

Experiment No. 4

Title: Pipes and filters in Linux

Batch: B2**Roll No: 16010420117****Experiment No: 4****Aim:** To implement pipes and filters in Linux.

Resources needed: Any open source OS/CoCalc online editor

Theory:**Pre lab/Prior concepts:****I/O Redirection**

In this experiment, we will explore a powerful feature used by many command line programs called input/output redirection. As we have seen, many commands such as `ls` print their output on the display. This does not have to be the case, however. By using some special notation we can redirect the output of many commands to files, devices, and even to the input of other commands.

Standard Output

Most command line programs that display their results do so by sending their results to a facility called standard output. By default, standard output directs its contents to the display. To redirect standard output to a file, the ">" character is used like this: `[me@linuxbox me]$ ls > file_list.txt`

In this example, the `ls` command is executed and the results are written in a file named `file_list.txt`. Since the output of `ls` was redirected to the file, no results appear on the display.

Each time the command above is repeated, `file_list.txt` is overwritten (from the beginning) with the output of the command `ls`. If you want the new results to be appended to the file instead, use ">>" like this:

```
[me@linuxbox me]$ ls >> file_list.txt
```

When the results are appended, the new results are added to the end of the file, thus making the file longer each time the command is repeated. If the file does not exist when you attempt to append the redirected output, the file will be created.

Standard Input

Many commands can accept input from a facility called standard input. By default,

standard input gets its contents from the keyboard, but like standard output, it can be redirected. To redirect standard input from a file instead of the keyboard, the "<" character is used like this:

```
[me@linuxbox me]$ sort < file_list.txt
```

In the above example we used the sort command to process the contents of file_list.txt. The results are output on the display since the standard output is not redirected in this example. We could redirect standard output to another file like this:

```
[me@linuxbox me]$ sort < file_list.txt > sorted_file_list.txt
```

As you can see, a command can have both its input and output redirected. Be aware that the order of the redirection does not matter. The only requirement is that the redirection operators (the "<" and ">") must appear after the other options and arguments in the command.

Pipe

A pipe is a way to connect the output of one program to the input of another program without any temporary file.

Pipe Defined as:

"A pipe is nothing but a temporary storage place where the output of one command is stored and then passed as the input for second command. Pipes are used to run more than two commands (Multiplecommands) from same command line."

Syntax:

```
command1 | command2
```

example :

```
me@linuxbox me]$ ls -l | less
```

In this example, the output of the `ls` command is fed into `less`. By using this "`| less`" trick, you can make any command have scrolling output. I use this technique all the time. By connecting commands together, you can accomplish amazing feats. Here are some examples you'll want to try:

Command What it does

`ls -lt | head` Displays the 10 newest files in the current directory.

`du | sort -nr` Displays a list of directories and how much space they consume, sorted from the largest to the smallest.

`find . -type f -print | wc -l` Displays the total number of files in the current working directory and all of its subdirectories.

Filters

One class of programs you can use with pipes is called filters. Filters take standard input and perform an operation upon it and send the results to standard output. In this way, they can be used to process information in powerful ways. Here are some of the common programs that can act as filters:

Command What it does

`sort` Sorts standard input then outputs the sorted result on standard output.

`uniq` Given a sorted stream of data from standard input, it removes duplicate lines of data (i.e., it makes sure that every line is unique).

`grep` Examines each line of data it receives from standard input and outputs every line that contains a specified pattern of characters.

`fmt` Reads text from standard input, then outputs formatted text on standard output.

`pr` Takes text input from standard input and splits the data into pages with page breaks, headers and footers in preparation for printing.

`head` Outputs the first few lines of its input. Useful for getting the header of a file.

`tail` Outputs the last few lines of its input. Useful for things like getting the most recent entries from a log file.

`tr` Translates characters. Can be used to perform tasks such as upper/lowercase conversions or changing line termination characters from one type to another (for example, converting DOS text files into Unix style text files).

`sed` Stream editor. Can perform more sophisticated text translations than `tr`.

`awk` An entire programming language designed for constructing filters. Extremely

powerful.

cut Cuts specific characters or fields from a file with options

paste Creates either rows or columns of data that are combined from two separate files.

Performing tasks with pipes

1. Printing from the command line. Linux provides a program called **lpr** that accepts standard input and sends it to the printer. It is often used with pipes and filters. Here are a couple of examples:

```
cat poorly_formatted_report.txt | fmt | pr | lpr
```

```
cat unsorted_list_with_dupes.txt | sort | uniq | pr | lpr
```

In the first example, we use **cat** to read the file and output it to standard output, which is piped into the standard input of **fmt**. **fmt** formats the text into neat paragraphs and outputs it to standard output, which is piped into the standard input of **pr**. **pr** splits the text neatly into pages and outputs it to standard output, which is piped into the standard input of **lpr**. **lpr** takes its standard input and sends it to the printer.

The second example starts with an unsorted list of data with duplicate entries. First, **cat** sends the list into **sort** which sorts it and feeds it into **uniq** which removes any duplicates. Next **pr** and **lpr** are used to paginate and print the list.

2. Viewing the contents of tar files Often you will see software distributed as a gzipped tar file. This is a traditional Unix style tape archive file (created with **tar**) that has been compressed with **gzip**. You can recognize these files by their traditional file extensions, ".tar.gz" or ".tgz". You can use the following command to view the directory of such a file on a Linux system:

```
tar tzvf name_of_file.tar.gz | less
```

Activities:

Question 1: Write command which produces a list of all the files on the system, such that:

0. their full pathname does not contain the word data
1. their filename does not contain the letter x
2. the script is Y2K compliant, so all `y's have been replaced by `k's.

