Operating Systems 2 – Shell Scripting

So far we have looked at executing individual commands in the shell such as pwd, ls, mkdir, etc. We can also write these commands in a file and have the shell execute all the commands listed inside the file.

For example, instead of writing the following 3 commands in the shell:

```
$ pwd
$ ls
$ who
```

We can write them inside a file using cat, vi, or any other shell command or program that allows us to write into a file or edit a text file:

```
$ cat > script
#!/bin/sh
pwd
ls
who
```

The first line (#!/bin/sh) written inside the script file tells the shell that this file should be treated as a program. We then wrote the 3 commands inside the file. To run these commands, we use the shell command **sh** that will treat the file as an executable program (a shell script) instead of a regular text file:

```
$ sh script
/home/std001
file1    file2    file2
haider   pts/0      2014-05-04 02:33 (10.33.18.101)
std001   pts/5      2014-05-01 01:24 (10.33.20.96)
```

What the sh command did was go through the script file line by line and treat everything written inside it as a command to execute.

To write comments inside a shell script we use the hashtag sign # before it:

```
# This is a comment
pwd
```

To display a statement on the screen we use the echo command:

```
echo “type this line”
```

To receive data from the user we use the read command:

```
echo “What is your name?”
read name
echo “$name, that’s a lovely name.”
```

Notice how we used name as a variable to store the name the user inputs, but when we wanted to display the name, we added the $ sign before it. This tells the shell not to treat name as a word to print out, but to print the value stored in the name variable (which the user entered).

We can also use conditional statements (if, else) inside the script command, as well as loops (while) and other statements that control the flow of the program, but we will not cover them here.

Exercise 1: Write a shell script that displays the current directory, a list of file/directories inside the current directory, then creates a directory called scripts:

Exercise 2: What is the output of the following shell script (assuming the current directory is empty):

```
#!/bin/sh
# Create 2 files
touch f1 f2
# Let the user know
echo “The files have been written”
ls
echo “Have a good day”
```

Output: