LABORATORY OF DATA SCIENCE

Python & Spyder- recap

Python

Python is a

- High-level
- Interpreted (Interpreters for many OS)
- Dynamically Typed
  - Verification of the type safety of a program a runtime
- Object oriented
- Cross-Platform
- Multi-purpose (WEB, GUI, Scripting)

computer programming language

https://www.python.org/

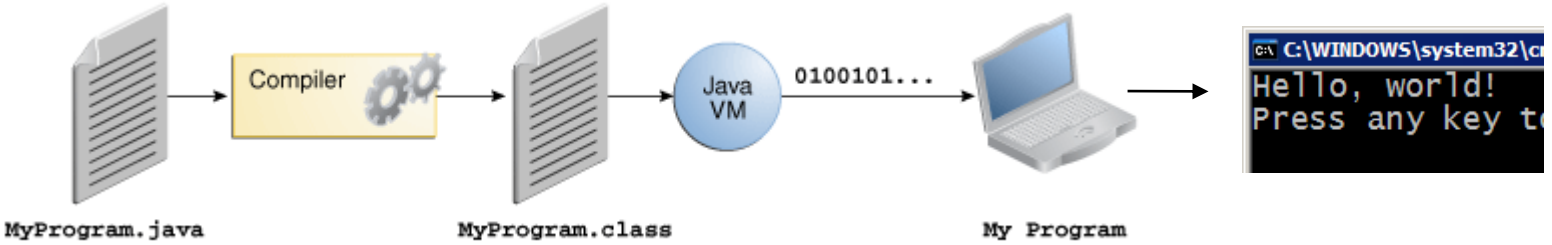
Version release dates [edit]

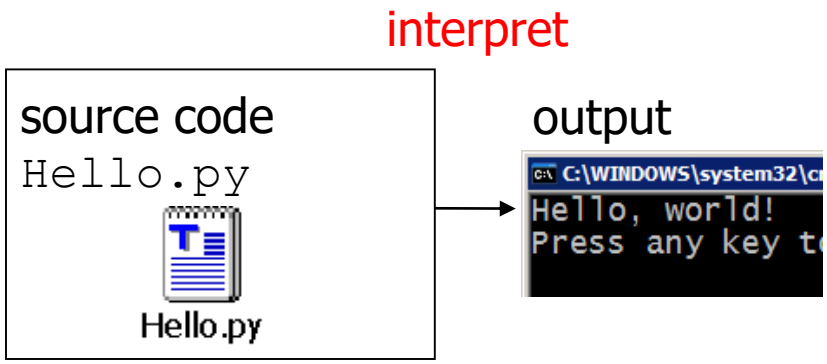
Release dates for the major and minor versions:[31][32]

- Implementation started - December, 1989
- Internal releases at Centrum Wiskunde & Informatica - 1990
- Python 0.9.0 - February 20, 1991
  - Python 0.9.1 - February, 1991
  - Python 0.9.2 - Autumn, 1991
  - Python 0.9.4 - December 24, 1991
  - Python 0.9.5 - January 2, 1992
  - Python 0.9.6 - April 6, 1992
  - Python 0.9.8 - January 9, 1993
  - Python 0.9.9 - July 29, 1993
- Python 1.0 - January 1994
  - Python 1.2 - April 10, 1995
  - Python 1.3 - October 12, 1995
  - Python 1.4 - October 25, 1996
  - Python 1.5 - December 31, 1997
  - Python 1.6 - September 5, 2000
- Python 2.0 - October 16, 2000
  - Python 2.1 - April 15, 2001
  - Python 2.2 - December 21, 2001
  - Python 2.3 - July 29, 2003
  - Python 2.4 - November 30, 2004
  - Python 2.5 - September 19, 2006
  - Python 2.6 - October 1, 2008
  - Python 2.7 - July 4, 2010
- Python 3.0 - December 3, 2008
  - Python 3.1 - June 27, 2009
  - Python 3.2 - February 20, 2011
  - Python 3.3 - September 29, 2012
  - Python 3.4 - March 16, 2014
  - Python 3.5 - September 13, 2015
  - Python 3.6 - December 23, 2016
  - Python 3.7 - June 27, 2018

