Introduction to python

Based in part on slides from:

http://www.seas.upenn.edu/~cis391/Lectures/python-review.pdf

http://www.seas.upenn.edu/~cis391/Lectures/python-tutorial.pdf

What is Python?

py·thon

ˈpīˌTHän,ˈpīTHən/

noun

1. a large heavy-bodied nonvenomous constrictor snake occurring throughout the Old World tropics.

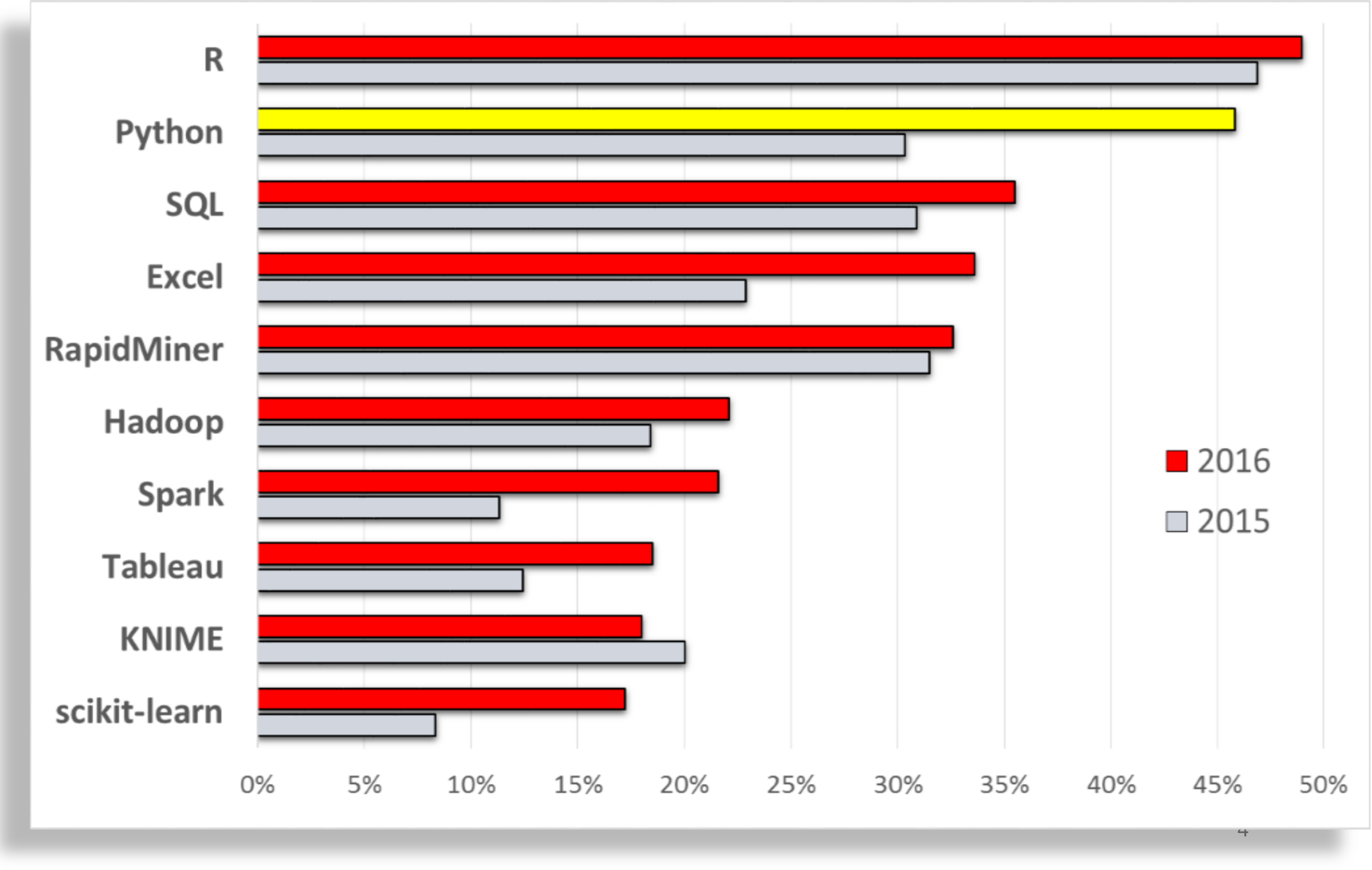
2. a high-level general-purpose programming language

Why Python?

- Free!
  - Available for download from www.python.org
- Relatively easy to use
  - Tons of online tutorials; documentation
- Powerful programming language
- One of the top 4 data science analytics software tools – it is great for your resume.

