Lab 3 Documentation and Community Development

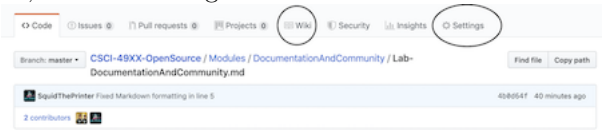
This Lab has two parts Documentation and Community

We talked about community and how people cooperate in order to contribute to a project. In this lab, we will explore this further using automatic tools, software and manually.

> This lab will be done in groups of 4-5. Each will be assigned a table number
>
> All the results should be documented in a lab report using your lab repository. Make sure you push it to your Github account before submitting to Submitty.

Part 1 - Documentation

1. Go to your lab repository and look for the wiki link at the top tab bar. If it isn't there, click on settings and check the wiki box near the bottom of the page.



2. Each person individually, on their wiki page (using Markdown or reStructuredText) writes a brief description of some area of interest that they might want to explore in their project - It should include:
    1. The problem area that your software is researching or a technology or technologies that you might want to explore.
    2. Existing Software that addresses similar problems and how yours will be different or better; or examples of projects in your area of interest. Check out LinksforPossibleProjects.md in the resources area to get ideas of both.
    3. Potential technologies you will use in your project; or any ideas about how where your technology might fit.
    4. The end product or goal.

    Note that all of this is preliminary. We don't need "good" answers, but I want you to be thinking about your project and this should be a first iteration of it. All of this together should take no more than one or two paragraphs. Here is a perfectly acceptable blurb.

    > *I am not sure exactly what project I would like to do, but I am interested in medical image processing. I have been looking around and found two open source resources, itk and open-access-medical-image-repositories. The first is an open source toolkit for working with medical images and the second is a list of open*

data archives. This seems to be an active area of research with projects like 3D Slicer being very popular. However, in looking through the site, 3D Slicer seems to be very difficult to learn.

I think I want to do something in this area that is maybe less powerful, but easier to use. I could consider using web technologies, which would required a web stack; or maybe just look at something that will help patients look at and understand their own data. What I would be focussing on is avoiding the complications of systems for research to make it more accessible to the patients themselves.

3. Use latex http://www.artofproblemsolving.com/texer to generate the formulae depicted in latex_formulae.png

4. Use latex to display a Hadamard Matrix of size 4 (equation 5). You can use just 1 and -1; although, if you want to make a matrix using the white and black boxes you are welcome to.

5. Now let's try something fun. The course notes for this class were developed over multiple years using different (generally open) technologies. In particular, the notes for many of the modules including Module 01.Introduction use a combination of Python (sphinx) and latex.

    1. Start up your Linux installation from Lab 1 and clone the class repository
    2. The build system for the notes uses the python module `sphinx`. Use `pip install sphinx` to install it.
    3. The output we are interested in also uses `latex`. Verify you have latex installed. If not, use `apt install texlive-latex-extra` to install it. You will also want to install `latexmk` using the same procedure.
    4. Change to the History directory 01.Introduction/History
    5. Type `make help`. You should see a list of targets that are available as the output of our build system. For the class presentation, I use `make slides` which generates an `html` file of slides that can be displayed in a browser such as `firefox` or `chrome`. You should use `make latexpdf` instead. It will probably fail with missing packages. Install any missing packages using `pip` and try again. Repeat until you successfully generate a pdf of the course notes. Take a screenshot of the successful make run and include it in your lab report. Can you find the file you generated? Where is it found on disk? Put the **full directory path** to the generated file in your lab report.

    General notes for part 1 of the lab:

    - If you get a warning stating that your installation is not part of the path, run: `export PATH=$PATH:/home/[YOUR_USERNAME]/.local/bin`
    - For the final step when running `make latexpdf`, you may get an error that reads: `LaTeX Error: File 'cmap.sty' not found`. To

fix this, run `sudo apt install texlive-latex-extra`. If you get an error after installing saying that it was unable to fetch some archives, follow the directions and run `sudo apt-get update`.
- When in doubt, restart Ubuntu.