At the end, print out the number of files that were found.

```

Activities Terminal Sep 26 03:37
os_it_b317@ubuntu: ~

os_it_b317@ubuntu:~$ ls
abc.txt      Documents  EXP5.txt   Music      sameer.c
a.out        Downloads  himanshu.c Pictures    snap
comment_exmple.sh exp4       kiara.c    Public     Templates
Desktop      Exp5.txt   kiara.txt  sameer     Videos
os_it_b317@ubuntu:~$ touch 'welcome to thunder oslab4.tern'
os_it_b317@ubuntu:~$ touch data
os_it_b317@ubuntu:~$ touch rintronFile.txt
os_it_b317@ubuntu:~$ mkdir Rintron && cd Rintron && touch fileInRintron
os_it_b317@ubuntu:~/Rintron$ ls
fileInRintron
os_it_b317@ubuntu:~/Rintron$ cd ..
os_it_b317@ubuntu:~$ ls
abc.txt      Exp5.txt   Rintron
a.out        EXP5.txt   rintronFile.txt
comment_exmple.sh himanshu.c sameer
data         kiara.c    sameer.c
Desktop      kiara.txt  snap
Documents    Music      Templates
Downloads    Pictures   Videos
exp4         Public     'welcome to thunder oslab4.tern'
os_it_b317@ubuntu:~$ ls -f $PWD/* | grep -v "data" | grep -v "x" | tr "y" "k"
/home/os_it_b317/a.out
/home/os_it_b317/himanshu.c
/home/os_it_b317/kiara.c
/home/os_it_b317/sameer
/home/os_it_b317/sameer.c
/home/os_it_b317/welcome to thunder oslab4.tern
..
..
/home/os_it_b317/Desktop:
..

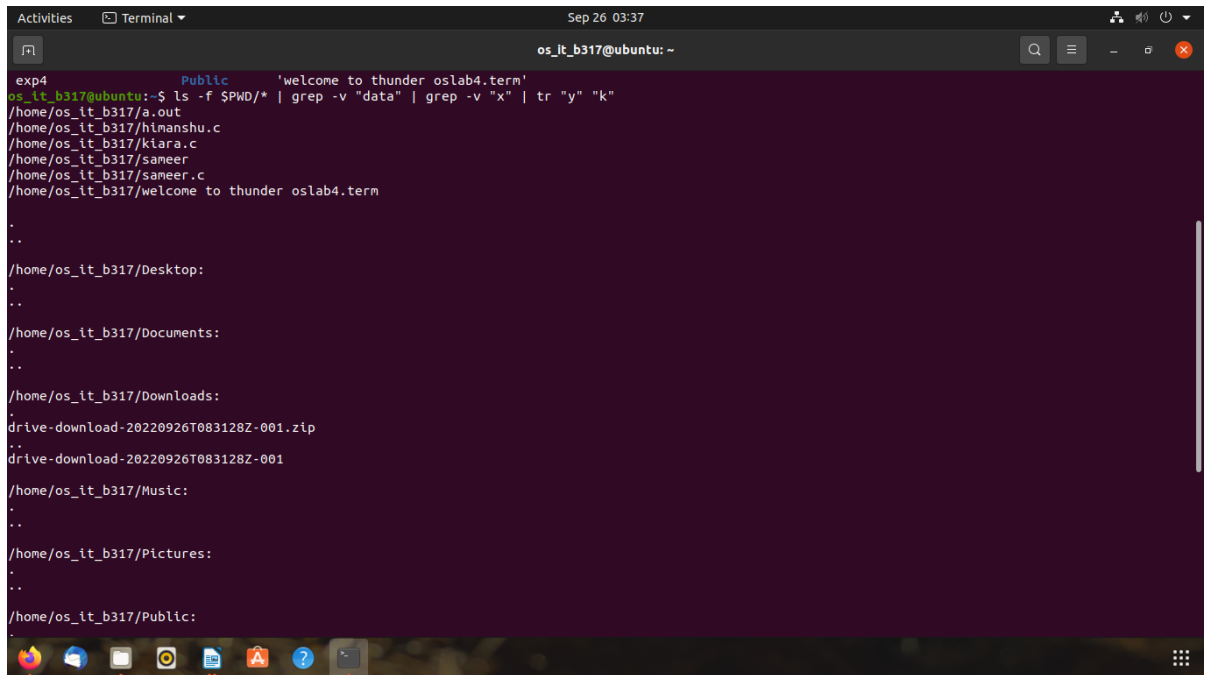
```

```

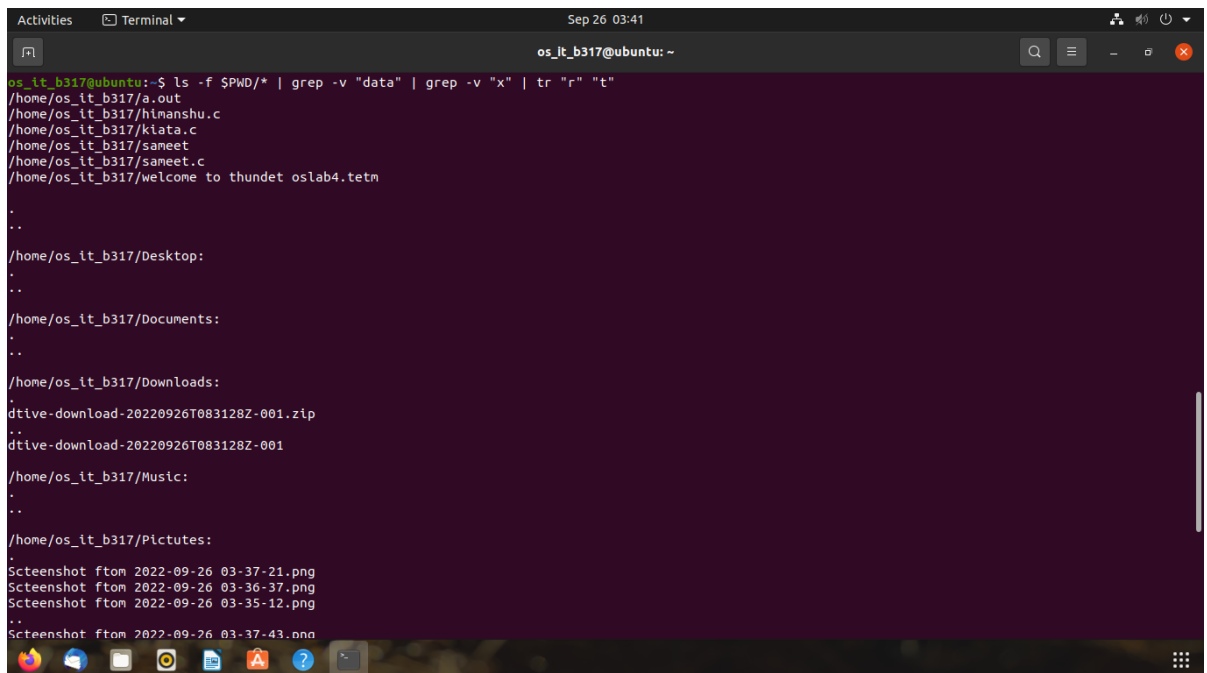
Activities Terminal Sep 26 03:38
os_it_b317@ubuntu: ~

..
/home/os_it_b317/Downloads:
drive-download-20220926T083128Z-001.zip
..
drive-download-20220926T083128Z-001
/home/os_it_b317/Music:
..
..
/home/os_it_b317/Pictures:
..
..
/home/os_it_b317/Public:
..
..
/home/os_it_b317/Rintron:
fileInRintron
..
/home/os_it_b317/snap:
..
snap-store
..
/home/os_it_b317/Templates:
..
..
/home/os_it_b317/Videos:

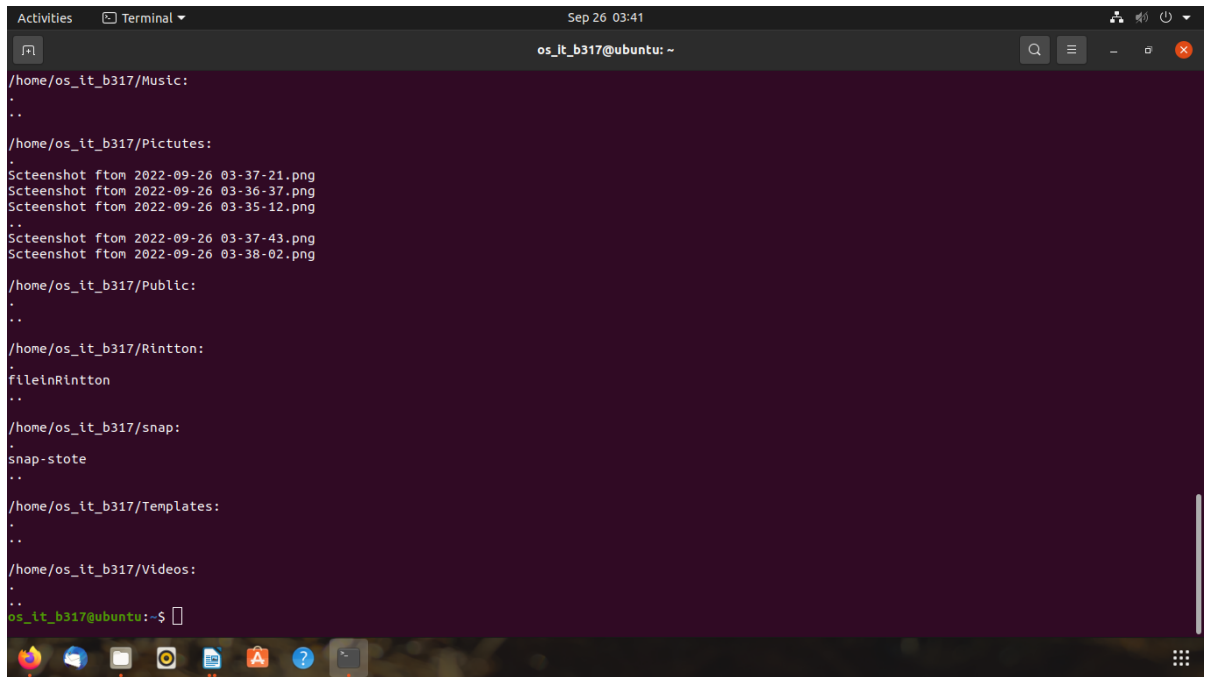
```



