**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: Adam Nguyen

Campus ID: anguyen117

Panther #: 900911012

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

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

grep is Global Regular Expression Print. Grep is the main search program on Unix that can search for any type of string on any file, list of files, or outputs of any command.

It uses basic regular expressions.

For example, $grep --color -n ‘sw.\*ng’ grepfile displays the lines containing the pattern /sw.\*ng/. “And you will be swimming there too.”

egrep or grep -E is the extended grep. It acts like grep except it also uses extended regular expression. It treats meta-characters as is and doesn’t substitute them as strings like in grep.

For example, s.\*w matches the sentence “Oh you sleepy young heads dream of wonderful things” with the red letters in this case.

fgrep or grep -F is the Fixed grep. Fgrep searches for complete string and doesn’t even recognize special characters as part of regular expression. fgrep is useful when you need to search for strings which contain lots of regular expression metacharacters.

For example, fgrep "support" myfile.txt finds the string “support” in myfile.txt.

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.

Taris is a utility that can be used to compress and decompress files.

tar -cvf tarfilename filelist.

tar -cvf ch6.tar ch6

ch6/

ch6/menu.csh

ch6/junk/

Is made into ch6.tar and it shows the three above content when viewing it by tar -tvf ch6.tar

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 and sort are the utilities that can break a line into multiple fields by defining a separator.

The default separator is tab or space.

In order to define a separator manually in the command line, do either awk -F or sort -t with a separator afterwards.

$ awk -F: '{ print NF, $1 }' /etc/passwd where -F: uses colon ‘:’ as the field separator, built-in variable, NF means number of fields, and $1 refers to filed 1. It prints the number of fields and first field in

the /etc/passwd file.

sort -tc -r {sortField -bfMn}\* {fileName}\* where -tc separator is c instead of blank. e.g. -t:

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

Sort command a file in ascending or descending order based

on one or more fields. It depends on one of the various sort commands.

**$ Cat test.txt**

**Apple**

**Cabbage**

**Banana**

**sort test.txt**

**Apple**

**Banana**

**Cabbage**

**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 <= NF { print $5 }

Prints out all values of 5th column for the number of fields greater than 1.

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

For this one, NR displays line number that is greater than or equal to 5. The first column is printed from row numbers greater than or equal to 5.

-- 1,5 { print $0 }

For all values in file, print file contents.

-- {print $1 }

It simply prints all values of the first column.

1. What is the output of 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' foo

sed '$(( $(wc -l <foo)-5 )),$ 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**

1:Wish

3:When

4:Now

The command prints the row number followed by the first item for any matched line that includes the pattern \*.ing where \* matches 0 or more characters before “.”

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

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

3:When everything seemed so clear.

Awk reads the awk program from h1.awk to be applied to float file.

NR means Row Number and for NR > 2 and NR < 4, the only Row Number available is 3 to meet both parameters. {print NR “:” $0} means to print the current row number (3) and current row {0} with : after Row Number.

**11.**

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

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

Start to scan file

Wish,strong,

And,days

When,clear.

Now,all…

END- float

The starting statement is “Start to scan file”, and then it prints from first line/sentence/field to last line/sentence/field with the first item and last item of each line/sentence/field. Each first and last item is separated with a comma (,). FILENAME prints the input file name (in this case, float).

**12. sed 's/\s/\t/g' 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…

**S stands for substitution and /g applies to the entire file.**

**Sed substitutes spaces (\s) encountered with Tab characters (\t) of the entire 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 *)*

**BEGIN** { print “Start to scan file”}

| means pipe symbol where the left side’s output is inputted into the right side’s input.

ls \*.awk list all the file with awk extension.

Therefore awk '{print "grep --color 'BEGIN' " $1 }' takes awk files as input and searches for matching word like ‘BEGIN’ and colors it and prints the current line and prints the output on the shell.

**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

Does not print anything?

It copies all the directories in test (the current directory) and add .bak to the copies.

**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.







