CSc 3320: Systems Programming

Spring 2021

Homework

2: Total points 100

Submission instructions:

1. Create a Google doc for each homework assignment submission.
2. Start your responses from page 2 of the document and copy these instructions on page 1.
3. Fill in your name, campus ID and panther # in the fields provided. If this information is missing in your document TWO POINTS WILL BE DEDUCTED per submission.
4. Keep this page 1 intact on all your submissions. If this *submissions instructions* page is missing in your submission TWO POINTS WILL BE DEDUCTED per submission.
5. Each homework will typically have 2-3 PARTS, where each PART focuses on specific topic(s).
6. Start your responses to each PART on a new page.
7. If you are being asked to write code copy the code into a separate txt file and submit that as well.
8. If you are being asked to test code or run specific commands or scripts, provide the evidence of your outputs through a screenshot and copy the same into the document.
9. Upon completion, download a .PDF version of the document and submit the same.

Full Name: Thach Huy Pham

Campus ID: tpham84

Panther #: 002416419

PART 1 (2.5 points each): 10pts

1. What are the differences among ***grep***, ***egrep*** *and* ***fgrep***? Describe using an example.
   -These commands are used for pattern matching. ***grep*** is used for basic regular expression to state what needed to be matched, whereas ***egrep*** is used for extended regular expression, which includes special character like | ? + (). On the other hand, ***fgrep*** is used when you want to match a fixed string rather than a regular expression.

2. Which utility can be used to compress and decompress files? And how to compress multiple files into a single file? Please provide one example for it.
   -The ***tar*** command can be used to compress and decompress files. For example:
   tar -cvf hw2tar file1 file2
   The line above compresses file1 and file2 into hw2tar file. You can also compress all the files in a directory by providing the directory's absolute path name.

3. Which utility (or utilities) can break a line into multiple fields by defining a separator? What is the default separator? How to define a separator manually in the command line? Please provide one example for defining the separator for each utility.
   -The ***awk*** command can do so. The default separator is a tab/space. When you need to change the separator, you must type ***awk -F*** followed by what you want to be the separator. For example:
   awk -F: '{print $2}' example1
   The command above will print the second field in example1 file for which each field is now separated by a ':'

4. What does the ***sort*** command do? What are the different possible fields? Explain using an example.

   -The ***sort*** command is used to sort the data in a file in some order defined by the command. Some of the fields are:

   sort -0 to write the output to an output file

   sort -r to sort in reverse

   sort -n to sort in numerical order

   sort -nr to sort numerical order in reverse.

   For example we have a file containing:

   apple
   banana
   zebra
   dog
   cat
   tiger
   monkey

   By using ***sort -r*** we will have:

   zebra
   tiger
   monkey
   dog
   cat
   banana
   apple

Part IIa (5 points each): 25pts

5. What is the output of the following sequence of bash commands: **echo 'Hello World' | sed 's/$/!!!/g'**

   Hello World!!!

6. What is the output for each of these awk script commands?

   -- 1 <= NF { print $5 } This will print out all the values of the 5th column
   -- NR >= 1 && NR >= 5 { print $1 } If NR displays line number, that is greater than 5, we print first column
   -- 1,5 { print $0 } Print all values of 1-5.
   -- {print $1 } Print value in the first column.

7. What is the output of following command line:
   echo good | sed '/Good/d'
   The output is good because the sed command only looks for 'Good' to delete.

8. Which **awk** script outputs all the lines where a plus sign + appears at the end of line?
   /\+$/{print $0}

9. What is the command to delete only the first 5 lines in a file "foo"? Which command deletes only the last 5 lines?
   -Delete first 5 lines: ***sed '1,5d' foo***
   -Delete last 5 lines: ***sed '(($(wc -l < foo)-5+1)),$ d' foo***

Part IIb (10pts each): 50pts

Describe the function (5pts) and output (5pts) of the following commands.

