Task:

This exercise covers:

* File manipulation from the command line.
* A very basic introduction to grep and regular expressions.
* Your first shell script
* A basic introduction to awk.

Review material provided to you by the instructor extracted from Chapter 4: UNIXLinux File Processing, Guide to UNIX Using Linux, Thompson Course Technology, ISBN 0-619-21562-3. These materials are presented in the form of a PDF file under fair usage.

Review the material provided covering Regular Expressions.

Print the Evaluation pages with the list of commands to execute and practice the contents. When you are ready for evaluation perform the commands indicated and record them on the Evaluation sheet. Submit the Evaluation pages to the instructor for marking.

Hints:

Points to ponder or watch for:

* When duplicating the two lines of the myhosts file, do them one at a time and make them adjacent. They will be used with the uniq command, which requires sorted files. Alternatively, you can sort myhosts and pipe the results to uniq to get the same results.
* The sort command needs to have its output saved for subsequent use. So when sorting any files redirect the output to new files and use these sorted files in subsequent commands (like comm).
* If using the tr command, create a new file with the upshifted text to use later with grep.

Evaluation:

This assignment is worth 35 marks. Please see the marking rubric below.

Assignment Notes:

The assignment must be demonstrated to the instructor on or before the due date during class. If your assignment is late please send an e-mail to the instructor, hal.o’connell@nscc.ca, to confirm submission. This e-mail will constitute the timestamp for evaluating any late penalty the assignment may incur.

**Part 1 – Basic File processing**

**Marking Rubric**

**Name: Jamie Lu, W0441213**

|  |  |  |
| --- | --- | --- |
| **Marks** | **Score** | **Part 1. Complete the following activities:** |
| **4** |  | Step 1. Some file setup:   1. fetch a copy of the system hosts file to your Documents directory and move it to a file called “myhosts”: (i.e. cp /etc/hosts ./myhosts 2. Use the touch command to create empty files called verbiage.txt, veg1 and veg2 |
| **1** |  | Step 2. Open myhosts in vi and duplicate each of the 2 lines that start with 127.0.0.1, one at a time (so you have two identical lines, followed by another two identical lines.) |
| **2** |  | Step 3.Use cat with redirection to append (>>) to the file verbiage.txt without using an editor. Enter the following text:  Linux was initially developed by Linus Torvalds.  There are many variants in use today.  End the entry by typing Ctrl-D (end of file) |
| **2** |  | Step 4. Append to the text file called veg1 using the cat command with redirection (>>) and enter the following text:  Potato  Tomato  Pea  Onion  <CTRL-D> to close the file      Use the same technique to append this text to the file veg2, containing these entries:  Potato  Bean  Onion  Corn  <CTRL-D> to close the file |
|  |  | You should now have 4 files on which to do today’s assignment: myhosts, veg1, veg2 and verbiage.txt. You are now ready to proceed with the remainder of the assignment. Conduct steps 5 through 14 by hand to test your use. Later you will build a script to do the same work. |
| **1** |  | Step 5. Run the uniq command on myhosts. How many lines does it display?  **It displays 9 lines.** |
| **1** |  | Step 6. Run the wc command, using the option to count lines in a file, on myhosts and note the results. Why are they different from above?  **It is different because the uniq command filters out lines that are repeated in a file. The wc command is simply a counter. It counts the number of lines, word count, byte, and characters count in the file.** |
| **1** |  | Step 7. Now run uniq on myhosts using the –d switch. What are the results?  **The uniq -d command displays the repeated lines in the file.** |
| **1** |  | Step 8. Sort each of the files veg1 and veg2 using the sort command and direct the output to files veg1a and veg2a respectively. |
| **1** |  | Step 9. Run the comm command on the two sorted veg?a files and record the results as a screen capture or demo to instructor |
| **1** |  | Step 10. Now use the diff command on the two sorted veg-a files and record the results |
| **1** |  | Step 11. Use the grep command to search in the verbiage.txt file for the string “linux”. Why did it not find any lines?  **The grep command did not find “linux” in the verbiage.txt file because it is case sensitive.** |
| **1** |  | Step 12. Modify the grep command to find all occurrences of “linux” regardless of case. List the modified command here:  **In order to find the specified text, disregarding the casing, we’d have to use grep -i <specified text> <file name>.**  **Example: grep -i “linux” verbiage.txt** |
| **1** |  | Step 13. Use the grep command on the file veg1 to display all lines that do not contain the word “Potato”. List the command below. |
| **Totals** |  |  |
| **18** |  |  |

|  |  |  |
| --- | --- | --- |
|  |  | **Part 2 –your first script** |
| **2** |  | Step 14. Create a new subfolder and repeat steps 1-4 to set up a fresh environment for your first script.  **CREATING A SUBFOLDER:**      **REPEATING STEPS 1-4**  **STEP1:**    **STEP2:**    **STEP 3:**    **STEP 4:** |
| **8** |  | Step 15. Write a bash script to perform steps 5 through 13 of part one of this assignment from a single script command of your own creation and submit the script listing with the assignment. Ensure you include comments in the script file describing what the script does. Demo the script by using the tee command to route output to a file called results.log while seeing the output on the screen at the same time. |
| **Totals** |  |  |
| **10** |  |  |

|  |  |  |
| --- | --- | --- |
|  |  | **Part 3 -awk** |
|  | Work through the online awk tutorial: [https://www.tutorialspoint.com //awk/](https://www.tutorialspoint.com/awk/)  Stop after the module on Built in Functions | |
| **2** | Using an editor of your choice, create a new file called coins.txt with the following column format: **(metal date-minted country-of-origin description(multi-column)) – Note the field separation by spaces. – This line does not go in the file!**  The actual file contents should be:  **gold 1986 USA American Eagle**  **gold 1908 Austria Franz Josef 100 Korona**  **gold 2003 Austria Philharmonic 100 euro**  **gold 1979 RSA Krugerrand**  **gold 1981 RSA Krugerrand**  **gold 1986 China Panda**  **silver 1986 USA Liberty dollar**  **gold 1986 USA Liberty 5 dollar piece**  **silver 1986 USA Liberty 50-cent piece**  **silver 1987 USA Constitution dollar**  **gold 1987 USA Constitution 5 dollar piece**  **gold 1988 Canada Maple Leaf**  **gold 1874 Norway 20 Crown Kroner**  **gold 2012 Canada Queen's Jubilee 300 dollar**  **gold 1925 Switzerland Vreneli 100 mark piece**  **gold 2013 UK Britannia**  **silver 2012 UK Britannia-silver** | |
| **1** |  | invoke awk to list all the gold pieces as follows:  **awk '/gold/' coins.txt**  Record the results on the back or in a separate screen capture |
| **1** |  | Now use awk to show only the description field of the file entries by typing:  **awk '/gold/ {print $4,$5,$6,$7}' coins.txt** |
| **1** |  | Analyze the command shown above and attempt to decipher what it did. Record the results on the back or in a separate screen capture.  **The command instructed AWK to print the columns 4,5,6,7, which are the description, from the input field** |
| **2** |  | Determine the awk command that will count the number of types of coins in the collection and print the total. Record the results on the back or in a separate screen capture.  **The AWK command that will count the number of types of coins in the collection and print the total is:**  **awk ‘/a/{++cnt} END {print “Total = “, cnt}’ coins.txt** |
| **Totals** |  |  |
| **7** |  |  |

0 - Assignment not submitted or work not original.