Introduction to Linux Shell

If you are using any major operating system you are indirectly interacting to **shell**. If you are running Ubuntu, Linux Mint, or any other Linux distribution or even MacOs, you are interacting to shell every time you use terminal.

What is Kernel

The kernel is a computer program that is the core of a computer's operating system, with complete control over everything in the system. It manages following resources of the Linux/MacOs system

- File management
- Process management
- I/O management
- Memory management
- Device management etc.

  Complete Linux System = Kernel + GNU system utilities and libraries + other managment scripts + installation scripts

Shell

A shell is special user program which provide an interface to user to use operating system services. Shell accept human readable commands from user and convert them into something which kernel can understand. It is a command language interpreter that execute commands read from input devices such as keyboards or from files. The shell gets started when the user logs in or start the terminal. SSH creates a channel for running a shell on a remote computer, in the manner of the Unix rsh command, but with end-to-end encryption between the local and remote computer.

Command Line Shell

Shell can be accessed by user using a command line interface. A special program called **Terminal** in linux/macOS or **Command Prompt** in Windows OS is provided to type in the human readable commands such as "cat", "ls" etc. and then it is being execute. The result is then displayed on the terminal to the user. Let us walk through an example

```
[alromero@srih0001 ~]$ cd BoronCarbide
```

```
[alromero@srih0001 BoronCarbide]$ pwd
/users/alromero/BoronCarbide
```

```
[alromero@srih0001 BoronCarbide]$ ls
12atoms_B3C  14atom_B5C2  14atoms_B4C3
4atoms_BC3   7atoms_B2C5  7atoms_B5C2
14atom_B1C6  14atom_B6C1  15atoms_B13C2_Done
6atoms_BC5   7atoms_B3C4  7atoms_B6C1
14atom_B2C5  14atoms_B3C4 16atoms_B3C
7atoms_B1C6  7atoms_B4C3
```

```
[alromero@srih0001 BoronCarbide]$ ls -la

total 2496

drwxrwxr-x 27 alromero alromero 32768 Aug 22 08:07 .
drwxr-xr-x 218 alromero aromero  32768 Aug 21 16:11 ..
drwxrwxr-x  3 alromero alromero 131072 Aug 20 20:18 12atoms_B3C
drwxrwxr-x  2 alromero alromero 32768 Aug 21 10:09 14atom_B1C6
drwxrwxr-x 44 alromero alromero 32768 Aug 22 17:38 14atom_B2C5
drwxrwxr-x 30 alromero alromero 32768 Aug 22 17:38 14atom_B5C2
drwxrwxr-x  8 alromero alromero 32768 Aug 22 17:38 14atom_B6C1
drwxrwxr-x 549 alromero alromero 65536 Aug 22 08:08 14atoms_B3C4
drwxrwxr-x 163 alromero alromero 32768 Aug 22 17:38 14atoms_B4C3
drwxrwxr-x  3 alromero alromero 32768 Aug 19 19:40 15atoms_B13C2
....
```

You can also find out where you in the directory tree

```
[alromero@srih0001 BoronCarbide]$ pwd
/users/alromero/BoronCarbide
```

You can create the wornership of a file

```
[alromero@srih0001 BoronCarbide]$ chmod 755 scriptname
```

This uses the octal numeric modes (as described in `man chmod`) to change the permissions for the file “scriptname” and give the user rwx permissions and all other users `r-x` permissions (they need Read permissions for their interpreters to interpret the script and execute permissions to run it as a program). This could have been done with symbolic symbols like this:

```
[alromero@srih0001 BoronCarbide]$chmod u=rwx,go=rx` scriptname
```

Other important commands

See content of a file without looking into

```
[alromero@srih0001 BoronCarbide]$cat`scriptname
```

Similar you can for example take whatever in in file2 a

```
[alromero@srih0001 BoronCarbide]$ tac afile bfile cfile > output
```

It will put all of “cfile”, then “bfile” and finally “afile” into the file called “output”. This is just handy for when you need to do a reverse concatenation

if you want to see only the last lines

```
[alromero@srih0001 BoronCarbide]$ last afile
```

You can also delete a file as

```
rm -ri afile
```

HERE we used "-ri" to allow the user to confirm.

If you do

```
rm -rf *
```

You will delete everything

A good plane to learns on some of the commands can be found in Line Shell Commands, most used shell commands.

Let us create a file with some content

```
[alromero@srih0001 BoronCarbide]$ vi content.txt
```

This editor have different comments, a good place to read is in Vim tutorial or in Other Vim tutorial.

Let us now think you are in you desktop and you want to go to spruce. For that we will use a secure shell.

If you are on campus, you can use

```
ssh <username>@spruce.hpc.wvu.edu
```

If you are outside campus, the best is to have VPN (Virtual Private Network). Here is a description on how to connect VPN connection. Either way, you do not need to connect in most cases, as you can use Google Colab for your projects or you own computer.