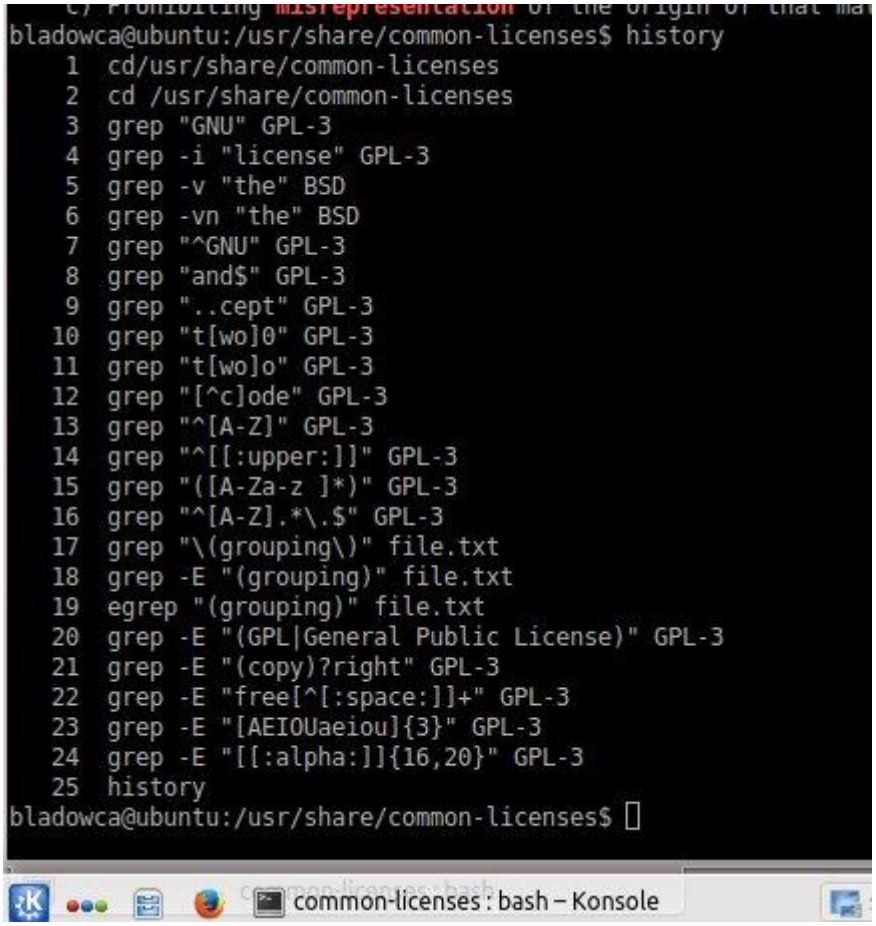
**Casey Bladow**
**CSIS 360 - Spring 2015**
**Assignment #12 - 25 points**
**Due Tuesday, May 5**

Follow the **Creating Assignments** handout while performing the below steps. The purpose of this assignment is to conclude with some very useful and powerful file processing tools (including *grep*, *sed*, and*awk*) that should be emphasized. Often, these are used in combination with a pipe. Going through the tutorials, you will in no way become an expert. In fact, you likely will not remember much of the syntax and options, but you will remember in a general sense the types of things you can do. In addition, this assignment extends the creation and use of shell scripts that were first introduced in assignment 10. It is my intent that this assignment not be too time intensive since this is your final assignment and I know everyone has plenty of stuff going on.

1. Learn about *grep & regular expressions*
    - Open a terminal window and type the following command to clear your history.
    - `history -cw`

