Tutorial 2: Unix Command Line (II)

CS 104

Spring, 2024-25

TA: Abhi Jain

Credits: TA-2023-2: Guramrit Singh

Topics

- Redirection and pipes
- Process Management
- Access Control
- Different Users
- Regular Expressions
- Some other commands
- Exercises

Redirection and pipes

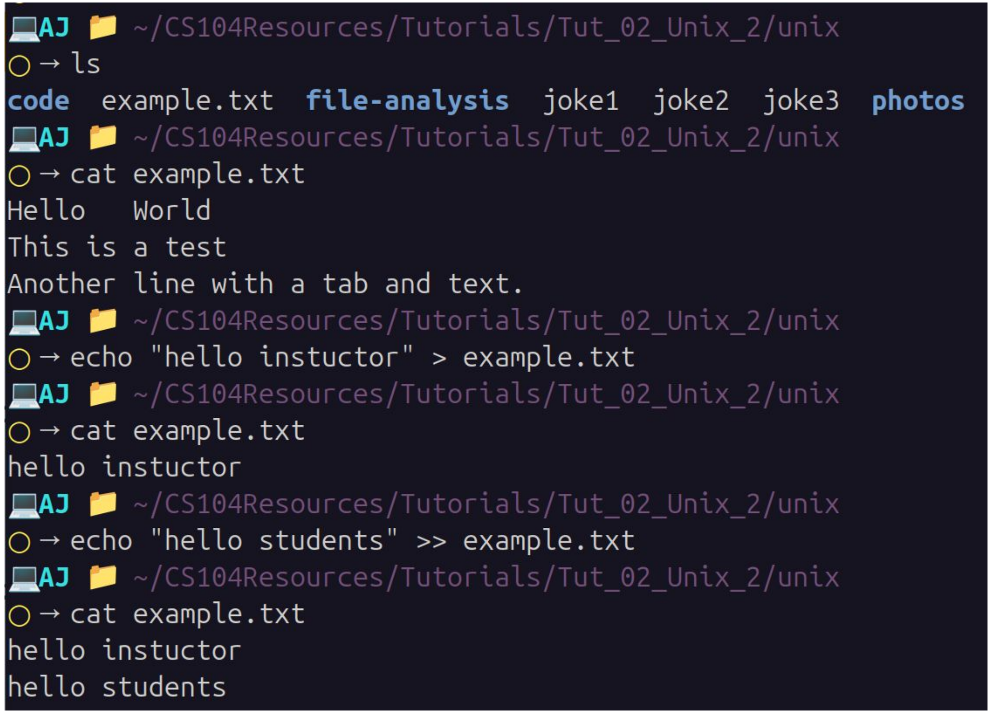
- >
- <
- <<
- |

Redirection

- Frequently, there's a desire to avoid manual input of command arguments or to save output to a file.
- Redirection operators offer a convenient way to accomplish these tasks.
- Types of redirection operators:
    - **>** : Redirects standard output to a file, overwriting the file if it already exists.
    - **>>** : Redirects standard output to a file, appending the output to the end of the file if it already exists.
    - **<** : Redirects standard input to come from a file.

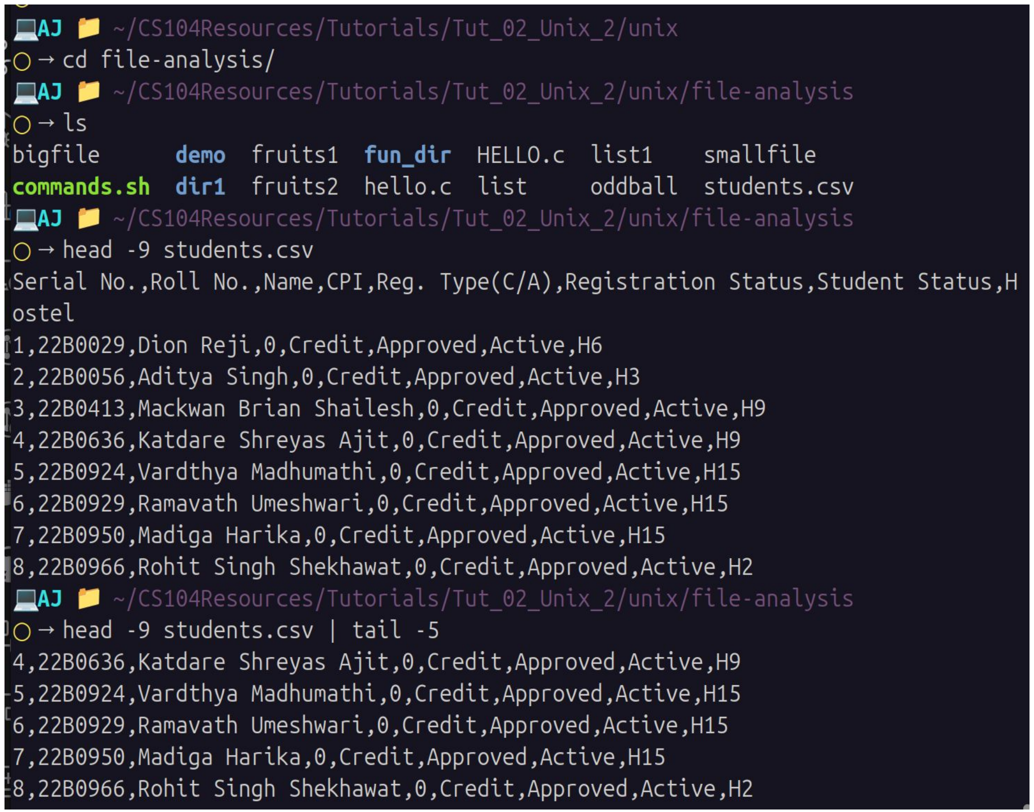
>, >>

- We saw echo command before. Now, we would like to redirect the output of echo command to a file rather than the terminal.
- In the example,
  - We first redirect “hello instructor” to example.txt
  - This overwrites the file
  - Then we append “hello students” to example.txt



```
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix
○ → ls
code  example.txt  file-analysis  joke1  joke2  joke3  photos
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix
○ → cat example.txt
Hello   World
This is a test
Another line with a tab and text.
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix
○ → echo "hello instuctor" > example.txt
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix
○ → cat example.txt
hello instuctor
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix
○ → echo "hello students" >> example.txt
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix
○ → cat example.txt
hello instuctor
hello students
```

Pipes



```
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix
○ → cd file-analysis/
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → ls
bigfile      demo  fruits1  fun_dir  HELLO.c  list1    smallfile
commands.sh  dir1  fruits2  hello.c  list     oddball  students.csv
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → head -9 students.csv
Serial No.,Roll No.,Name,CPI,Reg. Type(C/A),Registration Status,Student Status,H
ostel
1,22B0029,Dion Reji,0,Credit,Approved,Active,H6
2,22B0056,Aditya Singh,0,Credit,Approved,Active,H3
3,22B0413,Mackwan Brian Shailesh,0,Credit,Approved,Active,H9
4,22B0636,Katdare Shreyas Ajit,0,Credit,Approved,Active,H9
5,22B0924,Vardthya Madhumathi,0,Credit,Approved,Active,H15
6,22B0929,Ramavath Umeshwari,0,Credit,Approved,Active,H15
7,22B0950,Madiga Harika,0,Credit,Approved,Active,H15
8,22B0966,Rohit Singh Shekhawat,0,Credit,Approved,Active,H2
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → head -9 students.csv | tail -5
4,22B0636,Katdare Shreyas Ajit,0,Credit,Approved,Active,H9
5,22B0924,Vardthya Madhumathi,0,Credit,Approved,Active,H15
6,22B0929,Ramavath Umeshwari,0,Credit,Approved,Active,H15
7,22B0950,Madiga Harika,0,Credit,Approved,Active,H15
8,22B0966,Rohit Singh Shekhawat,0,Credit,Approved,Active,H2
```

