Python 0

Some material adapted
from Upenn cmpe391
slides and other sources

Overview

- History
- Significance
- Installing & Running Python
- Simple script examples

Brief History of Python

- Invented in the Netherlands, early 90s by Guido van Rossum
- Named after Monty Python
- Open sourced from the beginning, man-aged by [Python Software Foundation](#)
- Considered a scripting language, but is much more
- Scalable, object oriented and functional from the beginning
- Used by Google from the beginning

Python's Benevolent Dictator For Life

"Python is an experiment in how much freedom program-mers need. Too much freedom and nobody can read another's code; too little and expressive-ness is endangered."
   - Guido van Rossum

Python's place in the Market

- TIOBE has been collecting data on programming language "popularity" for many years
- Counts results for a query like "<language> programming" on popular search engines

Tiobe Programming Community Index
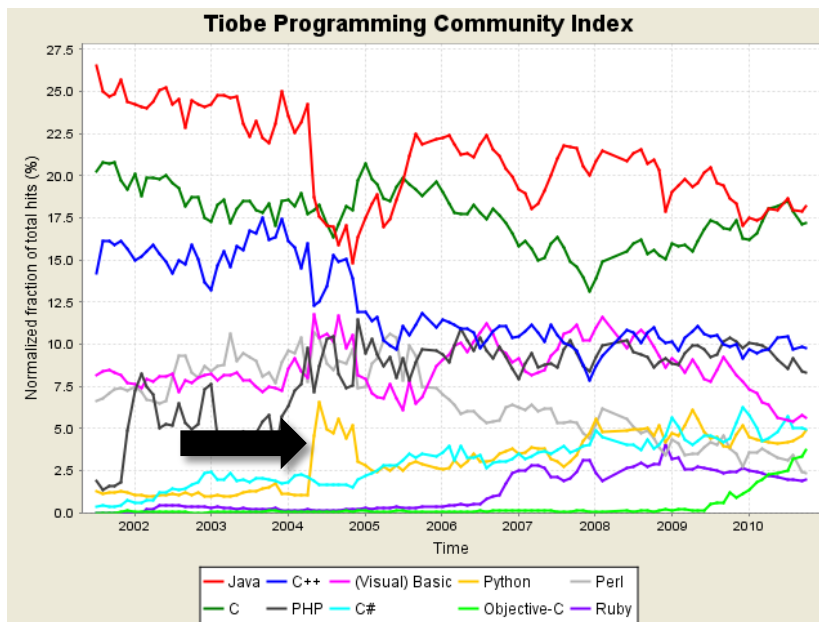
Position Oct 2010	Position Oct 2009	Delta in Position	Programming Language	Ratings Oct 2010	Delta Oct 2009	Status
1	1	=	Java	18.166%	-0.48%	A
2	2	=	C	17.177%	+0.33%	A
3	4	↑	C++	9.802%	-0.08%	A
4	3	↓	PHP	8.323%	-2.03%	A
5	5	=	(Visual) Basic	5.650%	-3.04%	A
6	6	=	C#	4.963%	+0.55%	A
7	7	=	Python	4.860%	+0.96%	A
8	12	↑↑↑↑	Objective-C	3.706%	+2.54%	A
9	8	↓	Perl	2.310%	-1.45%	A
10	10	=	Ruby	1.941%	-0.51%	A
11	9	↓↓	JavaScript	1.659%	-1.37%	A
12	11	↓	Delphi	1.558%	-0.58%	A
13	17	↑↑↑↑	Lisp	1.084%	+0.48%	A-
14	24	↑↑↑↑↑↑↑↑↑↑	Transact-SQL	0.820%	+0.42%	A-
15	15	=	Pascal	0.771%	+0.10%	A-
16	18	↑↑	RPG (OS/400)	0.708%	+0.12%	A-
17	29	↑↑↑↑↑↑↑↑↑↑↑↑	Ada	0.704%	+0.40%	A--
18	14	↓↓↓↓	SAS	0.664%	-0.14%	B
19	19	=	MATLAB	0.627%	+0.05%	B
20	-	↑↑↑↑↑↑↑↑↑↑↑	Go	0.626%	+0.63%	B

