Brian Duenas

CSE 460

Lab 2

20 points Total

1. Basic Shell Programming

A)

Q: What difference do you see when executing the script with $ ./ginfo and with $ . ./ginfo ?

A: $ ./ginfo opens the executable. $ . ./ginfo closed my ssh window for some reason. Using $. ginfo ran the script then closed my ssh window probably because I set up my environment to open executables without the "./" prefix.

B)

```
[005029683@csusb.edu@jb359-1 lab2]$ pwd
/home/csusb.edu/005029683/cse460/lab2
[005029683@csusb.edu@jb359-1 lab2]$ echo $USER
005029683@csusb.edu
[005029683@csusb.edu@jb359-1 lab2]$ echo $BASH_VERSION
4.4.19(1)-release
[005029683@csusb.edu@jb359-1 lab2]$ echo $HOME
/home/csusb.edu/005029683
[005029683@csusb.edu@jb359-1 lab2]$ echo $LINES
24
[005029683@csusb.edu@jb359-1 lab2]$
```

C)

Q: How do you define variable x with value 10 and print it on screen?

A:

$ x=10
$ echo $x

Q: How do you define variable xn with value 'Rani' and print it on screen?

A:

$ xn=Rani
$ echo $xn

Q: How do you print the sum of two numbers, say, 6 and 3?

A:

$ echo 6 + 3

Q: How do you define two variables x=20, y=5 and then print the quotient of x and y (i.e. x/y)?

A:

$x=20
$ y=5
$ expr x / y

Q: Modify the above question to store the result of dividing x by y to a variable called z.

A:

$ x=20
$ y=5
$ z=`expr x / y`
$ echo $z

Q: Write a script XYZ=2017. What do you see? Explain the difference you observe in the two cases.

A: I see no differences in my shell. I am using a ssh to the machine. See proof below.

```
[005029683@csusb.edu@jb359-1 lab2]$ ./testShell.sh
[005029683@csusb.edu@jb359-1 lab2]$ echo $XYZ
2017
[005029683@csusb.edu@jb359-1 lab2]$ testShell.sh
[005029683@csusb.edu@jb359-1 lab2]$ echo $XYZ
2017
[005029683@csusb.edu@jb359-1 lab2]$ . ./testShell.sh
[005029683@csusb.edu@jb359-1 lab2]$ echo $XYZ
2017
[005029683@csusb.edu@jb359-1 lab2]$
```

2. AWK

Q: Try the command: ***$ ps auxw | awk '{print $1 "\t\t" $2}'*** What do you see? What does the command do?

A: It print the processes running on the machine with the user that's running/ in control of it.

3. Viewing Processes

```
[2]+  Exit 1                  renice robot
[005029683@csusb.edu@jb359-1 lab2]$ ps -l
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S 387050 5728  5725  0  80   0 - 33249 -      pts/2    00:00:00 bash
0 R 387050 6460  5728 99  90  10 -  3351 -      pts/2    00:00:27 robot
0 R 387050 6475  5728  0  80   0 - 35760 -      pts/2    00:00:00 ps
[005029683@csusb.edu@jb359-1 lab2]$
```

4. Starting New Processes

Q: Compile and run the program given. What do you see? A: It shows system processes with variables TTY, STAT, TIME, and COMMANAD.

```
Running ps with system
  PID TTY      STAT   TIME COMMAND
    1 ?        Ss     0:01 /usr/lib/syste
    2 ?        S      0:00 [kthreadd]
    4 ?        I<     0:00 [kworker/0:0H]
    6 ?        I<     0:00 [mm_percpu_wq]
    7 ?        S      0:00 [ksoftirqd/0]
    8 ?        I      0:00 [rcu_sched]
    9 ?        I      0:00 [rcu_bh]
```

Q: Change the ***system*** statement in test_system.cpp to ***system ("ps -ax &");***. Compile and run it again. What happens? What does '&' here do?

A: It printed and process runs in background

5. Shell Programming Practice

```
[005029683@csusb.edu@jb359-1 lab2]$ robot &
[1] 6723
[005029683@csusb.edu@jb359-1 lab2]$ robot &
[2] 6724
[005029683@csusb.edu@jb359-1 lab2]$ robot &
[3] 6725
[005029683@csusb.edu@jb359-1 lab2]$ terminateProcess
[1]   Terminated              robot
[2]-  Terminated              robot
[3]+  Terminated              robot
```