A terminal window titled 'os_it_b317@ubuntu: ~' showing the output of the command `ls -f $PWD/* | grep -v "data" | grep -v "x" | tr "y" "k"`. The output lists files in the current directory and subdirectories, with some files having their names transformed by the `tr` command. The files listed are: `exp4`, `/home/os_it_b317/a.out`, `/home/os_it_b317/hlmanshu.c`, `/home/os_it_b317/ktara.c`, `/home/os_it_b317/sameer`, `/home/os_it_b317/sameer.c`, `/home/os_it_b317/welcome to thunder oslab4.tern`, `/home/os_it_b317/Desktop:`, `/home/os_it_b317/Documents:`, `/home/os_it_b317/Downloads:`, `drive-download-20220926T083128Z-001.zip`, `drive-download-20220926T083128Z-001`, `/home/os_it_b317/Music:`, `/home/os_it_b317/Pictures:`, and `/home/os_it_b317/Public:`.



A terminal window titled 'os_it_b317@ubuntu: ~' showing the output of the command `ls -f $PWD/* | grep -v "data" | grep -v "x" | tr "r" "t"`. The output lists files in the current directory and subdirectories, with some files having their names transformed by the `tr` command. The files listed are: `/home/os_it_b317/a.out`, `/home/os_it_b317/hlmanshu.c`, `/home/os_it_b317/ktara.c`, `/home/os_it_b317/sameet`, `/home/os_it_b317/sameet.c`, `/home/os_it_b317/welcome to thundet oslab4.tetrn`, `/home/os_it_b317/Desktop:`, `/home/os_it_b317/Documents:`, `/home/os_it_b317/Downloads:`, `dtive-download-20220926T083128Z-001.zip`, `dtive-download-20220926T083128Z-001`, `/home/os_it_b317/Music:`, `/home/os_it_b317/Pictutes:`, `Scteenshot fton 2022-09-26 03-37-21.png`, `Scteenshot fton 2022-09-26 03-36-37.png`, `Scteenshot fton 2022-09-26 03-35-12.png`, and `Scteenshot fton 2022-09-26 03-37-43.png`.



A terminal window titled 'Activities Terminal' with a timestamp of 'Sep 26 03:41'. The window shows the user 'os_it_b317@ubuntu: ~'. The terminal displays the following directory listings:

```
/home/os_it_b317/Music:
.
..

/home/os_it_b317/Pictures:
Screenshot fton 2022-09-26 03-37-21.png
Screenshot fton 2022-09-26 03-36-37.png
Screenshot fton 2022-09-26 03-35-12.png
..
Screenshot fton 2022-09-26 03-37-43.png
Screenshot fton 2022-09-26 03-38-02.png

/home/os_it_b317/Public:
.
..

/home/os_it_b317/Rintton:
fileInRintton
..

/home/os_it_b317/snap:
snap-store
..

/home/os_it_b317/Templates:
.
..

/home/os_it_b317/Videos:
.
..
os_it_b317@ubuntu:~$
```

The terminal window has a dark purple background. The bottom of the window shows a taskbar with various application icons.

Here in this whole List we will not find the files/Folders whose name contain the word “data” and the letter “x” (“data” and “rintronFile.txt” are not displayed)

Question 2. Write commands that will list the size of each directory given on the command line, sorted by size. The size includes disk space used by the directory and all the files and subdirectories inside it. The script should take options to sort with smallest first, and with largest first.

Ans:

#Sorting in Ascending order

```

Activities  Terminal  Sep 26 03:44
os_it_b317@ubuntu: ~
..
..
/home/os_it_b317/Videos:
..
..
os_it_b317@ubuntu:~$
os_it_b317@ubuntu:~$
os_it_b317@ubuntu:~$
os_it_b317@ubuntu:~$ du -sk * | sort -n
0      data
0      rintronFile.txt
0      welcome to thunder oslab4.term
4      abc.txt
4      comment_exmple.sh
4      Desktop
4      exp4
4      Exp5.txt
4      EXP5.txt
4      himanshu.c
4      kiara.c
4      kiara.txt
4      Music
4      Public
4      Rintron
4      sameer.c
4      Templates
4      Videos
20     a.out
20     sameer
64     Documents
1344   Pictures
6708   snap
7044   Downloads
os_it_b317@ubuntu:~$

```

#Sorting in Descending order

```

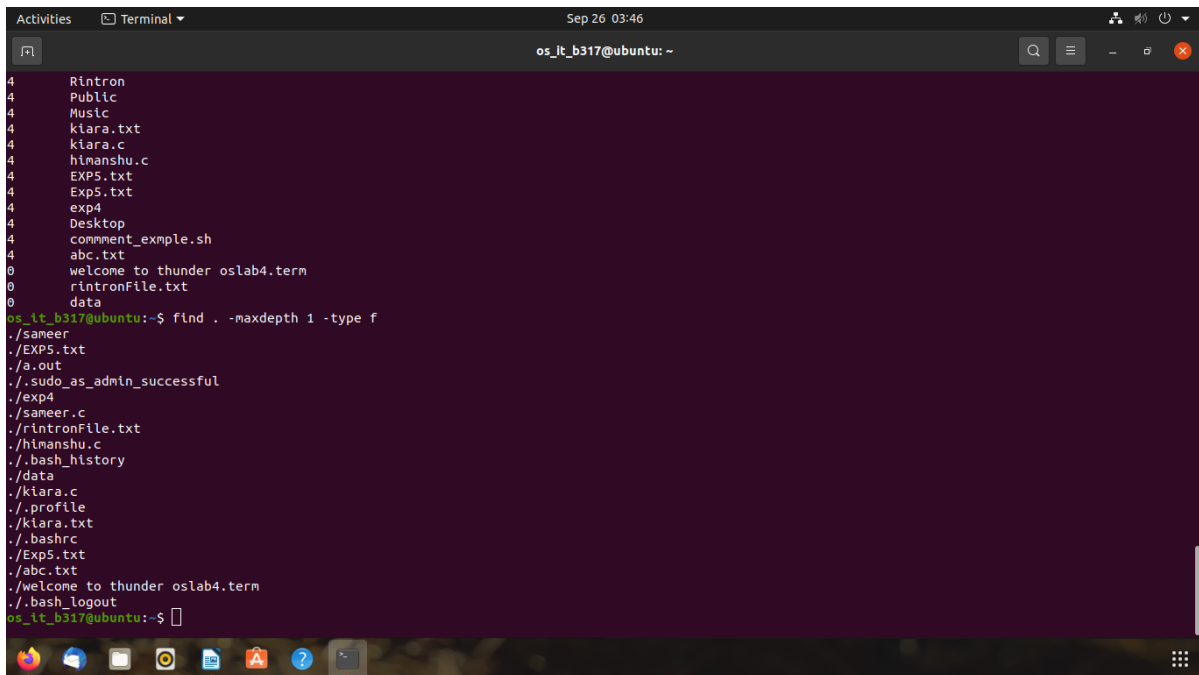
Activities  Terminal  Sep 26 03:45
os_it_b317@ubuntu: ~
4      sameer.c
4      Templates
4      Videos
20     a.out
20     sameer
64     Documents
1344   Pictures
6708   snap
7044   Downloads
os_it_b317@ubuntu:~$ du -sk * | sort -nr
7044   Downloads
6708   snap
1472   Pictures
64     Documents
20     sameer
20     a.out
4      Videos
4      Templates
4      sameer.c
4      Rintron
4      Public
4      Music
4      kiara.txt
4      kiara.c
4      himanshu.c
4      EXP5.txt
4      Exp5.txt
4      exp4
4      Desktop
4      comment_exmple.sh
4      abc.txt
0      welcome to thunder oslab4.term
0      rintronFile.txt
0      data
os_it_b317@ubuntu:~$