9. **$ cat float**

Wish I was floating in blue across the sky, my imagination is
strong, And I often visit the days
When everything seemed so clear.
Now I wonder what I'm doing here at all...

$ cat h1.awk

```
NR>2 && NR<4{print NR ":" $0
```

$ awk '/.*ing/ {print NR ":" $1}' float

*OUTPUT:
1:Wish
3:When
4:Now

*FUNCTION:
-The ***.*ing*** is a regular expression that matches any word of any length ending with ***ing***. So this ***awk*** command will find lines containing words that match this pattern. After the line is found, the first column will be printed out along with the line number and a colon before it.

10. As the next command following question 9,

$ awk -f h1.awk float

*OUTPUT:
3:When everything seemed so clear

*FUNCTION:
-The command above will print the file ***float*** with the condition in ***h1.awk.*** It means that the ***awk*** command will print the line with line numbers greater than 2 and less than 4 (which is 3), along with the line number and a colon in the front.

11.

$ **cat h2.awk**

```
                    "Start to scan file" }
```

BEGIN { print

{print $1 "," $NF}

END {print "END-" , FILENAME }

$ awk -f h2.awk float

*OUTPUT:
Start to scan file
Wish, strong,
And, days
When, clear
Now, all…
END- float

*FUNCTION:
-The command starts by printing 'Start to scan file' at the beginning. After that, it will print out the first field, followed by a comma and then the last field (NF). After it was done with scanning the file, it will print "END-" and then the file name (Because the variable FILENAME)

12. sed 's/\s/\t/g' float

*OUTPUT:

Wish	I	was	floating	in	blue	across	the
sky,	my	imagination	is	strong,			
And	I	often	visit	the	days		
When	everything	seemed	so	clear.			
Now	I	wonder	what	I'm	doing	here	at all..

*FUNCTION:
-The command finds and replaces all the space with tab (\s for \t) for the entire ***float*** file.

13.

$ ls *.awk| awk '{print "grep --color 'BEGIN' " $1 }' |sh *(Notes: **sh file** runs file as a shell script . $1 should be the output of* ' ls *.awk ' in this case, not the 1st field *)*

*OUTPUT
BEGIN {print "Start to scan file"}

*FUNCTION

The ***ls*** command lists all the files ending with ***.awk*** The result is piped into the second command, where it prints ***grep --color 'BEGIN' $1*** followed by the result from ***ls.*** The result in the ***awk*** command will be piped to ***sh*** and be executed as a shell script. So it will look for the word 'BEGIN' in h1.awk and h2.awk, but 'BEGIN' only exists in hw2.awk so it will print that line with the word 'BEGIN' colored.

14.

$ mkdir test test/test1 test/test2

$cat>test/testt.txt

This is a test file ^D

$ cd test

$ ls -l . | grep '^d' | awk '{print "cp -r " $NF " " $NF ".bak"}' | sh

*OUTPUT:

None

*FUNCTION:

The instruction will not have any output on screen. The first instruction will list all the files with the file detail. The output will be piped to the second command, where it looks for files beginning with 'd', which are directories. The result is piped into the third command. $NF means the last field, which is the file name. So we will have these following commands:

cp -r test1 test1.bak

cp -r test2 test2.bak

The file is piped to the second command, which is ***sh*** command to interpret the commands above as shell script. These commands will only copy the directory, so there is no output.

Part III Programming: 15pts

15. Sort all the files in your class working directory (or your home directory) as per the following requirements:

a. A copy of each file in that folder must be made. Append the string "_copy" to the name of the file
b. The duplicate (copied) files must be in separate directories with each directory specifying the type of the file (e.g. txt files in directory named txtfiles, pdf files in directory named pdffiles etc).
c. The files in each directory must be sorted in chronological order of months.
d. An archive file (.tar) of each directory must be made. The .tar files must be sorted by name in ascending order.
e. An archive file of all the .tar archive files must be made and be available in your home directory.

As an output, show your screen shots for each step or a single screenshot that will cover the outputs from all the steps.