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.

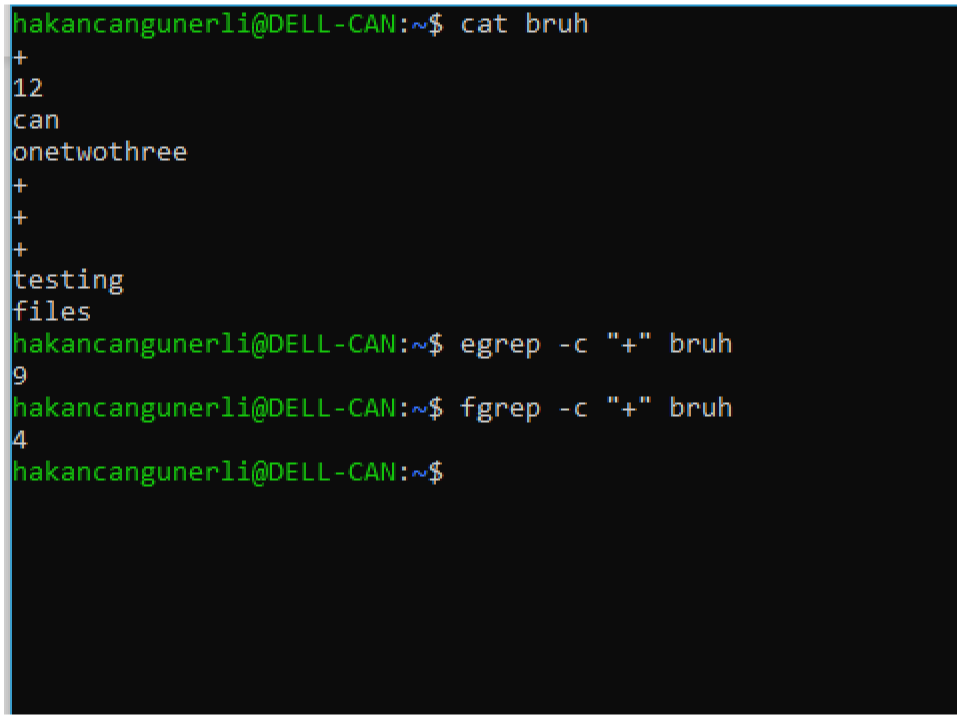
Full Name: Hakan Gunerli

Campus ID: hgunerli1

Panther #: 002504797

PART 1 (2.5 points each): 10pts

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



```
hakancangunerli@DELL-CAN:~$ cat bruh
+
12
can
onetwothree
+
+
+
testing
files
hakancangunerli@DELL-CAN:~$ egrep -c "+" bruh
9
hakancangunerli@DELL-CAN:~$ fgrep -c "+" bruh
4
hakancangunerli@DELL-CAN:~$
```

Grep will search for either regex or strings. Egrep is specialized in regex matching and fgrep is specialized in string matching.

egrep which is the same thing as grep -E which will allow you to accept certain characters such as + and? as metacharacters as supposed to be ignored by grep. It will do regex.

egrep "+" filename.txt

fgrep which is the same thing as grep -F does not recognize regex, it recognizes entire strings.

fgrep "+" filename.txt

thus in the example I have given the fgrep returns 4 because there are 4 plus signs whereas + regex does not mean anything so it represents the number of lines in the file we have.

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.

tar is the command for compressing/decompressing files. I suppose by SINGLE FILE, you mean SINGLE ARCHIVE FILE.

tar -cvf one_file.tar file_name_one file_name_two
for compression, we need
tar -xvf one_file.tar

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.

You can use awk or sed. The default separator is space for awk. With awk if you write awk -F ‘{statement $linerule} you can define a separator. Let’s say that we want : to be our separator, then we can set : as a line separator for each word of the line like;

awk -F: ‘{print $1}’

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

Sort will sort a text file's lines. There are various fields such as -n for numeric -R for random -r for reverse sorting.

sort file.txt

a

b

c

d

sort -r file.txt

d

c

b

a

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 word of each line.