Compiling and interpreting

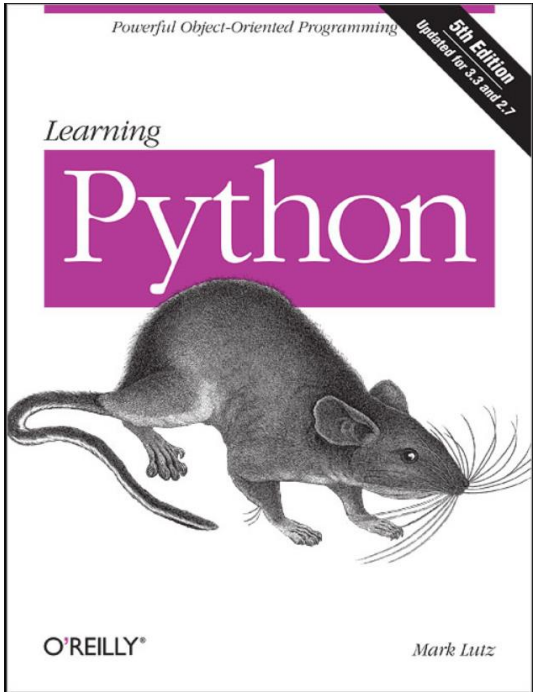
- Many languages require you to *compile* (translate) your program into a form that the machine understands.

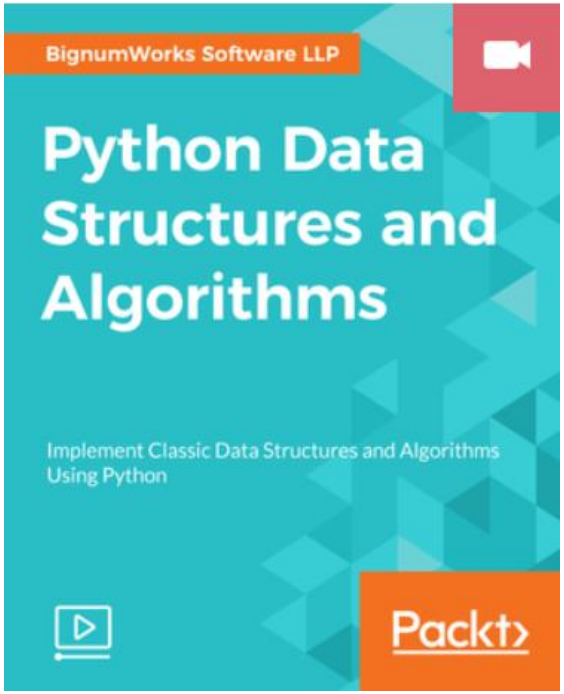


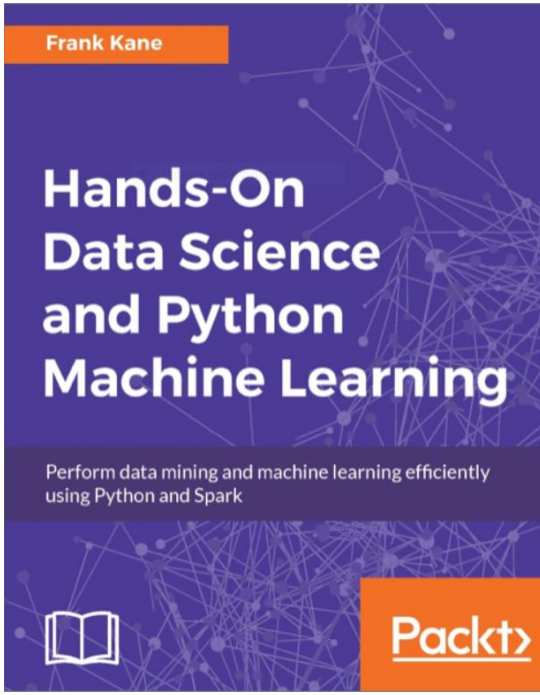


- Python is instead directly *interpreted* into machine instructions.

