Final Project

The end goal of this course is that you will be able to build your own software to solve your own problems. To work on this, you will write a program to solve a problem or answer a question from your own research.

I would rather you select an interesting problem than worry about your ability to code it. We'll help you scope the problem with a project proposal. (The earlier you turn this in, the better we can help you succeed). Further, the last two weeks of class will be devoted working on your programs in class. If you put in significant effort, but it becomes clear your program won't work, we will not dock you for this. Your program has to run and do something on the way to your final goal by the end of class; it does not have to achieve the stated goal.

Constraints

- The problem should be real.
- At the end of the day, the program you write should be useful.
- The approach should be feasible. This means it is not only mathematically possible, but that you can make significant progress on the project in a few weeks of coding.
- The problem should **not** be one you could solve better and faster using an existing software package. If you say you're going to write a transcriptome assembler, I'm going to point you to Trinity (unless you have reason to believe you can do better!)
- If you need data to develop the program (e.g. experimental images, sequencing runs, etc.) these should already have been collected.

Resources

- **Definitely** use software packages that help you achieve your goals. If you're analyzing colonies on a plate, use libraries like *PIL* and *skimage* --don't write your own image parser!
- Talk to your lab mates and PI about interesting problems and what software might be useful.
- Talk to Zach and me about strategies. For example, if you want to classify stuff, machine learning might be the way to go. We're not going to discuss it in detail in class, but Zach and I can point you to resources that will help you get started.

Project Ideas:

- Counting/classifying something in an automated fashion
- Fitting complicated model(s) to data
- Squeezing every last drop of information out of a high-throughput experiment
- Processing spectra
- Pulling shapes out images
- Tracking a particle in a video
- Extract some cool feature out of an MD simulation
- Machine learning to classify stuff

What you turn in:

Prospectus:

Due on (or, preferably, way before) Friday, May 12th.

It should have the following (bullet points are fine):

1. Identify a problem from your research that you will solve computationally.
2. Describe the strategy your program will use as a flow chart. (Can be hand drawn, as long as it's legible).
3. Describe the data you will use to develop your program (for example: images from a control and treatment condition).
4. Describe how you will verify the program is actually doing what you want it to do.
5. Describe existing software that you can use to help solve the problem.

The point of the prospectus is to help you think through the project, not to create busy work. I will review the proposal and discuss it with you one-on-one during one of the in-class workshops. The earlier you bring it in, the earlier you can get rolling on the final project. If there are issues that should be addressed, I might ask for a revised version.

Final project

Due on Monday, June 12th.

You will turn in EITHER a url for a github repo OR a zipped directory containing your project. The repo/directory should have the follwing structure:

```
README.md
python-y bits
demo.ipynb
data_files/
    data_file_for_demo
    data_file_for_demo
    ...
```

Assessment critera:

- Does the README.md file give sufficient information to know what the software does and how to use it?
- Does the software actually solve a meaningful problem or analyze data in an interesting way?
- Does `demo.ipynb` run without error?
- Does `demo.ipynb` effectively demonstrate the capabilities of the software?
- Is your python code well-documented and readable?