

Lab 1: Submitting problem set solutions

Ahmed Eldeeb

2023-08-21

Table of contents

Submitting problem set solutions (09/01/23)	1
Discussion on standard modern computer architecture	1
Discussion on Unix Architecture	1
Quick Intro to git and GitHub	2
Hands-on Lab Instructions	2
Steps to perform today:	2
Chunk options	3

Submitting problem set solutions (09/01/23)

By now you should already have access to the following 5 basic tools:

1. [Unix shell](#)
2. [Git](#)
3. [Quarto](#)
4. [Python](#)
5. A text editor of your choice

Today we will use all these tools together to submit a solution for Problem Set 0 (not a real problem set) to make sure you know how to submit solutions to upcoming (real) problem sets.

Here is a selection of some basic reference tutorials and documentation for [unix](#), [bash](#) and [unix commands](#), [git & GitHub](#), [quarto](#), [python](#) and [VS Code](#)

Discussion on standard modern computer architecture

[Figure](#) from “[Designing Embedded Hardware](#)”

Whiteboard and group discussion

Discussion on [Unix Architecture](#)

Whiteboard and group discussion.

Some books to [learn more](#).

Quick Intro to [git](#) and [GitHub](#)

1. Making changes 1.1 Editing and saving files 1.2 Staging changes 1.3 Committing changes locally 1.4 Pushing changes to remote repository
2. Undoing changes: 2.1 Local changes 2.2 Local staged changes 2.3 Local committed changes 2.4 Pushed changes
3. Merging divergent versions
4. Working with branches
5. GUI
6. Getting help

Hands-on Lab Instructions

[Problem Set submission](#) instructions:

1. Open the qmd file in any editor you like (e.g., Emacs, Sublime, ...). Use `quarto preview FILE` to show your rendered document live as you edit and save changes. You can put the preview window side by side with your editor, and the preview document should [automatically render](#) as you save your qmd file.
2. [Use VS Code](#) with the following extensions: Python, Quarto, and Jupyter Notebooks. This allows you to execute and preview chunks (and whole document) inside VS Code. This is currently deeb's favorite path due to how well it integrated with the Python debugger.
3. [Use RStudio](#) (yes, RStudio), which can manage Python code and will display chunk output in the same way it does with R chunks. This path seems to work quite well and is recommended if you are already familiar with RStudio.

Steps to perform today:

0. Clone your github repository to your development environment
1. Create a subdirectory in your github repository with the name ps0
2. In that subdirectory, create a quarto document (ps0.qmd) that has some simple code that creates a simple plot (you can follow this example/tutorial [here](#))
3. Use the quarto command line to render it into a pdf document (`quarto render FILE -to pdf`)
4. Commit the changes to your repository (`git add FILES`; `git commit -m MESSAGE`; `git push`)
5. Add another section to your quarto document (use your imagination), then preview and commit the changes
6. Use the quarto command line to render the updated document into a pdf document
7. Add the pdf document to the repository as well
8. Make sure that you can log into [gradescope](#) and upload a pdf document
9. [optional] Undo your last set of changes and regenerate the pdf file

We will also take today's lab as an opportunity to get familiar with the basic use of all the 5 basic tools listed above. For git and quarto, very basic knowledge should be sufficient, but for unix commands and python, the more you learn the more effective you will be at solving the problem sets (and at any computational task you take on after that).

Chunk options

Under development ...