CSC 2500: Unix Programming Lab
LAB 06
Name _____Bradley Harper__ _____

General Instructions

Using your book and previous lecture material, fill out this assignment sheet. **Use red text to signify your answers.** This assignment corresponds with chapter 10 of your textbook. You should utilize online resources to answer these questions as well.

Submission Instructions

To submit, **write your name** and save this document as a PDF. Attach your PDF document and each script you've written to the iLearn dropbox.

Lab Questions

1. The shell expands **$0** to the name of the calling program (Sobell, page 474), **$1–$n** to the individual command-line arguments (positional parameters; Sobell, page 475), **$*** (Sobell, page 478) to all positional parameters, and **$#** (Sobell, page 479) to the number of positional parameters.

Write a script named **all** that displays (sends to standard output) the name of the calling program, the number of positional parameters, and a list of positional parameters. Remember to make the file executable (Sobell, page 100). Test the script with 0, 1, and 5 positional parameters. What does the script print out in each of the cases? (20)

./all.sh

5

a b c d lego

2. Make a symbolic link (Sobell, p.114) named **linkto** to the **all** script you wrote in the previous step. Call **linkto** with two arguments. What does the script report as the name it was called as? (10)

./linkto

3. Write a script named **myname** that uses `echo` (most of the examples in Chapter 10 use this utility) to prompt the user with **Enter your name:** , reads into a variable the string the user types in response to the prompt, and then displays **Hello** followed by the string the user typed. When you run the program it should look like this: (10)

$ **./myname**

Enter your name: Max Wild

Bradley Harper

Hello Max Wild

4. Rewrite **myname** from the previous step, calling it **myname2**. Have this version of the program prompt the user for a string, but instead of displaying **Hello** and the string on the screen (sending it to standard output) redirect the output so the program writes only the string the user entered (and not **Hello**) to the temporary file named **PID.name** where PID is the process ID number (Sobell, page 480) of the process running the script. Display the contents of the **PID.name** file. (10)

idk

5. Write and run a script named **looper** that uses the **for** (Sobell, page 449) control structure to loop through the command-line arguments and display each argument on a separate line. (15)

6. Rewrite looper from the previous step, calling it **looper2**. Have this version of the program use the **for...in** control structure (Sobell, page 447) to perform the same task. (15)

7. Write a script named **ifthen** that prompts the user with >> and reads a string of text from the user. If the user enters a nonnull string, the script displays **You entered:** followed by the string; otherwise it displays **Where is your input?.** Use an **if...then...else** control structure (Sobell, page 439) to implement the two-way branch in the script. Use the **test** (Sobell, pages 435 and 1011) builtin to determine if the user enters a null string. What do you have to do to avoid getting an error message when you prompt with >>? (There are several ways to construct the test statement.) (20)

Bradley Harper