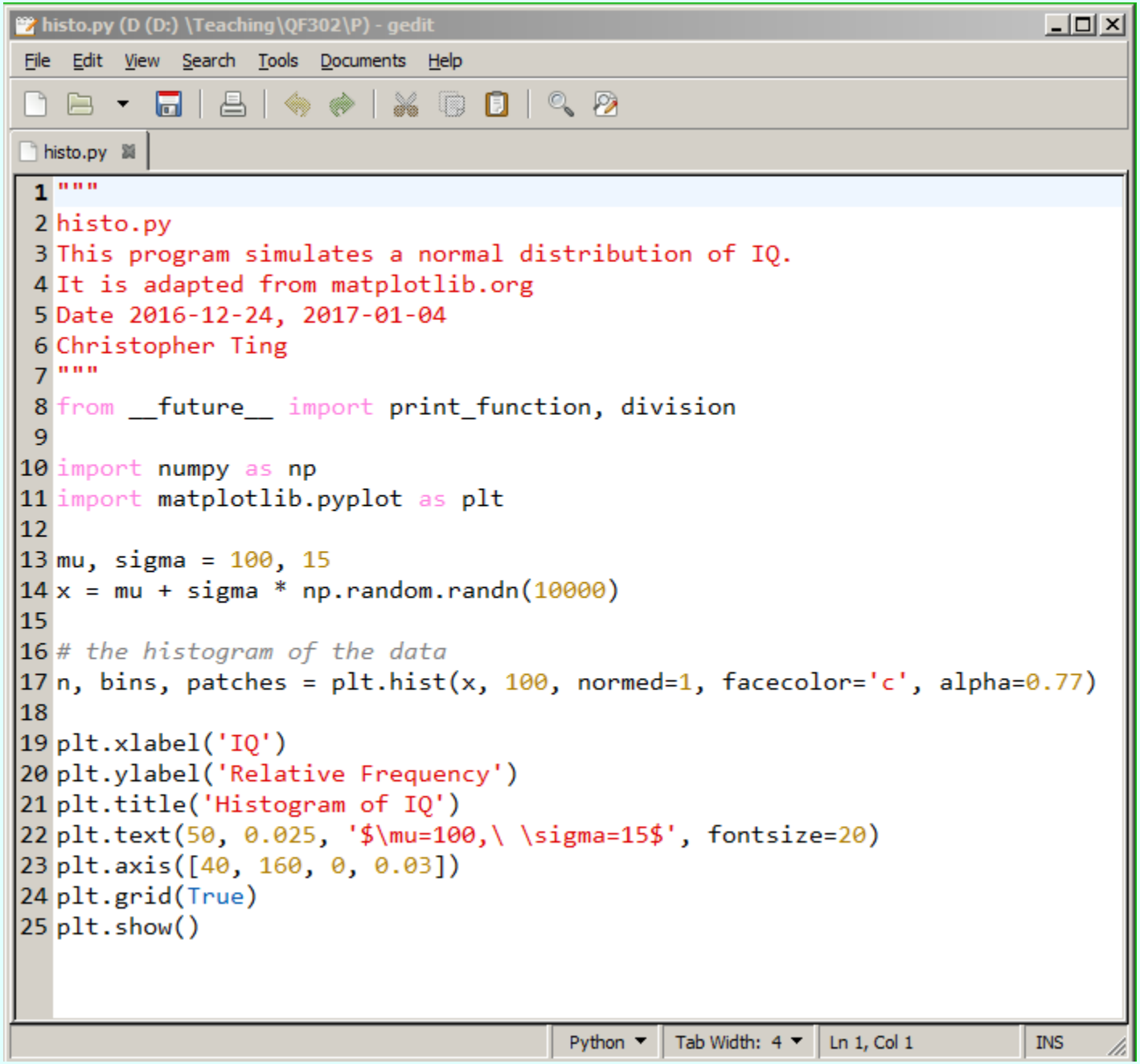
Example: Statistical Modeling of IQ

- I.Q. (intelligence quotient) in general, is an assessment of your ability to think and reason. IQ score is a standardized way of comparing this ability with the majority of people the same age as you are.

