Setting up your computer for scientific computing

I. DIVERSITY OF OPERATING SYSTEMS

Unfortunately, the writing of computer code and the running of programs is dependent on your operating system. There are two basic variants: Unix-based and Windows-based. Unix-based operating systems include all Linux variants (Fedora, Ubuntu, Red Hat, etc.) and Mac OSX. Windows is obviously Windows-based.

Unix-based operating systems are well-suited for scientific computing. Many tutorials and instructions (including the textbook) are written with Unix-based operating systems in mind. Mostly everyone working in scientific computing either uses a Unix-based operating system or knows how they work.

The crucial feature of the Unix-based operating system is its terminal, sometimes called a shell.

If you have a Mac or Unix operating system you have completed this assignment. If you are a Windows user, please continue to the next section.

II. FOR WINDOWS USERS ONLY

If you are a Windows user, you will need access to the terminal for accessing command-line programs. Most of the tools we will be using have significantly better support for Unix operating systems. This section will describe what you can do to make your environment more Unix-ish.

Recent versions of Windows come with the Windows Command Prompt, while this is closer to Unix my guess is its still far from what we will need. Instead, you should consider one of the following options

- (Recommended) Install the Anaconda package manager for Windows. This will give you access to a variety of programs including a shell.
    1. Visit https://www.anaconda.com/download and install the 64-bit version for Python 3 (any laptop manufactured since 2008 will run a 64-bit operating system)
    2. Once Anaconda is installed, launch the Anaconda Navigator
    3. Click on "Environments"
    4. Search for the package "git" and install it. You now have git and GitBash. Open GitBash... this is your terminal. **Warning**: Although this will give you some of the functionality of a Unix terminal, there will still be some differences as compared to Unix.
    5. **Online help video**: A student kindly posted a youtube video on how to do these steps. These steps are known to work as of Spring 2022: https://www.youtube.com/watch?v=-JJWJXrLF0A
- (Avoid) You may read on the internet that PuTTY is a good terminal for Windows. It is not! Its useful for some purposes, but not this class.
- (Advanced) Consider installing a dual boot linux-based operating system alongside Windows (I would recommend Ubuntu). **Backup your files first! You can accidentally delete all of your files.**

III. APPENDIX: INSTALLING GITBASH

If you cannot install GitBash with Anaconda, please follow the steps below.

Note: A student kindly posted a youtube video on how to do these steps. This is much better than the notes, which don't cover special topics like tinkering with the system path. These steps are known to work as of Spring 2022. Please let me know if you run into any issues.

https://www.youtube.com/watch?v=-JJWJXrLF0A

1. Google "git windows" and open the page (should be https://git-scm.com/download/win)
2. The installer exe file should automatically start downloading. If not, download it yourself.
3. Installation is easy; follow most of the default choices. Some suggested settings to override include the text editor (choose notepad++), and choose Unix commit format.

4. You should have a new program, GitBash, to open up.

Note: If you install GitBash this way, there is a chance GitBash will not recognize the Python command (see lab 1). If that's the case, either you can run your Python program using Anaconda Prompt or modify your Windows system (technically, you need to add Python as a system Path Variable) so GitBash automatically recognizes Python. This is yet another set of steps, which assumes you are running Windows 10 (please consult me if any of the steps sound intimidating...):

1. From the start button you should be able to search for "edit the system environment variables"
2. A system Properties box will pop up, click the Environment Variables in the bottom right corner
3. In the bottom section of the new dialog box there should be system variables
4. Find and select the one called Path and hit edit – DO NOT TOUCH ANY OTHERS
5. Now you need to find where Python is installed on your computer (probably /c/Users/YOURNAME/Anaconda3/)
6. Once you have the location of the folder containing python.exe, add this to the Path variable as a NEW entry, and hit OK
7. Relaunch Git Bash and it should work