10/10 TIOBE Programming Community Index

http://python.org/

http://docs.python.org/

Python v2.7 documentation »   modules | index

Download

Download these documents

Docs for other versions

Python 2.6 (stable)
Python 3.1 (stable)
Python 3.2 (in development)
Old versions

Other resources

FAQs
Guido's Essays
New-style Classes
PEP Index
Beginner's Guide
Book List
Audio/Visual Talks
Other Doc Collections
Report a Bug

Quick search

[   ]

Go

Enter search terms or a module, class or function name.
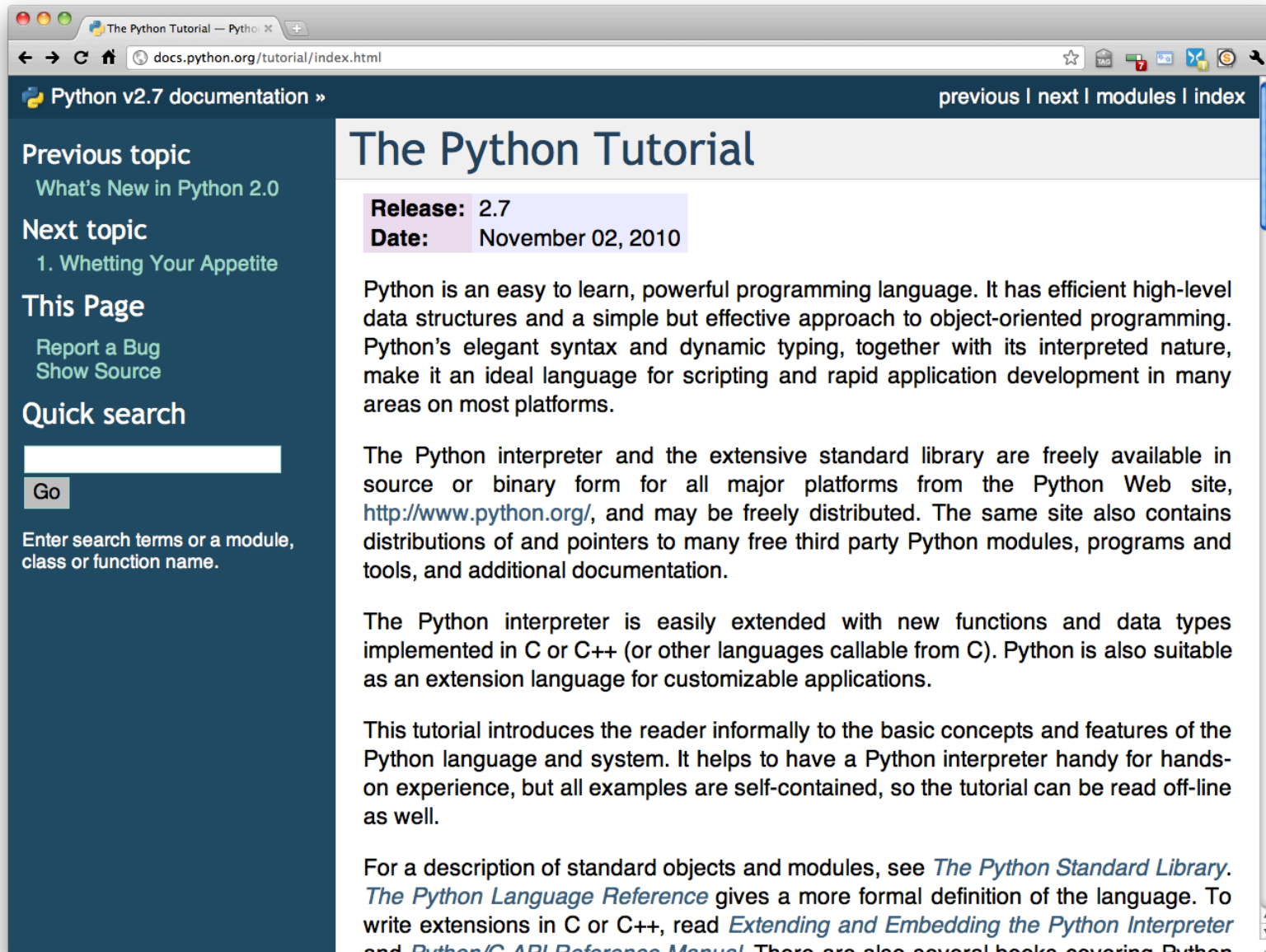
Python v2.7 documentation