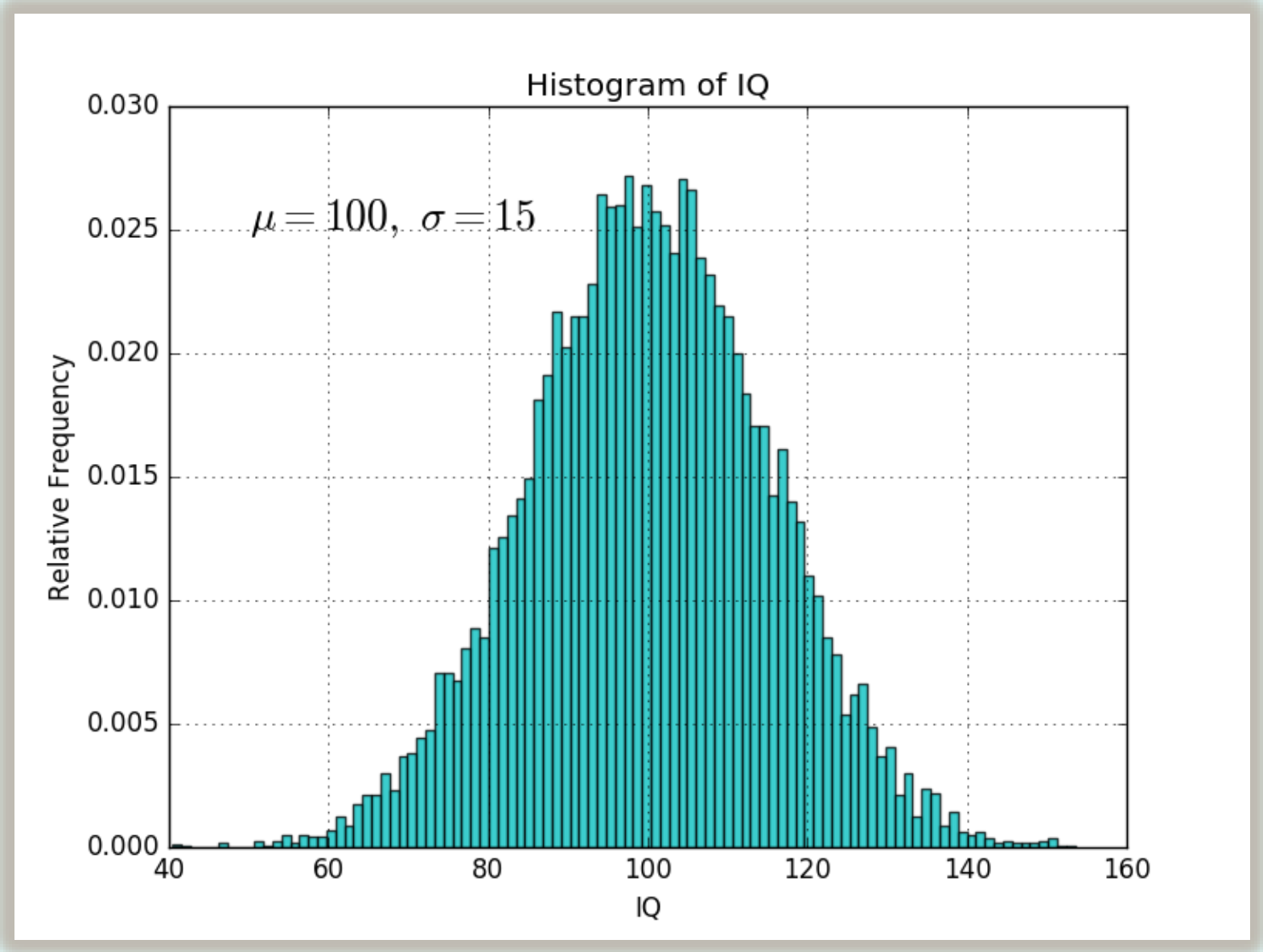
http://www.free-iqtest.net/



http://matplotlib.org/users/pyplot_tutorial.html



```
In [1]: run histo.py
```



Basic IPython Commands

- Clear the IPython console: `Ctrl L`
- Change the directory: `cd "d:"`
- Find the current directory: `pwd`
- Run a python script: `run histo.py`
- Change the default colors: `colors nocolor`
- Reset the IPython console: `reset -f`
- For more, check out https://ipython.org/ipython-doc/3/interactive/magics.html

Important: Computing Literacy

- It is always good to untick "Hide extensions for known file types".
- Keep your Python codes in a directory.
- `cd` the IPython console to that directory.
- Programming at the beginning stage is about finding out what goes wrong. You need
  - Patience
  - Perseverance
  - Proclivity to being a cool and sociable nerd
- Programming at post-beginning stages needs creativity.