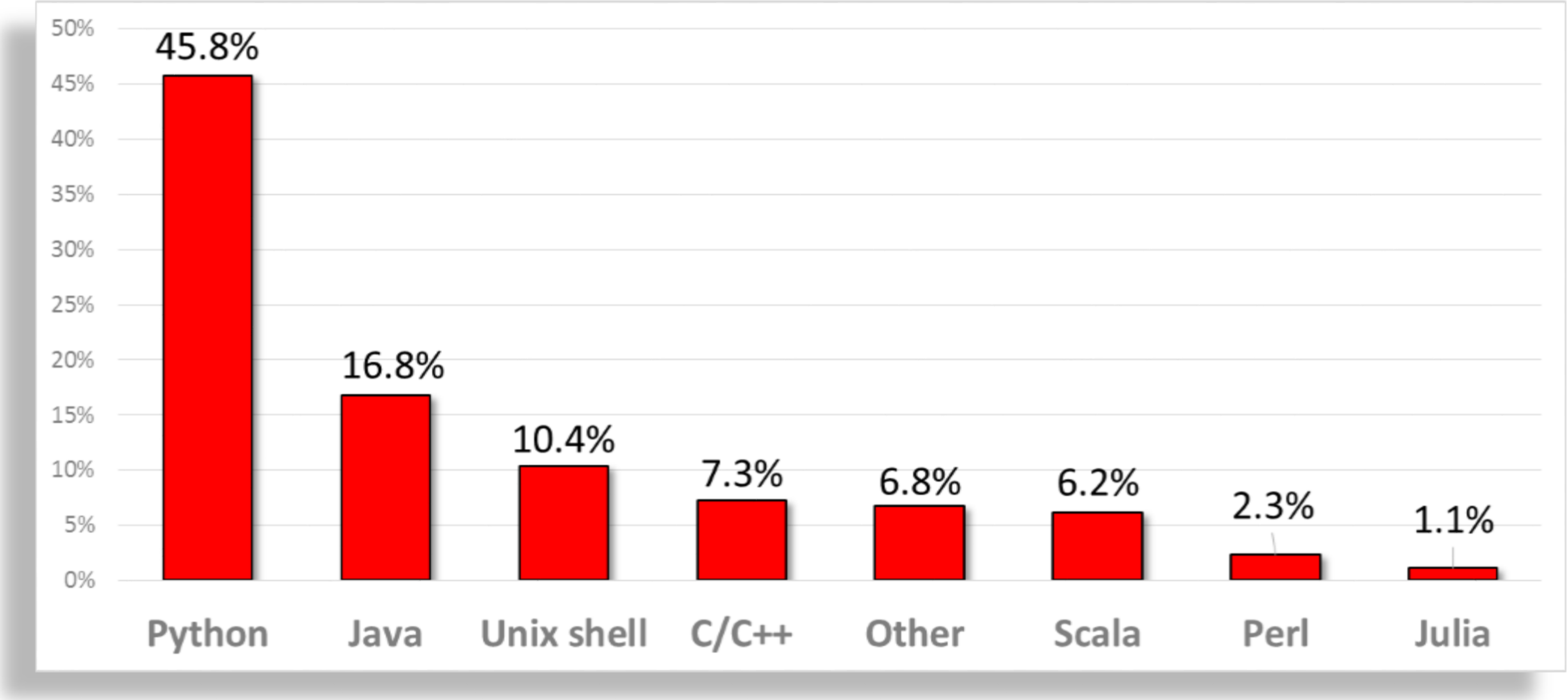
KDnuggets annual poll

What software you used for Analytics, Data Mining, Data Science, Machine Learning projects in the past 12 months?



KDnuggets annual poll

Among programming languages for data scientists: Python, Java, Unix tools, Scala grew in popularity, while C/C++, Perl, an Julia declined.

