ECE 282 Lab 6

Lab Report Due: 6 pm, Feb 21.2018

# Command practice (include in report)

**Tutorial:** [**http://eecs.mines.edu/Courses/csci274/Content/Slides/10\_searching.pdf**](http://eecs.mines.edu/Courses/csci274/Content/Slides/10_searching.pdf)

What do the following commands do?

1. grep –c: **Prints a count of how many lines are matching the input phrase**
2. grep –B: **Print the number of lines in-between matching phrases**
3. find ~ -user nhdo -name “\*.c” (Replace ‘nhdo’ with your username)  
   **Searches users home directory (and all subdirectories) for all .c files**

Explain the meaning of the following output of ‘diff’ command  
**Hint:** [**https://merchantprotocol.com/517/whats-the-diff/**](https://merchantprotocol.com/517/whats-the-diff/)

1. 2c2  
   **Line two of file one compared with line 2 of file 2**
2. 3a4  
   **Line added to file**

Write down the commands that can do the following tasks:

1. Count how many .txt file that you own in your home directory.  
   **Hint: wc**  
   **ls ~ | grep .txt | wc**
2. Turn off the “echo” of the terminal.  
   **Hint: Lecture slide  
   stty -echo**
3. Make your terminal output only UPPER CASE.  
   **Hint: man stty** “translate lowercase characters to uppercase”

**Hint:** You can search **man** using grep  
**stty olcuc**

1. Count how many numbers from 1 to 13000 that contain “123” (e.g. 1**123**5, **123**55 etc.)?  
   **Hint: seq** for generating numbers **seq 13000 | grep –e 123 | wc  
   Bonus:** How about counting the numbers that STARTS with “123” instead?  
   **Hint:** [**http://www.robelle.com/smugbook/regexpr.html**](http://www.robelle.com/smugbook/regexpr.html)

# Chit chat (include in report)

You can use the ‘**write’** command to chat with another friend. The usage is simple: just type “**write username**” and send messages! When you are done, press “**Ctrl+C**”

Now you can try to compile the source code **write0.c** and **write1.c** from the course companion website. Read the source code carefully and answer the following questions:

1. How does **write0** send the text to another terminal?  
   **opens the other terminal by using the username inputted then dumbs the contents of stdin to that terminal session.**
2. What are the differences between **write0** and **write1**?  
   **Fetches device that contains other users terminal session then writes to that device.**
3. What are the differences between **write1** and UNIX’s **write** command?  
   **The unix write command adds the sending user as well as a timestamp to the message. It also allows blocking of messages and the ability to specify which terminal session the user has open you would like to message to.**
4. How does **write1** get the tty from the username?

**Opens the utmp file then looks for the users name. When found it retrieves the terminal directory and then returns this value.**

**Hint: function get\_tty(). And do you remember the ‘who’ program?**

**Note: Did you know that you can chat to yourself? Just log in using another terminal (from another instance of Putty)**

**Hint: To use write0: ./write0 /dev/pts/123 (replace ‘pts/123’ with the terminal name you want)**

# Bit operation (include in report)

To do well in Lab 6 and the project 1, you will need to know bit operations, including set bit, clear bit, check bit and toggle bit. This knowledge will also be useful for ECE362 and when you interview for jobs.  
Tutorial: <http://blog.yastrebkov.com/2012/01/how-to-set-clear-toggle-and-check.html>

Let B = 11010101b in binary (which is 213 in decimal)

Use the bit operator such as &, |, or ^ to do the following tasks. Write down the 1 line C code that you can use.

Example: Set the 4th bit from the right of B, making B = 11011101b  
Answers: **B = B | (1<<3);** or **B |= 1 <<3;** or **B |= 8 (because 8decimal= 00001000binary)  
Note: You can run the source code bit.c in Oncourse/Resource/Lab5\_materals for testing purpose.**

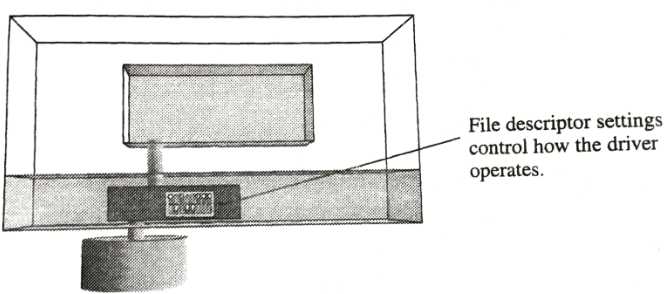
1. Set the 3rd bit from the left of B, making B = 11110101b **B |= 0x20**
2. Clear the 1st bit from the right of B, making B = 11010100b**B &= 0xFE**
3. Check if bit 2nd from the right of B is set  
   **if( (B && 0x02) != 0)**
4. Toggle the 2nd bit from the left of B. What is the value of B after that?

**B ^= 1 << 7**

# Get, modify and set bit

You can modify the attribute of the file descriptor by changing the “control variables” (that control how the file descriptor processes the in/out data). Typically, the procedure has 3 steps:

* Make a system call to copy the control variables from the file descriptor
* Modify it using bit check/set/clear/toggle
* Send the modified value back to the file descriptor.



By default the ‘\b’ character is the backspace character, which erases the last character on display. Write a program seterase.c which accepts a character as an argument, and sets that character as the erasing control character. If the passed argument is “0”, then the setting will be reset to the original behavior (‘\b’).

Ex:

./seterase x  
This command should set ‘x’ as the erase character

./seterase #  
This command should set ‘#’ as the erase character

./seterase 0  
This command should set ‘\b’ as the erase character, which is the default behavior.

See source code showtty.c to see how you can set the erase character (c\_cc[VERASE]). Also include proper functionality for --help and no argument cases.