**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: Tracy Michaels

Campus ID: tmichaels1

Panther #: 002430918

**PART 1 (2.5 points each): 10pts**

1. What are the differences among ***grep****,* ***egrep*** *and* ***fgrep***? Describe using an example.
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.
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.
4. What does the ***sort*** command do? What are the different possible fields? Explain using an example.

**Part IIa (5 points each): 25pts**

1. What is the output of the following sequence of bash commands: **echo 'Hello World' | sed 's/$/!!!/g'**
2. What is the output for each of these awk script commands?

-- 1 <= NF { print $5 }

-- NR >= 1 && NR >= 5 { print $1 }

-- 1,5 { print $0 }

-- {print $1 }

1. What is the output of following command line:

**echo good | sed** **'/Good/d'**

1. Which **awk** script outputs all the lines where a plus sign + appears at the end of line?
2. What is the command to delete only the first 5 lines in a file "foo"? Which command deletes only the last 5 lines?

**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**

**10.** As the next command following question 9,

**$ awk -f h1.awk float**

**11.**

|  |  |  |
| --- | --- | --- |
| $ **cat h2.awk** | | "Start to scan file" } |
| BEGIN { print | |
| {print $1 | "," | $NF} |
| END {print | "END-" , FILENAME } | |

* **awk -f h2.awk float**

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

**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 *)*

**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

**Part III Programming: 15pts**

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

1. A copy of each file in that folder must be made. Append the string “\_copy” to the name of the file
2. 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).
3. The files in each directory must be sorted in chronological order of months.
4. An archive file (.tar) of each directory must be made. The .tar files must be sorted by name in ascending order.
5. 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.

**PART 1 (2.5 points each): 10pts**

1.

* grep evaluates basic regex patterns when searching for matches
* egrep evaluates extended regex patterns when searching for matches
* fgrep evaluates a fixed string as the pattern when searching for matches

2.

* tar, cpio, dump
* tar -cvf archiveFile.tar fileToArchive1 fileToArchive2… etc

3.

* awk, sort
* default separator is spaces and tabs
* to specify separator use -F option for awk and -t for sort
* -F: this uses ‘:’ as a separator
* -t: uses ‘:’ as separator

4.

* Sorts a file in ascending or decending order based on one or more fields that can be specified using options for the sort command
* -t defines the separator for fields, default is space/tabs
* After definig or using default separators you can define which field to sort by giving the command the index starting from 0 on the left of where that field is located, and specify in which way you want the command to sort by using one of the following options
  + -r reverse order
  + -b ignores leading space
  + -f ignores case
  + -M sorts by month order
  + -n sorts numerically
* sort +0 -2 -rbf fileToSort.txt
  + this command sorts the file fileToSort.txt based on the first 2 fields (fields 0 and 1). It will sort in reversed order ignoring case and leading blank space

**Part IIa (5 points each): 25pts**

5.

* This command will append three ! characters to the end of the string ‘Hello World’ and print the result in the terminal
* Hello World!!!

6.

-- 1 <= NF { print $5 }

* + This will print the 5th field of the current line if 1 is less than or equal to the Number of Fields in the current line

-- NR >= 1 && NR >= 5 { print $1 }

* + If the line number is greater than or equal to 1 and the line number is greater than or equal to 5, print the first field

-- 1,5 { print $0 }

* + Print lines 1 through 5

-- {print $1 }

* + Print field one

7.

* good

8.

* awk ‘/\+$/ { print $0 }’ fileName

9.

* sed ‘1,5 d’ foo
* tac foo | sed “1,5 d” | tac

**Part IIb (10pts each): 50pts**

10.

* searches for lines with words ending in ‘ing’ then prints the line number assoicated with a match followed by a colon then the first field of that line
* 1:Wish

3:When

4:Now

11**.**

* if the current line number is greater than 2 and less than 4 it prints the line number followed by a colon followed by the line
* 3:When everything seemed so clear.

12.

* First prints “Start to scan file” followed by printing the first word in each line followed by a comma followed by the last field in that line. Then lastly it prints “END- “ followed by the name of the file
* Start to scan file

Wish,is

Strong,,days

When,clear.

Now,all…

13**.**

* Does a global substitution replacing spaces with tabs
* 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...

14**.**

* makes a list of all files that end in .awk and passes that list to an awk command. The awk command prints the output of a grep command that searches for the word “BEGIN” in $1 which in this case is the first field of each line in the list created by the ls command which would be the files that are passed from the ls and would now be the file name argument for the grep command. The output is then piped to sh which as stated in the prompt means that it is run as a shell script. Grep would find nothing in h1.awk but it would in h2.awk and would then print the line containing the matching pattern with the matched word in red due to the --color flag
* BEGIN { print “Start to scan file” }

15**.**

* makes directory tree with test as parent directory to subdirectories test1 and test2
* creates a testt.txt file with “This is a test file ^D” in it
* move into test directory
* make a long list of all files and directories in current directory and pass output into a grep command that searches for the character ‘d’ at the beginning of each line, so in this case it will only return directories since they have a ‘d’ character in the very beginning of a long listing denoting that they are directories preceding their permissions. The output of that is then passed to an awk command that “prints” (though not to console) a command that creates a copy of the directory with a ‘.bak’ appended to the end of it. This is then piped into sh which will run the inputs (in this case the commands that were created by the awk command) as actual commands that create copies of only the directories and appends .bak to the name
* no output is printed to console, but if you were to use the ls command again you would see the new directories.

**Part III Programming: 15pts**

