CSC3320 System Level Programming
Lab Assignment 5 - out-of-Lab

Purpose: Learn how to write basic shell script.

In Chapter 2 and 3, you have learned a list of utilities. However, each time we could only type a single command on command line in terminal. It is inconvenient sometimes when a task has to been accomplished by multiple commands. For example, if the task needs to be repeated, you may have to restart the execution of the list of commands by typing the command one by one. For this reason, the shell script file is used to store the commands interpreted by shell. It is more than a regular file containing only the command. You can even write for loop, if else and switch case statement in the shell script. The shell script file can be executed directly by providing the name of it on command line.

Write a report by answering the questions and upload the report (named as Lab5_P1_FirstNameLastName.pdf or Lab5_P1_FirstNameLastName.doc) to google classroom.

This lab assignment is related to the slides #12 to #14 in chapter 4 Part 1:

Now it is your turn to create your first shell script file by following the steps below.

Step 1: Go to your home directory and create a new file named as simple.sh (vi simple.sh or nano simple.sh), then include following lines in your simple.sh.

Question 1) : What did you see in the output of step 3?

```
#!/bin/bash
#Simple Script
#
Echo Congratulations! Now you know shell script!
Echo -n “The current time and date are: “
```

Step 2: Save your file and exit editor.

Step 3: Execute this file by invoking its name.

date

Step 4: Execute this file by adding ./ before the its name.
$ ./simple.sh

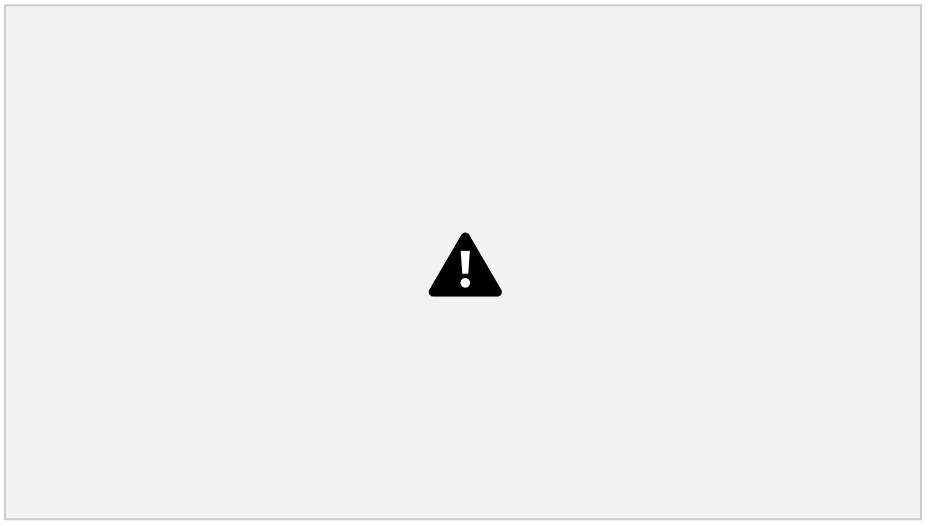
Question 2) : What did you see in the output of step 4?

Congratulations! Now you know shell script!
The current time and date are: Thu Sep 23 18:46:58 EDT 2021 Step 5: Try

following command to make simple.sh executable. $chmod a+x simple.sh

Step 6: Execute this file again.
$ ./simple.sh Note: you must type ./ before the name. This is because current working directory is not in PATH. However, you can modify the value of PATH variable and add current working directory into it by referring to next part.

Question 3): Attach a screenshot of the output in step 6.



Question 4): Describe the meaning of -n option in echo command. Read

the slides #13 in Chapter 4 and then answer the following two questions.

-n option is used with echo command and it deletes any trailing spaces

from the following line by putting the date in line with the echoed line

2
Question 5): Is "Simple Script" a comment? If not, what is the meaning of it or why we use it?

No it is not a comment because if you don't put it in a file then it is not going to run

Question 6): Is "#!/bin/bash" a comment? If not, what is the meaning of it or why we use it in first line?
It is a script to be ran and it is not a comment, but it rather specifies which shell script itll be interpreted by. It has to be in the first line or else the script will be ran by the bourne shell

Part 2:
To discard the ./ before the script file name when executing it, we need to change the PATH variable's value and add current working directory into it.

Step 7: Print out the value stored in PATH variable.
Question 7) : How many directories you can find in the output? Note: the directories are separated by colon.
6 separate directories

Step 8: Try command below to insert current working directory at the beginning of the string value stored in PATH variable. $PATH=.:$PATH Step 9: Execute simple.sh again by trying following command. $simple.sh $ simple.sh

Question 8) : Can you find errors prompted in step 9 ? If not, please briefly describe why there is no need to put ./ before the file name.

There are not any errors prompted in step 9. The PATH variable has been updarted with the present working directory. So the command could be ran without the preceding ./.

Step 10: Log out the connection to the snowball server and reconnect to it. Or simply close your terminal and then reopen your terminal.

Step 11: Print out the value stored in PATH variable again. Question 9: Can you find the current working directory . in the PATH variable?

No the directory is not included in the PATH variable

Step 11: Execute simple.sh again by trying following command. $simple.sh
$ simple.sh

Question 10) : Can you find errors prompted in step 11 ? If yes, please explain why?

The command is not found as the PATH variable no longer containes the directory

Part 3 - Optional:

This part is optional, but you will find more questions about this part in your next lab.

Step 1:
Create a new file named as checkError.sh (vi checkError.sh or nano checkError.sh), then include following lines in your checkError.sh.

```
$#/bin/bash

/* Check Error Script */ echo "Try to find out some

errors!!!"

# Seach for the words which can be matched by regex [^a]*ce
# And save the output to file "Result"
echo "The regex [^a]*ce can match the string(s):" > Result grep '^[^a]*ce$' <<
END >> Result lance ace brace decide piece -ENDHERE

# Check the existence of file "Result"
# Send the content in "Result" to your emailbox
# $1 is replaced by your campusID ls mail $1 < Result

# $1 is replaced by your campusID echo "The result has been sent to
${1}@student.gsu.edu" echo "Congratulations! You have corrected all the errors!"
```

Or you can directly copy this file from my public directory to your current working directory by following command and then skip step 2.

```
$cp /home/bbello1/public/checkError.sh checkError.sh
```

Step 2: Save

your file and exit editor.

Step 3: Try following command to make checkError.sh executable.
$chmod a+x checkError.sh
Step 4: Execute this file by following command.
$./checkError.sh campusID
Note: Replace campusID with your own campus ID.
E.g. $./checkError.sh bbello14

Questions:
Can you find some errors when executing the command in step 4? If yes, please point out which lines contain errors. Think about the correction in your next lab. Before the correction, you could pre-view the slides #15 - #24 in Chapter 4.

Hints:

• Following is a sample of the output once all the errors are corrected

$ ./checkError.sh bbello1 Try to find out
some errors!!! checkError.sh Result
The result has been sent to bbello1@student.gsu.edu Congratulations! You have corrected all the errors!

• You can use cat -n checkError.sh to check line numbers.

• You may need to use CTRL-C to terminate the execution of the command, especially for the script file with errors.