- Often, there's a preference to bypass manually specifying command arguments or saving output to a file, opting instead to seamlessly pass the output of one command as the input to another using a pipe.
- | : Redirects the output of one command as the input to another command (pipe).
- In the example, we want to find the lines 5-9 line of the students.csv file in file analysis directory.
  - We first use head on students.csv file to get the first 9 lines
  - Then, we pass the above output to tail, which extracts last 5 lines and gives us the desired output

Process Management

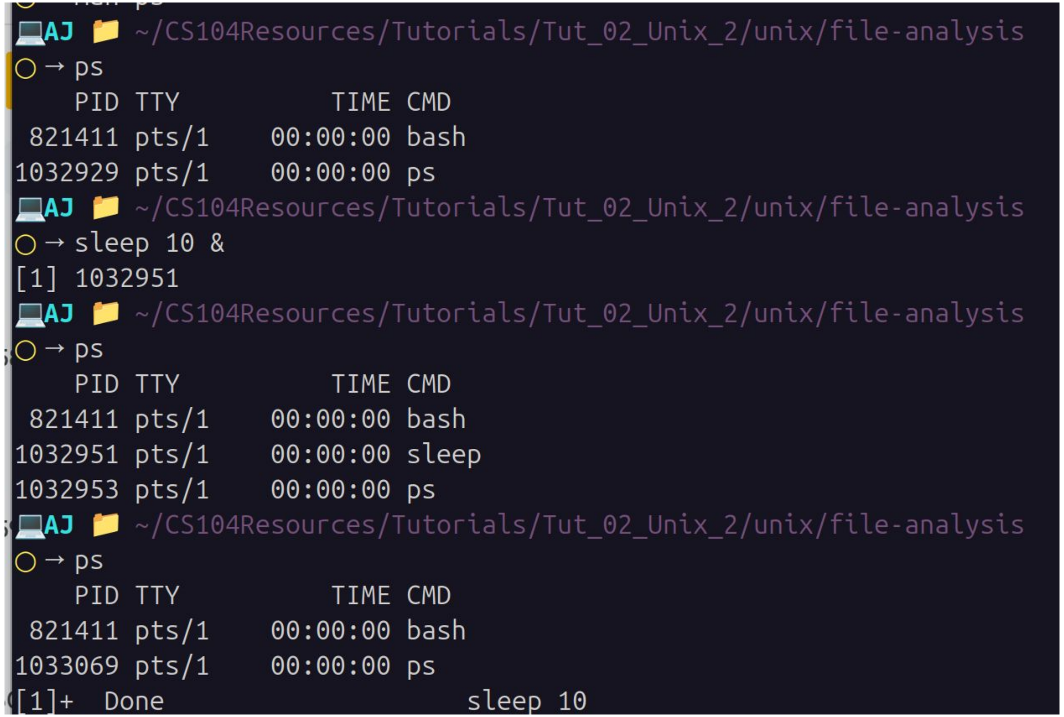
- ps
- pkill

Processes

- ➔ A program in execution is referred to as process.
- ➔ It consists of several components, including data retrieved from files, user input, program instructions, etc.
- ➔ Same program can be executed any number of times, each execution instance becomes a new process.
- ➔ Each process has a unique id referred to as pid.
- ➔ init is the first process that is created by OS during boot up and usually has pid 1.
- ➔ Every other process is created by another process referred to as parent process and it's pid referred to as ppid.
- ➔ Interested in processes?? More on this in your OS course!!!
- ➔ Next we discuss two frequently used tools ps and kill dealing with processes

ps

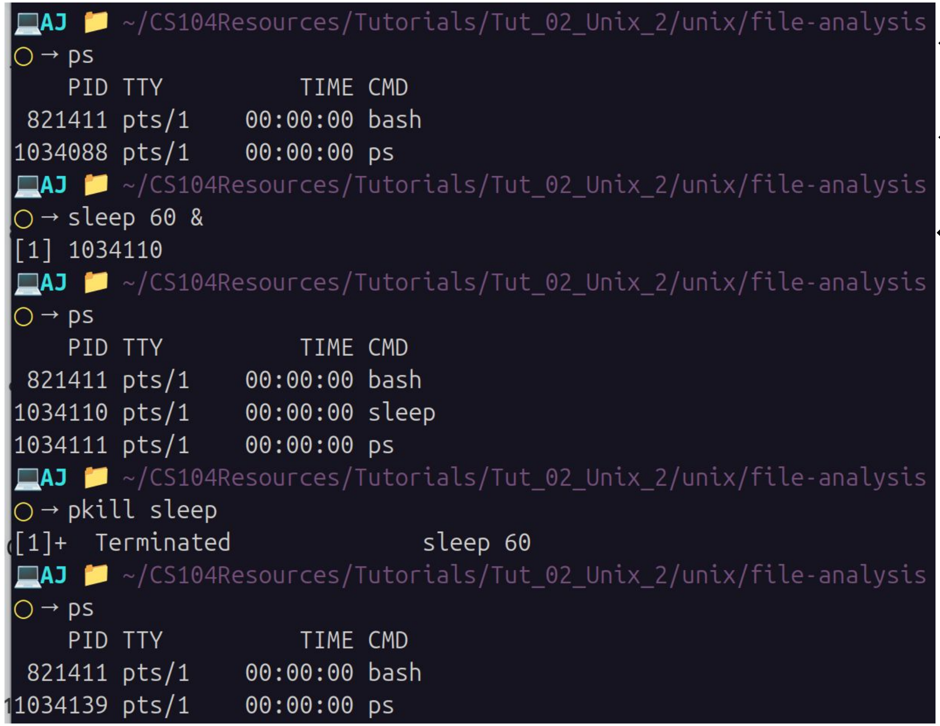
- ps : process status, displays a header line, followed by lines containing information about all of your processes that have controlling terminals.
- Some of the useful options are:
  - -a : Display info about processes of other users' as well as yours.
  - -u : Display the processes belonging to the specified usernames.
  - Checkout : -f, -x, -j, -m



```
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → ps
    PID TTY          TIME CMD
 821411 pts/1    00:00:00 bash
1032929 pts/1    00:00:00 ps
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → sleep 10 &
[1] 1032951
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → ps
    PID TTY          TIME CMD
 821411 pts/1    00:00:00 bash
1032951 pts/1    00:00:00 sleep
1032953 pts/1    00:00:00 ps
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → ps
    PID TTY          TIME CMD
 821411 pts/1    00:00:00 bash
1033069 pts/1    00:00:00 ps
[1]+  Done                    sleep 10
```

Example: We first use ps and see that only shell process is running on the current terminal, then we execute sleep for 10 seconds in background and see the ps output before and after sleep process terminates.

pkill



```
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → ps
    PID TTY          TIME CMD
 821411 pts/1    00:00:00 bash
1034088 pts/1    00:00:00 ps
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → sleep 60 &
[1] 1034110
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → ps
    PID TTY          TIME CMD
 821411 pts/1    00:00:00 bash
1034110 pts/1    00:00:00 sleep
1034111 pts/1    00:00:00 ps
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → pkill sleep
[1]+  Terminated              sleep 60
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → ps
    PID TTY          TIME CMD
 821411 pts/1    00:00:00 bash
1034139 pts/1    00:00:00 ps
```