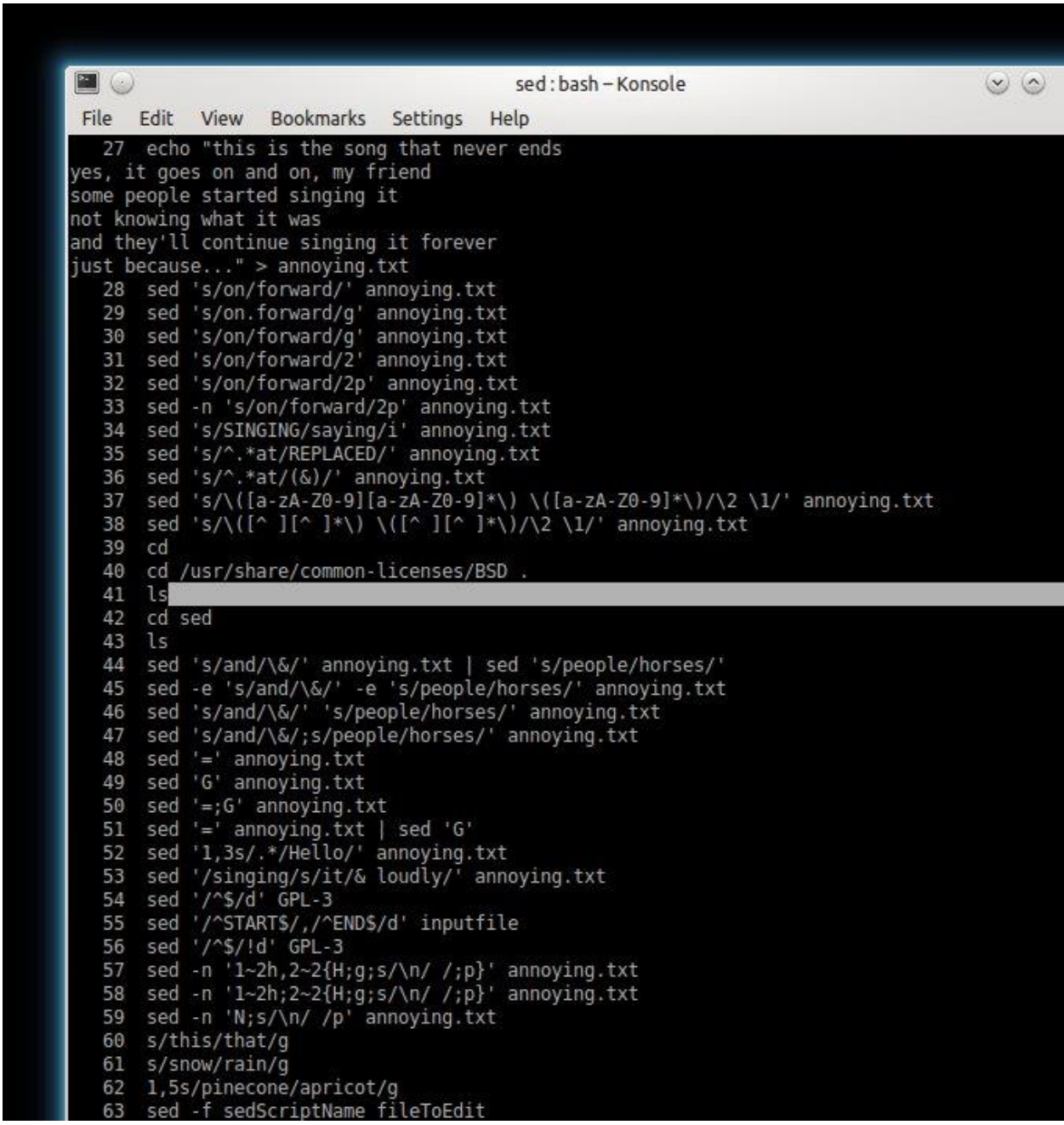
    - Read the following article while at the keyboard, typing each command as you go.
        - **https://www.digitalocean.com/community/tutorials/using-grep-regular-expressions-to-search-for-text-patterns-in-linux**
    - Once you've completed the tutorial, type the command *history* and ***copy and paste the results to your assignment 12 document.*** If you wanted to save your commands for future reference, you could give the command *history > grepExamples* (or whatever you want to call it, plus you would need to be in your own directory or give a path to it).




2. Learn about *sed*
   - Open a terminal window and type the following command to clear your history.
   - `history -cw`

   - Make a directory and change to it as the tutorial will have you copy some files.
   - Read the following articles while at the keyboard, typing each command as you go.
     - **https://www.digitalocean.com/community/tutorials/the-basics-of-using-the-sed-stream-editor-to-manipulate-text-in-linux** Note that you already read the article linked by this article*learn about regular expressions*. Also, there is a link to the article below (*Editing with Sed - Article #2*) at the bottom of this article.

