1 Introduction to Bash Script:

Bash Script (also called Bash Shell Script) is a Linux-terminal-based language used to submit jobs. "submit jobs" means you are asking the machine to execute the set of instructions you provided it, in the form of Bash Script. These are powerful tools to handle files and folders. To illustrate it, for example, you have 1000 folders on your computer and you realized that 990 of them are duplicate copies of the same folder. We can of course manually check the contents of each folder, determine which folders are duplicates, and finally delete the ones that are duplicates. However, it is a painful and a lot of time-consuming tasks requiring hours if not a day. That same task can be done using a bash script within a few seconds. Before learning the bash script we need to know that it is designed to execute at the Linux terminal. Therefore, it naturally understands the Linux terminal-based commands. These commands are in-built and perform a specific task. Usually, the bash script sequentially utilizes those commands. I have attached a link here Click Me which introduces the important and most frequently used Linux commands.

Getting started
Copy every files from GitHub Click Me to a folder in your Linux machine. Go to that folder using your terminal. In the terminal type pwd and press enter (Does it display the path of the folder you were looking for??). If so it's perfect. Let us know otherwise.

Running the first script *[start.sh]*
In your terminal type as indicated below followed by pressing enter symbol

cat start.sh

You should be able to see the contents of the file start.sh. Please go through the lines. Anything after the hash symbol (#) in each line is ignored by bash. We are using it to clarify the meaning of what we are trying to do on that particular line. Read it carefully. After you are done reading them, execute the file by type as indicated below followed by pressing enter symbol

bash start.sh

I hope you can successfully run your first script. You have learned how to print and assign different variables. Please refer to an online resource for more (Google).

Second script *[if.sh]*
This file gives a simple demonstration of how to use if conditions in the bash script. Take a look at it. Follow the comments added to the lines for more information. Finally, Run the following

bash if.sh

There are a lot of ways of handling files and folders using if conditions in bash. Please refer to the online sources for more information.

Third script *[writing_functions.sh]*
Similar to FORTRAN, Python, and C/C++, we can define a function in bash script. Please take a look at the file writing_functions.sh, read the commented lines for more clarity. I have written five functions there which can be very useful in the future. There is not an easy way to do algebra with floating numbers in bash. The functions I have attached there could be helpful in that regard. Feel free to run it and modify it, depending upon your intended use.

bash writing_functions.sh

Fourth script *[use_python_with_bash.sh]*
One of the coolest features of bash script is that you can control and run other programs like FORTRAN/C/C++/Python etc. Please take a look at the script use_python_with_bash.sh and check for commented lines for more info. It calls and runs a python script named add.py to add two arguments. This is especially useful if you are an expert at other programming languages and want to use bash for just handling the files and folders. This is slightly slower but can be effective for numerical calculations where bash alone is difficult to deal with.

bash use_python_with_bash.sh

Fifth script *[multiple_job_submission.sh]*
The other cool feature about bash scripting is that it allows multiple job submissions to cluster or even the terminal. So, you can run multiple jobs in parallel. Take a look at the file multiple_job_submission.sh and check commented lines for more description. This file creates a folder, prepares or copies necessary files into it, and executes the desired program. Then creates another folder and do the same. At the final execution, it has six folders with calculations running on each. For the executable of quantum chemical calculations in a cluster, you need to check for a sample script designed to run on that cluster. With little practice, you can easily modify them in your script. Also, the name and nature of input files differ with each quantum chemical packages. Like the input files for VASP are different than those for Quantum Expresso or QChem. Therefore check for sample scripts files and the requirements of the packages before executing the following

bash multiple_job_submission.sh

Thank you for your interest!!
Please write to me at SANTOSHA@mailbox.sc.edu if you have more questions/comments/suggestions.