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

After line 5, print the first word of each line.

-- 1,5 { print $0 }

This will print everything between lines 1 to 5.

-- {print $1 }

Print first word of each line.

7. What is the output of following command line:

   echo good | sed '/Good/d'

this will print the word good.

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

awk /\+$/{print $0}
I put the /\ because it shouldn’t be ignored.

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 -I ‘1,5d’ foo
for the first 5 lines in the file foo.

head -b -5 foo
for the last 5 lines.

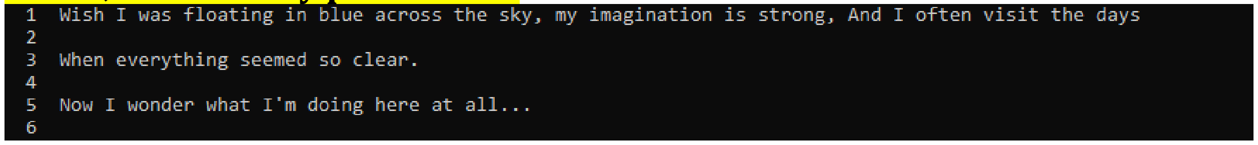
Part IIb (10pts each): 50pts

Describe the function (5pts) and output (5pts) of the following commands.

9. $ cat float

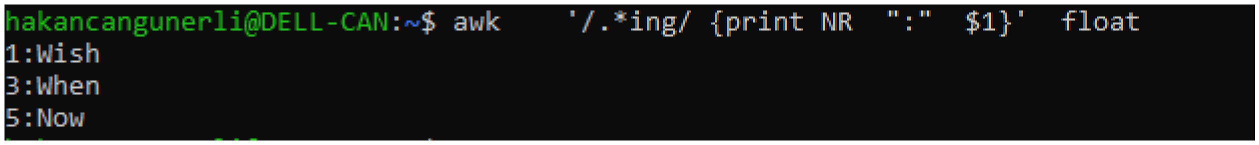
Wish I was floating in blue across the sky, my imagination is strong, And I often visit the days
When everything seemed so clear.
Now I wonder what I'm doing here at all...

$ cat h1.awk
NR>2 && NR<4{print NR ":" $0}
This is the structure of the file, which means that when we say NR>2 and NR< 4, we will only print line 3.



```
1  Wish I was floating in blue across the sky, my imagination is strong, And I often visit the days
2
3  When everything seemed so clear.
4
5  Now I wonder what I'm doing here at all...
6
```

$ **awk '/.*ing/ {print NR ":" $1}' float**

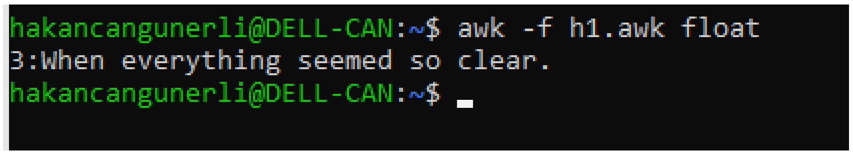


```
hakancangunerli@DELL-CAN:~$ awk    '/.*ing/ {print NR  ":"  $1}'  float
1:Wish
3:When
5:Now
```

This will print the first word of lines that have ing on the same line.

10. As the next command following question 9,
   $ **awk -f h1.awk float**

awk file separator, get data from both h1.awk and float and depending on the specifications of the h1.awk file it will print the info between record 2 and 4 not inclusive, meaning 3.