```

Question 3. Write a command to count total number of the files in present working directory.

Ans:

#printing the total number of files in current directory



```

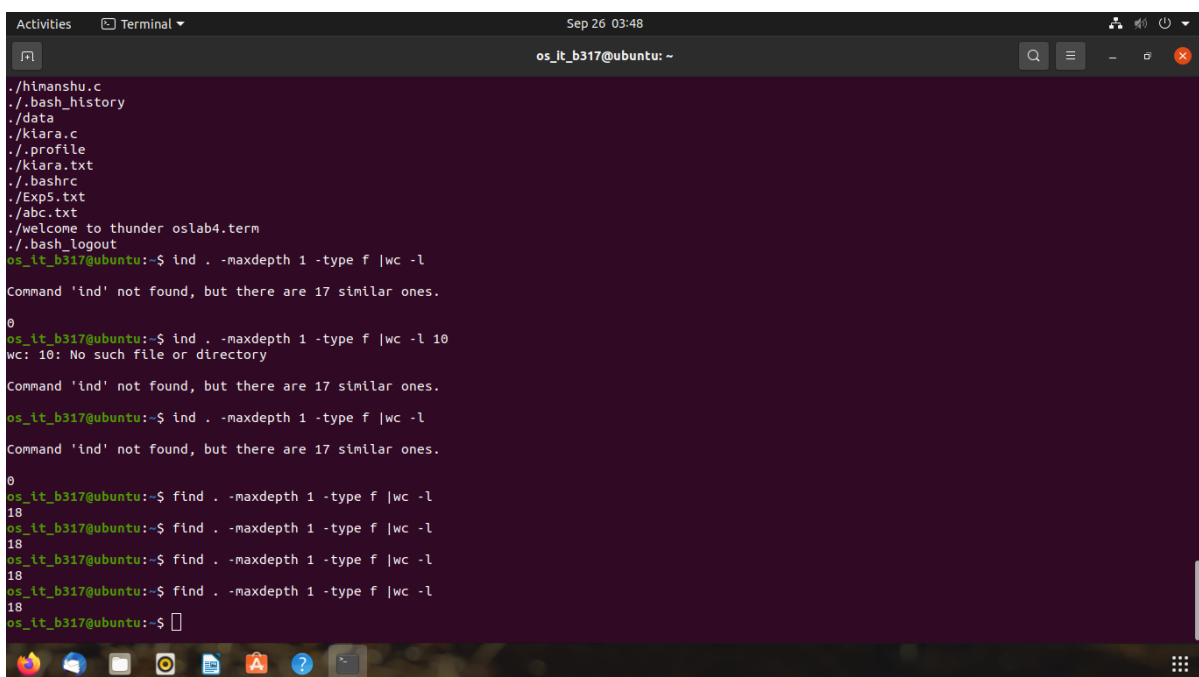
Activities  Terminal  Sep 26 03:46
os_it_b317@ubuntu: ~

4 Rintron
4 Public
4 Music
4 klara.txt
4 klara.c
4 himanshu.c
4 EXP5.txt
4 Exp5.txt
4 exp4
4 Desktop
4 comment_exmple.sh
4 abc.txt
0 welcome to thunder oslab4.term
0 rintronFile.txt
0 data

os_it_b317@ubuntu:~$ find . -maxdepth 1 -type f
./sameer
./EXP5.txt
./a.out
./sudo_as_admin_successful
./exp4
./sameer.c
./rintronFile.txt
./himanshu.c
./bash_history
./data
./klara.c
./profile
./klara.txt
./bashrc
./Exp5.txt
./abc.txt
./welcome to thunder oslab4.term
./bash_logout
os_it_b317@ubuntu:~$