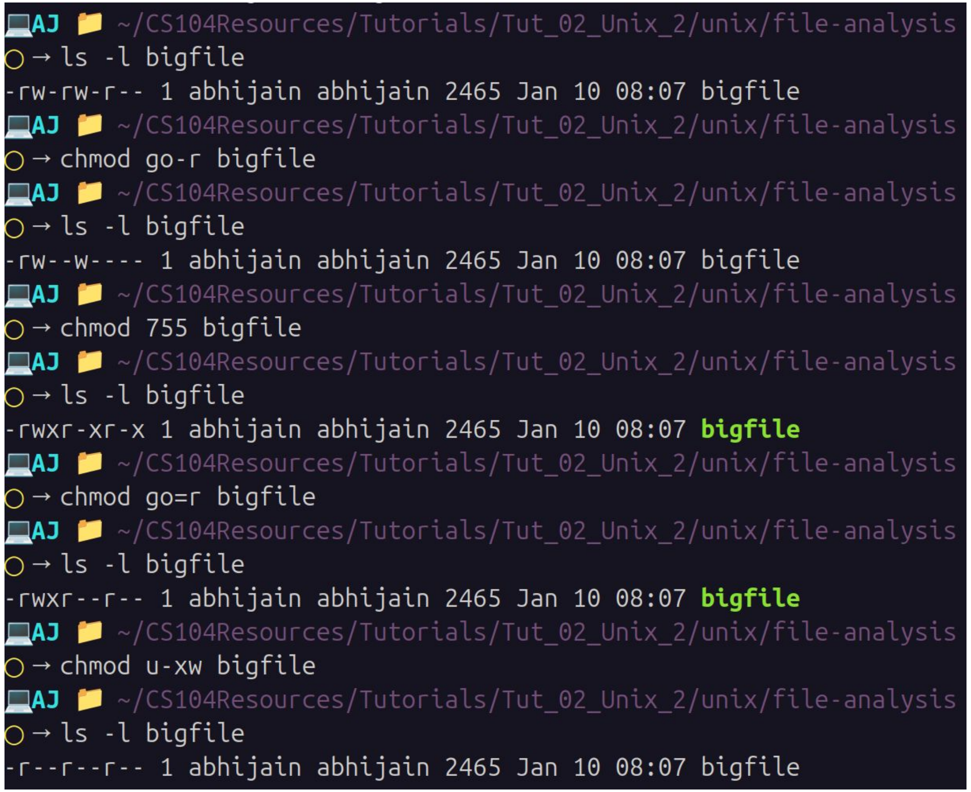
Example: We execute sleep for 60 seconds in background and see the ps output before and after kill command terminates the sleep process.

- Used to send signals to processes to request actions like termination, suspension, or restarting
- Targets processes based on their names or other attributes, rather than PID.
- Some of the useful options are:
  - -s : signal name specifying the signal to be sent.
  - If omitted, the default signal is SIGTERM
  - To know what SIGNALs available, use kill -l
  - -f: Matches against the full command line of the processes, not just the process name
  - -u USER: Targets processes owned by the specified user
  - -t TTY: Targets processes associated with the specified terminal
  - -P PID: Targets child processes of the specified parent PID

Access Control

- chmod

chmod



```
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → ls -l bigfile
-rw-rw-r-- 1 abhijain abhijain 2465 Jan 10 08:07 bigfile
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → chmod go-r bigfile
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → ls -l bigfile
-rw--w---- 1 abhijain abhijain 2465 Jan 10 08:07 bigfile
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → chmod 755 bigfile
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → ls -l bigfile
-rwxr-xr-x 1 abhijain abhijain 2465 Jan 10 08:07 bigfile
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → chmod go=r bigfile
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → ls -l bigfile
-rwxr--r-- 1 abhijain abhijain 2465 Jan 10 08:07 bigfile
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → chmod u-xw bigfile
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → ls -l bigfile
-r--r--r-- 1 abhijain abhijain 2465 Jan 10 08:07 bigfile
```

Example: 1. By using go-r, we removed read access from group and others.
2.By using 755 (111 101 101), we gave rwx to user, and rx to group and others.
By using go=r, we changed access of group, others to read only. 4. Finally, by using u-xw we removed execute and write permission from user.

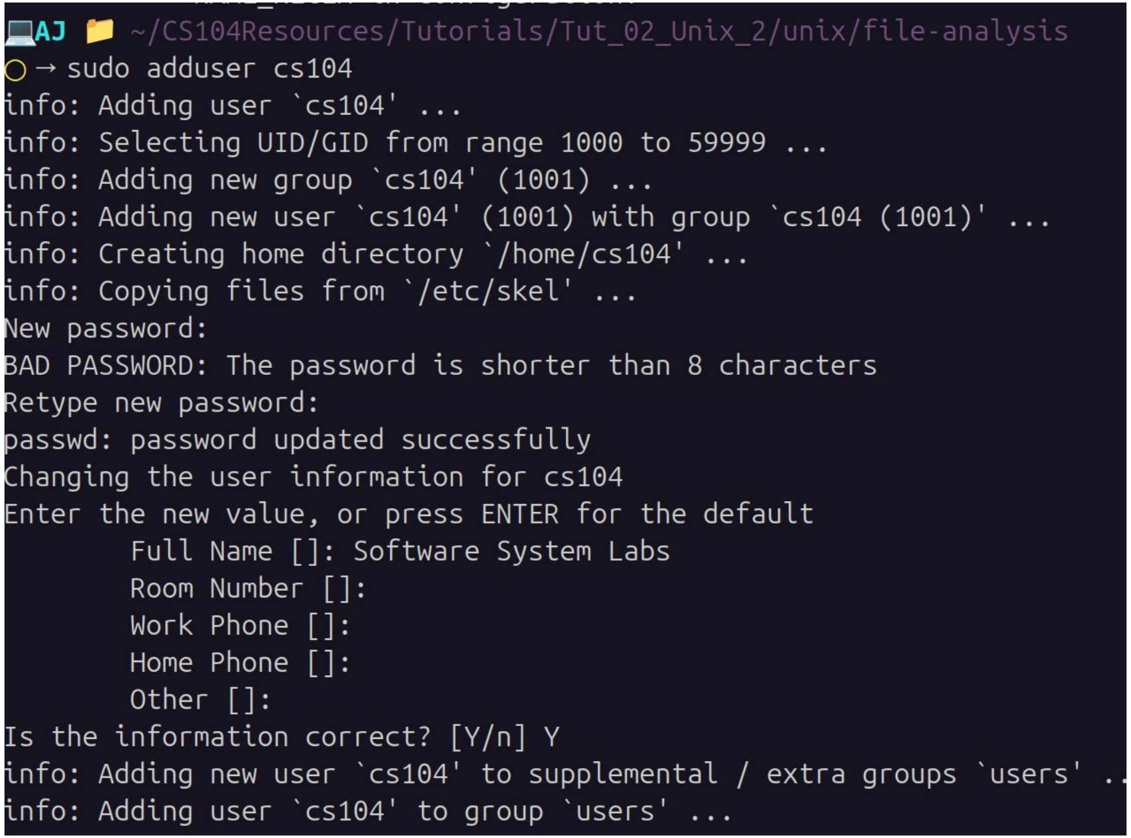
- The chmod utility modifies the file mode bits of the listed files as specified by the mode operand.
- Symbolic Mode:
  - u - The file owner.
  - g - The users who are members of the group.
  - o - All other users.
  - a - All users, identical to ugo
  - - Removes the specified permissions.
  - + Adds specified permissions.
  - = Changes the current permissions to the specified permissions
    - If no permissions are specified after the = symbol, all permissions from the specified user class are removed

