**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: Mohammad Osama

Campus ID: mosama1

Panther #: 002-61-6232

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

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

Ans) Filter out all lines that do not contain a specified pattern (i.e. output all lines containing a specified pattern) grep stands for global regular expression, egrep stands for extend global regular expression, fgrep stands for fixed global regular expression.

Assume we have a file called grepfile

$cat -n grepfile

1: Well you know it’s your bedtime,

2: So turn off the light,

3: Say all your prayers and then,

4: Oh you sleepy young heads dream of wonderful things,

5: Beautiful mermaids will swim through the sea,

6: And you will be swimming there too.  
grep -inw ‘pattern’ {fileName}\* basic regular expression  
  
grep example: Display the lines containing the pattern /sw.\*ng/

$grep --color -n ‘sw.\*ng’ grepfile

6: And you will be swimming there too.  
  
grep example 2: Display the lines containing the pattern /a./

$grep --color -n ‘a.’ grepfile  
3: Say all your prayers and then,

4: Oh you sleepy young heads dream of wonderful things,

5: Beautiful mermaids will swim through the sea,

egrep -inw ‘pattern’ {fileName}\* extended regular expression

egrep example: display the lines containing the pattern /sw.\*ng/

$egrep –color -n ‘sw.\*ng’ grepfile

6: And you will be swimming there too.

egrep example 2: display the line containing the pattern /s.+w/

$egrep –color -n ‘s.+w’ grepfile

4: Oh you sleepy young heads dream of wonderful things,

5: Beautiful mermaids will swim through the sea,

fgrep -inw ‘pattern’ {fileName}\* fixed string

fgrep example: display the line containing the fixed-character string separated by a line /will/

$fgrep –color -n ‘will’ grepfile

And you will be swimming there too.

Beautiful mermaids will swim through the sea,

Example 2:

]$ fgrep --color 'l' -n grepfile.txt

1: Well you know it’s your bedtime,

2: So turn off the light,

3: Say all your prayers and then,

4: Oh you sleepy young heads dream of wonderful things,

5: Beautiful mermaids will swim through the sea,

6: And you will be swimming there too.

-i : ignore case

-n : display line numbers

-w :matches only whole words only

--color: highlight the matched string

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.

Ans2) tar creates the archives and extract the archives too.

**Modes:**

* 1. Create a “tape archive” format file from the file list.
     1. Tar -cvf tarFileName fileList
  2. Extract files from a “tape archive” format file to current directory
     1. Tar -xvf tarFileName fileList
  3. Show the content of a “tape archive” format file
     1. Tar -tvf tarfileName

f- enables you to give a tar file name   
 Default name is /dev/rmto

v- verbose

z- compressed file gzip

j- highly compressed file

Example – regarding tar   
  
$tar -cvf ch6.tar ch6

ch6/

ch6/menu.csh

ch6/junk/

ch6/junk/junk.csh

ch6/junk.csh

ch6/menu2.csh

ch6/multi.csh

ch6/expr1.csh

ch6/expr3.csh

ch6/expr4.csh

ch6/if.csh

ch6/menu3.csh

$tar -tvf ch6.tar

drwxr-xr-x raj/raj 0 2007-06-03 09:57 ch6/

-rwxr-xr-x raj/raj 403 2007-06-02 14:50 ch6/menu.csh

drwxr-xr-x raj/raj 0 2007-06-03 09:57 ch6/junk/

-rwxr-xr-x raj/raj 1475 2007-06-03 09:57 ch6/junk/junk.csh

-rwxr-xr-x raj/raj 1475 2007-06-03 09:56 ch6/junk.csh

-rw-r--r-- raj/raj 744 2007-06-02 15:59 ch6/menu2.csh

-rwxr-xr-x raj/raj 445 2007-06-02 15:26 ch6/multi.csh

-rwxr-xr-x raj/raj 279 2007-06-02 15:18 ch6/expr1.csh

-rwxr-xr-x raj/raj 98 2007-06-02 15:20 ch6/expr3.csh

-rwxr-xr-x raj/raj 262 2007-06-02 15:21 ch6/expr4.csh

-rwxr-xr-x raj/raj 204 2007-06-02 15:22 ch6/if.csh

-rw-r--r-- raj/raj 744 2007-06-02 16:01 ch6/menu3.csh

-rw-rw-r-- raj/raj 29 2007-06-21 11:06 date.txt

$tar -xvf ch6.tar

ch6/

ch6/menu.csh

ch6/junk/

ch6/junk/junk.csh

ch6/junk.csh

ch6/menu2.csh

ch6/multi.csh

ch6/expr1.csh

ch6/expr3.csh

ch6/expr4.csh

ch6/if.csh

ch6/menu3.csh

date.txt

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.

* Ans3) awk and sort.
* The default separator is whitespace or tab.
* To define separator manually for awk, you can do awk-F followed by the separator you want. You can use OFS (output field separators), FS (Field separator) , RS(Record Separator)
* To define a separator manually for sort you can do sort -t, followed by the separator.

Example 3: awk -F ":" '/1/ {print $1}'

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

Ans) sort a file in ascending or descending order based on one or more fields. Individual fields are ordered lexicographically, which means that corresponding characters are compared based on their ASCII value.