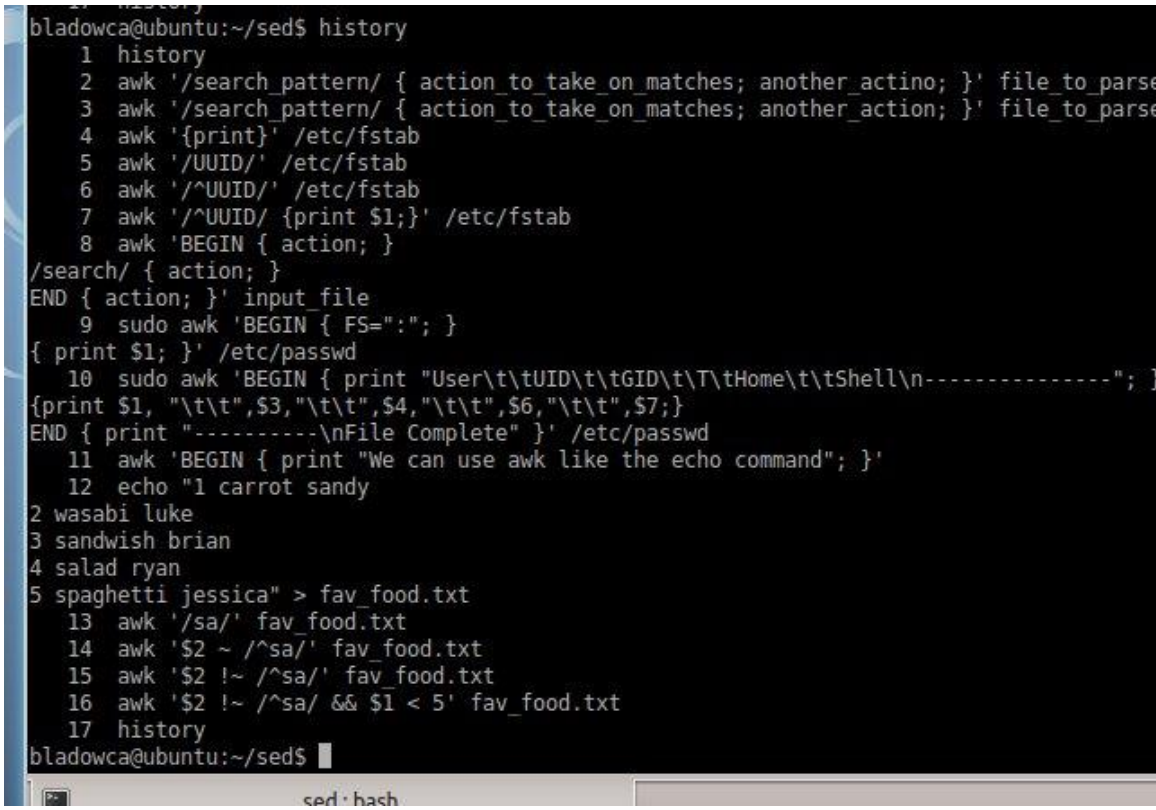
        - **https://www.digitalocean.com/community/tutorials/intermediate-sed-manipulating-streams-of-text-in-a-linux-environment**
    - Once you've completed the tutorial, type the command *history* and ***copy and paste the results to your assignment 12 document.*** If you wanted to save your commands for future reference, you could give the command *history > sedExamples* (or whatever you want to call it).
    - One final note about *sed*... the *vim* editor uses *sed* syntax for various file manipulation commands. For example, if you wanted to replace all occurrences of the of the word *thare* with *there* in the first 5 lines of the file, you could give the following command in *vim*
    - `:1,5s/thare/there/g`




3. Learn about *awk*
    - Open a terminal window and type the following command to clear your history.
    - `history -cw`

    - Read the following article while at the keyboard, typing each command as you go.

