**Console Foundations**

* Console- text interface that allows you to communicate with the computer
  + Type in the command and the computer prints out a response
* List command: ls
  + Long list: ls –l
    - Gives you more information on the list
  + List all: ls – a
    - Shows the hidden files as well
  + List a certain file: ls /(folder name)
* clear: clears the command screen
  + Can still scroll up to see previous commands
* Find out in what directory you’re in: pwd
  + Changing the directory: cd (directory name)
  + ~ refers to the home directory
* Changing directories:
  + Moving up a directory: cd ..
  + Can cd into an absolute path: cd /home/user
    - Regardless of where you are
* Reading a file: less (file name)
  + q: quit and takes you back to the command line
  + interactive
  + cat (file name): prints out the contents of the file without being interactive
    - Can view multiple files: cat (file name) (file name)
* Editing a file: nano (file name)
* Moving a file: mv
  + mv (old file name) (new file name)
    - Renames the file within a directory
  + mv (file name) (directory)
    - Moving to another directory without renaming it
  + mv (file name) (directory)/(new file name)
    - Moving to a different directory and renaming it
  + Can rename a directory or move a directory
* Copying a file: cp
  + cp (file name) (new file name)
    - Copies a file name and names the new copy (new file name)
  + cp –r (directory name) (new directory name)
    - copies a directory, their files, and any sub-directories
* Removing a file: rm
  + rm (file name)
  + rm –r (directory name)
    - Removes a directory and anything within it
* Creating a directory: mkdir (directory name)
  + mkdir (directory)/(new directory name)
    - Creates a nested directory
  + mkdir –p (directory name)/(new directory name)/(new directory name)
    - Creates new nested directories
* Users:
  + Creating users: sudo adduser (user name)
  + Switching users: su (user name)
    - Switching to the previous user: exit
  + whoami tells you which user you’re currently in
* Permissions:
  + Read, write, execute (r w x)
    - r = 4, w = 2, x - 1
    - Can be defined on the user, group, or other—everyone else (u g o)
  + chmod will let you modify the permissions of a directory
    - chmod o+w (adding the writing permission by the other)
    - chmod o-w (removing the writing permission by the other)
    - Can also sum the corresponding numbers for the group:
    - rwx--x-w- = 712
* Changing the owner: sudo chown (new owner) (file name)
  + Changes the user
  + Changing the user and the group: sudo chown (new user):(new group) (file name)
* sudo: allows you to run a command with the privileges of the root user
  + Only works if your user is allowed to use it
* !! represents the most recently run command
* Processes- an instance of a program
  + Ex. nano executes a program
  + top: shows running processes
  + ps: lists out the processes
    - ps aux: view a full list of all processes
  + signal- a message sent to a process by the operating system
    - TERM (terminate)- requests that the process terminates after any cleanup
* Job- a process that belongs to you and your session
  + Ctrl + z: pauses your job
  + fg: brings the most recently stopped program to the foreground
    - resumes the last stopped process we have
    - fg (job number) opens up your stopped program
  + jobs: prints the list of jobs in the session
  + (job name) &: runs the job in the background
* Killing processes:
  + Crtl + c: sends a terminate signal
  + kill (process id): will kill the process
    - May need to use sudo
    - Cannot undo. Make sure the process you are killing is what you truly intended
* Environment variables- store a value associated with a name
  + $(environment variable) will allow you to use the variable
  + changing a variable: (variable name)=”(value)”
* bash: will open a new session of bash while still running the previous session in the background
  + exit: exits that session of bash and takes you to the previous session
  + if you create a variable in one session of bash, it will not be passed down to a new session
    - can be exported with: export (variable name)=”(value)”
      * will be available to the program rather than just the session of bash
* which (program name): tells you where the program is located
* find (directory name) –name “(file name)”: used to locate files based on the file name
  + find / -name “(file name)”: searches the entire system for the file
  + can put in multiple directory names to search
    - find (1st directory name) (2nd directory name) –name”(file name)”
* grep “(pattern)” (file name): search inside a file for a pattern
  + grep –n “(pattern)” (file name): tells you what line the pattern appears
  + grep –i “(pattern)” (file name): makes the search case insensitive
  + grep –v “(pattern)” (file name): Will show you what lines do NOT contain the pattern
  + You can redirect the standard input and standard output
    - Changing the standard input (using a file instead of the keyboard)
      * grep “(pattern)” < (file name)
    - changing the standard output (writing the output to a file instead of the console)
      * grep “(pattern)” (file to search) > (file where you want it to be written)
        + Overrides the contents of the file
      * grep “(pattern)” (file to search) >> (file where you want it to be written)
        + Appends the output to the end of the file
* Redirecting error messages to another file: (command) 2> (file where you want it to be written)
  + Only the outputs that do not contain error message will be printed to the console
* /dev/null: a special file that will delete anything written into it
* The pipe command | - takes the standard output of the process to the left of it and makes it the standard input of the process to the right of it
* sort: sorts the lines of standard input and sends it to the standard output
* Installing Software manually:
  + (install tools) sudo apt-get update
    - update the package manager on your computer
  + sudo apt-get install (package name)
    - installs the package
  + Get the source code of the software you want
  + curl –O (source code)
    - downloads the software
  + tar –xvf (source code)
    - unzips the file
  + installing a program from source:
    - ./configure
      * runs the configure program in the current directory
      * looks through the system and prepares more configuration files that allows us to build the program
      * creates a Makefile
  + execute the Makefile: make
  + sudo make install: installs the software
* Installing Software with a package manager:
  + sudo apt-get install (package)
* sudo apt-get upgrade: updates the software you have installed to the latest version
* Uninstalling a package: sudo apt-get remove (package)