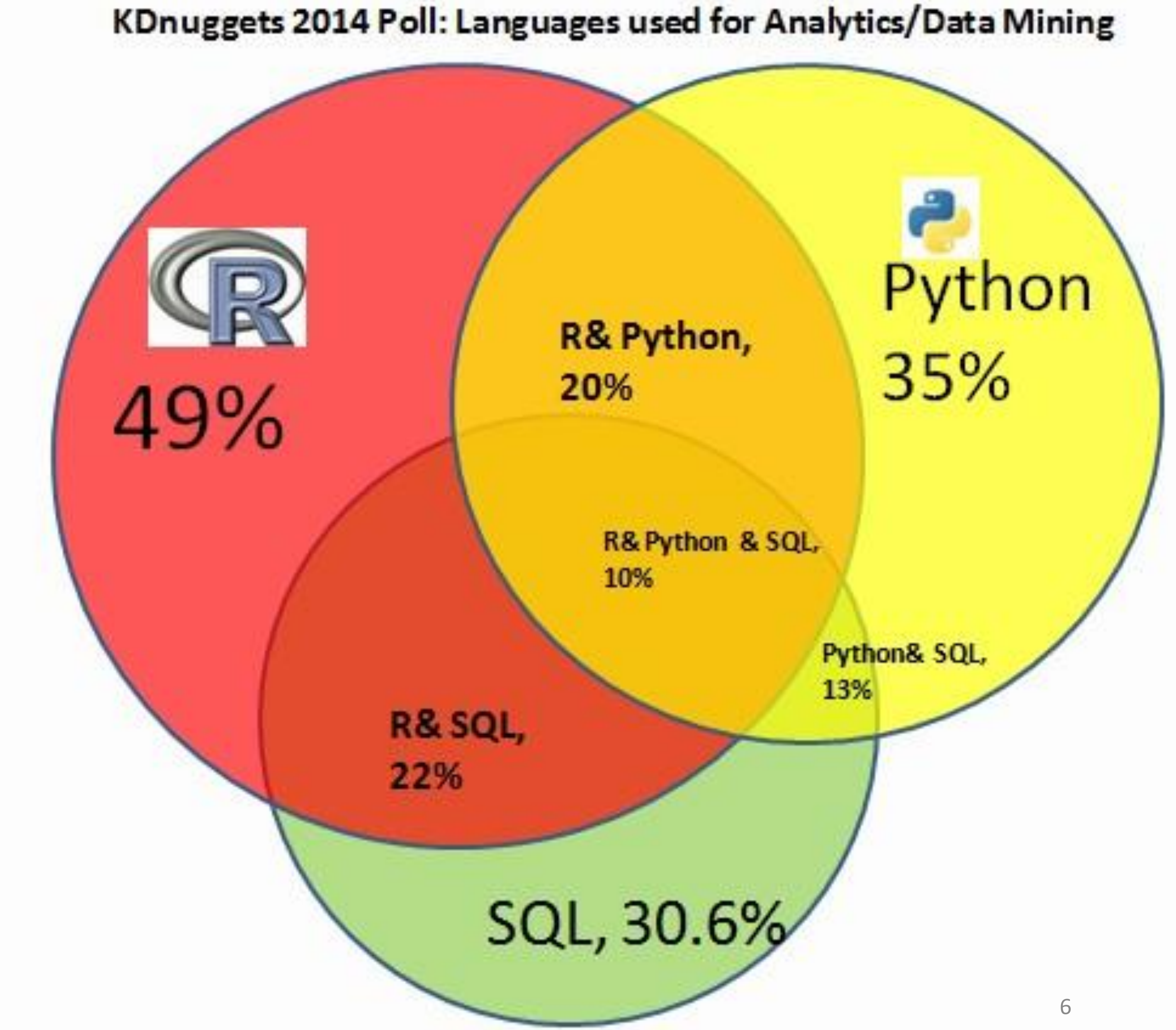


Data scientists do not just use 1 tool…

Only 1.4% said they only use R

Only 0.1% said they only use Python

No one said they only use SQL



Python Introduction

- Named after Monty Python
- Interpreted language
- Dynamically typed
- Portable
- Clean syntax
- Rich, built-in collection types:
  - Lists, Tuples, Dictionaries, Sets
- Large collection of support libraries: e.g. NumPy

Disclaimer: You may need to learn more on your own.

Things to check out

- https://docs.python.org/2/tutorial/index.html
- http://www.tutorialspoint.com/python/index.htm
- https://docs.python.org/2.7/
- http://www.davekuhlman.org/python_book_01.pdf

Lynda.com: Python 3 Essential Training

Which Python?

- **Python 2**
  - Last stable release before version 3
- **Python 3**
  - Not backward compatible…
  - Most third party software is now compatible with Python 3

