1. Semester and Year	: 2012 FALL
2. Course Number	: CS-227
3. Course Title	: Linux & System Software
4. Work Number	: LA-09
5. Work Name	: Lab 09
6. Work Version	: Version1
7. Long Date	: Sunday, 28, October, 2012
8. Author(s) Name(s)	: Jake Waffle

Exercise 9

Write a sequence of commands or a script that demonstrates variable expansion occurring before pathname expansion.

```
#!/bin/bash

Var = *
Var2 = Var
cat “$Var2”
```

Because the wild card is within a variable. When using variable expansion on the variable with the wild card, path name expansion is done after the variable is expanded.

Exercise 11

Explain the behavior of the following shell script:

```
#!/bin/bash
twoliner=”This is line 1.
This is line 2.”
echo “$twoliner”
echo $twoliner
```

a) How many arguments does each echo command see in this script? Explain.

After writing the script with vi and executing it in the terminal, I found that the output of the first echo command resulted with two lines of text. While the second echo command yielded a single line of text despite there looking to be a newline (\n) character in the string.

Since the only difference between the two echo commands is the double quotes around the variable. I think that the quotes may separate the variable into different pieces using a new line character and then give them to the command in a bundle (in this case of two.) Assuming echo is anything like Python's print command, then after each call there will be a newline character. So if echo uses a separate command for each of its arguments (like I think it might,) then there will be a newline character in between the arguments' output on the terminal. And I guess newline characters just don't get recognized by echo commands by default. So only by making the IFS variable a certain character, can you have arguments separated and in turn displayed with newline characters between them.

So for the first echo command I think that there are two arguments. And for the second echo command I think that there was only one argument.

b) Redefine the IFS shell variable so that the output of the second echo is the same as the first.

To make the second echo command result with the same thing as the first echo command, the second echo command needs to register the newline character within the variable. So by making the IFS variable equal to '\n' the literal newline character, we can split the variable into different arguments divided by the newline. And then the echo command will display the arguments with a newline in between them.

Script Task:

Write a script that logs off of all users, updates the system, reboots machine and set it up to run at 02:00AM every day.

#script.txt

#!/bin/bash

killall -u

sudo apt-get update

sudo apt-get upgrade

reboot

The above script simple does most of the tasks on the above task. The killall command kills all processes, but with the -u option it will only kill (log off) user processes. The apt-get commands are used for updating the package list and upgrading your software. Then the reboot command reboots the system.

0 2 * * * ./script

Finally the above line if placed in a crontab will execute the above script every day at 02:00AM. But keep in mind that the script has to be in one of the directories in the $PATH variable for the crontab to be able to find it.