Python language: books



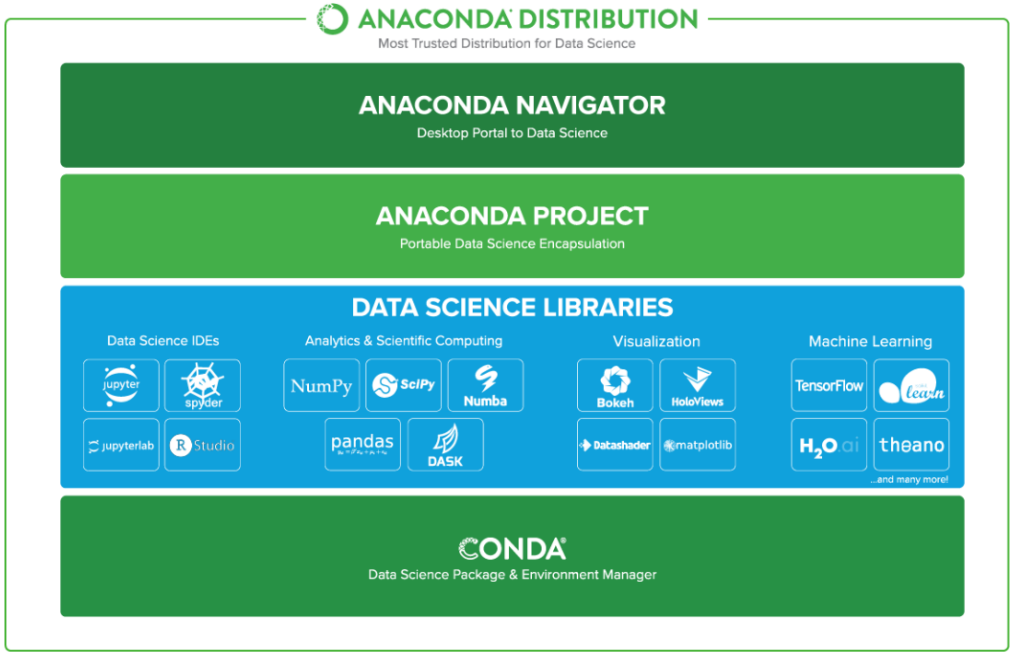




.... These are only examples ...

Anaconda - www.anaconda.com

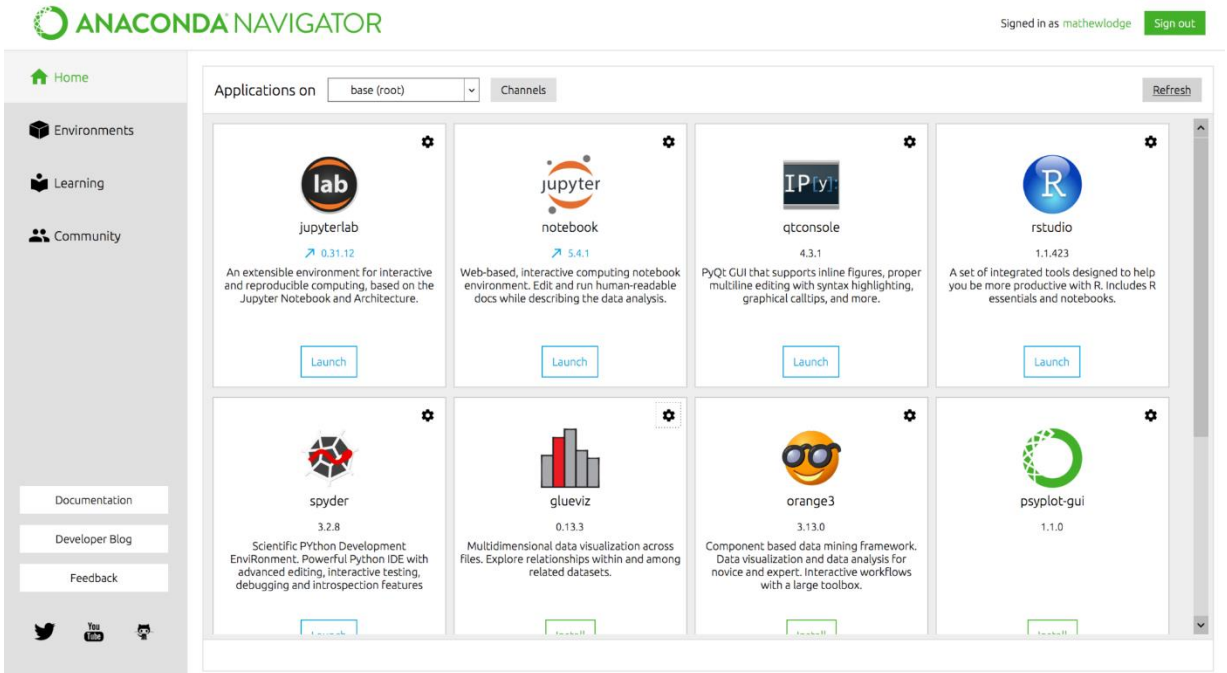
- Manage your DS packages, dependencies, and environments
- Develop DS projects using Jupyter, JupyterLab, Spyder



