Exercise : Due by class time Wednesday, Feb 20, 2013 (no class Feb 18)

`Journal file:` **YourName_ex4.pro** (in your `~/ASTR2600_assignments/assignment4/` directory)
(Create an `assignment4` subdirectory under your `ASTR2600_assignments` directory if necessary.)
Exercises from Chapter 8 and Chapter 9 involve journal files. This is the last assignment with long journal files as exercises. It's probably a good idea to break this up into multiple journal files as described in Assignment 3.

Exercise 8.0 : Experimenting with various plot keywords
Exercise 8.1: Title & `linestyle` keywords, using `xyouts` for legends
Exercise 8.2: A little zooming
Exercise 8.3: 2D plotting
Exercise 8.4: Multiple plots
Exercise 9.0: Playing with color definitions.
Exercise 9.1 : Like Exercise 8.1 but with colors

Whuduzitdo? All Whuduzitdo's from Chapters 8 and 9

Turn in via github.
Journal files of the exercises from Chapters 8 & 9.

Graded Homework: Due by Wednesday, February 20th, 11:59:59 PM
(Readability is 10% of your grade)
All homeworks in this assignment & 8.3 can and should be done by writing *script* files instead of interactively and creating a journal file.

Homework 8.2: Script file name: `YourName_hw8.2.pro`
Postscript file name: `YourName_multiPlanck.ps`
Homework 8.3: Script file name: `YourName_hw8.3.pro`

Homework 8.4 (Not in book): Script file, `YourName_hw8.4.pro` that does the following:
Read in the lif2b data files used in assignment 3. (Read them from the shared `assignment3` directory using the full path. Do not copy them into yours.)
Draw the following plots and save them into JPEG files as described. The plots should all have *appropriate x & y axes labels and an appropriate title* .
If the yaxis label appears clipped off the left edge of the window, use the `xmargin` keyword to remedy this. (Read about it in the book or online documentation.)
a) Plot the flux vs. wavelength. (Déjà vu. You did this in Homework 6.4)
Save an image of this plot as `YourName_fullSpectrum.jpg`
b) The plot has a double peak near 990 angstroms.
Edit your script to use the `xrange` keyword (rather than the `where` function used in Homework 7.1) to plot about a 5-angstrom range roughly centered on this double peak so that we see it clearly. (This is like "zooming in" on that part of the spectrum.)
Save an image of this plot as `YourName_doublePeak.jpg`
c) The plot has a large, thin peak near 1026 angstroms.
Edit the script to use the range keywords to plot about a 1-angstrom range roughly centered on this double peak. This plot should also have a flux range of about `1e-12` flux units so that we can clearly see the structure at this peak. (This means setting a `yrange` as well.)
Save an image of this plot as `YourName_tallPeak.jpg` .

Homework 9.2 (python; Not in book) :
Create a script file `YourName_hw9.2.py` that does the following:
Read a binary 500 `x` 500 array of floats from the file:

`/home/shared/astr2600/data/temperature_iy45.dat` The format is simply an array of floats. There is no first-number "header" with the array size. I'm telling you the size. It's 500 `x` 500. In order to read data from a binary file in python, you need to use `np.fromfile`. In order to figure this function out, look at `help(np.fromfile)`. The first line,
`fromfile(file, dtype='float', count=-1, sep='')`
tells you the order of parameters. `file` should be the filename, and `dtype` is the data type.

HINT: Check to make sure the data look sane! The default `numpy` data type `float` is a `float64`. There is also a `float32` and `float128`. You should figure out which of these is correct. These are `numpy` data types, so they have to be referred to by `np.float32` etc, or you can pass them in with quotes as in the above example (`'float'`), or the letter `f` followed by the number of *bytes*, e.g. `'f4'`, `'f8'`,`'f16'`.

`fromfile` will read in the data as a 1D array. Check this using the `array.shape` method (i.e., if you call your array `x`, print `x.shape`). I've told you the correct shape already, so you'll need to `reshape` the array. Here's an example where I reshape a length-100 array into a 10×10 array:

```
x = np.arange(100)
xr = x.reshape([10,10])
```

Display that array as a monochrome image with `imshow`. Monochrome means one color, so you should use one of the black-and-white colormaps. Save the figure with `savefig("YourName_solarTemp_monochrome.png")`.

Open another python plotting window using the `figure` function. Plot the data as a filled color contour map with 10 contours. The `contourf` command fills in the colors (`contour` just shows lines). The help for `contourf` is confusing, but the last unnamed keyword argument is the number of levels, e.g. `contour(image,number_of_levels)`. Save a figure using a reasonable colormap (for the sun's surface) using savefig: `savefig("YourName_solarTemp_auto.png")`

Now, in a new `figure`, define a range of colors where the largest contour values are the brightest. In `contour`, if you want to specify your own levels, you use a list of levels, e.g. `contourf(image,levels=[0,1,2,3])`. Look at the min and max values of your array to figure out appropriate level values.

Specify these colors in the `contourf` command to match the levels. You can specify colors in any way shown in the lecture (color vector, hexadecimal, name), but you need to use 10 different colors. Use multiple hues, e.g. blue to red to yellow. The 10 colors must correspond to 10 levels that you can specify 'by hand' or with a convenience function (e.g., `linspace`).

The title of the plot should be "Solar Temperature". You don't need to label the axes. Save this contour plot as a JPG file with the name `YourName_solarTemp.jpg`

Run this script with `%run YourName_solarTemp.py`. Make sure it runs flawlessly!

Turn in via github: Your IDL & python script files:
YourName_hw8.2.pro
YourName_hw8.3.pro
YourName_hw8.4.pro (for *one* of the plots)
YourName_hw9.2.py
Your ps, jpg & png files:
YourName_multiPlanck.ps
YourName_fullSpectrum.jpg
YourName_doublePeak.jpg
YourName_tallPeak.jpg
YourName_solarTemp_monochrome.png
YourName_solarTemp_auto.png
YourName_solarTemp.jpg