Welcome! This is the documentation for Python 2.7, last updated Nov 02, 2010.

Parts of the documentation:

What's new in Python 2.7?
or all "What's new" documents since 2.0

Tutorial
start here

Library Reference
keep this under your pillow

Language Reference
describes syntax and language elements

Python Setup and Usage
how to use Python on different platforms

Python HOWTOs
in-depth documents on specific topics

Extending and Embedding
tutorial for C/C++ programmers

Python/C API
reference for C/C++ programmers

Installing Python Modules
information for installers & sys-admins

Distributing Python Modules
sharing modules with others

Documenting Python
guide for documentation authors

FAQs
frequently asked questions (with answers!)

Indices and tables:

The Python tutorial is good!

docs.python.org/tutorial/index.html

Python v2.7 documentation »

previous | next | modules | index

The Python Tutorial

Previous topic

What's New in Python 2.0

Next topic

1. Whetting Your Appetite

This Page

Report a Bug
Show Source

Quick search

Go

Enter search terms or a module,
class or function name.

| **Release:** | 2.7 |
| **Date:** | November 02, 2010 |

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

The Python interpreter and the extensive standard library are freely available in source or binary form for all major platforms from the Python Web site, http://www.python.org/, and may be freely distributed. The same site also contains distributions of and pointers to many free third party Python modules, programs and tools, and additional documentation.

The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.

This tutorial introduces the reader informally to the basic concepts and features of the Python language and system. It helps to have a Python interpreter handy for hands-on experience, but all examples are self-contained, so the tutorial can be read off-line as well.

For a description of standard objects and modules, see *The Python Standard Library*. *The Python Language Reference* gives a more formal definition of the language. To write extensions in C or C++, read *Extending and Embedding the Python Interpreter* and *Python/C API Reference Manual*. There are also several books covering Python

Running Python
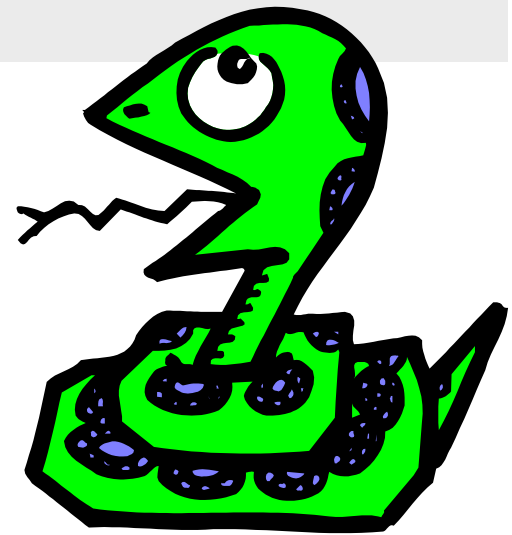
The Python Interpreter

- Typical Python implementations offer both an interpreter and compiler
- Interactive interface to Python with a read-eval-print loop

```
[finin@linux2 ~]$ python
Python 2.4.3 (#1, Jan 14 2008, 18:32:40)
[GCC 4.1.2 20070626 (Red Hat 4.1.2-14)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> def square(x):
...   return x * x
...
>>> map(square, [1, 2, 3, 4])
[1, 4, 9, 16]
>>>
```