- Automatically manages all packages, including cross-language dependencies
- Works across all platforms: Linux, macOS, Windows
- Create virtual environments

Anaconda Navigator

- Desktop Portal to Data Science
- Install and launch applications and editors including Jupyter, RStudio, Visual Studio Code, and Spyder
- Manage your local environments and data science projects from a graphical interface



Spyder

Spyder is the Scientific Python Development EnviRonment

- advanced editing
- interactive testing
- Debugging

Spyder websites:

- Downloads, bug reports and feature requests: https://github.com/spyder-ide/spyder
- Discussions: http://groups.google.com/group/spyderlib

Python Recap

Indentation


```
/* Bogus C code */
if (foo) {
    if (bar) {
        baz(foo, bar);
}
else {
    qux();
}}
```

```
# Python code
if foo:
    if bar:
        baz(foo, bar)
    else:
        qux()
```

Strings


```
#This is a string
name = "Anna Monreale (that\"s me)"

#This is also a string
city = 'Pisa'

#This is a multi-line string
office = """My office is at the department
of Computer Science, University of Pisa"""

#This is also  a multi-line string
other = '''My office hours is on Tuesday in the
afternoon, however it is always better to take
an appointment'''
```

String manipulation


```
animals = "Cats " + "Dogs "
animals += "Rabbits"
# Cats Dogs Rabbits

fruit = ', '.join(['Apple', 'Banana', 'Orange'])
# Apple, Banana, Orange

date = '%s %d %d' % ('Sept', 11, 2010)
# Sept 11 2010
```

Numbers


```
# Integers Numbers
year = 2010
year = int("2010")

# Floating Point Numbers
pi = 3.14159265
pi = float("3.14159265")

# Fixed Point Numbers
from decimal import Decimal
price = Decimal("0.02")
```

Arithmetic


```
a = 10          # 10
a += 1          # 11
a -= 1          # 10

b = a + 1       # 11
c = a - 1       # 9

d = a * 2       # 20
e = a / 2       # 5
f = a % 3       # 1
g = a ** 2      # 100
```

Lists


```
# Lists can be heterogeneous
favorites = []

# Appending
favorites.append(42)

# Extending
favorites.extend(["Python", True])

# Equivalent to
favorites = [42, "Python", True]
```

Lists


```
numbers = [1, 2, 3, 4, 5]

len(numbers)
# 5

numbers[0]
# 1

numbers[0:2]
# [1, 2]

numbers[2:]
# [3, 4, 5]
```

Dictionary


```
person = {}

# Set by key / Get by key
person['name'] = 'Nowell Strite'

# Update
person.update({
    'favorites': [42, 'food'],
    'gender': 'male',
    })

# Any immutable object can be a dictionary key
person[42] = 'favorite number'
person[(44.47, -73.21)] = 'coordinates'
```