Different Users

- su
- sudo

Adding a new user

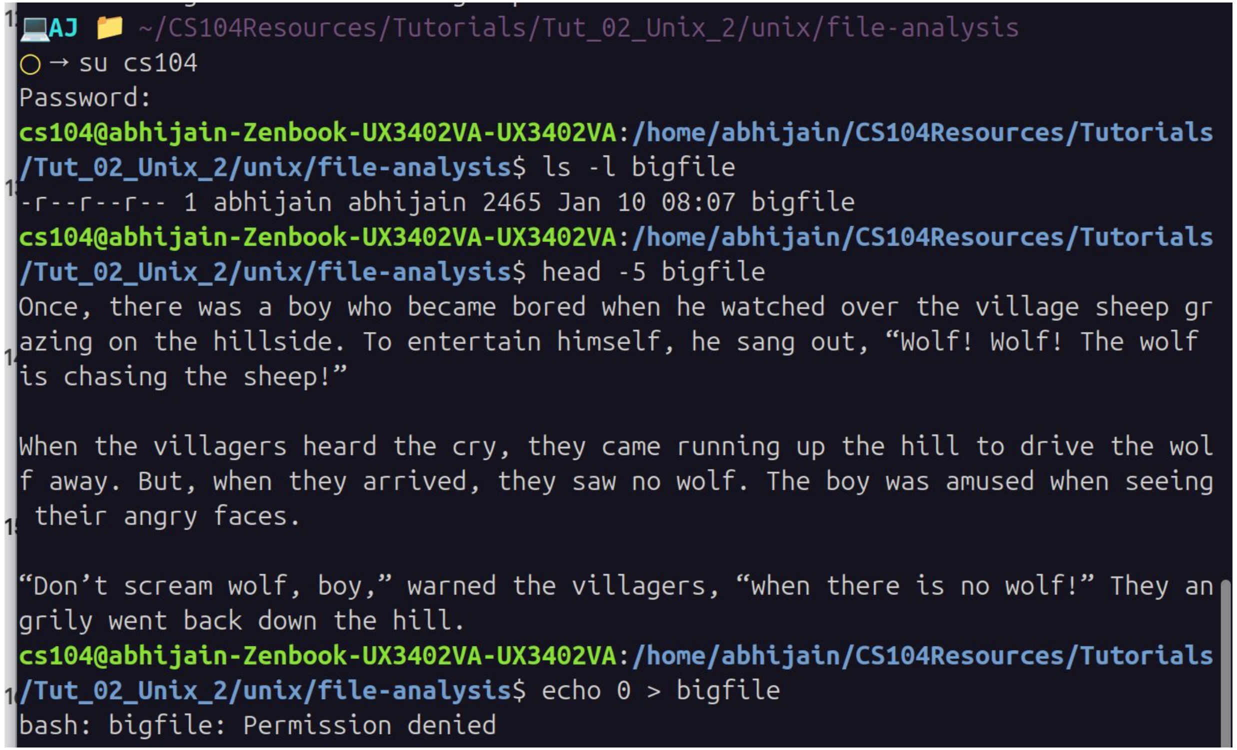
- Not part of course. Just part of Tutorial to know how to add a new user.
- Created a new user named cs104 using sudo adduser.
- The process is interactive, just do as instructed.



```
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → sudo adduser cs104
info: Adding user `cs104' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group `cs104' (1001) ...
info: Adding new user `cs104' (1001) with group `cs104 (1001)' ...
info: Creating home directory `/home/cs104' ...
info: Copying files from `/etc/skel' ...
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: password updated successfully
Changing the user information for cs104
Enter the new value, or press ENTER for the default
        Full Name []: Software System Labs
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] Y
info: Adding new user `cs104' to supplemental / extra groups `users' ..
info: Adding user `cs104' to group `users' ...
```

su

The su utility requests appropriate user credentials and switches to that user ID (the default user is the superuser). A shell is then executed.

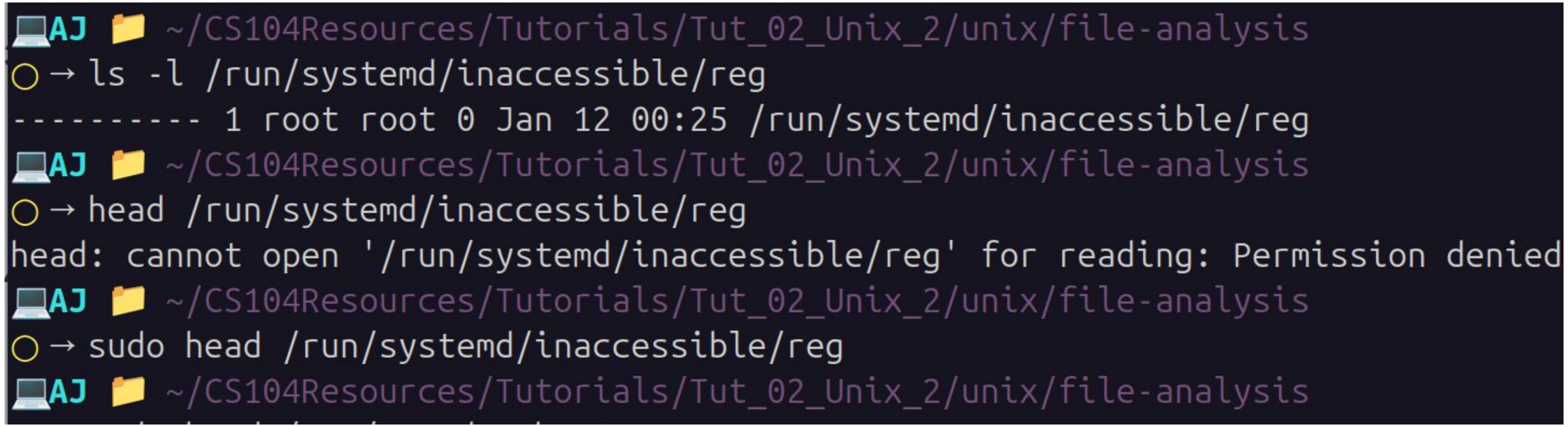


```
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → su cs104
Password:
cs104@abhijain-Zenbook-UX3402VA-UX3402VA:/home/abhijain/CS104Resources/Tutorials
/Tut_02_Unix_2/unix/file-analysis$ ls -l bigfile
-r--r--r-- 1 abhijain abhijain 2465 Jan 10 08:07 bigfile
cs104@abhijain-Zenbook-UX3402VA-UX3402VA:/home/abhijain/CS104Resources/Tutorials
/Tut_02_Unix_2/unix/file-analysis$ head -5 bigfile
Once, there was a boy who became bored when he watched over the village sheep gr
azing on the hillside. To entertain himself, he sang out, “Wolf! Wolf! The wolf
is chasing the sheep!”

When the villagers heard the cry, they came running up the hill to drive the wol
f away. But, when they arrived, they saw no wolf. The boy was amused when seeing
 their angry faces.

“Don’t scream wolf, boy,” warned the villagers, “when there is no wolf!” They an
grily went back down the hill.
cs104@abhijain-Zenbook-UX3402VA-UX3402VA:/home/abhijain/CS104Resources/Tutorials
/Tut_02_Unix_2/unix/file-analysis$ echo 0 > bigfile
bash: bigfile: Permission denied
```

Example: We saw the access controls, now lets test them. We are logged in as abhijain, we will use su to switch to cs104 account. Note that bigfile (owned by abhijain/staff) file had read only permissions for everyone and so we will be able to use head but won’t be able to write to it using echo.

sudo

The sudo allows a permitted user to execute a command as the superuser or another user, as specified by the security policy.



```
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → ls -l /run/systemd/inaccessible/reg
---------- 1 root root 0 Jan 12 00:25 /run/systemd/inaccessible/reg
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → head /run/systemd/inaccessible/reg
head: cannot open '/run/systemd/inaccessible/reg' for reading: Permission denied
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → sudo head /run/systemd/inaccessible/reg
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
```

Example: We will try to read a file with owner as root and no access to group and others and as you would expect, we fail to read the file. Trying the same command with sudo, lets us read the file (Though in this case nothing printed on terminal as file was empty, but if there was something it would have got printed)

Regular Expressions

- grep

Basics of regex

➔ Regex is a powerful tool for finding text according to a particular pattern in a variety of situations. We will use it in grep (today) and sed, awk (in some later week).