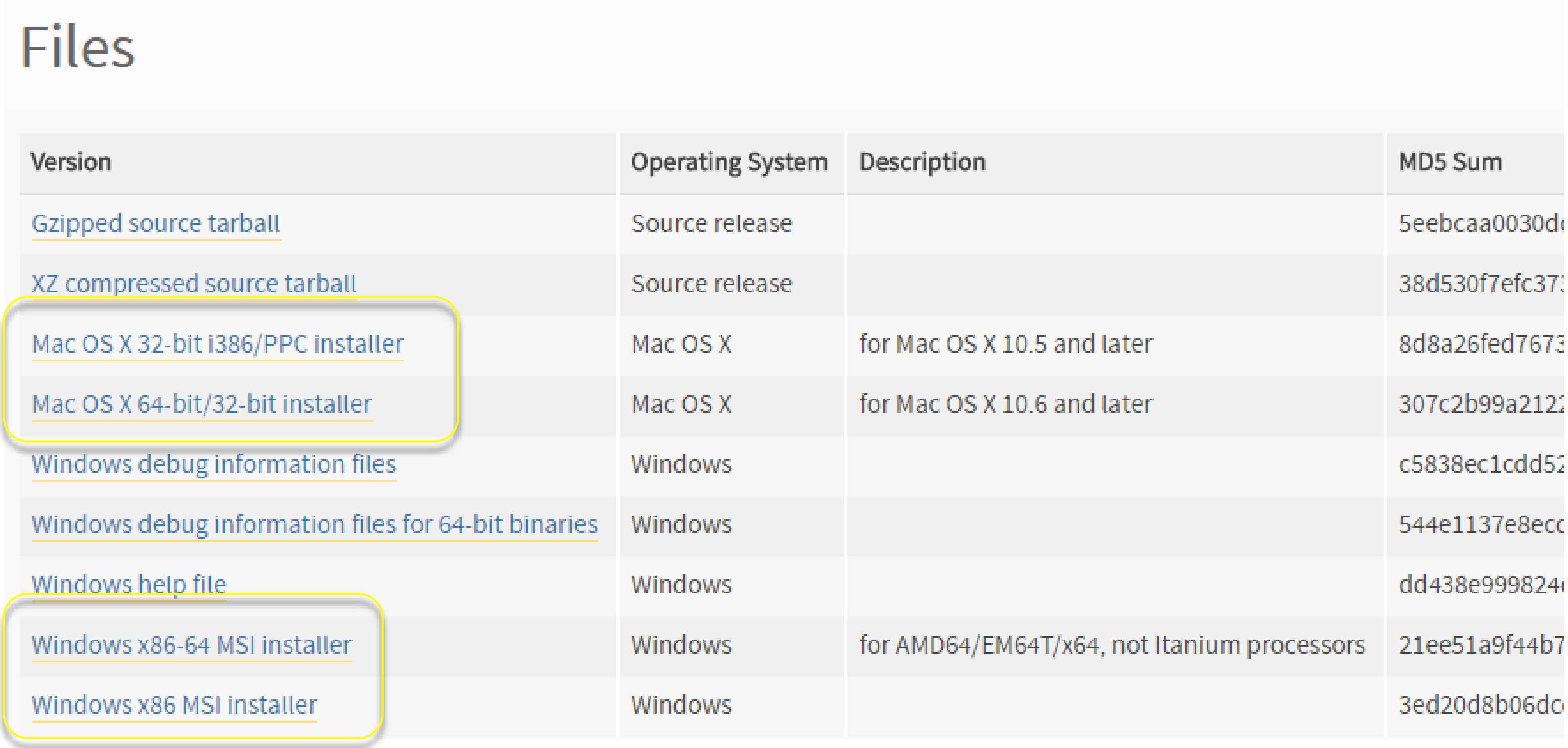
Installation

- You need the language
  - Python 3
- And some important packages…
- And you will want an IDE
  - IDLE
  - WingIDE

To install…

You need the language…

- https://www.python.org/downloads/release/python-279/

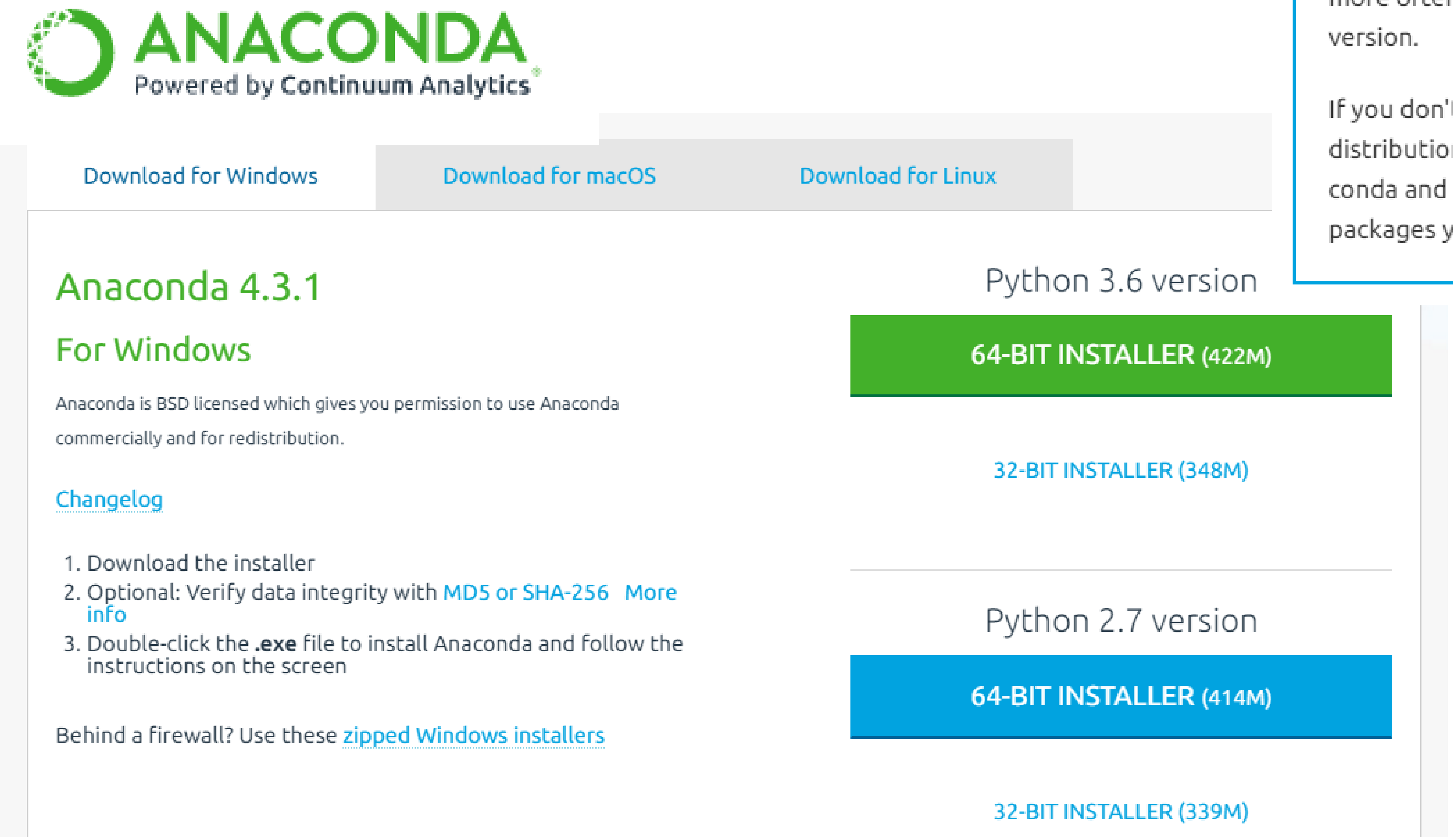


Files

Version	Operating System	Description	MD5 Sum
Gzipped source tarball	Source release		5eebcaa0030d
XZ compressed source tarball	Source release		38d530f7efc37
Mac OS X 32-bit i386/PPC installer	Mac OS X	for Mac OS X 10.5 and later	8d8a26fed7673
Mac OS X 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.6 and later	307c2b99a212
Windows debug information files	Windows		c5838ec1cdd52
Windows debug information files for 64-bit binaries	Windows		544e1137e8ecc
Windows help file	Windows		dd438e999824
Windows x86-64 MSI installer	Windows	for AMD64/EM64T/x64, not Itanium processors	21ee51a9f44b7
Windows x86 MSI installer	Windows		3ed20d8b06dc