Dictionary


```
person = {'name': 'Nowell', 'gender': 'Male'}

person['name']
person.get('name', 'Anonymous')
# 'Nowell Strite'

person.keys()
# ['name', 'gender']

person.values()
# ['Nowell', 'Male']

person.items()
# [['name', 'Nowell'], ['gender', 'Male']]
```

If-then-else


```
grade = 82
if grade >= 90:
    if grade == 100:
        print 'A+'
    else:
        print "A"
elif grade >= 80:
    print "B"
elif grade >= 70:
    print "C"
else:
    print "F"

# B
```

For Loop


```
for x in range(10): #0-9
    print( x )
```

```
fruits = ['Apple', 'Orange']

for fruit in fruits:
    print( fruit)
```

```
states = {
    'VT': 'Vermont',
    'ME': 'Maine',
    }

for key, value in states.items():
    print('%s: %s' % (key, value))
```

Function Definition


```
def my_function():
    """Function Documentation"""
    print("Hello World")
```

```
# Positional
def add(x, y):
    return x + y

# Keyword
def shout(phrase='Yipee!'):
    print( phrase)

# Positional + Keyword
def echo(text, prefix=''):
    print( '%s%s' % (prefix, text))
```

Import packages


```
# Renaming imports
from datetime import date
from my_module import date as my_date

# This is usually considered a big No-No
from datetime import *
```

Error Handling


```
import datetime
import random

day = random.choice(['Eleventh', 11])
try:
    date = 'September ' + day
except TypeError:
    date = datetime.date(2010, 9, day)
else:
    date += ' 2010'
finally:
    print( date)
```

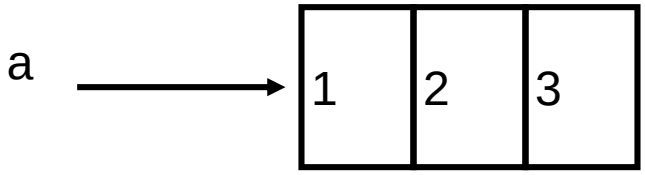
Reference Semantics

- Assignment manipulates references
  - x = y **does not make a copy** of y
  - x = y makes x **reference** the object y references
- Very useful; but beware!
- Example:

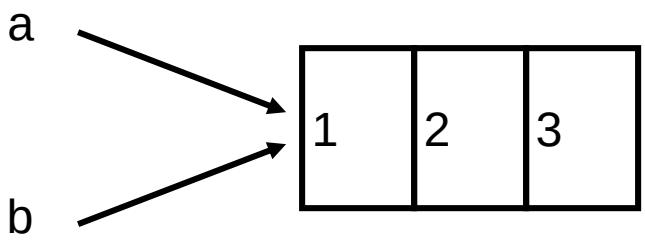
```
>>> a = [1, 2, 3]
>>> b = a
>>> a.append(4)
>>> print b
[1, 2, 3, 4]
```

Changing a Shared List

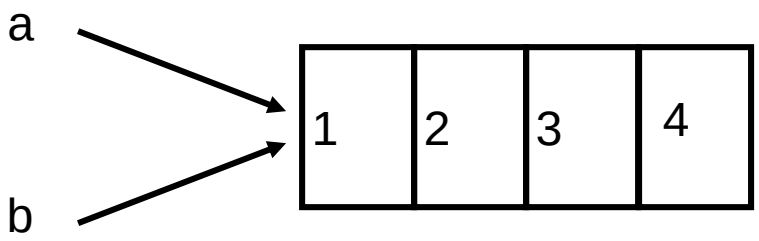
a = [1, 2, 3]



b = a



a.append(4)