Great Places to Start Learning

- learnpython.org

https://www.learnpython.org/

- Google

https://developers.google.com/edu/python/introduction

- Quantopian

https://www.quantopian.com/lectures

- Python for Econometrics

https://www.kevinsheppard.com/Python_for_Econometrics

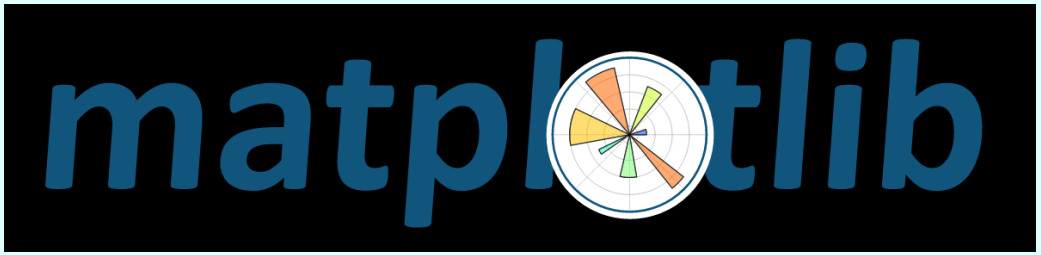
- Quantitative Economics

https://lectures.quantecon.org/py/

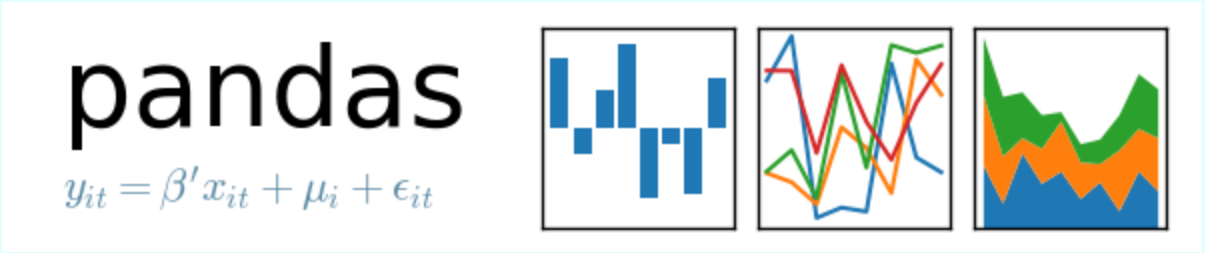
Important Packages aka Toolboxes

- https://docs.scipy.org/doc/numpy/user/quickstart.html

- https://matplotlib.org/users/pyplot_tutorial.html



- http://pandas.pydata.org/pandas-docs/stable/tutorials.html



The `for` loop

http://homes.cs.washington.edu/~stepp/bridge/2007/

- **`for` loop**: Repeats a set of statements over a group of values.
  - Syntax:

```
for variableName in groupOfValues:
    statements
```

    - Note that ***statements*** is a tab or a few spaces from **`for`**.
    - ***variableName*** gives a name to each value, so you can refer to it in the ***statements***.
    - ***groupOfValues*** can be a range of integers, specified with the `range` function.

