Final Project Due Friday, May 3 by 11:59 PM or possibly Saturday, May 4th at final time

Worth **200** points in homework category (equivalent of two homeworks).
These projects are going to be challenging, but that's why you have 2+ weeks with multiple class periods in between to complete them. We recommend getting started early, asking plenty of questions, and saving often.

Solar System N-body simulation:
- Create a simulation of our solar system with the Sun and 8 major planets. Incorporate a keyword to choose the duration of the simulation, and have an if statement to create a default if the user doesn't specify. Make sure you get the direction of orbit right! `[20%]`

- Simulation is reasonably accurate and representative of reality, default system stays together for at least 100 years (a counter, either as a title or legend, would be helpful). This will require accurate initial positions (especially velocity) and a sufficiently small timestep. Get polar coordinate positions from `initial_solar_positions.pdf` (you will need to convert to cartesian coordinates). The direction of the velocity is always at a right angle from the (current) heliocentric longitude (angle from the sun). `[15%]`

- Implement at least two of the following options (or your own idea approved by Cameron and Adam), choosable by the user with read or keyword prompts: `[25%] 8%` extra credit per extra implementation up to 24%

- Interactive way for the user to define mass, position and velocity of a given object, as well as gravitational constant G, preferably in terms of ratios. User does not open the file and modify variables. Can use series of read prompts, or keyword arguments.
- Simulate what happens if a rogue stellar object like a brown dwarf, white dwarf or black hole came through the solar system. The user should be able to specify mass, position and velocity, but have well-functioning defaults if they don't. You'll probably want to have a separate initial conditions function.
- Model a real exoplanetary system with at least 5 planets and the star.
  Go to: `http://en.wikipedia.org/wiki/List_of_multiplanetary_systems`, most star pages linked from there have a table with the exoplanet's mass and position (you may have to guess at the angles. Don't worry about eccentricity). You'll probably want to have a separate initial conditions function.
- Incorporate the asteroid belt and/or trojan asteroids into the simulation (this can run by default, doesn't need a keyword). You'll probably want to make use of the random functions instead of explicitly defining the properties of each asteroid.
- Change the angle of perspective from 90 deg (looking down on solar system) to 0 deg (looking at it from flat to elliptical) and back as simulation runs (make it look good)

- Understandable, visually pleasing output. Use plot symbol keyword settings or `draw_circle` function to make Sun and planets easily recognized with size, shape and color. (Rough size scale, doesn't have to be realistic). Animation is not required, but probably useful. `[20%]`

- Simulation has an informative, understandable and managable user interface. Code is well formatted and commented so it could be easily understood by your peers. `[20%]`

Image Analysis/Photometry:
- Create several procedures and functions (called by one parent procedure) that take an image, finds the position, brightness/magnitude and radius of all stars above a brightness threshhold, and outputs all that information and other meaningful data to the terminal. Have at least 1 plot window with meaningful data in addition to terminal output. You will find chapter 20 section 6 (pages 33-47) extremely helpful. You will definitely want to use recursion to accomplish this task. `[45%]`

- Implement at least two of the following options (or your own idea approved by Cameron and Adam), choosable by the user with read or keyword prompts: `[25%] 8%` extra credit per extra implementation up to 24%

- Keywords or prompts to sort output from min-¿max and vis-versa for all categories (position, brightness, radius, etc), and also save data into text files for later use.
- Make a color-magnitude plot from the two provided image files.
- Merge two or more images from different wavelengths and specify levels to make a new image.
- Adjust the parameters of the chosen image, such as min/max levels, hue, saturation, value, and save it as a jpeg (make a few cool alterations).
- Fit a profile to the star data you collected
- Statistical analysis of star data. Mean and standard deviation of distance, brightness, size.
- Given multiple images of the same piece of sky, analyze each image and compare images to find comets, asteroids or supernovae.

- Understandable, easy to parse, visually pleasing output. `[10%]`

- Simulation has an informative, understandable and managable user interface. Code is well formatted and commented so it could be easily-understood by your peers. `[20%]`