Anaconda and miniconda

https://www.continuum.io/downloads



Anaconda 4.3.1

For Windows

Anaconda is BSD licensed which gives you permission to use Anaconda commercially and for redistribution.

Changelog

1. Download the installer
2. Optional: Verify data integrity with MD5 or SHA-256 More info
3. Double-click the **.exe** file to install Anaconda and follow the instructions on the screen

Behind a firewall? Use these zipped Windows installers

Python 3.6 version

Python 2.7 version

Which version should I download and install?

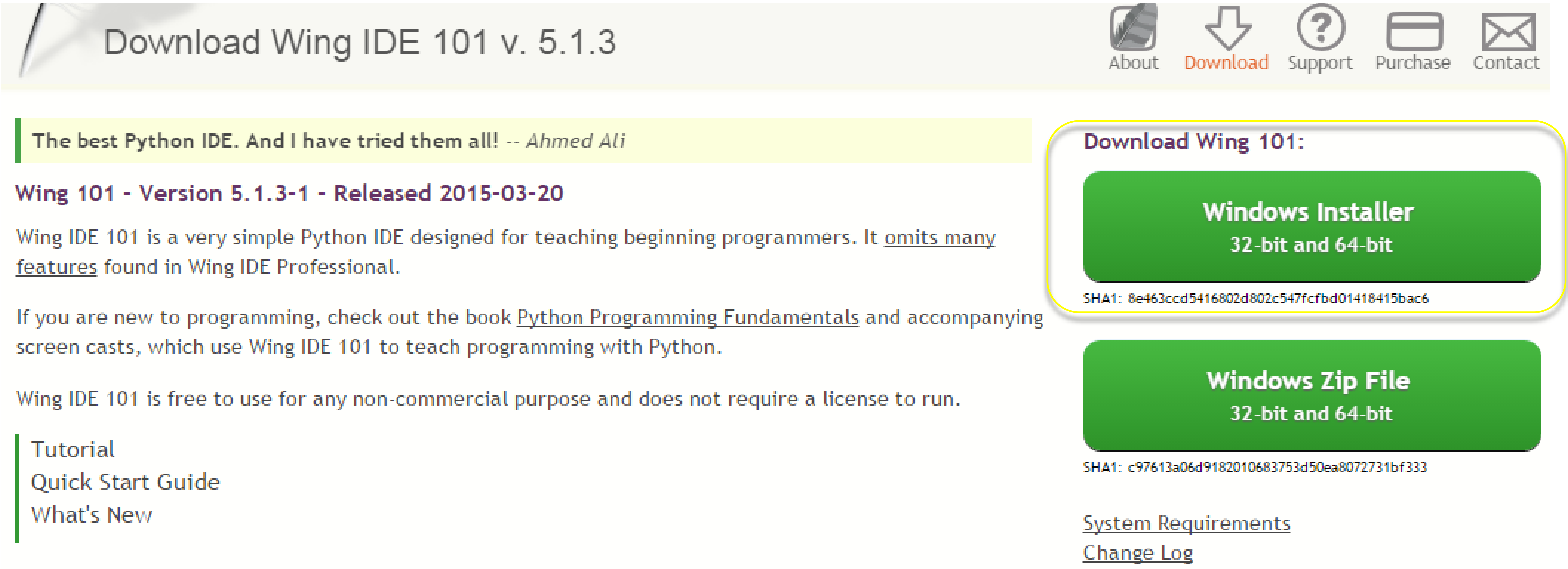
With Anaconda you can run multiple versions of Python in isolated environments, so choose the download with the Python version that you use more often, as that will be your default Python version.

If you don't have time or disk space for the entire distribution, try Miniconda which contains only conda and Python. Then install just the individual packages you want through the conda command.

To install…

You'll want an IDE…

- https://wingware.com/downloads/wingide-101



Download Wing IDE 101 v. 5.1.3

The best Python IDE. And I have tried them all! -- *Ahmed Ali*

Wing 101 - Version 5.1.3-1 - Released 2015-03-20

Wing IDE 101 is a very simple Python IDE designed for teaching beginning programmers. It omits many features found in Wing IDE Professional.

If you are new to programming, check out the book Python Programming Fundamentals and accompanying screen casts, which use Wing IDE 101 to teach programming with Python.

Wing IDE 101 is free to use for any non-commercial purpose and does not require a license to run.