  - Example:

```
for x in range(1, 6):
    print x, "squared is", x * x
```

Output:

```
1 squared is 1
2 squared is 4
3 squared is 9
4 squared is 16
5 squared is 25
```

range

http://homes.cs.washington.edu/~stepp/bridge/2007/

- The `range` function specifies a range of integers:
  - `range(`***start***`, `***stop***`)` - the integers between ***start*** (inclusive) and ***stop*** (exclusive)
- It can also accept a third value specifying the change between values.
  - `range(`***start***`, `***stop***`, `***step***`)` - the integers between ***start*** (inclusive) and ***stop*** (exclusive) by ***step***
- Example:

```
for x in range(5, 0, -1):
    print x
print "Blastoff!"
```

Output:

```
5
4
3
2
1
Blastoff!
```

Cumulative Loops

http://homes.cs.washington.edu/~stepp/bridge/2007/

- Some loops incrementally compute a value that is initialized outside the loop. This is sometimes called a *cumulative sum*.

```
sum = 0
for i in range(1, 11):
    sum = sum + (i * i)
print "sum of first 10 squares is", sum

Output:
sum of first 10 squares is 385
```

- **Exercise:** Write a Python program that computes the factorial of an integer.

if

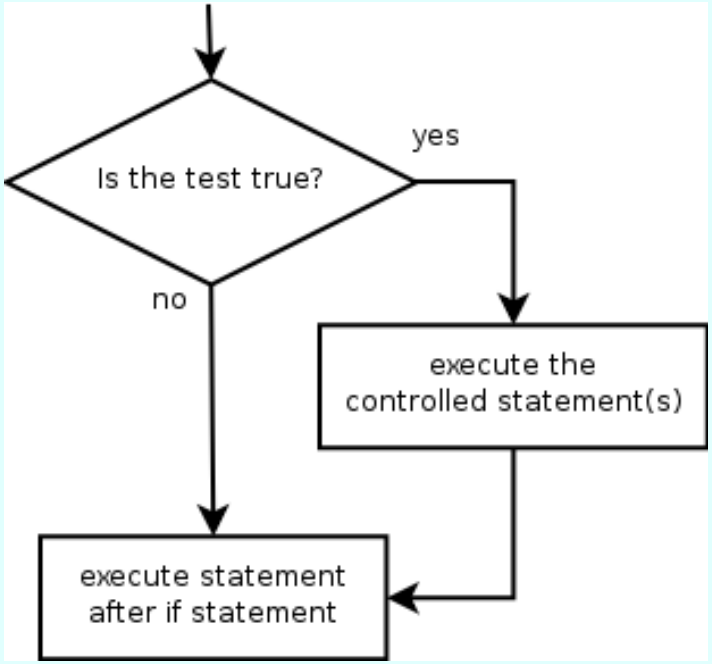
http://homes.cs.washington.edu/~stepp/bridge/2007/

- **`if` statement**: Executes a group of statements only if a certain condition is true. Otherwise, the statements are skipped.
  - Syntax:

```
if condition:
    statements
```

  - Example:

```
gpa = 3.4
if gpa > 2.2:
    print "You passed."
```

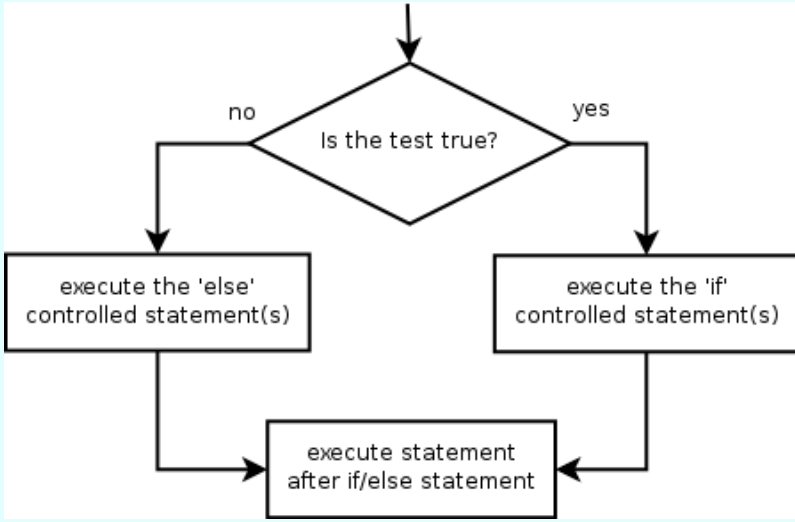


if/else

http://homes.cs.washington.edu/~stepp/bridge/2007/

- **`if/else` statement**: Executes one block of statements if a certain condition is True, and a second block of statements if it is False.
  - Syntax:

```
if condition:
    statements
else:
    statements
```