**GitHub Basics**

* Repository (repo) – works as a folder for your project code
  + Contains documentation and history of your project
* Push (git push) – pushing your project from your computer onto your github repo
  + git status
    - tells you whether there are any changes to your project
  + git add .
    - adds all the unsaved changes
  + git commit –m “(commit message)”
    - Commits your changes
    - Your commit message should be a brief description of what you’ve changed. Make sure it is meaningful!
  + git remote add origin (repo)
    - Connect your project to your repo
    - Can get the command line from the github website
  + git push origin master
    - Sends your commit to the master branch
    - git push origin (branch name): push to another branch
* Branching- allows you to make changes to your project without changing the original version
  + You can merge your branch with the master branch
  + Adding a branch:
    - git branch (branch name)
  + git checkout (branch name) will switch you to a different branch
    - git checkout –b (branch name): will create the branch and switch you to it
* Pull requests- used to post changes to a project file. Many times used for code review
  + Merging the pull request will add your changes to the overall project
  + Make sure your titles and your comments describe what changes you made
  + Reviewing pull requests:
    - Clone the branch
    - cd into the branch and make any necessary changes
    - commit and push your changes up
* Merge conflict- caused when changes are made to the same line of code
  + When you try to push your changes but someone else has already changed that portion of the code
  + Make sure you git pull before making changes to avoid them!
* GitHub pages
  + Allows you to show off you projects and documentation
* Open-Source Repos:
  + Owner can give push and pull access to collaborators and maintainers
  + Maintainer-someone the owner trusts and can push and pull code
  + Contributors normally can only pull the code
  + Community members- uses the code but do not contribute into the code

**Git Basics**

* Benefits of version control system (VCS)
  + Helps you control all the different versions of your project
  + Commit your changes to your repository to keep track of your changes
* git init (repo name): creates a new repo
* git add (file name): adds a file to your repo
* committing a change: git commit –a –m “(commit message)”
* git status: will tell you if there are any changes that have been made
* git log: will show you a log of all of your commits
  + commit identifier: unique identifier for that specific commit
  + commit author: the person who committed the change
  + commit message: the commit message
* git checkout (commit identifier): allows you to see a previous version of your code
  + don’t need the whole commit identifier. Only need about the first 5 characters
* git diff (commit identifier of the first commit) (commit identifier of the second commit): will show you the differences between the two commits
* Branching- like alternate realities of your project, allowing you to make changes and then later merge them to the head branch if you want
  + Master branch- the “trunk” or the main branch of your project.
  + git branch (branch name): creates a new branch
  + git checkout (branch name) will switch you into that branch
  + git checkout –b (branch name): will create the branch you checkout if it doesn’t exist
    - branches off of the current branch you’re in
  + git branch: will list all your branches and will lable the one you’re in with a\*
  + git branch –D (branch name): will delete the branch
* Merging: combines changes and commits into one cohesive timeline
  + git merge (branch name): will merge the branch with the current branch
  + merge conflicts: in resolving it, you want to edit the file with the conflict and choose which version you want to keep or keep both
    - Make sure to delete the conflict markers
    - Then add and commit your change
* Pushing and pulling changes from remote repos allows you to merge history between different copies of the same repo
* Remote repository- a repo that is on another computer
  + git clone (repo name) (optional name for repo): clones the repo onto your computer
  + git push (remote) (branch): will send all your changes from your local files to the remote repo
    - have to add and commit your changes first
  + git pull (what remote you want to specify from) (what branch): will pull any changes from the remote repo onto your local files
    - Pull before you push!
* Feature branch- making any kind of substantial changes to your project on another branch called the feature branch.