```

#printing the total number of files in the directory



```

Activities  Terminal  Sep 26 03:48
os_it_b317@ubuntu: ~

./himanshu.c
./bash_history
./data
./klara.c
./profile
./klara.txt
./bashrc
./Exp5.txt
./abc.txt
./welcome to thunder oslab4.term
./bash_logout

os_it_b317@ubuntu:~$ ind . -maxdepth 1 -type f |wc -l
Command 'ind' not found, but there are 17 similar ones.

0
os_it_b317@ubuntu:~$ ind . -maxdepth 1 -type f |wc -l 10
wc: 10: No such file or directory

Command 'ind' not found, but there are 17 similar ones.

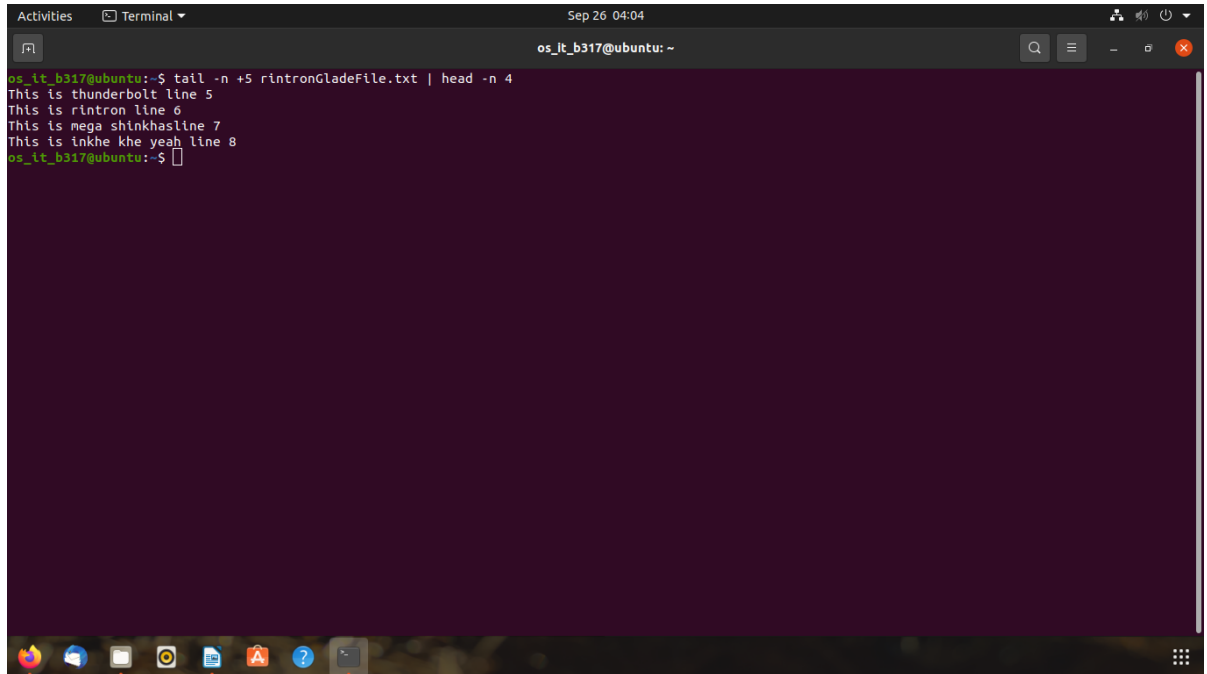
os_it_b317@ubuntu:~$ ind . -maxdepth 1 -type f |wc -l
Command 'ind' not found, but there are 17 similar ones.

0
os_it_b317@ubuntu:~$ find . -maxdepth 1 -type f |wc -l
18
os_it_b317@ubuntu:~$ find . -maxdepth 1 -type f |wc -l
18
os_it_b317@ubuntu:~$ find . -maxdepth 1 -type f |wc -l
18
os_it_b317@ubuntu:~$ find . -maxdepth 1 -type f |wc -l
18
os_it_b317@ubuntu:~$

```

Question 4. Write command to extract 4 line starting from line number 5 to line number 8 from a file which contains 10 lines in it.

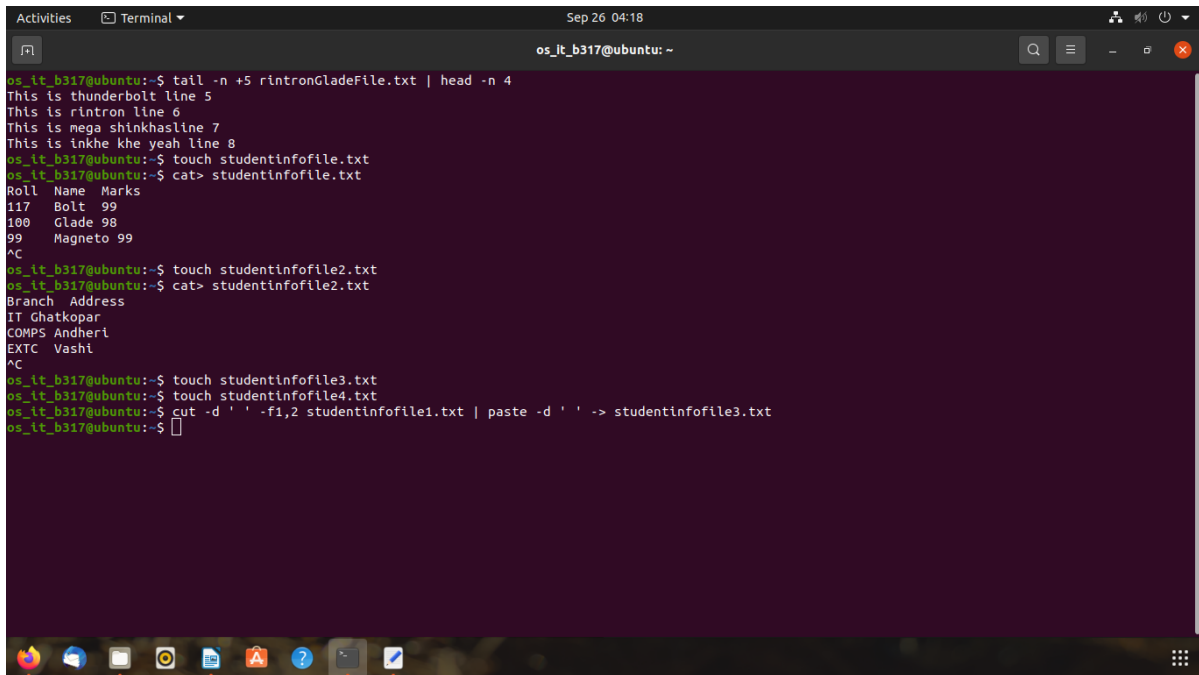
Ans:



```
Activities Terminal Sep 26 04:04 os_it_b317@ubuntu: ~
os_it_b317@ubuntu:~$ tail -n +5 rintronGladeFile.txt | head -n 4
This is thunderbolt line 5
This is rintron line 6
This is mega shinkhasline 7
This is inkhe khe yeah line 8
os_it_b317@ubuntu:~$
```

