UCF Physics: AST 5765/4762: (Advanced) Astronomical Data Analysis

Fall 2019 Homework 3

Due Tuesday 17 September 2019

Work:

Become sufficiently familiar with Python to:

1. Display 2D images and surface plots,
2. Write and document functions and procedures with good coding style.

Resources:

1. Chapters 1 and 2 of Bevington (**DUE** before class Tuesday, 17 September 2019)
2. Greenfield and Jedrzejewski, *Using Python for Interactive Data Analysis*, ch. 1 – 4 (ch. 4 may be scary; skim what you can if you're not comfortable with it). **DUE** before class Tuesday, 17 September 2019)
3. `homework+coding.pdf` handout

   For general NumPy help:

4. Docstrings of Python routines, as needed, note References and See Also sections
5. `Files/python/module_template_with_docstring.py` in WebCourses
6. `np.lookfor("keyword")`: look up functions by word or phrase in docstring
7. Numpy doc wiki: http://docs.scipy.org, especially:
   (a) *NumPy User Guide*, sections "How to find documentation" and "Numpy basics", especially early parts of each subsection, especially "Broadcasting"
   (b) Cookbook
   (c) Additional documentation, especially "Numpy Functions by Category" and "NumPy Example List"

Hand in:

Again, no loops!

Except where the problems ask you to write a function, put the commands to do all of this into the main homework file, labeled with the problem numbers in comments. This should be a clean file of commands to do the problems. It is not a log file. Put the functions in one or more separate files named for their functionality and import them into the Python session. For example, you might have a file named `square.py` containing both `square()` and `squareplot()`. You'd import them and then call `square.square()` and `square.squareplot()`. If you did it this way, the other function asked for would not be appropriate for this module, and you'd

have to put it in a separate file (which is fine). Or, name it something like `hw3funcs.py` and put everything in it. Just make the name somehow categorize the functions in the file.

Keep your main homework file under Git in a **local** repository (**not** on GitHub or another internet archive). This will require you to execute `git` commands in the shell and to store the output. Record these commands and their output in your log for this assignment. (Copying the output there is not required for future assignments.) This may be a messy process as you learn. That's ok. The log is not a work of art.

1. Write a function called `square` that returns the square of its input (which may be a scalar or an array of any dimension or numerical type).

   (a) (2 points) Copy the template in `module_template_with_docstring.py` to create your function file. Edit out anything extraneous.

   (b) (4 points) Create a **local** Git repository and check the function file into it. Record the shell command in your log.

   (c) (4 points) Document the function following the template as well as you can. Be sure to include an example! Check that into Git and record the shell command in your log.

   (d) (4 points) Write the function code. Test and iterate until it works. Check that into Git and record the shell command in your log.

   (e) (4 points) Update the example and docs. Confirm that the function does what this assignment asks, that the docs and example are consistent with that, and that the example says exactly what Python produces. Check that into Git. Print the Git log for the file, recording the shell command and output in your log.

   (f) (2 points) Create an array with the integers from 0-9, square them with your function, and print the result.

2. (20 points) Write a function called `squareplot` that plots the squares of numbers using your function from Problem 1. It should have three positional arguments: the low end of the range, the high end of the range (inclusive!), and the number of points to plot over the range. It should have one optional argument, `savename=False`. The points in your plot should be evenly spaced. The function should call the function in Problem 1 exactly once. The horizontal and vertical axes should be labeled "Input" and "Output", respectively, and the title should be "Square Function". Again, document the function, include an example in the documentation, and check your function into Git as soon as you have a file and several times as you write it, including the final version. If `savename` is not `False`, the function should save the plot image as PNG or PDF (according to the name) in the file name given by `savename`. In the main homework file, give a command line to plot the squares of only the numbers 1, 2.5, 4, 5.5, and 7 using this function, and save to an appropriate name (see `homework+coding.pdf`). Print the Git log for the file and record it in your log.

3. (10 points) List the resources (books, web sites, etc.) that you have used to learn Python. Give full publication and URL reference information. Critique each in a paragraph (how has it helped you, what is the level, what is it good/bad at, etc.). If you have used more than two, pick the best, or favor those not included in the course materials and recommendations. Doing more than two is ok but you don't have to.

4. (10 points) Write a function that implements the linear scaling given in class:

$$A'_{ij} = 255 \times \frac{A_{ij} - \min(A)}{\max(A) - \min(A)} \tag{1}$$

   It should take a floating array and return an array of 8-bit, unsigned integers of the same shape. (See demo for NumPy data types and sizes, converting types, etc. Beware of the deceptively named `bytearray()` function! It does something unrelated.) Print the Git log for the file and record it in your log.

5. (10 points) Read `pix.fits` from the Astro tutorial examples directory (see lecture demo #5 and `Files/python/doc/pydatatut/examples/pix.fits`). Scale it with your routine. Put it on screen USING MATPLOTLIB (not `ds9`). Annotate the image by placing your name completely within the image boundary. Be sure the lower-left corner is 0,0 and that you have axis labels and a title. Produce a PNG file with the result.

6. (10 points) Include a copy of your class log file in your handin. Print the Git log for your main homework file.