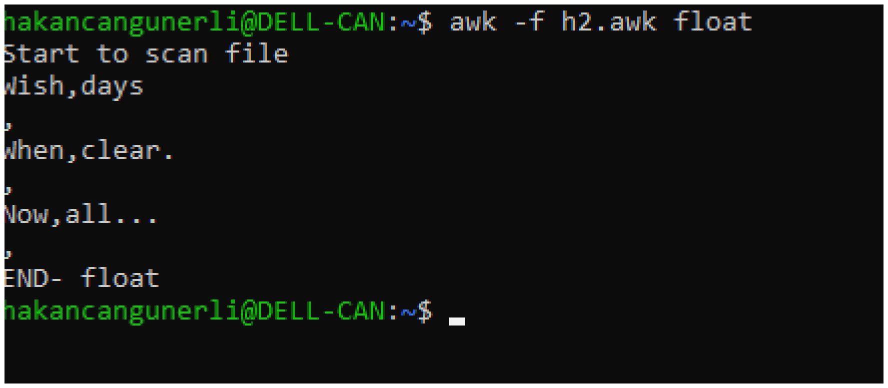
```
hakancangunerli@DELL-CAN:~$ awk -f h1.awk float
3:When everything seemed so clear.
hakancangunerli@DELL-CAN:~$
```

11.

```
$ cat h2.awk
BEGIN { print    "Start to scan file" }
{print  $1     ","  $NF}

END {print    "END-" , FILENAME }
$ awk -f h2.awk float
```

this will print "Start to scan file", and then print the endings and the starting words of the file float.

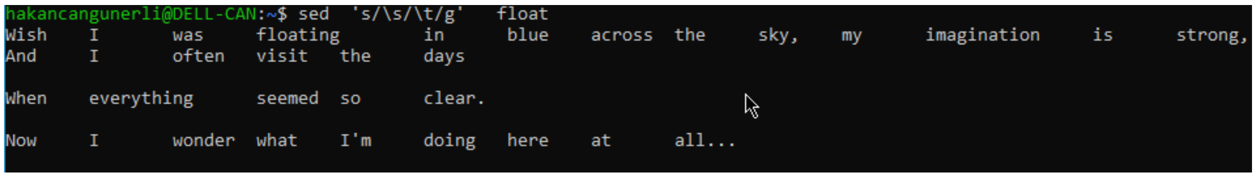


```
hakancangunerli@DELL-CAN:~$ awk -f h2.awk float
Start to scan file
Wish,days
,
When,clear.
,
Now,all...
,
END- float
hakancangunerli@DELL-CAN:~$
```

12.

s/ means substitute, \s means match whitespace, \t means insert a tab and \g means global (per line there's multiple ways that thing exists.)

so essentially, match regex whitespace with tabs globally.



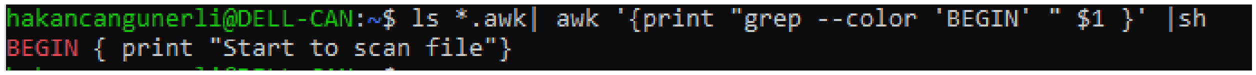
```
hakancangunerli@DELL-CAN:~$ sed  's/\s/\t/g'   float
Wish    I       was     floating        in      blue    across  the     sky,    my      imagination     is      strong,
And     I       often   visit   the     days

When    everything      seemed  so      clear.

Now     I       wonder  what    I'm     doing   here    at      all...
```

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 *)*

Find the word BEGIN and give it a color.