Question 5. Create a file containing rollno, name and marks of 3 students and another file containing branch and address of same 3 students. Use space as delimiter in both files. Write commands to cut rollno and name files first file and address field from second file and paste result in new file and display it.

Ans:

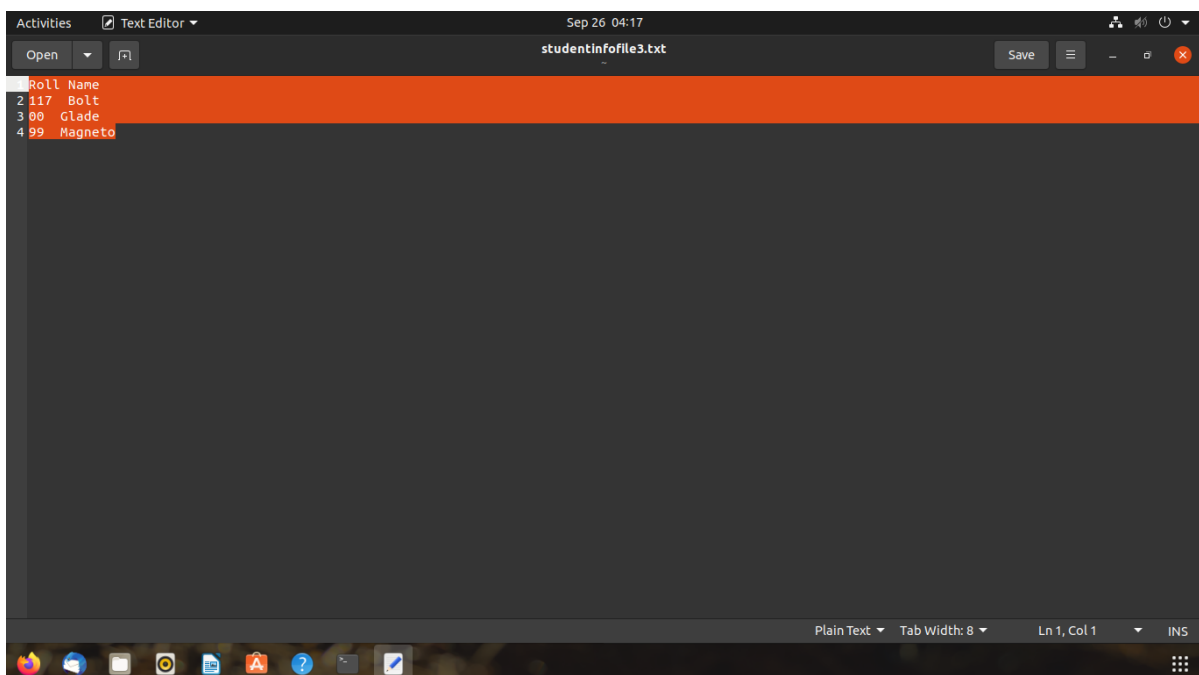


```

os_it_b317@ubuntu:~$ tail -n +5 rintronGladeFile.txt | head -n 4
This is thunderbolt line 5
This is rintron line 6
This is mega shinkhasline 7
This is inkhe khe yeah line 8
os_it_b317@ubuntu:~$ touch studentinfofile1.txt
os_it_b317@ubuntu:~$ cat > studentinfofile1.txt
Roll Name Marks
117 Bolt 99
100 Glade 98
99 Magneto 99
^C
os_it_b317@ubuntu:~$ touch studentinfofile2.txt
os_it_b317@ubuntu:~$ cat > studentinfofile2.txt
Branch Address
IT Ghatkopar
COMPS Andheri
EXTC Vashi
^C
os_it_b317@ubuntu:~$ touch studentinfofile3.txt
os_it_b317@ubuntu:~$ touch studentinfofile4.txt
os_it_b317@ubuntu:~$ cut -d ' ' -f1,2 studentinfofile1.txt | paste -d ' ' -> studentinfofile3.txt
os_it_b317@ubuntu:~$
  
```

#Here the columns 1 and 2(Roll No and Name) are cut and pasted from file “studentinfofile1.txt” to a new file “studentinfofile3.txt”

