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: Ramey Serdah

Campus ID: rserdah1

Panther #: 002 48 9675

PART 1 (2.5 points each): 10pts

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

   grep searches for a basic regular expression, egrep searches for an extended regular expression, and fgrep searches for a fixed string and disregards any special characters of regex. For example, egrep “(abc)” would search for any instances of “abc”, fgrep “(abc)” would search for any instances of “(abc)”, and grep “(abc)” would also search for “(abc)” (because parentheses are not used in basic regex).

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 utility can compress and decompress files. You can compress multiple files into a single file by using

   ```
   tar -cvf tarFileName compressFile1
   compressFile2 compressFile3
   ```

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 utility can break a line into multiple fields using a separator. The default separator is a space. You can manually define a separator, for example to a comma, by using `awk 'BEGIN { FS="," } ; { print $2 }' fileName`.

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

   The sort command arranges lines based on a certain filter. The different possibilities for sort are numerically, reverse numerically, alphabetically, and reverse alphabetically. You can sort a file numerically by using `sort -g fileName` or you can sort reverse numerically by using `sort -gr fileName`. Using sort without any options sorts according to ASCII value.

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 } The fifth field of each line if the line has at least one field.
   -- NR >= 1 && NR >= 5 { print $1 } The first field if the current record is greater than or equal to 1 and greater than or equal to 5
   -- 1,5 { print $0 } This prints each line
   -- {print $1 } The first field of each line

7. What is the output of the following command line:
   echo good | sed '/Good/d'
   good
8. Which **awk** script outputs all the lines where a plus sign + appears at the end of line?
   ```
   awk '/\+$/{print $0}' fileName
   ```
9. 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.txt
   ```

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…
This prints the contents of a file named float. The output is what the file contains.

$ cat h1.awk
NR>2 && NR<4{print NR ":" $0
This command is printing the contents of an awk script named h1.awk. The script contains the options for an awk command to only find the third line of a file, print the line number, followed by a semicolon, and then print the whole third line.

$ awk '/.*ing/ {print NR ":" $1}' float
This prints the line number where any occurrence of "ing" was found, a semicolon and then prints the first field of that line (each field being separated by spaces by default). This command would output: `1:Wish`

```
3:When
4:Now
```

10. As the next command following question 9,
$ awk -f h1.awk float
This command uses the awk script from above to print the third line number, semicolon, and then the whole third line from a file named float.
This command outputs:

```
3:When everything seemed so clear.
```

11.
$ **cat h2.awk**
BEGIN { print

"Start to scan file" }
{print $1 "," $NF}

END {print "END-" , FILENAME }
This prints the contents of an awk script named h2.awk. This script contains the options for an awk command to print "Start to scan file" followed by the first field of every line, then a comma, then the last field in that line. After the last line, it prints "END-" followed by the file name.

$ awk -f h2.awk float

This command calls awk using the options from the h2.awk script from above to print using the format from the h2.awk script on the file float. The output is:

```
 Start to scan file
Wish,strong,
And,days
When,clear.
Now,all...
END- float
```

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

This replaces all whitespaces with a tab in the float file. The output would be:

```
  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...
```

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 *)*

This uses ls to get all .awk scripts, pipes the result into awk, then pipes into sh command to run the scripts.

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
```

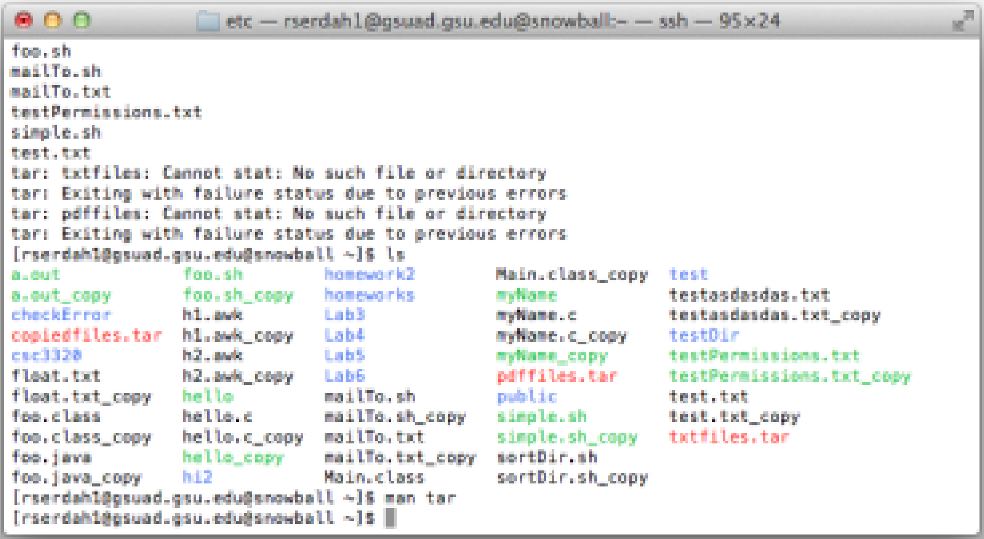
This uses ls to print the details of each file and directory, pipes it into grep to get each line with a directory (each line starting with "d"), pipes into awk to add command to copy each folder recursively and adding ".bak" to the folder name, then pipes into sh in order to run what awk printed as a command in order to actually copy.

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.


etc — rserdahl1@gsuad.gsu.edu@snowball:~ — ssh — 95×24

```
foo.sh
mailTo.sh
mailTo.txt
testPermissions.txt
simple.sh
test.txt
tar: txtfiles: Cannot stat: No such file or directory
tar: Exiting with failure status due to previous errors
tar: pdffiles: Cannot stat: No such file or directory
tar: Exiting with failure status due to previous errors
[rserdahl1@gsuad.gsu.edu@snowball ~]$ ls
a.out            foo.sh          homework2        Main.class_copy   test
a.out_copy       foo.sh_copy     homeworks        myName            testasdasdas.txt
checkError       h1.awk          Lab3             myName.c          testasdasdas.txt_copy
copiedfiles.tar  h1.awk_copy     Lab4             myName.c_copy     testDir
csc3320          h2.awk          Lab5             myName_copy       testPermissions.txt
float.txt        h2.awk_copy     Lab6             pdffiles.tar      testPermissions.txt_copy
float.txt_copy   hello           mailTo.sh        public            test.txt
foo.class        hello.c         mailTo.sh_copy   simple.sh         test.txt_copy
foo.class_copy   hello.c_copy    mailTo.txt       simple.sh_copy    txtfiles.tar
foo.java         hello_copy      mailTo.txt_copy  sortDir.sh
foo.java_copy    hi2             Main.class       sortDir.sh_copy
[rserdahl1@gsuad.gsu.edu@snowball ~]$ man tar
[rserdahl1@gsuad.gsu.edu@snowball ~]$
```

```
                                                                 ls -l
. | grep '^[^d]' | awk 'NF>3 {print "cp " $NF " " $NF"_copy"}' | sh ls -l
. | grep '(.txt_copy)$' | awk 'NF>3 {print "mv " $NF " " "./txtfiles"}' |
sh
ls -l . | grep '(.pdf_copy)$' | awk 'NF>3 {print "mv " $NF " "
"./pdffiles"}' | sh
ls -p --sort=time | grep -v /
tar -cf txtfiles.tar txtfiles
tar -cf pdffiles.tar pdffiles
tar -cf copiedfiles.tar txtfiles.tar pdffiles.tar
```