        - **https://www.digitalocean.com/community/tutorials/how-to-use-the-awk-language-to-manipulate-text-in-linux**
    - Once you've completed the tutorial, type the command *history* and ***copy and paste the results to your assignment 12 document.*** If you wanted to save your commands for future reference, you could give the command *history > awkExamples* (or whatever you want to call it).
    - I would also encourage you to read the following articles as the above article barely touched the surface of *awk* and the following will allow you to at least scratch the surface! In addition, you may want to search for more articles. If you find any good ones, let me know in your assignment 12 document.
        - **http://www.ibm.com/developerworks/library/l-awk1/**
        - **http://www.ibm.com/developerworks/library/l-awk2/**




4. The purpose of this activity is to give you more familiarity with writing shell scripts. You were introduced to this in assignment 10. Shell scripts can be very powerful and can be written in different languages. The *bash* shell has it's own programming language and it is decribed in chapter 27 in your textbook. Scripts can also be written in other programming languages such as Perl

(chapter 28) and Python (CSIS 152 and CSIS 153). One certainly cannot become proficient in writing scripts in any language for one assignment, so like I say, the purpose is to give you *some* familiarity and hopefully an interest and ability go beyond this assignment if you have an interest or need to do so in the future.

There is a trick to writing Python shell scripts which I learned about from the below articles.

- **http://www.dreamsyssoft.com/python-scripting-tutorial/shell-tutorial.php**
- **http://www.linuxjournal.com/content/python-scripts-replacement-bash-utility-scripts**
- **http://magazine.redhat.com/2008/02/07/python-for-bash-scripters-a-well-kept-secret/**

Read through these articles as well as browsing through chapter 27 in preparation for this activity.

So here's the prologue for this activity. As you are aware, when you are using the command line interface, files can be overwritten by accident and there is no trash where deleted files go if you accidentally delete them. This activity will have you create a set of scripts for safe copying and deleting including a trash where files can be recovered. There are 5 scripts included in this set.

4. *safecopy fromFile toFile*
   If the *toFile* file already exists, it will prompt to make sure that's what the user wants to prevent accidental overwriting. Yes, I know you can use a *-i* option with the *cp* utility that will do the same thing.

5. *safedelete File*
   The directory /home/youraccount/Trash will be created if it doesn't exist and the *File* will first be copied to /home/youraccount/Trash before it is deleted. If there is already a file with the same name in the trash, a numeric extension is appended. For example, if *File* already exists in the trash, *safedelete File* would create *File.1* in the trash. If both *File* and *File.1* exist in the trash,*safedelete File* would create *File.2* in the trash, etc.

6. *viewtrash*
   List the directory contents of /home/youraccount/Trash

7. *recover File*
   It will move *File* from /home/youraccount/Trash/File to the current directory if it exists else outputs an error message if it doesn't exist

8. *emptytrash* or *emptytrash File*
   It will either delete the entire /home/youraccount/Trash directory (if an accompanying file isn't given) or delete *File* from /home/youraccount/Trash.

I would suggest that you use one of the accounts that you created in an earlier assignment as deleting files and directories in a script during testing can wreak havoc. These scripts should be placed in the account's *bin* directory and the *bin* directory should be included in the path so you could execute these scripts from anywhere. The default in the accounts that I created was that the *bin* directory didn't exist but *.profile* would include it in the path if it did exist. If that's the case (as you know from previous assignments) you would need to log out after creating the *bin* directory because *.profile* is only executed at login (you knew that, right?). If adding the *bin* directory to the path is not in *.profile*, you will need to add it.

My original intent was to have you write all these scripts. However, writing all of them could be pretty ambitious for many of you so here are the scripts that I wrote for the more complicated ones. These will also help you see how to do certain things including various checks for parameters and existance of files and directories.

