**Useful Third-Party Packages**

Being able to install and import third party libraries is useful, but to be an effective programmer you also need to know what libraries are available for you to use. People typically learn about useful new libraries from online recommendations or from colleagues. If you're a new Python programmer you may not have many colleagues, so to get you started here's a list of packages that are popular with engineers at Udacity.

* [**IPython(opens in a new tab)**](https://ipython.org/) - A better interactive Python interpreter
* [**requests(opens in a new tab)**](http://docs.python-requests.org/) - Provides easy to use methods to make web requests. Useful for accessing web APIs.
* [**Flask(opens in a new tab)**](http://flask.pocoo.org/) - a lightweight framework for making web applications and APIs.
* [**Django(opens in a new tab)**](https://www.djangoproject.com/) - A more featureful framework for making web applications. Django is particularly good for designing complex, content heavy, web applications.
* [**Beautiful Soup(opens in a new tab)**](https://www.crummy.com/software/BeautifulSoup/) - Used to parse HTML and extract information from it. Great for web scraping.
* [**pytest(opens in a new tab)**](http://doc.pytest.org/) - extends Python's builtin assertions and unittest module.
* [**PyYAML(opens in a new tab)**](http://pyyaml.org/wiki/PyYAML) - For reading and writing [**YAML(opens in a new tab)**](https://en.wikipedia.org/wiki/YAML) files.
* [**NumPy(opens in a new tab)**](http://www.numpy.org/) - The fundamental package for scientific computing with Python. It contains among other things a powerful N-dimensional array object and useful linear algebra capabilities.
* [**pandas(opens in a new tab)**](http://pandas.pydata.org/) - A library containing high-performance, data structures and data analysis tools. In particular, pandas provides dataframes!
* [**Matplotlib(opens in a new tab)**](http://matplotlib.org/) - a 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments.
* [**ggplot(opens in a new tab)**](https://pypi.org/project/ggplot/) - Another 2D plotting library, based on R's ggplot2 library.
* [**Pillow(opens in a new tab)**](https://python-pillow.org/) - The Python Imaging Library adds image processing capabilities to your Python interpreter.
* [**pyglet(opens in a new tab)**](http://www.pyglet.org/) - A cross-platform application framework intended for game development.
* [**Pygame(opens in a new tab)**](http://www.pygame.org/) - A set of Python modules designed for writing games.
* [**pytz(opens in a new tab)**](http://pytz.sourceforge.net/) - World Timezone Definitions for Python

**How to Search**

Here are some techniques for effective web searching:

* Try using "Python" or the name of the library you're using as the first word of your query. This tells the search engine to prioritize results that are explicitly related to the tools you're using.
* Writing a good search query can take multiple attempts. If you don't find helpful results on your first attempt, try again.
* Try using keywords found on the pages you found in your initial search to direct the search engine to better resources in the subsequent search.
* Copy and paste error messages to use as search terms. This will lead you to explanations of the error and potential causes. An error message might include references to specific line numbers of code that you wrote. Only include the part of the error message that comes before this in your search.
* If you can't find an answer to your question, ask it yourself! Communities like StackOverflow have etiquette rules you must learn if you want to participate, but don't let this stop you from using these resources.

## Hierarchy of Online Resources

While there are many online resources about programming, not all of the them are created equal. This list of resources is in approximate order of reliability.

1. [**The Python Tutorial(opens in a new tab)**](https://docs.python.org/3/tutorial/) - This section of the official documentation surveys Python's syntax and standard library. It uses examples, and is written using less technical language than the main documentation. Make sure you're reading the Python 3 version of the docs!
2. [**The Python Language and Library References(opens in a new tab)**](https://docs.python.org/3/index.html) - The Language Reference and Library Reference are more technical than the tutorial, but they are the definitive sources of truth. As you become increasingly acquainted with Python you should use these resources more and more.
3. **Third-Party Library Documentation** - Third-party libraries publish their documentation on their own websites, and often times at [**https://readthedocs.org/(opens in a new tab)**](https://readthedocs.org/). You can judge the quality of a third-party library by the quality of its documentation. If the developers haven't found time to write good docs, they probably haven't found the time to polish their library either.
4. **The websites and blogs of prominent experts** - The previous resources are primary sources, meaning that they are documentation from the same people who wrote the code being documented. Primary sources are the most reliable. Secondary sources are also extremely valuable. The difficulty with secondary sources is determining the credibility of the source. The websites of authors like [**Doug Hellmann(opens in a new tab)**](https://doughellmann.com/blog/) and developers like [**Eli Bendersky(opens in a new tab)**](http://eli.thegreenplace.net/) are excellent. The blog of an unknown author might be excellent, or it might be rubbish.
5. [**StackOverflow(opens in a new tab)**](http://stackoverflow.com/) - This question and answer site has a good amount of traffic, so it's likely that someone has asked (and someone has answered) a related question before! However, answers are provided by volunteers and vary in quality. Always understand solutions before putting them into your program. One line answers without any explanation are dubious. This is a good place to find out more about your question or discover alternative search terms.
6. **Bug Trackers** - Sometimes you'll encounter a problem so rare, or so new, that no one has addressed it on StackOverflow. You might find a reference to your error in a bug report on GitHub for instance. These bug reports can be helpful, but you'll probably have to do some original engineering work to solve the problem.
7. **Random Web Forums** - Sometimes your search yields references to forums that haven't been active since 2004, or some similarly ancient time. If these are the only resources that address your problem, you should rethink how you're approaching your solution.