Installing
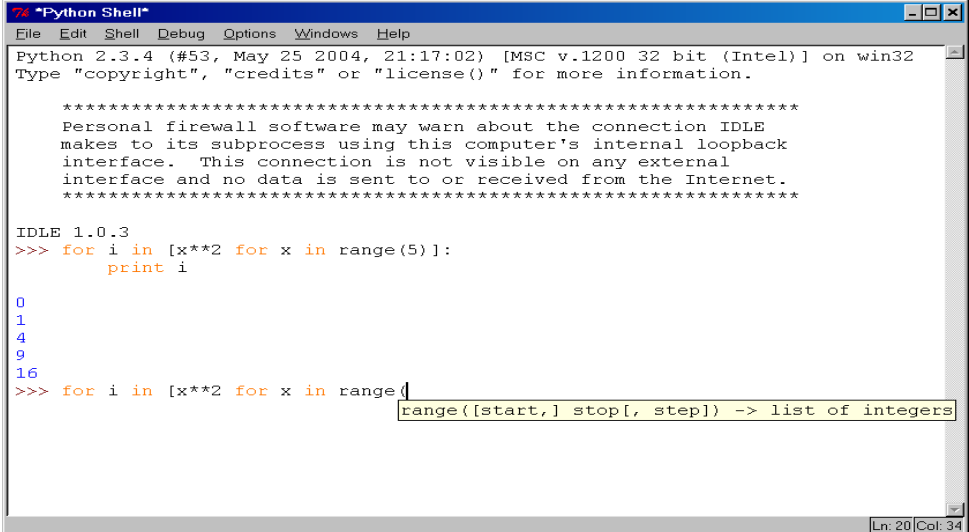
- Python is pre-installed on most Unix systems, including Linux and MAC OS X
- The pre-installed version may not be the most recent
- Two "latest versions"
  - 2.7 released 7/10 & 3.2 released 10/10
  - Python 3 is a non-backward compatible version which you should not use for 331
- Download from http://python.org/download/
- Python comes with a large library of standard modules

Python IDEs and Shells

- There are many Integrated Development Environments
  - IDLE
  - Eclipse + PyDev
  - Emacs
- As well as enhanced shells