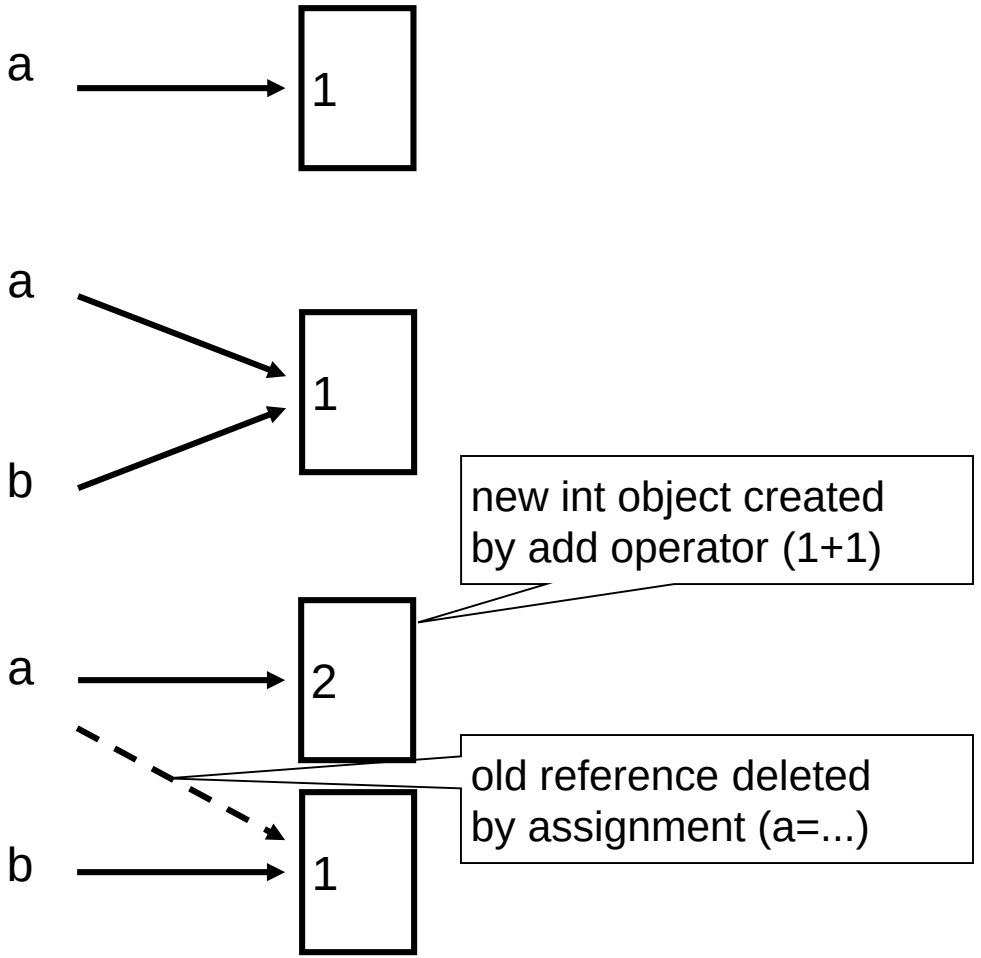


Changing an Integer

a = 1

b = a

a = a+1



Exercise: maximal subsequence

- Given an array of integers, e.g.
  - a = [−2, 1, −3, 4, −1, 2, 1, −5, 4];
- and called
  - S(h, k) = $\sum_{i=h}^{k} a[i]$
- the sum of subsequence from k to k, find the maximal S(h, k)
  - max S(h, k)
- For the array above max S(h, k) = S(3, 6) = 4-1+2+1= 6

- Variants: array of integers
  - passed on the command line
  - read from a text file (one int per line)

Exercise: lists and dictionaries

- Given the list: l = [12, 3, -4, 6, -5, 9]
- Given the dictionary:
  - d = {'apple':3, 'orange':4, 'tomato':-5, 'meat':6, 'potato':15, 'strawberry':9}
- If a value in the dictionary is found in the list, add the corresponding key to a string named 'to-buy' and print it at the end.
- If a value in the dictionary is not found in the list, chose another value in the list, that is not present in the dictionary, and assign it to the corresponding key. Print the updated dictionary at the end.

Exercise: lists

- Given 2 lists:

a = [12, 3, 4, 6, 5, 9]

b = [10, 3, 2, 6, 3, 7]

Compute the Pearson's correlation.

Try everything also generating lists with random elemements.