**CSc 3320: Systems Programming**

Fall 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: Iven Souffrant

Campus ID: isouffrant1

Panther #: 002-39-7180

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

1. What are the differences among ***grep****,* ***egrep*** *and* ***fgrep***? Describe using an example.

Grep can search for any type of string on any file or list of files or even output of any command. Ex. grep string file\_name

Egrep is efficient and fast when it comes to searching for a regular expression pattern as it treats meta-characters as is and doesn’t substitute them as strings like in grep. Ex. egrep debian file.txt

Fgrep is fast when it comes to search for the entire string instead of regular expression as it doesn’t recognize the regular expressions, neither any meta-characters. Ex. fgrep “learni\ng” para

1. 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.

Tar or Zip. Ex. tar -czvf archive..gz /path/to/file

1. 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.

Awk. ’\t’. You can define a field separator by using the "-F" switch under the command line or within two brackets with “FS=” . Ex. awk '{ print $2 }' FS='\t' infile.

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

The sort command sorts the contents of a file, in numeric or alphabetic order, and prints the results to standard output. Sort options: -b,-r,-o,-n,-M,-u,-k,-t.

Ex.

01 Priya

04 Shreya

03 Tuhina

02 Tushar

$ sort -r file1.txt

04 Shreya

03 Tuhina

02 Tushar

01 Priya

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

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

Hello World!!!

1. 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. -- 1 <= NF { print $5 } # prints out all values of 5th column.
2. NR displays line number, that is greater than 5, we are printing first column.
3. For all values in file simply print file file contents.
4. It simply prints all values of first column.
5. What is the output of the following command line:

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

good

1. Which **awk** script outputs all the lines where a plus sign + appears at the end of line?

/\+$/{print $0}

1. What is the command to delete only the first 5 lines in a file "foo"? Which command deletes only the last 5 lines?

sed ‘1,5d’ file

**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**

Function: NR means the row number so the command is looking at row 3 in the file. Then it will print row number ‘3’ with the ‘:’ and with the statement on row 3.

The second command search each row that has an ‘ing’ in it and print the row number ‘1’ with a colon ‘:’ and first word on that line.

Output:

3:When everything seemed so clear.

1:Wish

3:When

4:Now

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

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

Reads the awk program from the program file and apply it to the input file contents of h1.awk.

**11.**

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

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

Function: The command prints the first word of each line with a ‘,’ after then prints the last word of each line and prints the input file name at the end.

Output:

Wish,strong,

And,days

When,clear.

Now,all…

END- float

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

Function: The command with sed will substitute (s) spaces with tab (\t) of the whole file (/g).

Output:

Wish I floating in blue across the sky,my imagination is strong, (part of line 1)

And I often visit the days

When everything seemed so clear.

Now I wonder what I’m doing here at all...

**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 *)*

Function: The command has 3 parts in it since it is separated by ‘|’. (ls \*.awk) lists all the files with extensions .awk, (awk ‘(print “grep --color ‘BEGIN’”$1)’) will go through the file and will on show lines with the word ‘BEGIN’ on it colored with the rest of the other words on that line as well, (sh) will make the file a shell script.

Output:

BEGIN { print “Start to scan file”}

**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

Function: First it will make a directory called ‘test’ with two subdirectory called ‘test1’ and ‘test2’. The cat command will create a text file called ‘testt.txt’ in the test directory. The cd command will go inside the test directory. The last command will create the backup file with the .bak extension to all the directory and the subdirectory and will print cp -r then a space then $NF which is 'test1' then again a white space then again 'test1' is appended with .bak.

Output:

drwxrwxr-x 2 isouffrant1 isouffrant1 4096 Sept 20 13:25 test1

drwxrwxr-x 2 isouffrant1 isouffrant1 4096 Sept 20 13:25 test1.bak

drwxrwxr-x 2 isouffrant1 isouffrant1 4096 Sept 20 13:25 test2

drwxrwxr-x 2 isouffrant1 isouffrant1 4096 Sept 20 13:25 test2.bak

**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.

I do not have any pdf files in my user directory so I will be showing the text files only :)



