ESCI 386 – Scientific Programming, Analysis and Visualization with Python

Lesson 1 – Introduction to Python

Python is Interpreted, not Compiled

- Python is an *interpreted* programming language.

- The difference between an interpreted language and a compiled language (such as Fortran or C++) is:
  - In a compiled language the entire source code text file is read and then and then converted into an executable code (in machine language) that can be directly executed by the operating system. This executable code is often optimized for the operating system..
  - In an interpreted language the source code text file is read line-by-line and each statement converted into machine language and executed before the next line is read, or the text-file is read completely and converted into an intermediate code that is then further converted into machine language for execution by the operating system.
  - Compiled languages are generally more efficient and faster than interpreted languages, since the conversion to machine language only occurs one time, and then the machine language code can be executed whenever the program needs to be run. In an interpreted language, the interpretation process occurs every time the program is run.
  - Interpreted languages are often more flexible and changes can be easily made to the program at runtime.

About Python

- Compiled languages are generally more efficient and faster than interpreted languages, since the conversion to machine language only occurs one time, and then the machine language code can be executed whenever the program needs to be run.

- In an interpreted language, the interpretation process occurs every time the program is run.

- Interpreted languages are often more flexible and changes can be easily made to the program at runtime.

Python is Dynamically Typed

- Most variables do not need to be declared as real, integer, string, etc.
- The type of a variable is defined by the value assigned to it.
- The type of a variable can change throughout the program execution.
- The opposite of dynamically-typed is *statically-typed*. Fortran and C are examples of statically-typed languages.

Python is Object-oriented

- Although Python is an object-oriented language, we will often use it in a more traditional procedural or functional programming paradigm.

- For those who want to truly learn about what object-oriented means, and how to use it to its fullest, I recommend a book titled *The Object-oriented Thought Process* by Matt Weisfeld.

Python is Open-source and Free

- Unlike with MATLAB or IDL there are no licenses required.
- There are companies such as Enthought that put together high quality Python bundles and support them for a fee. The advantage is that the libraries supported by these bundles work together with few bugs.
- However, the same libraries can be assembled and installed freely from other sources, though there may be some additional headaches sorting out dependencies, etc.

Getting Python for Your Own Computer

- As an academic user you can get a fully functional Enthought Python package for free. See this link for the academic download information: https://www.enthought.com/products/canopy/academic

- Enthought has two different Python products:
  - Canopy: This is the newest product
  - EPD: This is what is on the computers in our classroom.

- Either Canopy or EPD will work just fine for our purposes, so get whichever you want. My examples in class will all be using EPD.

Versions of Python

- Python 3.0 does not maintain backward compatibility with the older versions of Python
  - Thus, code developed for Python 2.X may not work with Python 3.X, and vice-versa.

- So much code and so many libraries have been developed for Python 2.X that most persons in the scientific community still use 2.X.

- Python 2.7, includes many features of Python 3.X, but maintains backward compatibility with older versions.

- We will be using Python 2.7 in this course. I will try to point out which features are from 3.X and which ones are from older versions.

The IDLE

- Python comes with a nice interactive development environment called IDLE.

- You are free to use the command line, IDLE, or any other text editor or interactive development environment you choose.

Documentation and Books

- One book that I use more than any other for reference is:
  - *Python: Essential Reference* (4th ed.) by David M. Beazley. In fact I have two copies: one at work and one at home.
  - Another book that may be helpful is: *Python Pocket Reference* by Mark Lutz.

Online Documentation

- Much of the Python documentation, as well as that for its libraries, is online. Websites that I frequently visit for documentation are:
  - Python v2 documentation page: http://docs.python.org/genindex.html
  - MATPLOTLIB documentation page: http://matplotlib.sourceforge.net
  - Numerical Python (NumPy) documentation page: http://docs.scipy.org/doc/numpy/reference
  - ScientificPython (SciPy) documentation page: http://docs.scipy.org/doc/scipy/reference

Online Documentation

- A web search for “How do I [*whatever you want to do*] in Python?” will often yield fruitful results.

- There is a growing community of Python users in the atmospheric and oceanic sciences community. They have their own webpage/blog called PyAOS, which can be found at http://pyaos.johnny-lin.com.