IDLE Development Environment

- [IDLE](#) is the "official" IDE distributed with Python
- Preinstalled on MAC OS X
- Written in Python with the [Tkinter](#) GUI package
- Multi-window text editor with syntax highlighting, auto-completion, smart indent and other features
- Python shell with syntax highlighting, line recall, …
- Integrated debugger with stepping, persis- tent breakpoints, and call stack visi- bility

```
7% *Python Shell*                                          _ □ ×
File  Edit  Shell  Debug  Options  Windows  Help
Python 2.3.4 (#53, May 25 2004, 21:17:02) [MSC v.1200 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.

      ***********************************************************
      Personal firewall software may warn about the connection IDLE
      makes to its subprocess using this computer's internal loopback
      interface.  This connection is not visible on any external
      interface and no data is sent to or received from the Internet.
      ***********************************************************

IDLE 1.0.3
>>> for i in [x**2 for x in range(5)]:
        print i

0
1
4
9
16
>>> for i in [x**2 for x in range(
                              range([start,] stop[, step]) -> list of integers
                                                              Ln: 20 Col: 34
```
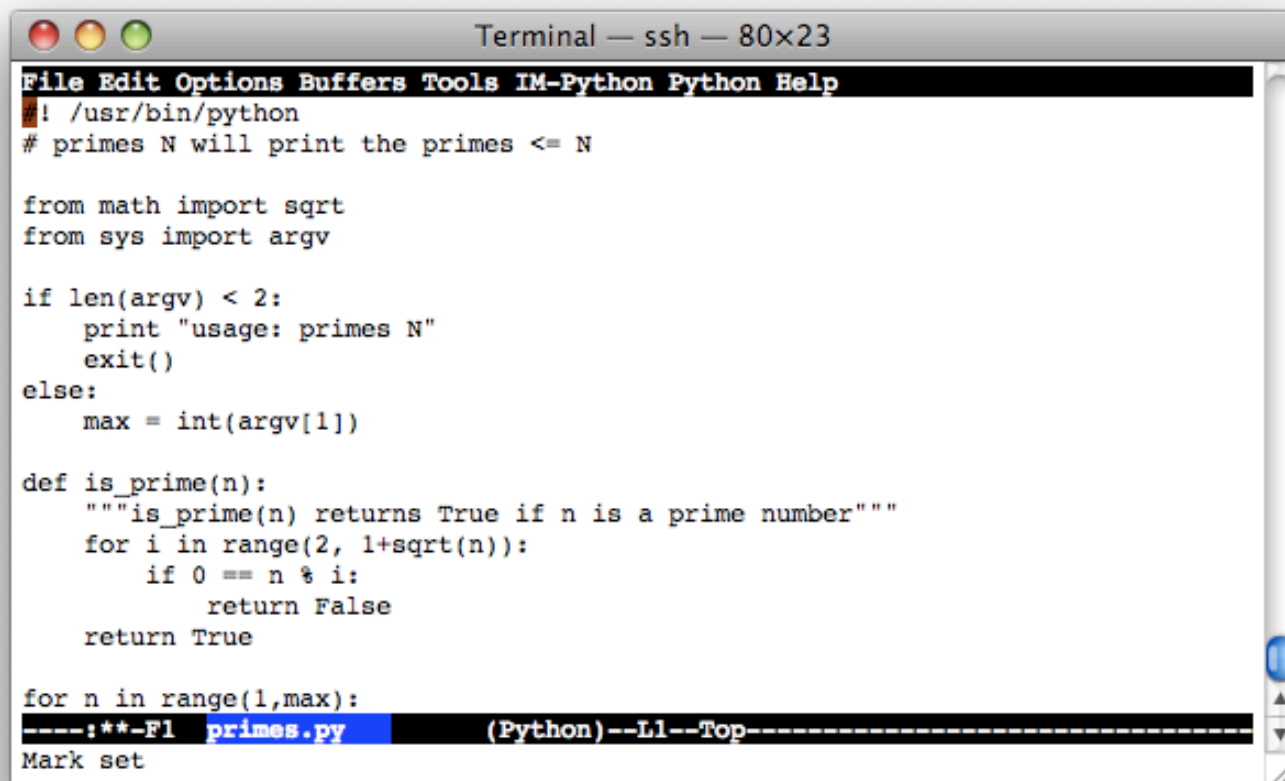
Eclipse + Pydev

- Pydev is an Eclipse plugin for Python

- Download from http://pydev.org/

- Syntax highlighting, code completion, goto function, debugger, …

Editing Python in Emacs

- Emacs *python-mode.el* has good support for editing Python, enabled enabled by default for .py files
- Features: completion, symbol help, eldoc, and inferior interpreter shell, etc.

Emacs as a Python IDE

- You can fire up a shell in emacs via M-x python-shell
- You can also set up a more powerful Python IDE environment in EMACS
  - Pymacs allows two-way communication between Emacs Lisp and Python
  - Ropemacs provides advanced features such as completion, refactoring, etc

Running Interactively on UNIX

On Unix…

```
% python
>>> 3+3
6
```

- **Python prompts with '>>>'.**
- **To exit Python (not Idle):**
  - In Unix, type CONTROL-D
  - In Windows, type CONTROL-Z + <Enter>
  - Evaluate exit()

Running Programs on UNIX

- Call python program via the python interpreter

  ```
  % python fact.py
  ```

- Make a python file directly executable by

  - Adding the appropriate path to your python interpreter as the first line of your file