➔ There are three basic building blocks when working with regular expressions: *regular characters*, *metacharacters*, and *patterns.*

➔ Meta-characters (some of them are listed below):

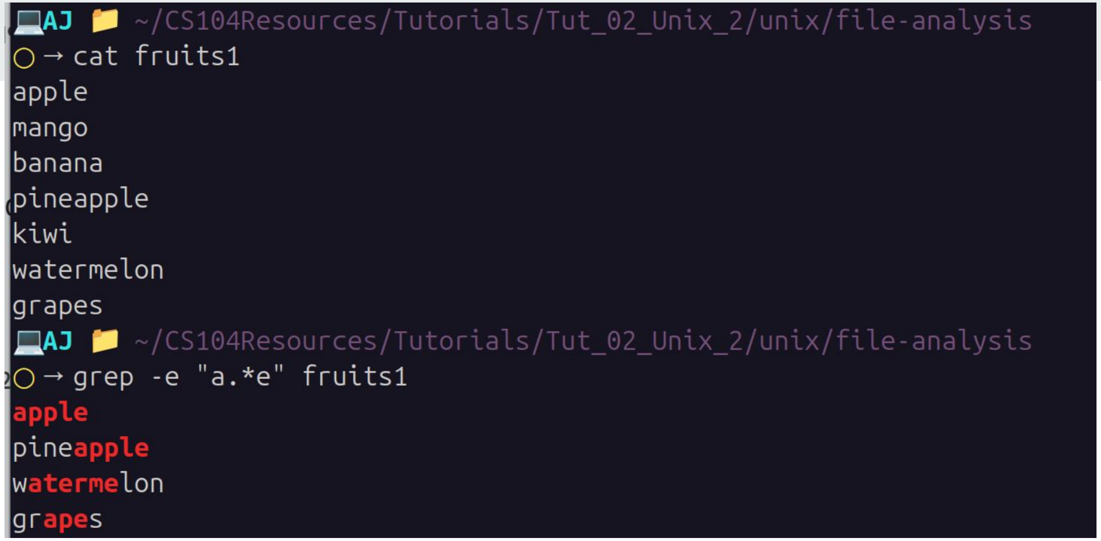
- ^ : start of a line (NOTE: Can also mean "not" if used inside [])
- $: end of line
- . : match any single character
- \ : escape a special character
- | : logical OR operation i.e. match a particular character set on either side
- * : search for a character that occurs zero or more times as defined by the preceding character
- + : search for a character that occurs one or more times as defined by the preceding character
- ? : search for a character that occurs zero or one time as defined by the preceding character
- \d : represents any single numeral, 0 through 9
- \s : represents space

Basics of regex

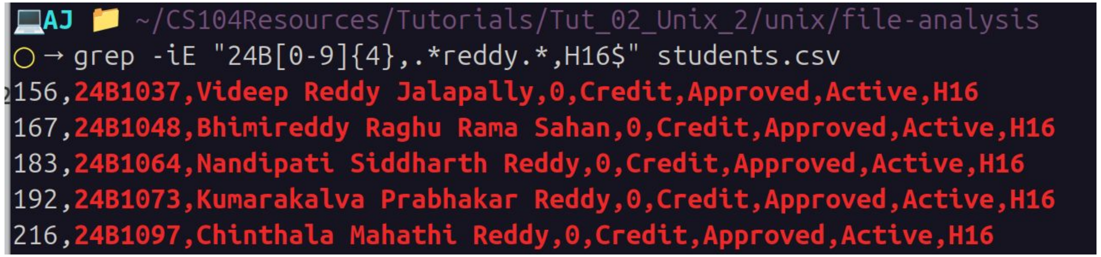
- ➔ A quantifier is a syntactic structure in regular expressions that indicates the number of times a character occurs in sequence in the input text.
- ➔ Some of them are listed below:
  - ◆ {n} : the preceding character needs to occur exactly n times
  - ◆ {n,} : the preceding character needs to occur at least n times
  - ◆ {n,m} : the preceding character needs to occur between n and m times
- ➔ Groups and ranges:
  - ◆ (<def>) : a group of characters declared according to a specific definition
  - ◆ [<range>] : match any character from range of given characters in the []
    - [0-9] : match any digit from 0-9
    - [a-z] : match any lowercase letter from a-z
    - [A-Z] : match any uppercase letter from A-Z
  - ◆ [^<range>] : match any character not in the range of given characters in the []
    - [^adA-Z] : match any character which is not a, d or lies in A-Z

grep

- ❖ The grep utility searches any given input files, selecting lines that match one or more patterns.
- ❖ Some of the useful options are:
  - ➢ -i : Case insensitive match (default is case sensitive)
  - ➢ -e : Specify a pattern used during the search of the input (you can use -e any number of times)
  - ➢ -o : Prints only the matching part of the lines.
  - ➢ Checkout options: -n, -v, -m



```
AJ ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → cat fruits1
apple
mango
banana
pineapple
kiwi
watermelon
grapes
AJ ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → grep -e "a.*e" fruits1
apple
pineapple
watermelon
grapes
```



```
AJ ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → grep -iE "24B[0-9]{4},.*reddy.*,H16$" students.csv
156,24B1037,Videep Reddy Jalapally,0,Credit,Approved,Active,H16
167,24B1048,Bhimireddy Raghu Rama Sahan,0,Credit,Approved,Active,H16
183,24B1064,Nandipati Siddharth Reddy,0,Credit,Approved,Active,H16
192,24B1073,Kumarakalva Prabhakar Reddy,0,Credit,Approved,Active,H16
216,24B1097,Chinthala Mahathi Reddy,0,Credit,Approved,Active,H16
```

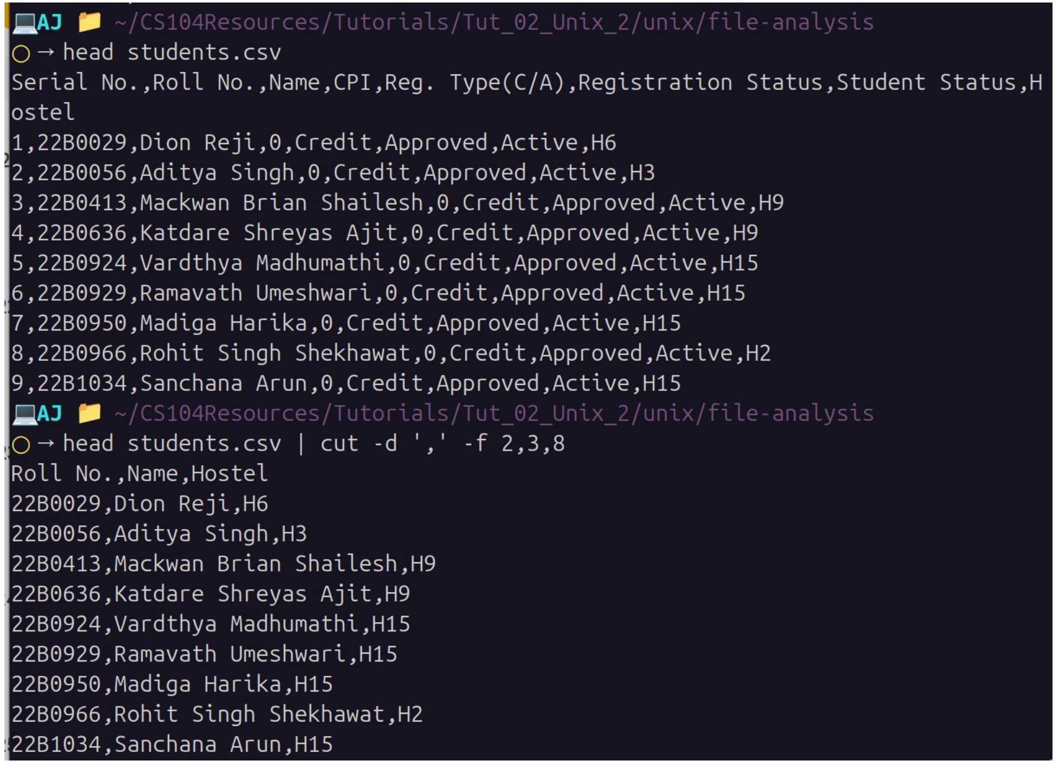
Example:
1. All fruits with a followed by an e.
2. Students from 24 batch, having "reddy" in their name and are from H16.

