The Python community is amazing. It was one of the first to adopt a code of conduct, first for the [Python Software Foundation](https://www.python.org/psf/conduct/) and then for [PyCon](https://us.pycon.org/2019/about/code-of-conduct/). There is a real commitment to diversity and inclusion: blog posts and conference talks on this theme are frequent, thoughtful, and well-read by Python community members. For everything that is not in the standard library, there is an enormous ecosystem to support the new Pythonista, from exciting packages to text editor plugins specifically for the language. With around 200,000 projects hosted on PyPi (at the time of writing) and growing, there is something for everyone: [data science](https://pypi.org/project/pandas/), [async frameworks](https://pypi.org/project/Twisted/), [web frameworks](https://pypi.org/project/Django/), or just tools to make [remote automation](https://pypi.org/project/paramiko/) easier. Python has a great standard library with many hidden gems I did not know about until I took the time to [walk through the list of all available](https://docs.python.org/3/library/) functions, constants, types, and much more. One of my personal favorites is the itertools module, which is listed under the functional programming modules. In programming language design, a primitive is the simplest available element. The fact that Python is easy to read does not mean it is not a powerful language, and that stems from its use of primitives. My favorite example of what makes Python both easy to use and advanced is its concept of generators. Pseudocode is the concept of writing out programming logic without it following the exact syntax and grammar of a specific language. I have stopped writing much pseudocode since becoming a Python programmer because its actual design meets my needs.