Introduction To Programming: Part 2

Why Python again?

Ease of use

Rapid Development

Readability

One Way to do things

Builtins

Standard Library

Third Pary Libraries

Language Features

The Zen of Python, by Tim Peters

```
>>>import this
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

Builtin Functions

abs()	dict()	help()	min()	setattr()
all()	dir()	hex()	next()	slice()
any()	divmod()	id()	object()	sorted()
ascii()	enumerate()	input()	oct()	staticmethod()
bin()	eval()	int()	open()	str()
bool()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
set()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	__import__()
complex()	hasattr()	max()	round()	classmethod()
delattr()	hash()	memoryview()		

Builtin Functions

abs()	dict()	help()	min()	setattr()
all()	dir()	hex()	next()	slice()
any()	divmod()	id()	object()	sorted()
ascii()	enumerate()	input()	oct()	staticmethod()
bin()	eval()	int()	open()	str()
bool()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
set()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	__import__()
complex()	hasattr()	max()	round()	classmethod()
delattr()	hash()	memoryview()		

Builtin Functions

`help(<object>)`

`Get help with an object`

`dir(<object>)`

`List all variables associated with an object`

`ord(<char>)`

`Convert a character to its unicode integer value`

`chr(<number>)`

`Convert a number to its unicode character value`

`abs(<number>)`

`Convert a number to its absolute value`

`round(<number>,<integer>)`

`Round a number to a certain amount of digits`

`complex(<number>, <number>)`

`Make a complex number from a real and imaginary part`

Builtin Functions

`all(<iterable>)`

`Return True if all elements of the iterable are true`

`any(<iterable>)`

`Return True if any elements of the iterable are True`

`sum(<iterable>)`

`Returns the sum of all of the variables in the iterable`

`sorted(<iterable>)`

`Returns a sorted version of the iterable`

`reversed(<iterable>)`

`Returns the a reversed version of the iterable`

`len(<iterable>)`

`Returns the length of the iterable`

Builtin Functions

`bool(<object>)`

`Convert to object to a True or False value`

`float(<object>)`

`Convert the object to a float`

`int(<object>)`

`Convert the object to an int`

`str(<object>)`

`Get the string representation of the object`

`repr(<object>)`

`Get the canonical representation of the object`

`map(<function>, <iterables>)`

`Make a function that consumes values in the iterables to evaluate the function`

`filter(<function>, <iterables>)`

`Run the function on each of the objects in the iterable and return the ones which are True`

Builtin Functions

`min(<iterable>)`

`Return the smallest value in the iterable`

`max(<iterable>)`

`Return the largest value in the iterable`

`zip(<iterable>,<iterable>)`

`Zip iterables together into a combined iterable`

`enumerate(<iterable>)`

`Enumerate an iterable so that you can reference the object and its position`

`open(<filepath>,<mode>)`

`Opens a file for reading, 'r' or writing 'w' or both 'r+w'`

`input(<prompt string>)`

`Takes string input from a user`

`print(<string>)`

`Prints a string to a file, typically the file is the terminal stdout`

Builtin Functions

`list()`

`make a list object`

`tuple()`

`make a tuple object`

`dict()`

`make a dict object`

`vars(<dict>)`

`list all of the variables in a dict object`

`set()`

`make a set object`

`frozenset()`

`make a frozenset object`

PyPI – pypi.python.org

Libraries to do almost anything you would want to do in python.

Don't reinvent the wheel, get it from PyPI instead.

The Python standard library contains mostly “static” libraries. PyPI has libraries that need to change to stay up to date.

PyPI is also known as the cheese shop

PyPI – pypi.python.org

Installing pip

<u>Debian/Ubuntu/Mint:</u>

```
sudo apt-get install python-pip
```

<u>Fedora:</u>

```
sudo yum install python-pip
```

<u>Arch:</u>

```
sudo pacman -S python-pip
```

<u>OSX:</u>

```
sudo easy_install pip
```

PyPI – pypi.python.org

Installing pip

Windows:

Get the setuptools installer from

https://bitbucket.org/pypa/setuptools/raw/bootstrap/ez_setup.py

Get the pip instaler from

https://raw.github.com/pypa/pip/master/contrib/get-pip.py

Change directory to wherever you downloaded these files

Type: `python ez_setup.py`

When that's finished installing

Type: `python get-pip.py`

PyPI – pypi.python.org

Using pip

Pip is really simple to use.

```
pip install <packagename>
pip list –outdated
pip install –upgrade <packagename>
pip uninstall <packagename>
pip freeze > requirements.txt
```

Virtualenv

“It worked on my machine!”

Virtualenv allows you to define which versions of packages should be used with your program.

Solves problems with dependencies and makes your program distributable

Great for webapps and games!

Virtualenv

```
# Make a new empty virtualenv
virtualenv my_project –no-site-packages
# Use the virtualenv
source my_project/bin/activate
# Add a library to the virtualenv
pip install yolk
# List the packages that are currently in the virtualenv
yolk -l
```

More Resources

Learning

http://learnpythonthehardway.org/book/

http://www.diveintopython3.net/

http://www.codecademy.com/tracks/python

http://swaroopch.com/notes/Python_en-Preface/

http://inventwithpython.com/chapters/

Advanced Learning

http://newcoder.io/

http://www.checkio.org

http://www.reddit.com/r/dailyprogrammer

http://www.learningpython.com/

http://pleac.sourceforge.net/pleac_python/index.html

More Resources

Idiomatic Python

http://python.net/~goodger/projects/pycon/2007/idiomatic/handout.html

https://intermediate-and-advanced-software-carpentry.readthedocs.org/en/latest/idiomatic-python.html

https://speakerdeck.com/pyconslides/transforming-code-into-beautiful-idiomatic-python-by-raymond-hettinger-1

http://www.jeffknupp.com/blog/2012/10/04/writing-idiomatic-python/

More Resources

Web Development

http://flask.pocoo.org/

https://www.djangoproject.com/

http://www.web2py.com/

http://www.appscale.com/

http://bottlepy.org/docs/dev/

http://twistedmatrix.com/trac/

http://www.pylonsproject.org/

More Resources

Games

http://www.pygame.org/

https://www.panda3d.org/

http://www.pyglet.org/

http://ignifuga.org/

http://arcticpaint.com/projects/rabbyt/

http://blender.org

More Resources

Science

www.scipy.org/

www.numpy.org/

http://pandas.pydata.org/

http://ipython.org/

http://rpy.sourceforge.net/

http://matplotlib.org/

http://biopython.org/wiki/Main_Page