Part 2 - Community

1. Break up into "tables" of 4-5 students.
2. Randomly (or purposefully) choose some projects from https://observatory.rcos.io/projects/past and clone them locally. All people at the table need to record the information for all projects, but each person can do one project and share the results with the group.

   Adjust the numbers to the number of people at your table, but choose at least 4 projects to review. If you can't clone a specific project simply choose another.
3. For each of your projects, look up by hand and record in your `lab report`:
   - the number of contributors
   - number of lines of code
   - the first commit
   - the latest commit
   - the current branches

   To get the lines of a project, try something like `git ls-files -z | xargs -0 wc -l` in the cloned project directory

   Note: if the project you are assigned has no commits to its repository, pick a different project.
4. Gitstats
   1. Install - This can be done in Windows or OSX, but it will be easiest on Linux. I got it working on WSL, but it required installing gnuplot and (possibly) installing an X11 windows server. **Here are some instructions**.
      - Clone the project `https://github.com/hoxu/gitstats` locally `cd gitstats`, and run `make install`
      - Homebrew / Linuxbrew users can use `brew install --HEAD homebrew/head-only/gitstats`
      - If you have issues, an alternate version, specific to Python 3 can be found in RCOS. Use either the `futurized_gitstats` or `python3` branches.
      - Gitstats requires gnuplot. To install, run `sudo apt-get install gnuplot-x11` for Ubuntu (or the appropiate command for your platform, see https://sourceforge.net/projects/gnuplot/).
   2. Running

- From the command line, run `gitstats <path to project1 git repo> <output path>` inside the cloned project directory
  - If you get an error with `gnuplot` not being able to generate a .png file, you can safely ignore it. Your output won't be quite as nice, but you will be able to answer all of the questions.
  - If you really want the graphs, edit the file `gitstats` and change all occurences of `.png` to `.svg` and remove all occurences of `transparent`. Then rerun.
- You can see the output in a browser by typing `file:///<output path>/index.html` in the address bar (use `pwd` from the command line to get the current path)
- You may also be able to open it from the command line using `xdg-open <output path>/index.html`, `sensible-browser <output path>/index.html`, or `sensible-browser <output path>/index.html`
- Repeat this for each of the five projects.

3. Compare to your results from the github exercise to these results and comment on your findings in your `lab report`.

   If you are curious, please read and try to understand the python code for gitstats. (It's even better if you suggest some improvements!)

5. Streaming Contribution Visualizations

   - Read the webpage for gource.
   - Download gource using `sudo apt-get install gource` (or `brew install gource`) or install it via your favorite binary installer.
   - Go to each of the five cloned repository directories and execute the command `gource`
   - You will get a streaming video of the activity in that project.
   - Now, create a video of these projects.
   - Install ffmpeg using `sudo apt-get install ffmpeg` or install avconv using `sudo apt-get install avconv` (or `brew install <package name>`).
   - Execute the following two commands from each of the cloned repositories:

```
gource -1280x720 -o gource.ppm --time-scale 3

ffmpeg -y -r 60 -f image2pipe -vcodec ppm -i gource.ppm  -vcodec mpeg4 -b:v 3000k -s hd
```

or

```
gource -1280x720 -o gource.ppm --time-scale 3

avconv -y -r 60 -f image2pipe -vcodec ppm -i gource.ppm  -vcodec mpeg4 -b:v 3000k -s hd
```

or for a more fun gource, generate the ppm file with:

```
gource -1280x720 -o gource.ppm --auto-skip-seconds 1 --max-files 0 --time-scale 3 --cam
```

- (Optional) If you would like, upload your video to Youtube and add a link in your `lab report`.

    Example youtube videos - Observatory (old) and CSCI 2961-01 Intro to Open Source

Make sure to include screenshots and comments in your lab report.