$sort -tc -r {sortField -bfMn}\* {fileName}\*

-tc separator is c instead of blank e.g. -t:

-r descending instead of ascending.

sortField  : **+POS1 [-POS2]** key positions start [up to end]

-b   ignore leading blanks

-f    ignore case

**-M** month sort (3 letter month abbreviation)‏

**-n** numeric sort

**$ cat sort.dat**

John Smith 1222 20 Apr 1956

Tony Jones 1012 20 Mar 1950

John Duncan 1111 20 Jan 1966

Larry Jones 1223 20 Dec 1946

**$ sort +4  -5  -M  sort.dat**

John Duncan 1111 20 Jan 1966

Tony Jones 1012 20 Mar 1950

John Smith 1222 20 Apr 1956

Larry Jones 1223 20 Dec 1946

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

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

Ans) Hello World!!!

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

-- 1 <= NF { print $5 }

Ans)First command simply prints out 5th column, where we are specifying NF to print all row values in 5th column if present, NF is used to display last field.

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

**Ans)NR displays line number, that is greater than 5, we are printing first column.**

-- 1,5 { print $0 }

Ans) **For all values in file simply print file contents.**

-- {print $1 }

Ans) **It simply prints all values of first column.**

1. What is the output of the following command line:

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

Ans) good

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

Ans) /\+$/{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?

Ans) sed -i "$(($(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**

**Ans9)   
Output:**   
1: Wish

3:When

4: Now   
  
**Explaination:** awk [condition] {action} filename

So basically the condition is lines having /.\*ing/ which means . any word that starts from anything but ends with ing then the next is the action print NR, NR meaning Current line #. {print NR “:” $1} “:” means have this between the Current line # and the answer $1 means the first field in the file. If you do $5 and then it will show the fifth word. In this case we do $1 that means show the first word or field as they say. First item of the matching line

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

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

**Ans10)   
Output:**3: When everything seemed so clear.

**Explanation:**

This command meant awk -f programfile inputfile

Awk reads the awk program from program file to be applied to the input file.

Contents of h1.awk:  
NR>2 && NR<4{print NR “:” $0)

NR means stands for row number or Current line# (NR>2 && NR<4) This command means Current line greater than 2 and Current line less than 4 meant. It could only choose line 3 as that was the condition. Action was {printNR “:” $0} print the current line # and $0 signifies the entire record.

**11.**

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

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

**Ans11) Output:**Start to scan file

Wish,strong,

And,days

When,clear.

Now,all...

END- float

**Explanation:**

awk -f programfile inputfile

awk reads the awk program from program file to be applied to the input file.

print ($1 “,” $NF), $1 prints the first item of each and every line followed by ,(separator) and $NF means Number of fields. In this context means NF last item of each and every line. So it prints the first word with “,” in the middle and in the end, the last word.

**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...

**Explanation:**

Sed ‘s/old/new/g’ filename

Sed 's/\s/\t/g' float

Where first ‘s’ stands for substituting. The ‘\s’ means white spaces(old) then / towards new which is \t(tab characters) for the entire file and then finally /g stands for global. Also the filename in the end.

**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"}

**Explanation:**In the above statement | represents pipe symbol

x|y|z where the output from x is fed as the input to y and output from y is fed as input to z

ls -\*.awk – List all the files with awk extension. Output from this went as input to

awk ‘{print “grep –color ‘BEGIN’ “$1}’ – It takes all awk files as input and searches for matching ‘BEGIN’ and colors it and will print the line as it finds it. This goes as input to SH

Sh- is a command interpreter that takes your input , provides output back to the screen. Hence we can see the line as output.

**14.**

$ mkdir test test/test1 test/test2

$cat>test/test.txt This is a test file ^D

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

**Output:**

It creates back for the specified directories Text

Description automatically generated

**Explanation:**ls -l lists long files from this folder then this output goes as input then grep ‘^d’ searches for ^ in the start d which is directories then this output goes as input to awk command which cp all the readable files number of fields and in the end put .bak as extension to show that this is as a back up file. Then this ouput is put through Sh to be properly interpreted and executed by the proper shell script.

**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).

Copied and seperated them into 2 different directories. Calendar

Description automatically generated with medium confidence

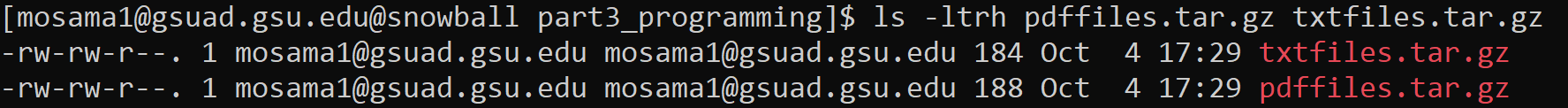
1. The files in each directory must be sorted in chronological order of months.

Used sort -M to make sure of that.

1. An archive file (.tar) of each directory must be made. The .tar files must be sorted by name in ascending order.

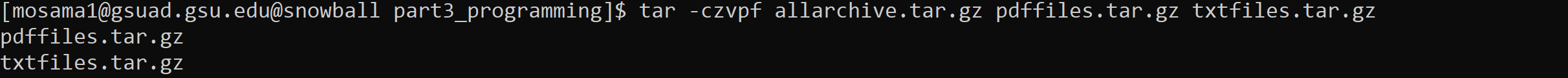
A screenshot of a computer

Description automatically generated with medium confidence



1. An archive file of all the .tar archive files must be made and be available in your home directory.

Archiving all



Copying it to the 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.

Text

Description automatically generated