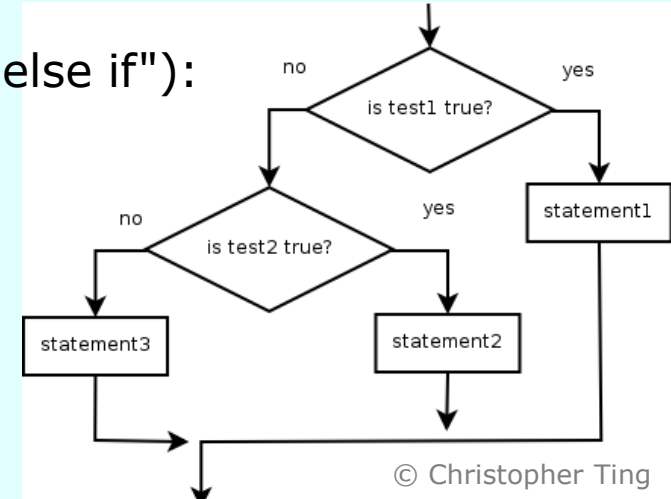


- Example:

```
gpa = 1.4
if gpa > 2.0:
    print "Doing fine toward graduation"
else:
    print "You need to improve your GPA."
```

- Multiple conditions can be chained with `elif` ("else if"):

```
if condition:
    statements
elif condition:
    statements
else:
    statements
```



while

http://homes.cs.washington.edu/~stepp/bridge/2007/

- **`while` loop**: Executes a group of statements as long as a condition is True.
  - good for *indefinite loops* (repeat an unknown number of times)

- Syntax:

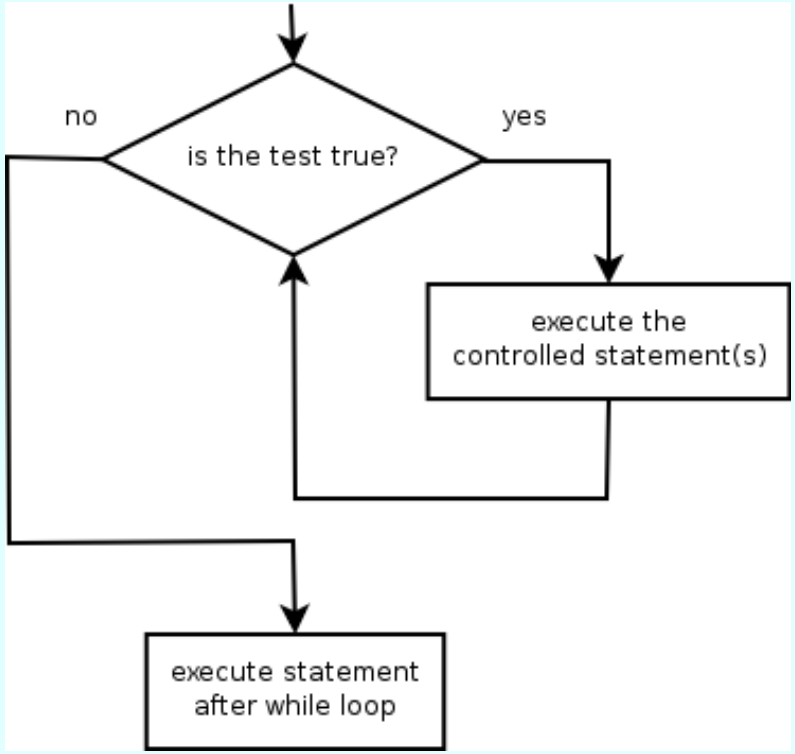
```
while condition:
    statements
```

- Example:

```
number = 1
while number < 200:
    print number,
    number = number * 2
```

  - Output:

```
1 2 4 8 16 32 64 128
```



Logic

http://homes.cs.washington.edu/~stepp/bridge/2007/

- Many logical expressions use *relational operators*:

Operator	Meaning	Example	Result
==	equals	`1 + 1 == 2`	`True`
!=	does not equal	`3.2 != 2.5`	`True`
<	less than	`10 < 5`	`False`
>	greater than	`10 > 5`	`True`
<=	less than or equal to	`126 <= 100`	`False`
>=	greater than or equal to	`5.0 >= 5.0`	`True`

- Logical expressions can be combined with *logical operators*:

Operator	Example	Result
`and`	`9 != 6 and 2 < 3`	`True`
`or`	`2 == 3 or -1 < 5`	`True`
`not`	`not 7 > 0`	`False`

- **Exercise:** Write code to display and count the factors of a number.