- *safecopy* **bash version** / **python version**
- *safedelete* **bash version** / **python version**
- *emptytrash* **bash version** / **python version**

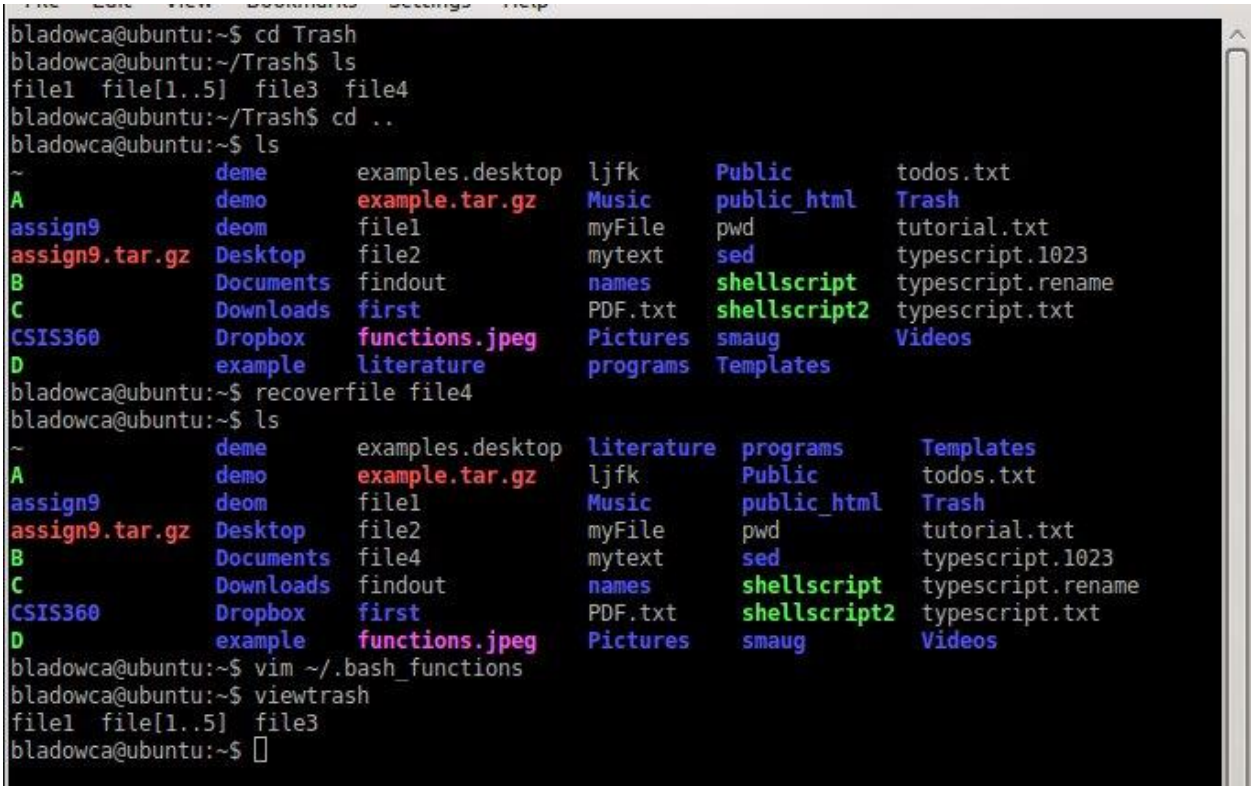
Put one of each of these scripts (you obviously can't put both with the same name) in your *bin* directory and experiment with them. Note that I wrote them so that it doesn't matter if it's a file or a directory that's being copied or deleted. I will gladly sit down with any of you in class to explain each of the scripts if you need it. If you would like the challenge, I would encourage you to try to write them yourself and tell me about it in your assignment 12 document. Make your terminal window full height and demonstrate the scripts and ***take a window grab and include it in your assignment 12 document.***

You could have multiple window grabs if things are scrolling out of the window.

Your task then becomes writing *viewtrash* and *recover*. Write one in bash and the other in Python (your choice as to which). Make your scripts robust that include error messages when necessary. Note that page 957 lists tests for files in bash scripts.

Once your scripts are working, ***take a window grab of the code of each script and include them in your assignment 12 document.***


Next, make your terminal window full height and demonstrate your scripts working (you could have multiple windows) and ***take a window grab(s) of the demononstration and include it in your assignment 12 document.***




5. ***What did you think of this assignment? How long did it take you? Do you have any suggestions for the next time I teach this class?*** Pretty easy. Basically just followed the tutorials… about 3 hours

6. ***What did you learn from this assignment?*** functions are pretty handy

7. ***Finally, are there any topics that were not covered this semester that you feel should be included in this class?*** Nope