    ```
    #!/usr/bin/python
    ```

  - Making the file executable

    ```
    % chmod a+x fact.py
    ```

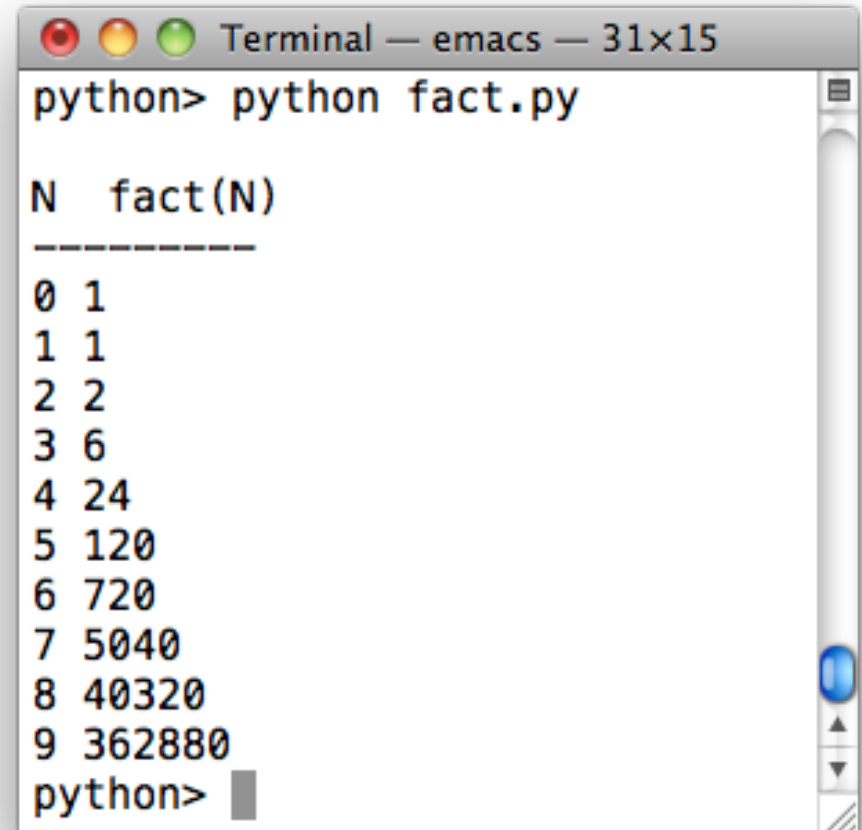 - Invoking file from Unix command line

    ```
    % fact.py
    ```

Example 'script': fact.py

```python
#! /usr/bin/python

def fact(x):
    if x == 0:
        return 1
    return x * fact(x - 1)

print "\nN  fact(N)"
print "---------"
for n in range(10):
    print n, fact(n)
```

```
Terminal — emacs — 31×15
python> python fact.py

N  fact(N)
---------
0 1
1 1
2 2
3 6
4 24
5 120
6 720
7 5040
8 40320
9 362880
python>
```

Python Scripts

- When you call a python program from the command line the interpreter evaluates each expression in the file

- Familiar mechanisms are used to provide command line arguments and/or redirect input and output

- Python also has mechanisms to allow a python program to act both as a script and as a module to be imported and used by another python program

Another Script Example

```
#! /usr/bin/python

""" reads text from stdin and outputs any email
    addresses it finds, one to a line """

import re

from sys import stdin

# a regular expression for a valid email address
pat = re.compile(r'[-\w][-.\w]*@[-\w][-\w.]+[a-zA-Z]{2,4}')

for line in stdin:
    for address in pat.findall(line):
        print address
```

results

```
python> python email0.py <email.txt
bill@msft.com
gates@microsoft.com
steve@apple.com
bill@msft.com
python>
```

Getting a unique, sorted list

```
import re
from sys import stdin

pat = re.compile(r'[-\w][-.\w]*@[-\w][-\w.]+[a-zA-Z]{2,4}')
# found is an initially empty set (a list w/o duplicates)
found = set()
for line in stdin:
    for address in pat.findall(line):
        found.add(address)

# sorted() takes a sequence, returns a sorted list of its elements
for address in sorted(found):
    print address
```

results

```
python> python email2.py <email.txt
bill@msft.com
gates@microsoft.com
steve@apple.com
python>
```

Conclusion: Python is ..

- Popular as a scripting language
- Popular as a general purpose language
- Open sourced
- Interesting from a program language perspective
- Easy to learn and use, so being used in many CS 101 courses