Strings

http://homes.cs.washington.edu/~stepp/bridge/2007/

- **string**: A sequence of text characters in a program.
  - Strings start and end with quotation mark `"` or apostrophe `'` characters.
  - Examples:

```
"hello"
"This is a string"
"This, too, is a string.   It can be very long!"
```

- A string may not span across multiple lines or contain a " character.

```
"This is not
a legal String."
"This is not a "legal" String either."
```

- A string can represent characters by preceding them with a backslash.
  - `\t` tab character
  - `\n` new line character
  - `\"` quotation mark character
  - `\\` backslash character
  - Example: `"Hello\tthere\nHow are you?"`

Indexes

http://homes.cs.washington.edu/~stepp/bridge/2007/

- Characters in a string are numbered with *indexes* starting at 0:
  - Example:

```
name = "P. Diddy"
```

index	0	1	2	3	4	5	6	7
character	P	.		D	i	d	d	y

- Accessing an individual character of a string:
  variableName `[` ***index*** `]`
  - Example:

```
print name, "starts with", name[0]
```

Output:

```
P. Diddy starts with P
```

String Properties

http://homes.cs.washington.edu/~stepp/bridge/2007/

- `len(`***string***`)` - number of characters in a string

  (including spaces)
- `str.lower(`***string***`)` - lowercase version of a string
- `str.upper(`***string***`)` - uppercase version of a string

- Example:

```
name = "Martin Douglas Stepp"
length = len(name)
big_name = str.upper(name)
print big_name, "has", length, "characters"
```

Output:

```
MARTIN DOUGLAS STEPP has 20 characters
```

Text Processing

http://homes.cs.washington.edu/~stepp/bridge/2007/

- **text processing**: Examining, editing, formatting text.
  - often uses loops that examine the characters of a string one by one

- A `for` loop can examine each character in a string in sequence.
  - Example:

```
for c in "booyah":
    print c
```

Output:

```
b
o
o
y
a
h
```

Strings and Numbers

http://homes.cs.washington.edu/~stepp/bridge/2007/

- `ord(`***text***`)` - converts a string into a number.
  - Example: `ord("a")` is `97`, `ord("b")` is `98`, ...
  - Characters map to numbers using standardized mappings such as *ASCII* and *Unicode*.
- `chr(`***number***`)` - converts a number into a string.
  - Example: `chr(99)` is `"c"`
- **Exercise:** Write a program that performs a rotation cypher.
  - e.g. `"Attack"` when rotated by 1 becomes `"buubdl"`

File Processing

http://homes.cs.washington.edu/~stepp/bridge/2007/

- Many programs handle data, which often comes from files.

- Reading the entire contents of a file:

variableName `= open("`***filename***`").read()`

Example:

```
file_text = open("bankaccount.txt").read()
```

Line-by-Line Processing

http://homes.cs.washington.edu/~stepp/bridge/2007/

- Reading a file line-by-line:

```
for line in open("filename").readlines():
    statements
```

Example:

```
count = 0
for line in open("bankaccount.txt").readlines():
    count = count + 1
print "The file contains", count, "lines."
```

- **Exercise:** Write a program to process a file of DNA text, such as:

```
ATGCAATTGCTCGATTAG
```

  - Count the percent of C+G present in the DNA.

Concluding Remarks

- Python is free, and open source.
- Python is general purpose and comes with many powerful tool boxes. Check out Python Software Foundation https://pypi.python.org/pypi/
- Mastery of Python may be necessary for jobs in algorithmic trading, risk management, quantitative research, etc.
- Above all, it is fun and cool to apply Python!

Python Inventor: It is an Experiment

"Python is an experiment in how much freedom programmers need. Too much freedom and nobody can read another's code; too little and expressiveness is endangered."

- Guido van Rossum