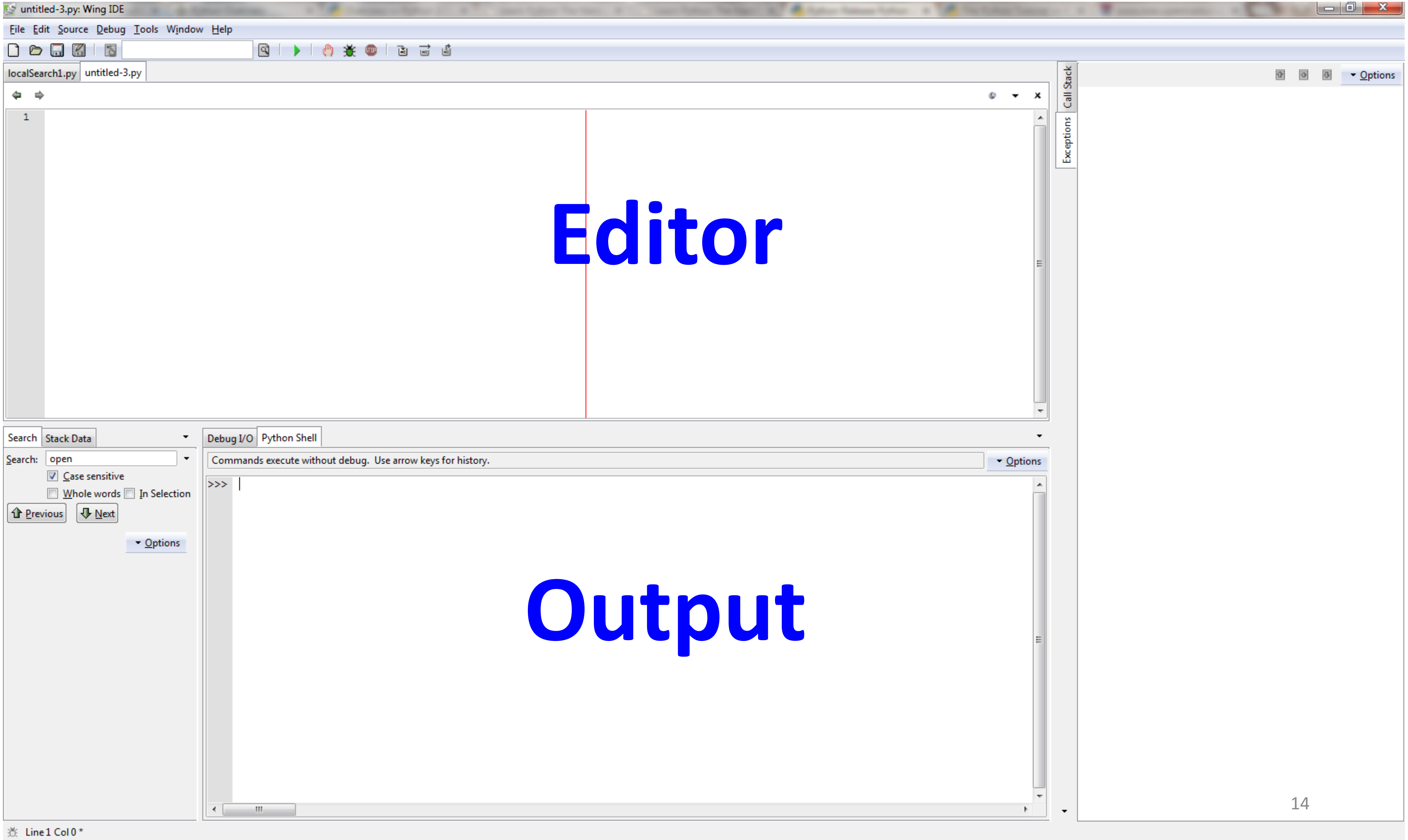
Tutorial
Quick Start Guide
What's New

Download Wing 101:

SHA1: 8e463ccd5416802d802c547fcfbd01418415bac6

SHA1: c97613a06d9182010683753d50ea8072731bf333

System Requirements
Change Log



Search: open

☑ Case sensitive

□ Whole words □ In Selection

Commands execute without debug. Use arrow keys for history.

