Computer Setup, Git, and Python

Local/laptop computer Setup

1. We will not be using Jupyter notebooks, so choose a text editor to use for writing code. I use MacVim, but Atom and Sublime Text are other popular options.

2. Select a text editor and prepare it for class.
   - You may want to alias it, for example:
     csh: alias atom ‘open -a atom’
     bash: alias atom=‘open -a atom’

3. Install the VPN Pulse Secure software (see links page).

Python Setup

1. Complete the Python Primer. Key things to note:
   - Create the class conda environment.
   - Always use if __name__==“__main__”!
   - Familiarize yourself with list of best practices and naming conventions.

   Stop here with pre-class preparation.

Python Rec Arrays

Rec arrays

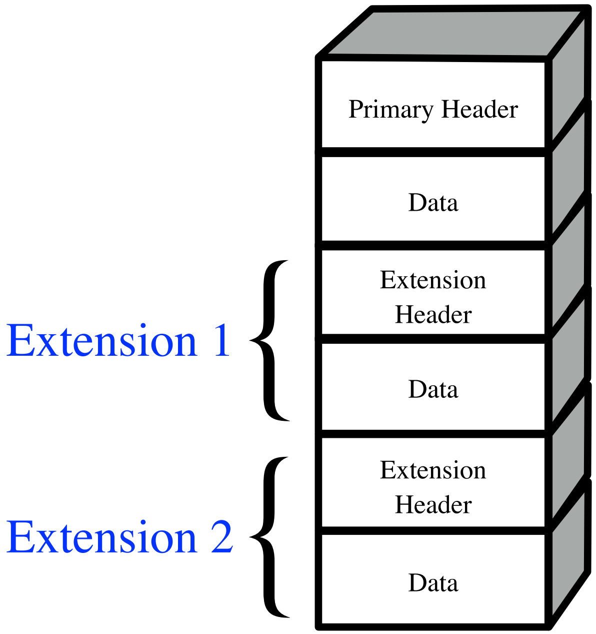
- Rec arrays thought of as single entities that can contain any number of variables (or arrays) by name. Similar to a Python structured array, but with different calling options.
- rec arrays are very useful. They keep track of information (i.e. which column of a file contains the right ascension, which contains the declination)
- They make reading files and sharing files extremely easy (*PyFITS* can read a file with millions of rows in a few seconds)
- rec arrays are single objects. So, for instance, a function can return one entire rec array that contains a complex set of variables and arrays

Rec arrays

- To learn how to make rec arrays and write them out as fits files, consult the documentation for *PyFITS* in as*tropy,* linked from the syllabus, under week 1.

FITS files

- Moving forward, we will start to work with FITS files, which are a binary file format for storing rec arrays
- Although originally developed to transfer digital images FITS (Flexible Image Transport System) files are highly convenient for storing “tagged” information.
- They have “layers” of logical header/data units (HDUs) and are based on the concept of a record, or “rec” array



Schematic of a FITS file

The point of a rec array

- I’ve put a rec array “struc.fits” in my week 1 Git directory and on the website. To read it using *PyFITS*:
  - *from astropy.io import fits*
  - *fx = fits.open(file)*
- To see what the fits file contains try printing fx.info*()*
- To access the data in the binary table, try *objs = fx[1].data* and to get its header *hdr= fx[1].header*
- To use the variables (as you have used other arrays) you can try (after importing *matplotlib.pyplot* as *plt*)
  - *plt.plot(objs[“RA”], objs[“DEC”], “bx”)*
  - *plt.show()*

Python tasks (Remember to commit to Git!!!)

1. Write your code with proper structure (i.e., if __name__==“__main__”).
2. Read in my ‘struc.fits’ file and plot δ vs. α (Declination against Right Ascension) for objects in the file
3. The *extinction* tag in ‘struc.fits’ is a 5-array. To access its first column you can use *objs[“EXTINCTION”][:,0]*
4. On your plot, overplot the (α,δ) of just those objects in ‘struc.fits’ where the first column of extinction is more than 0.22...the *numpy.where* function will be useful

Python tasks (Remember to commit to Git!!!)

5. Generate 3 different sets of 100 random integers (see *numpy.random.randint*)

6. Create a rec array with the tags *ra*, *dec*, and *randomnum* to store this information. Take *ra, dec* from *struc.fits*. Make *randomnum* a 3-array (see *numpy.reshape* if necessary). Write your rec array to a fits file.

7. Experiment with docstrings versus comments:
   - *import polycalc*
   - *print(help(polycalc))*
   - *print(polycalc.get_poly_o3.__doc__)*
   - *Remove if __name__==“__main__” and import.*