Some other commands

- cut
- wc
- diff
- sort
- tar
- zip/unzip

cut

- ❖ cut out selected portions of each line of a file
- ❖ Some of the useful options are:
  - ➢ -d : Used to specify field delimiter character (default is tab)
  - ➢ -f : Used to specify the fields desired in the output and separated in the input by field delimiter character.

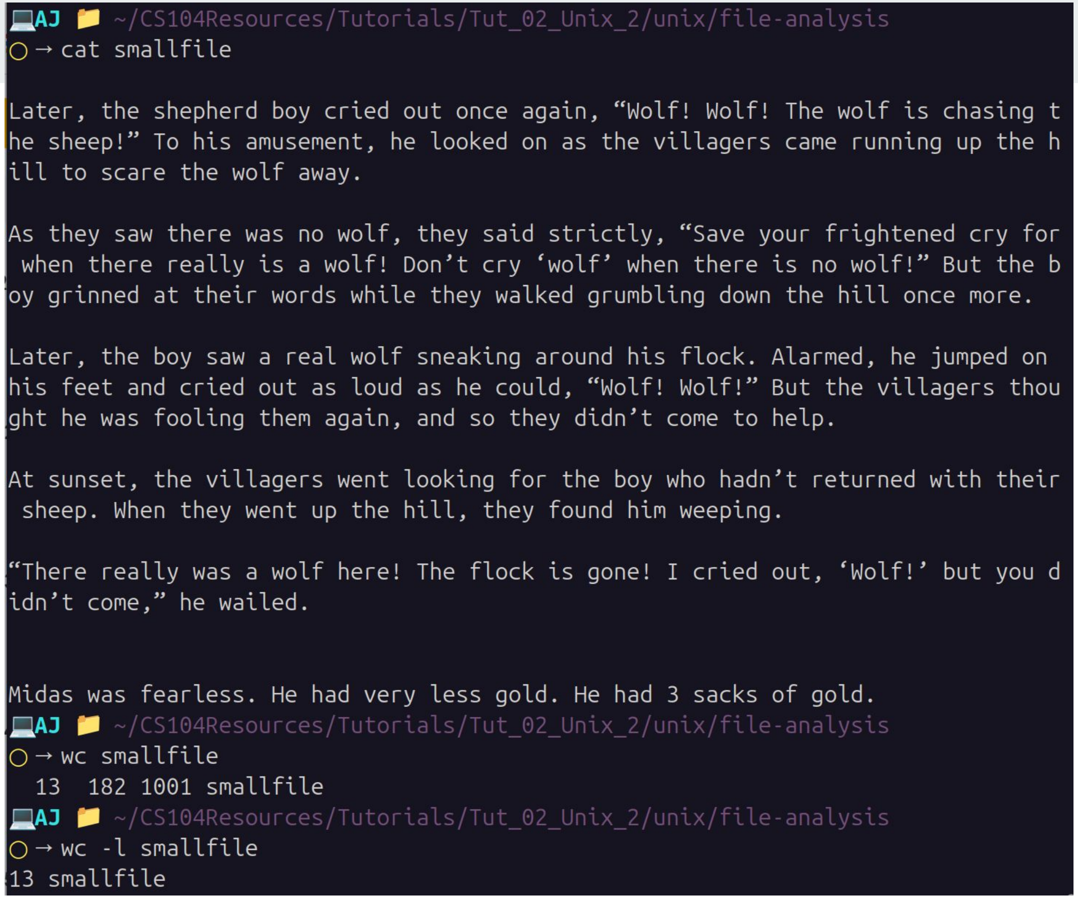


```
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → head students.csv
Serial No.,Roll No.,Name,CPI,Reg. Type(C/A),Registration Status,Student Status,H
ostel
1,22B0029,Dion Reji,0,Credit,Approved,Active,H6
2,22B0056,Aditya Singh,0,Credit,Approved,Active,H3
3,22B0413,Mackwan Brian Shailesh,0,Credit,Approved,Active,H9
4,22B0636,Katdare Shreyas Ajit,0,Credit,Approved,Active,H9
5,22B0924,Vardthya Madhumathi,0,Credit,Approved,Active,H15
6,22B0929,Ramavath Umeshwari,0,Credit,Approved,Active,H15
7,22B0950,Madiga Harika,0,Credit,Approved,Active,H15
8,22B0966,Rohit Singh Shekhawat,0,Credit,Approved,Active,H2
9,22B1034,Sanchana Arun,0,Credit,Approved,Active,H15
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → head students.csv | cut -d ',' -f 2,3,8
Roll No.,Name,Hostel
22B0029,Dion Reji,H6
22B0056,Aditya Singh,H3
22B0413,Mackwan Brian Shailesh,H9
22B0636,Katdare Shreyas Ajit,H9
22B0924,Vardthya Madhumathi,H15
22B0929,Ramavath Umeshwari,H15
22B0950,Madiga Harika,H15
22B0966,Rohit Singh Shekhawat,H2
22B1034,Sanchana Arun,H15
```

Example: From first 10 lines of students.csv file, we extracted 2nd, 3rd and 8th columns corresponding to roll number, hostel and department.

wc

- The wc utility displays the number of lines, words, and bytes contained in each input file
- Some of the useful options are:
  - -l : Used to get the number of lines in each input file
  - -w : Used to get the number of words in each input file
  - Checkout options : -L, -c



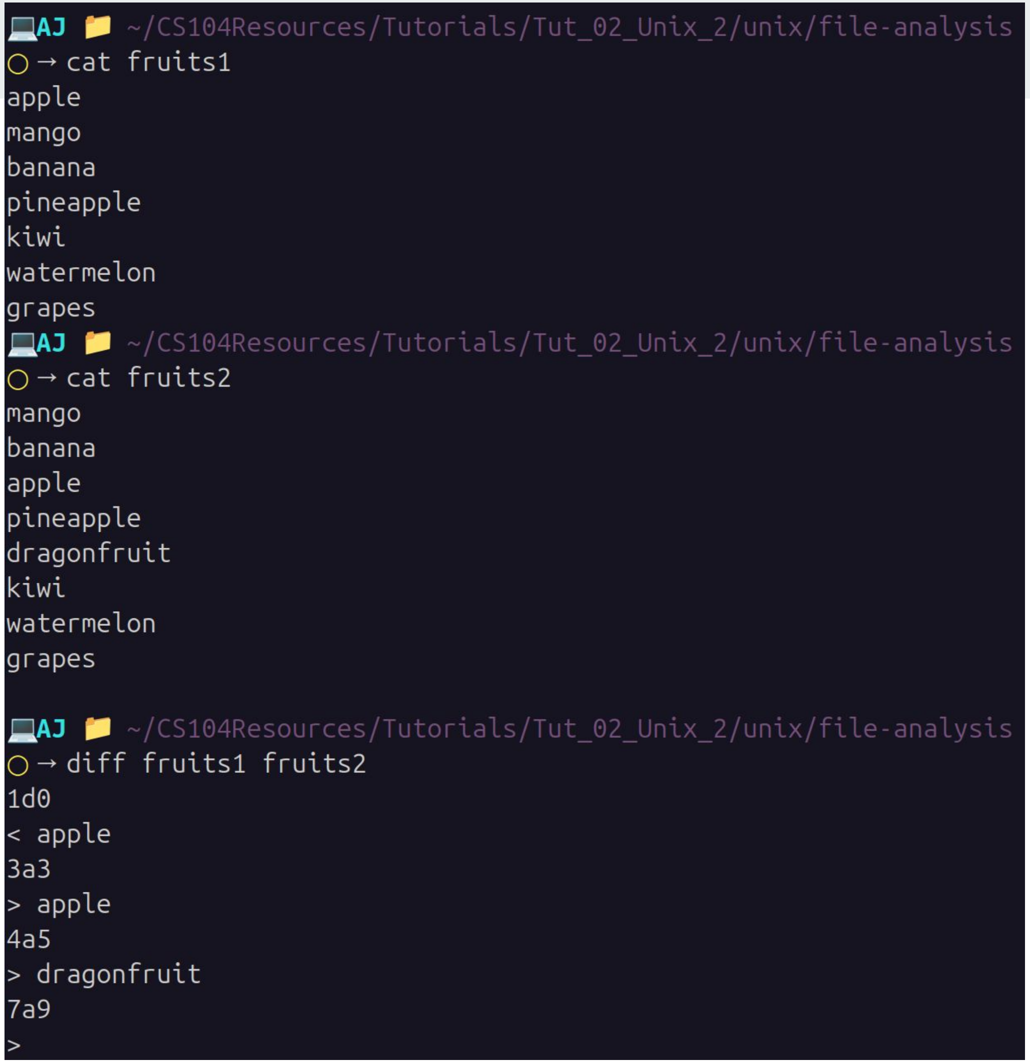
```
AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → cat smallfile

Later, the shepherd boy cried out once again, “Wolf! Wolf! The wolf is chasing t
he sheep!” To his amusement, he looked on as the villagers came running up the h
ill to scare the wolf away.

As they saw there was no wolf, they said strictly, “Save your frightened cry for
 when there really is a wolf! Don’t cry ‘wolf’ when there is no wolf!” But the b
oy grinned at their words while they walked grumbling down the hill once more.

Later, the boy saw a real wolf sneaking around his flock. Alarmed, he jumped on
his feet and cried out as loud as he could, “Wolf! Wolf!” But the villagers thou
ght he was fooling them again, and so they didn’t come to help.

At sunset, the villagers went looking for the boy who hadn’t returned with their
 sheep. When they went up the hill, they found him weeping.

“There really was a wolf here! The flock is gone! I cried out, ‘Wolf!’ but you d
idn’t come,” he wailed.


Midas was fearless. He had very less gold. He had 3 sacks of gold.
AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → wc smallfile
  13  182 1001 smallfile
AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → wc -l smallfile
13 smallfile
```