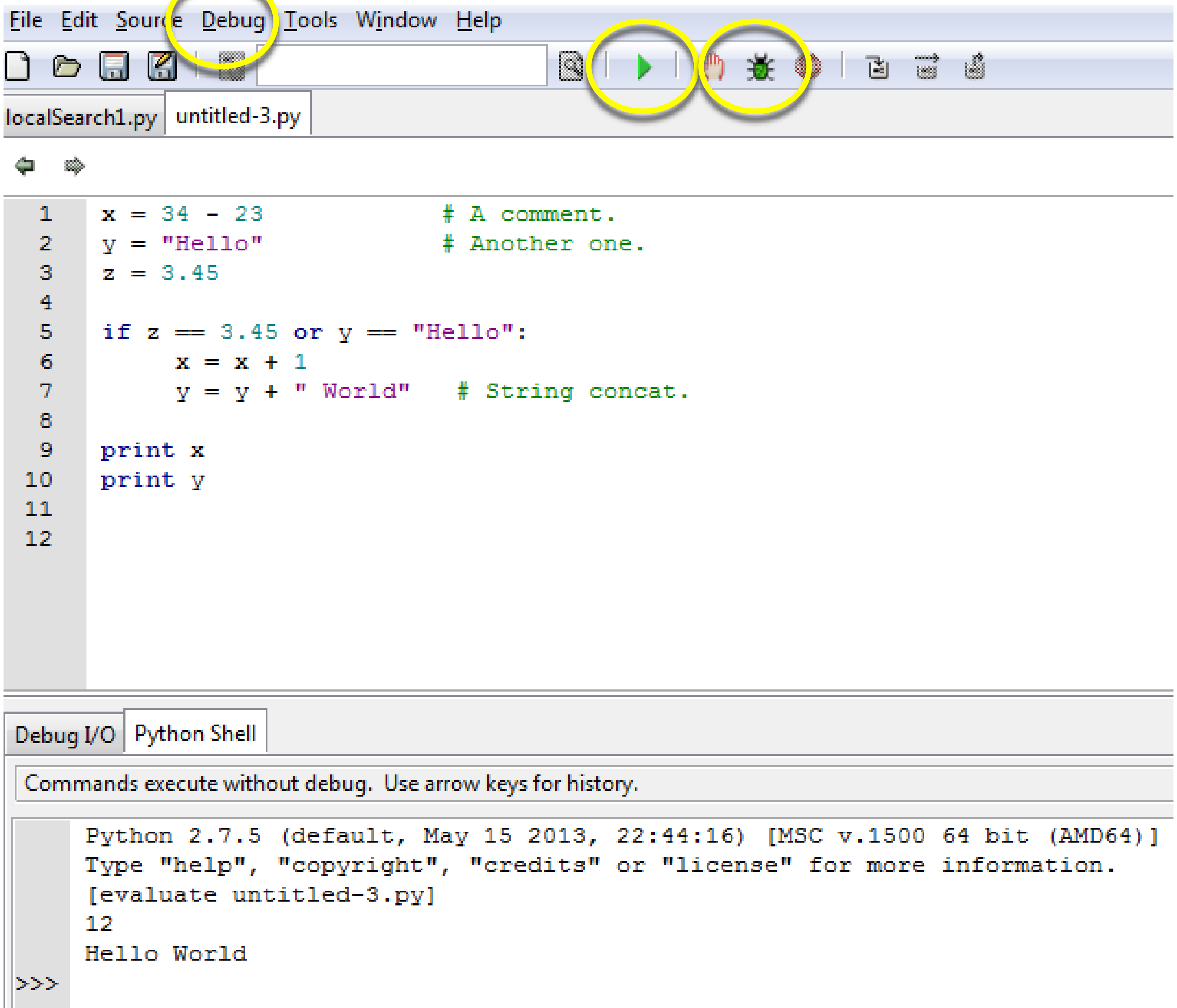
```
>>> 
```

Look at a sample of code…

```
x = 34 - 23            # A comment.
y = “Hello”            # Another one.
z = 3.45
if z == 3.45 or y == “Hello”:
    x = x + 1
    y = y + “ World”   # String concat.
print (x)
print (y)
```

Enough to Understand the Code

- Start comments with # (rest of line is ignored)
- Assignment uses = and comparison uses ==
- For numbers + - * / % are as expected.
    - Special use of + for string concatenation.
    - Special use of % for string formatting.
- Logical operators are words (**`and, or, not`**) *not symbols (&&, ||, !)*
- The basic printing command is “print.”
- First assignment to a variable will create it.
    - Variable types don’t need to be declared.
    - Python figures out the variable types on its own.



```
x = 34 - 23             # A comment.
y = "Hello"             # Another one.
z = 3.45

if z == 3.45 or y == "Hello":
    x = x + 1
    y = y + " World"    # String concat.

print x
print y
```

Commands execute without debug. Use arrow keys for history.

```
Python 2.7.5 (default, May 15 2013, 22:44:16) [MSC v.1500 64 bit (AMD64)]
Type "help", "copyright", "credits" or "license" for more information.
[evaluate untitled-3.py]
12
Hello World
>>>
```

Basic Datatypes

- Integers (default for numbers)

```
z = 5 / 2    # Answer is 2, integer division.
```

- Floats

```
x = 3.456
```

- Strings
  - Can use “ or ‘ to specify strings, e.g. “abc” ‘abc’
  - Use triple double-quotes for multi-line strings

Whitespace

Whitespace is meaningful in Python

- Use a newline to end a line of code.
  (Not a semicolon like in C++ or Java.)
  (Use \ when must go to next line prematurely.)
- No braces { } to mark blocks of code in Python…
  Use consistent indentation instead.
  - The first line with a new indentation is considered outside of the block.

Look at a sample of code…

```
x = 34 - 23            # A comment.
y = “Hello”            # Another one.
z = 3.45
if z == 3.45 or y == “Hello”:
    x = x + 1
    y = y + “ World”   # String concat.
print x
print y
```

Python and Types

Python determines the data types
in a program automatically. “Dynamic Typing”

But Python’s not casual about types, it enforces them after it figures them out. “Strong Typing”

So, for example, you can’t just append an integer to a string. You must first convert the integer to a string itself.

```
x = “the answer is ”   # Decides x is string.
y = 23                 # Decides y is integer.
print x + y            # Python will complain!
```

Naming Rules

- Names are case sensitive and cannot start with a number. They can contain letters, numbers, and underscores.

```
bob   Bob   _bob   _2_bob_   bob_2   BoB
```

- There are some reserved words:

```
and, assert, break, class, continue, def,
del, elif, else, except, exec, finally,
for, from, global, if, import, in, is,
lambda, not, or, pass, print, raise,
return, try, while
```

Multiple Assignment

You can also assign to multiple names at the same time.

```
x, y = 2, 3
```

String Operations

- We can use some methods built-in to the string data type to perform some formatting operations on strings:

```
“hello”.upper()
‘HELLO’
```

- There are many other handy string operations available. Check the Python documentation for more.