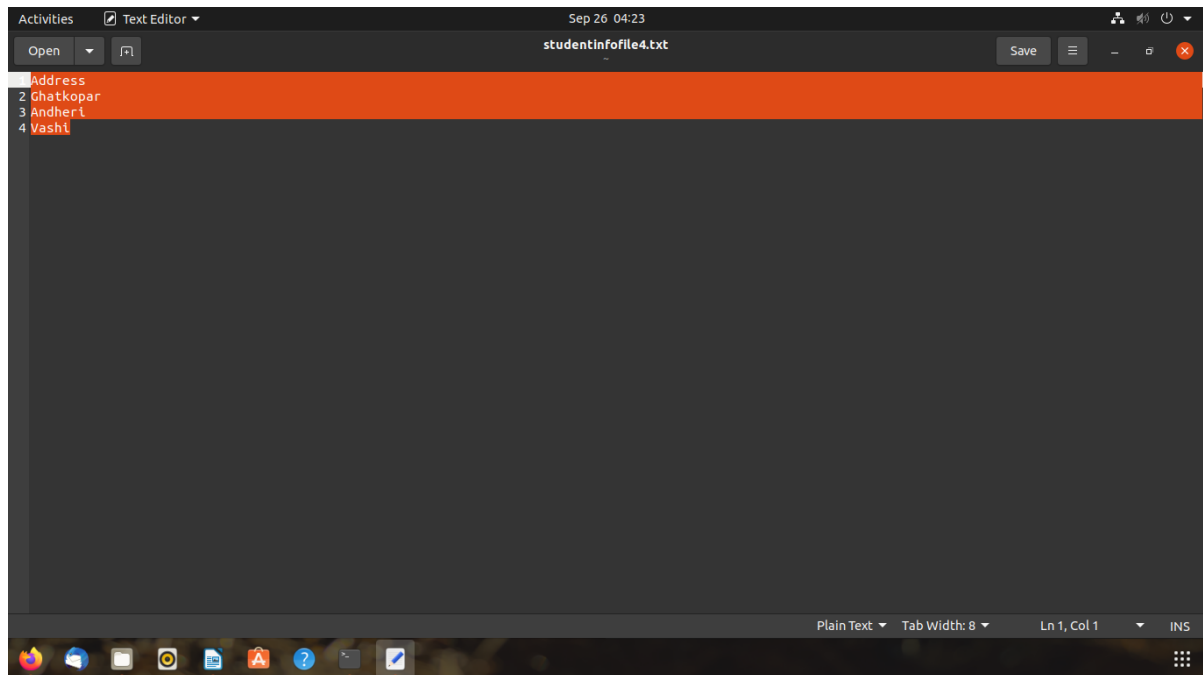
#Contents of file studentinfofile3.txt



```

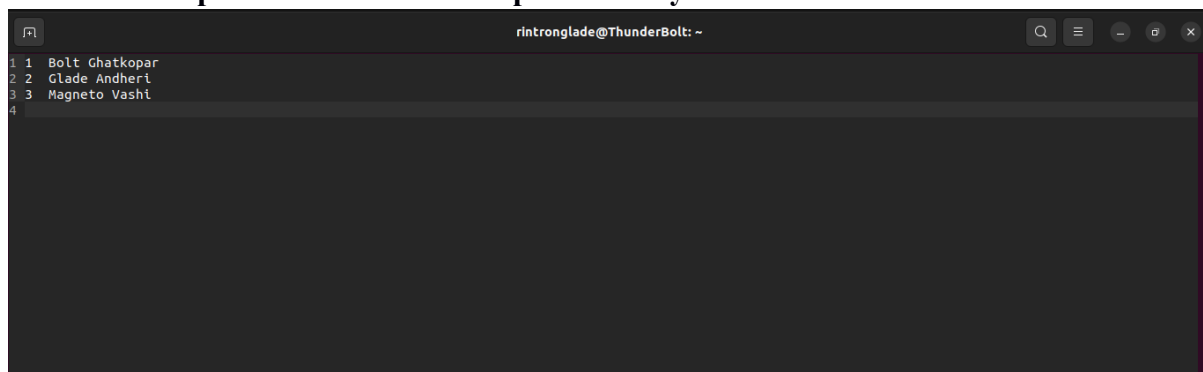
Roll Name
1 117 Bolt
2 100 Glade
3 99 Magneto
  
```

#Here the column 2(Address) is cut and pasted from file “studentinfofile2.txt” to a new file “studentinfofile4.txt”



The screenshot shows a text editor window titled "studentinfofile4.txt" with a dark background. The first four lines of the file are highlighted in orange: "1 Address", "2 Ghatkopar", "3 Andheri", and "4 Vashi". The window's title bar includes "Activities", "Text Editor", and the date/time "Sep 26 04:23". The bottom status bar indicates "Plain Text", "Tab Width: 8", "Ln 1, Col 1", and "INS". The system tray at the bottom shows various application icons.

We can also paste the combined output directly to a new file “studentinfofile5.txt”.



The screenshot shows a terminal window titled "rintronglade@ThunderBolt: ~" with a dark background. The first four lines of the file are displayed: "1 Bolt Ghatkopar", "2 Glade Andheri", "3 Magneto Vashi", and "4". The window's title bar includes a search icon, a menu icon, and window control buttons. The bottom status bar shows the user's name and the terminal's title.

Results: Perform the activity. No snapshots to be taken.

The assignment submitted should be e- media saved as <Roll No_Batch No_Date>

This file must contain on the top:

Name:

Roll No.

Exp No.

Batch:

Date:

And Students have to upload this document electronically.

Outcomes: CO4: Demonstrate open source standards usage

Conclusion: We learnt the implementation of pipes and filters in Linux and understood the use of various commands and functionalities related to them.

<Students should write about the concluding remarks of the experiment themselves>.

Grade: AA/AB/BB/BC/CC/CD/DD

Signature of faculty in-charge with date

References:

Books/ Journals/ Websites:

1. Richard Blum and Christine Bresnahan, "Linux Command Line & Shell Scripting",
II Edition edition, Wiley, 2012.