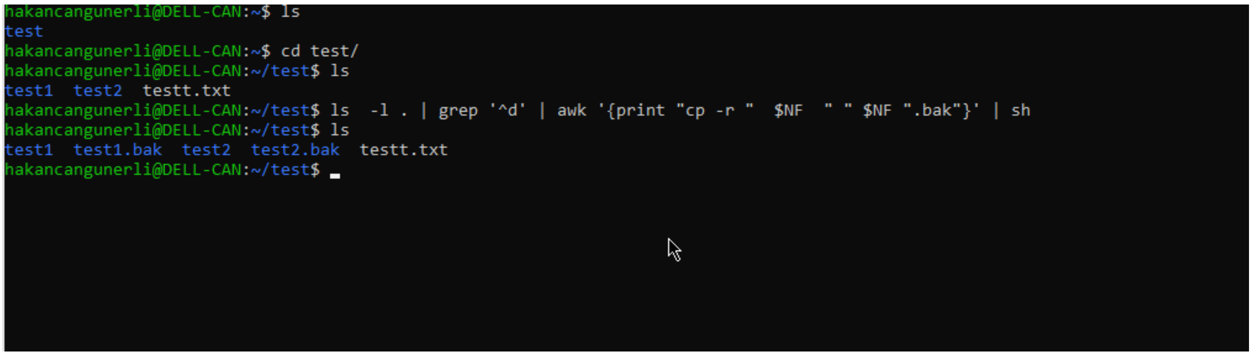
```
hakancangunerli@DELL-CAN:~$ ls *.awk| awk '{print "grep --color 'BEGIN' " $1 }' |sh
BEGIN { print "Start to scan file"}
```

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

this will create something called backup directory, which will make test1 and test2 have the same names including the things inside of the directories, the names will be followed by test1.bak and test2.bak instead.



```
hakancangunerli@DELL-CAN:~$ ls
test
hakancangunerli@DELL-CAN:~$ cd test/
hakancangunerli@DELL-CAN:~/test$ ls
test1  test2  testt.txt
hakancangunerli@DELL-CAN:~/test$ ls  -l . | grep '^d' | awk '{print "cp -r "  $NF  " " $NF ".bak"}' | sh
hakancangunerli@DELL-CAN:~/test$ ls
test1  test1.bak  test2  test2.bak  testt.txt
hakancangunerli@DELL-CAN:~/test$ 
```

Part III Programming: 15pts

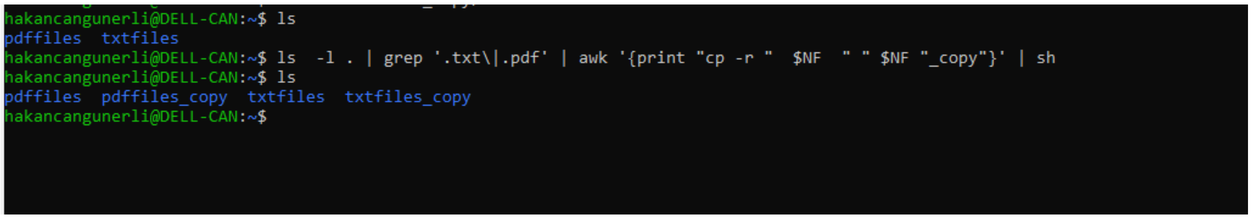
15. Sort all the files in your class working directory (or your home directory) as per the following requirements:

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.

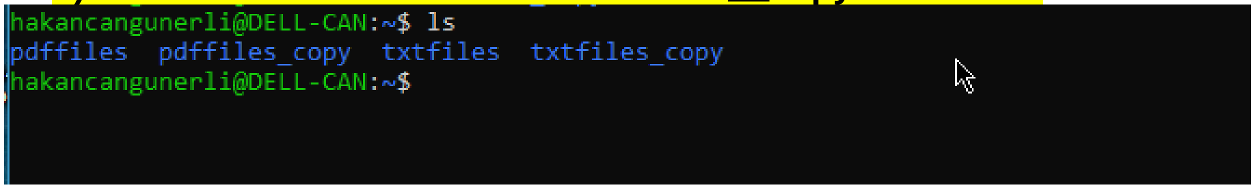
I'll just create my own files.

