*S****hell Workshop***

***Introduction and Setup***

*A* ***shell*** *is simply the outermost layer of an operating system. It’s designed to provide a way for you to interact with the tools and services that your operating system provides.*

*It could be in two forms:*

***Graphic User interface or GUI-*** *Interaction with the operating system through like windows and buttons.*

***Command Line Interface or CLI-*** *Entering of text commands instead of clicking on buttons.*

*I will however focus on the use of CLI because we are already familiar with GUI from daily usage e.g creating a new folder, deleting a file etc by clicking on mouse or buttons.*

*There are several available types of shell, however the one that seems to be most common is* ***BASH*** *which I will be sticking with for this workshop.*

*First, download and install Git BASH from* [*https://git-scm.com/download/win*](https://git-scm.com/download/win) *for Window’s user, follow up the installation procedures and then open up the terminal (which is the bash). Terminal is simply the interface you work with to give commands to the shell, this already comes with* ***Linux and Mac.***

***Commands***

*There are several commands available and these are best known with practice over time. To become a terminal guru, you have to live in the terminal and one day soon from now that you are starting of, it will be part of you. I will however run through the most common and fundamentals of working with shell and for clarity, commands will be enclosed as```code```, so let us get down to it.*

***echo:*** *It is used to output strings on the terminal by simply repeating the string back to you on the next line* ***e.g ```****echo “Hello Abdullah”```. I would always recommend you put the strings you want to output in quote as with* ***Hello Abdullah*** *above, this will prevent unexpected output as you move on.*

***!!****: This simply re-enters your last command and execute it,* ***go ahead and try it.***

*\_****Quick tip\_: You can use the up-arrow and down-arrow keys on your keyboard to navigate through your previous commands and you can type history too to see the list.***

***Variables:*** *Just as in programming languages, you can assign variable on the command line of which the variable can be used for another function. To do this, simply assign value for x by entering ```x=100```, notice the absence of space: any form of spacing such as ```x = 100``` in between will be misinterpreted. So you can go ahead and reuse the variable .* ***Try```echo x```.***

***NB:*** *We did not put x in quote because it is a variable that holds another value. Also, if the variable x is to be used in a sentence, you have to put a dollar ($) as a prefix so it will not be interpreted as ordinary character.*

***Compare: ``****`echo “welcome to class x”``` and ```echo “welcome to class $x”```,* ***go ahead and try it!***

***Terminal screen size:*** *Simply enter ```echo “$COLUMNS x $LINES”```. Note the dollar signs, they indicates variables, those are self-defining variables in Bash which gets the terminal size in number of columns and lines.*

***Navigating directories:*** *There are a bunch of commands in this aspect:*

***ls****: list the contents of the current directory*

***cd ‘-particular directory\_’*** *: Move to a particular directory e.g ```cd Donwloads```*

***cd -****: Move back to the former directory that you were on.*

***ls ..*** *: list the contents of the parent directory.*

***cd ..*** *: Go backward one directory(this is called the* ***parent directory****)*

***\_Quick tip\_: Consider the directory ```c/Users/abdulloooh/Desktop```, if you are currently in Desktop directory i.e the output of running `pwd`(****will be covered next)* ***is the directory given above (for my own computer, might be different from yours but will definitely lead to Desktop)****. SO, “****Desktop” is the current directory also known as working directory, “abdullooh” is the parent directory while Users is the grandparent directory,***

***So, Working directory is the current directory you are inside, parent directory is the immediately above your current directory and home directory is user’s top level directory that has all of the other directories inside of it.***

***I hope you figure that out,*** *take time to ruminate over it!*

***pwd:*** *shows the current directory you are on. Go ahead and check!*

***\_Exercise\_: Read up on absolute and relative path*** *of specifying directories****!!***

***\_Tips\_: In Absolute path, you provide full path from the home directory while in Relative path, you provide only relative path 😑*** *. Consider the example directory we used initially (****c/Users/abdulloooh/Desktop****), if there is a folder named “lesson” inside desktop, to navigate into lesson, you either type:* ***```cd*** *‘****c/Users/abdulloooh/Desktop/lesson’```*** *or ```****cd ‘lesson’```. The first is absolute path while the second is relative. Note that the current directory was “Desktop”, so we moved from Desktop to Lesson.***

***ls -l:*** *long formatted list, it gives more details than* ***ls*** *like inclusion of the date and time the files were last modified and there sizes.*

***ls \*.bear*** *,* ***ls ade\**** *and* ***ls ade\*.txt :*** *the first lists all files that ends with “.bear”, the second lists all files that start with “ade” while the third list all files that start with “ade” and end in “.txt”. This is particularly important when you have several files in a directory and you need to filter some out. Go ahead and try several other options using the great tool (****\*)****.*

***Organizing your files***

***mkdir:*** *create a new directory (folder) e.g ```mkdir photos```.*

***touch:*** *create a new file e.g ```touch mydoc.txt```*

*There is different between folder(directory) and file 😣*

***mv:*** *moves a file from the current directory to another. To specify the directory for the file to be moved to, you can use relative or absolute directory. Example: Remember we created a folder named* ***photos*** *inside Desktop, if there already exist another file (say, myfile.txt) in Desktop, we can move this file into Photos directory (of course if you want to) by ```mv myfile.txt Photos```. Note that the Photos directory and the file were* ***in the working directory <note this>.***

***What if we want to move it back to Desktop?*** *Try to figure it out then come back to check how I did mine.*

*Because we want to move the file into Desktop,* ***cd*** *back into the parent of* ***Desktop*** *which is* ***abdulloooh*** *in this case, then provide relative path of the file as well as the destination directory i.e ```mv Desktop/Photos/myfile.txt Desktop```. The secret behind it is that the destination directory and the file to be moved should be daughters of current directory.*

*Wondering what does* ***cd*** *actually means? It stands for* ***change directory 😜***

***Up next: Downloading files, viewing files and removing things***