Example: By default wc outputs #lines, #words, #characters present in the input files.

diff

- The diff utility compares the contents of file1 and file2 and writes to the standard output the list of changes necessary to convert one file into the other.
- No output is produced if the files are identical.
- Checkout options : -B, -w

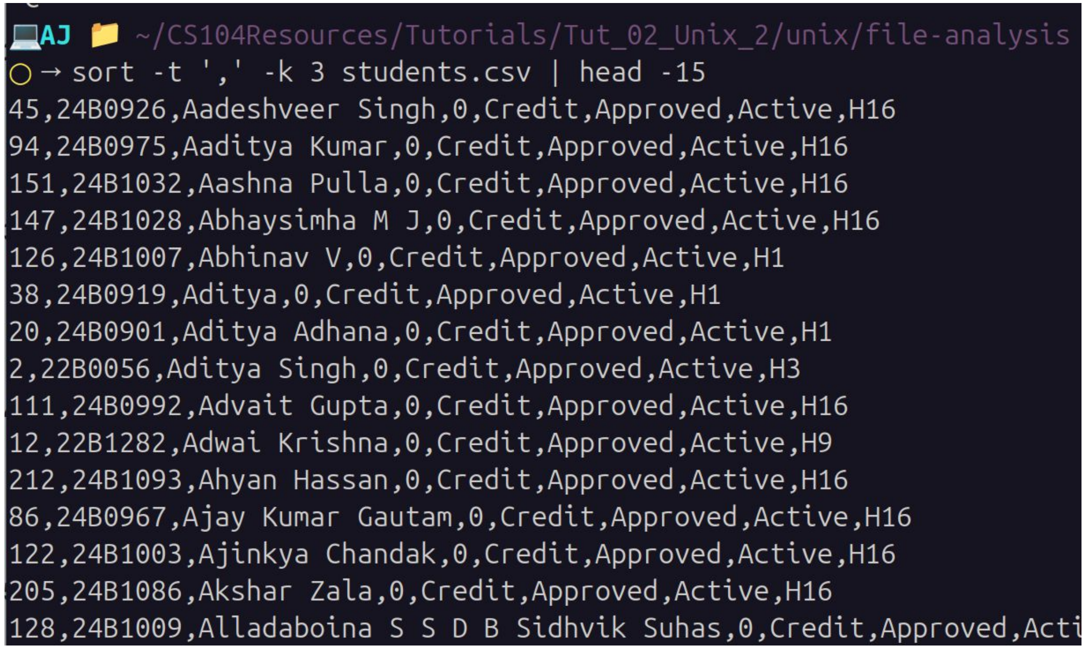
We compared the files fruits1 and fruits2, the diff command instructs how to convert file1 to file2.



```
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → cat fruits1
apple
mango
banana
pineapple
kiwi
watermelon
grapes
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → cat fruits2
mango
banana
apple
pineapple
dragonfruit
kiwi
watermelon
grapes

💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → diff fruits1 fruits2
1d0
< apple
3a3
> apple
4a5
> dragonfruit
7a9
>
```

sort

- The sort utility sorts text and binary files by lines. A line is a record separated from the subsequent record by a newline (default).
- Some of the useful options are:
  - -t : Used to specify field separator character.
  - -r : Used to sort in reverse order.
  - -k : Used to specify the field(s) that will be used as sort key(s).
  - Checkout options : -c

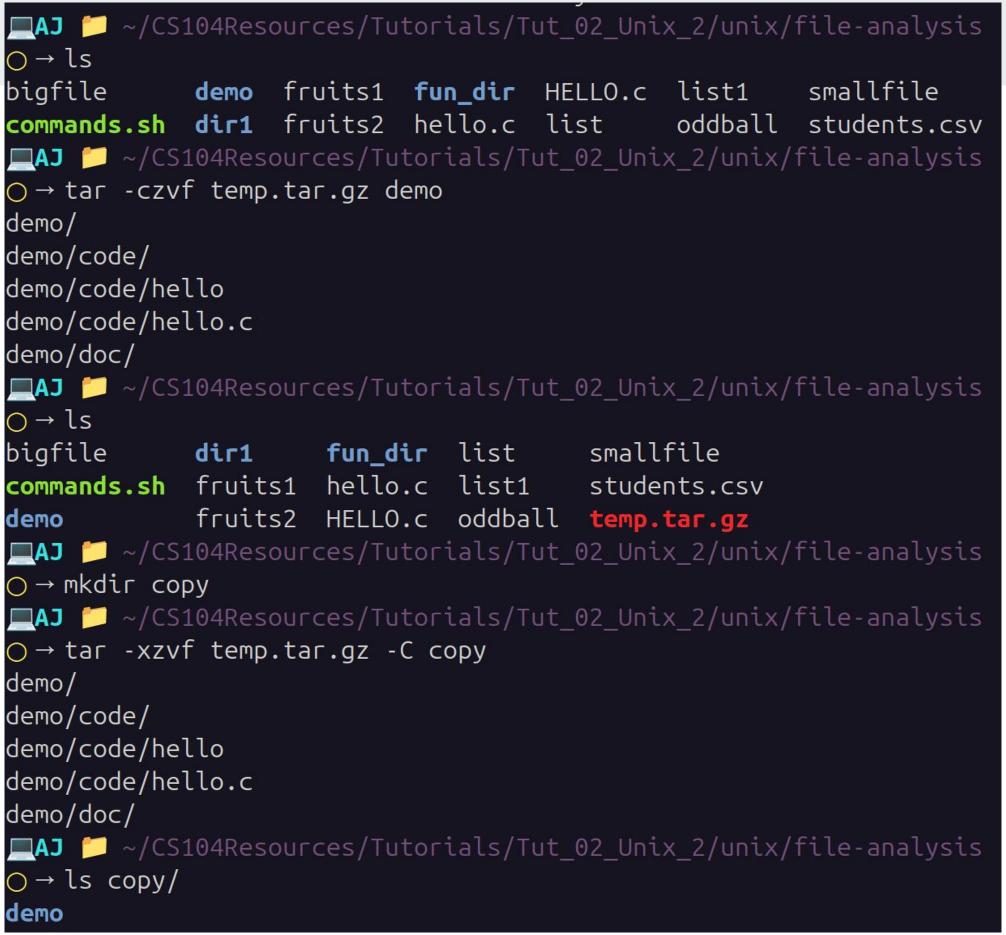


```
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → sort -t ',' -k 3 students.csv | head -15
45,24B0926,Aadeshveer Singh,0,Credit,Approved,Active,H16
94,24B0975,Aaditya Kumar,0,Credit,Approved,Active,H16
151,24B1032,Aashna Pulla,0,Credit,Approved,Active,H16
147,24B1028,Abhaysimha M J,0,Credit,Approved,Active,H16
126,24B1007,Abhinav V,0,Credit,Approved,Active,H1
38,24B0919,Aditya,0,Credit,Approved,Active,H1
20,24B0901,Aditya Adhana,0,Credit,Approved,Active,H1
2,22B0056,Aditya Singh,0,Credit,Approved,Active,H3
111,24B0992,Advait Gupta,0,Credit,Approved,Active,H16
12,22B1282,Adwai Krishna,0,Credit,Approved,Active,H9
212,24B1093,Ahyan Hassan,0,Credit,Approved,Active,H16
86,24B0967,Ajay Kumar Gautam,0,Credit,Approved,Active,H16
122,24B1003,Ajinkya Chandak,0,Credit,Approved,Active,H16
205,24B1086,Akshar Zala,0,Credit,Approved,Active,H16
128,24B1009,Alladaboina S S D B Sidhvik Suhas,0,Credit,Approved,Acti
```