a) $ ls -l . | grep '.txt\|.pdf' | awk '{print "cp -r" $NF " " $NF "_copy"}' |sh



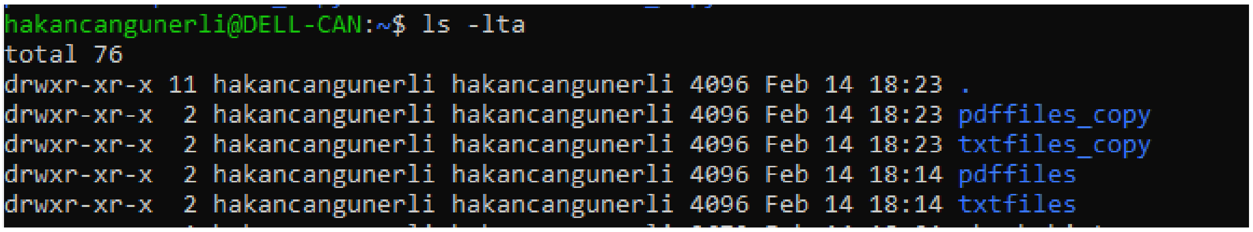
```
hakancangunerli@DELL-CAN:~$ ls
pdffiles  txtfiles
hakancangunerli@DELL-CAN:~$ ls  -l . | grep '.txt\|.pdf' | awk '{print "cp -r "  $NF  " " $NF "_copy"}' | sh
hakancangunerli@DELL-CAN:~$ ls
pdffiles  pdffiles_copy  txtfiles  txtfiles_copy
hakancangunerli@DELL-CAN:~$
```

b) here are the created directories with __copy at the end.



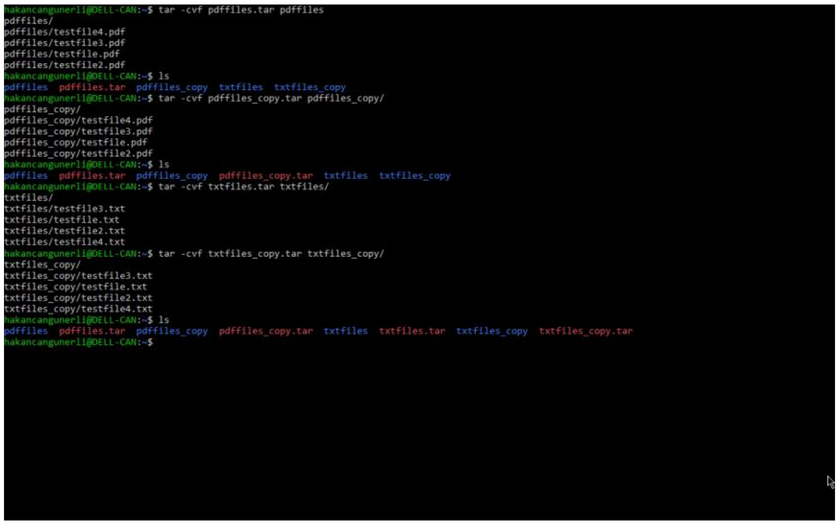
```
hakancangunerli@DELL-CAN:~$ ls
pdffiles  pdffiles_copy  txtfiles  txtfiles_copy
hakancangunerli@DELL-CAN:~$
```

c) ordered in chronological order of Months first using $ls-lta.



```
hakancangunerli@DELL-CAN:~$ ls -lta
total 76
drwxr-xr-x 11 hakancangunerli hakancangunerli 4096 Feb 14 18:23 .
drwxr-xr-x  2 hakancangunerli hakancangunerli 4096 Feb 14 18:23 pdffiles_copy
drwxr-xr-x  2 hakancangunerli hakancangunerli 4096 Feb 14 18:23 txtfiles_copy
drwxr-xr-x  2 hakancangunerli hakancangunerli 4096 Feb 14 18:14 pdffiles
drwxr-xr-x  2 hakancangunerli hakancangunerli 4096 Feb 14 18:14 txtfiles
```