Math commands

Command name	Description
`abs(`***value***`)`	absolute value
`ceil(`***value***`)`	rounds up
`cos(`***value***`)`	cosine, in radians
`floor(`***value***`)`	rounds down
`log(`***value***`)`	logarithm, base *e*
`log10(`***value***`)`	logarithm, base 10
`max(`***value1***`,` ***value2***`)`	larger of two values
`min(`***value1***`,` ***value2***`)`	smaller of two values
`round(`***value***`)`	nearest whole number
`sin(`***value***`)`	sine, in radians
`sqrt(`***value***`)`	square root

Constant	Description
`e`	2.7182818...
`pi`	3.1415926...

Python has useful commands for performing calculations.

To use many of these commands, you must write the following at the top of your Python program: `from math import *`

Data types: Lists

- A compound data type:

```
x = [0]
y = [2.3, 4.5]
myList = [5, "Hello", "there", 9.8]
z = []
```

Note: use brackets [ ] to define a list

- Lists are *mutable*
- Use len() to get the length of a list

```
names = [“Ben", “Chen", “Yaqin"]
len(names)
3
```

Use [ ] to index items in the list

```
>>> names[0]
‘Ben'
>>> names[1]
‘Chen'
>>> names[2]
‘Yaqin'
>>> names[3]
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
IndexError: list index out of range
>>> names[-1]
‘Yaqin'
>>> names[-2]
‘Chen'
>>> names[-3]
‘Ben'
```

[0] is the first item.
[1] is the second item

...

Out of range values
raise an exception

Negative values
go backwards from
the last element.

Fun with lists…

.append(value) - appends element to end of the list

.count('x') - counts the number of occurrences of 'x' in the list

.insert('y','x') - inserts 'x' at location 'y'

.pop() - returns last element then removes it from the list

.remove('x') - finds and removes first 'x' from list

.reverse() - reverses the elements in the list

.sort() - sorts the list alphabetically in ascending order, or numerical in ascending order

Lists: Modifying Content

```
>>> x = [1,2,3]
>>> y = x
>>> x[1] = 15
>>> x
[1, 15, 3]
>>> y
[1, 15, 3]
>>> x.append(12)
>>> y
[1, 15, 3, 12]
```

- **two named lists, x and y**
- **x[i] = a** reassigns the i^{th} element to the value **a**
- **Since x and y point to the same list object, *both* are changed**
- The method **append** also modifies the list

Data types: Tuples

- A tuple is the same as a list, except for one difference: the tuples cannot be changed.
- Creating a tuple is as simple as follows:

```
tup1 = ('physics', 'chemistry', 1997, 2000)
tup2 = (1, 2, 3, 4, 5 )
```

Note: use parenthesis ( ) to define a tuple

Sets

- A python **set** is an unordered collection of unique items.
- Since sets are defined to be composed of unique items; they will automatically eliminate duplicates.
- With sets you can perform operations like **union**, **intersection**, **difference**.
- Since sets are unordered, however, you cannot access their elements using the slicing operator.
- Sets are defined using braces.

Data types: Dictionaries

- Dictionaries are lookup tables.
- They map from a “key” to a “value”.

```
symbol_to_name = {
    "H": "hydrogen",
    "He": "helium",
    "Li": "lithium",
    "C": "carbon",
    "O": "oxygen",
    "N": "nitrogen"
}
```

Note: use curly brackets { } to define a dictionary

- Duplicate keys are not allowed
- Duplicate values are just fine

Dictionary

```
>>> symbol_to_name["C"]
'carbon'
```

Get the value for a given key

```
>>> "O" in symbol_to_name
True
```

Test if the key exists
(“in” only checks the keys,
not the values.)

```
>>> "oxygen" in symbol_to_name
False

>>> symbol_to_name["P"]
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
KeyError: 'P'
```

Copying Dictionaries and Lists

- The built-in list function will copy a list
- You can also use the slicing operator
- The dictionary has a method called copy

```
>>> L1 = [1]
>>> L2 = list(L1)
>>> L1[0] = 22
>>> L1
[22]
>>> L2
[1]

>>> L2 = L1[:]
```

```
>>> d = {1 : 10}
>>> d2 = d.copy()
>>> d[1] = 22
>>> d
{1: 22}
>>> d2
{1: 10}
```

Data Type Summary

- **Lists**, **Tuples**, and **Dictionaries** can store any type (*including other lists, tuples, and dictionaries!)*
- Only lists and dictionaries are mutable
- **All variables are references**

Data Type Summary

- Integers: 2323, 3234L
- Floating Point: 32.3, 3.1E2
- Complex: 3 + 2j, 1j
- Lists: l = [1,2,3]
- Tuples: t = (1,2,3)
- Dictionaries: d = {‘hello’ : ‘there’, 2 : 15}