Lab 0(b): Introduction to Shell Scripting and Script Files:

A script file is a plain text file that contains commands that can be interpreted by the bash shell. Since it is a plain text file, you may edit or create it using your favorite text editor. Conventionally bash script files are named with the extension .sh. In order to let the OS know that a file is in fact a bash script file, it has to be:

1. Set the file to be executable, for instance

```
chmod u+x myscript.sh
```

This tells the OS that this file contains a program and is executable. The next step tells the OS how to actually execute it.

2. Set the first line of your script to be

```
#!/bin/bash
```

This will tell the OS to use bash to execute the commands in your file

- **How to start Scripting**

The simplest form of scripting is to simply put a sequence of commands in a file. These are the same commands that you would type at the shell prompt. You can create a simple script file script1.sh using any editor with the following lines in it and save it.

```
#!/bin/bash

echo "Listing of current directory"

ls

echo "Listing of root directory"

cd /

ls
```

You can then execute the script file using ./script1.sh after making it executable with *chmod u+x script1.sh*. You are already familiar with the ls command. You will find that after executing the last individual command at the prompt, your current working directory (which can be found using the pwd command) is now /. Whereas after the script has run, the current working directory remains unchanged. This is because, when a script file is executed, the shell starts a new session of the shell (called a subshell), and it is the subshell that interprets and executes the commands in the script. The subshell process terminates once the script is complete and we are returned to our original shell prompt. Since the cd command in the script changes the current working directory in the subshell and not the shell session we are in, it does not affect the state of our current shell.

- **Command Piping and output redirection**

In the bash shell, the pipe or ‘|’ symbol connects two commands together. For example the line command1 | command2 connects the standard output (stdout) of command1 to the standard input (stdin) of command2. Consider a simple example. The sort command sorts the lines of a text file alphabetically and writes it to stdout. The uniq command omits repeated lines in files and writes out the result to stdout (read their man pages to know more).

Consider the text file fruits.txt containing the following lines

```
apple
mango
orange
pineapple
mango
strawberry
strawberry
apple
grapes
```

We wish to remove duplicate lines in this file. Since uniq will only omit consecutive duplicate lines, we need to sort the file alphabetically first. We would like to take advantage of shell pipes to do this in one step. Our script to perform this uniquefruits.sh will look like:

```
#!/bin/bash
sort fruits.txt | uniq
```

- Redirection

Suppose you would like the output of a command to be written to a file rather than the screen. The shell provides the redirection operator to achieve that. Redirection has several variants, but right now we will use the output

redirection operator ‘>’. If we use command > file.txt

This redirects the stdout of command so that its contents is written to the file file.txt rather than to the screen. You can modify uniquefruits.sh so that its output goes to the file uniquefruits.txt. The script will look like:

```
#!/bin/bash
sort fruits.txt | uniq > uniquefruits.txt
```

More than two commands can be joined in a pipeline. For example

```
#!/bin/bash

sort fruits.txt | uniq | nl > uniquefruits.txt
```

Assignments:

1. Print the details of the system and perform the actions using a shell script
    a. Display the kernel version.
    b. Hostname & Username, Current date and calendar
    c. Check Ethernet connectivity and IP address
    d. how long system is running
    e. Show memory status
    f. Display processor activity of system
    g. Display user's activity in the system

After showing these details of convey a message “System is going for Rebooting”. Then restart the system.

2. Find the processes and their details related with networking.