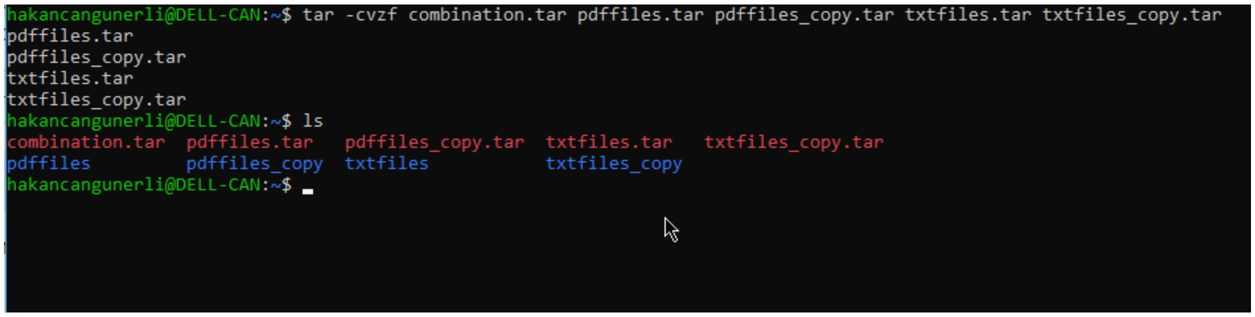
d) tar compression of the files. Essentially you type
$tar -cvf thetarname.tar directoryname
e.g. $tar -cvf pdffiles.tar pdffiles/
to create the pdffiles archive.



```
hakancangunerli@DELL-CAN:~$ tar -cvf pdffiles.tar pdffiles
pdffiles/
pdffiles/testfile4.pdf
pdffiles/testfile3.pdf
pdffiles/testfile.pdf
pdffiles/testfile2.pdf
hakancangunerli@DELL-CAN:~$ ls
pdffiles  pdffiles.tar  pdffiles_copy  txtfiles  txtfiles_copy
hakancangunerli@DELL-CAN:~$ tar -cvf pdffiles_copy.tar pdffiles_copy/
pdffiles_copy/
pdffiles_copy/testfile4.pdf
pdffiles_copy/testfile3.pdf
pdffiles_copy/testfile.pdf
pdffiles_copy/testfile2.pdf
hakancangunerli@DELL-CAN:~$ ls
pdffiles  pdffiles.tar  pdffiles_copy  pdffiles_copy.tar  txtfiles  txtfiles_copy
hakancangunerli@DELL-CAN:~$ tar -cvf txtfiles.tar txtfiles/
txtfiles/
txtfiles/testfile3.txt
txtfiles/testfile.txt
txtfiles/testfile2.txt
txtfiles/testfile4.txt
hakancangunerli@DELL-CAN:~$ tar -cvf txtfiles_copy.tar txtfiles_copy/
txtfiles_copy/
txtfiles_copy/testfile3.txt
txtfiles_copy/testfile.txt
txtfiles_copy/testfile2.txt
txtfiles_copy/testfile4.txt
hakancangunerli@DELL-CAN:~$ ls
pdffiles  pdffiles.tar  pdffiles_copy  pdffiles_copy.tar  txtfiles  txtfiles.tar  txtfiles_copy  txtfiles_copy.tar
hakancangunerli@DELL-CAN:~$
```

e) **$tar -czvf combination.tar pdffiles.tar pdffiles_copy.tar txtfiles.tar txtfiles_copy.tar**

finally, all tar files in a big tar archive named combination.tar



```
hakancangunerli@DELL-CAN:~$ tar -cvzf combination.tar pdffiles.tar pdffiles_copy.tar txtfiles.tar txtfiles_copy.tar
pdffiles.tar
pdffiles_copy.tar
txtfiles.tar
txtfiles_copy.tar
hakancangunerli@DELL-CAN:~$ ls
combination.tar  pdffiles.tar   pdffiles_copy.tar  txtfiles.tar   txtfiles_copy.tar
pdffiles         pdffiles_copy  txtfiles           txtfiles_copy
hakancangunerli@DELL-CAN:~$
```