Example: Sort the students in students.csv according to name.
Exercise: Write commands to not sort header and make it appear in first line irrespective of sort.

tar

- tar creates and manipulates streaming archive files.
- To tar a folder, we will use the following command
  `tar -cvzf <output file> <folder>`
- To untar, we will use the following command
  `tar -xvzf <tar file>`
- Checkout the above options if you are curious, namely
  -c, -v, -x, -z, -f

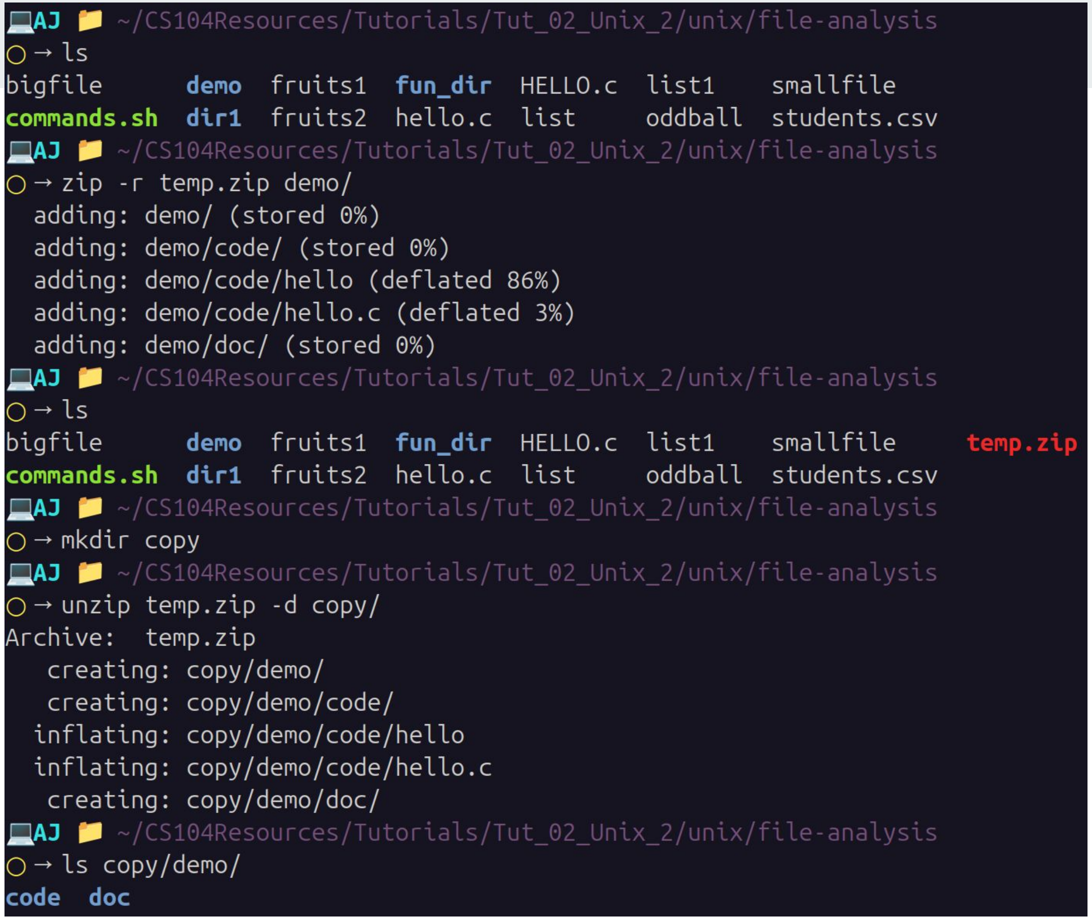


```
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → ls
bigfile      demo  fruits1  fun_dir  HELLO.c  list1    smallfile
commands.sh  dir1  fruits2  hello.c  list     oddball  students.csv
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → tar -czvf temp.tar.gz demo
demo/
demo/code/
demo/code/hello
demo/code/hello.c
demo/doc/
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → ls
bigfile      dir1     fun_dir  list     smallfile
commands.sh  fruits1  hello.c  list1    students.csv
demo         fruits2  HELLO.c  oddball  temp.tar.gz
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → mkdir copy
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → tar -xzvf temp.tar.gz -C copy
demo/
demo/code/
demo/code/hello
demo/code/hello.c
demo/doc/
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → ls copy/
demo
```

Example: We first tar demo, then we untar it into copy folder using -C option, (by default pwd is used for untarring)

zip/unzip

- zip is a compression and file packaging utility for Unix
- To zip a folder recursively, we will use the following command
  `zip -r <zip file> <folder>`
- unzip - list, test and extract compressed files in a zip archive
- To unzip, we will use the following command
  `unzip <zip file>`



```
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → ls
bigfile      demo   fruits1  fun_dir  HELLO.c  list1    smallfile
commands.sh  dir1   fruits2  hello.c  list     oddball  students.csv
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → zip -r temp.zip demo/
  adding: demo/ (stored 0%)
  adding: demo/code/ (stored 0%)
  adding: demo/code/hello (deflated 86%)
  adding: demo/code/hello.c (deflated 3%)
  adding: demo/doc/ (stored 0%)
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → ls
bigfile      demo   fruits1  fun_dir  HELLO.c  list1    smallfile     temp.zip
commands.sh  dir1   fruits2  hello.c  list     oddball  students.csv
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → mkdir copy
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → unzip temp.zip -d copy/
Archive:  temp.zip
   creating: copy/demo/
   creating: copy/demo/code/
  inflating: copy/demo/code/hello
  inflating: copy/demo/code/hello.c
   creating: copy/demo/doc/
💻AJ 📁 ~/CS104Resources/Tutorials/Tut_02_Unix_2/unix/file-analysis
○ → ls copy/demo/
code  doc
```

Example: We first zip demo, then we unzip it into copy folder using -d option, (by default pwd is used for unzipping)

Exercises

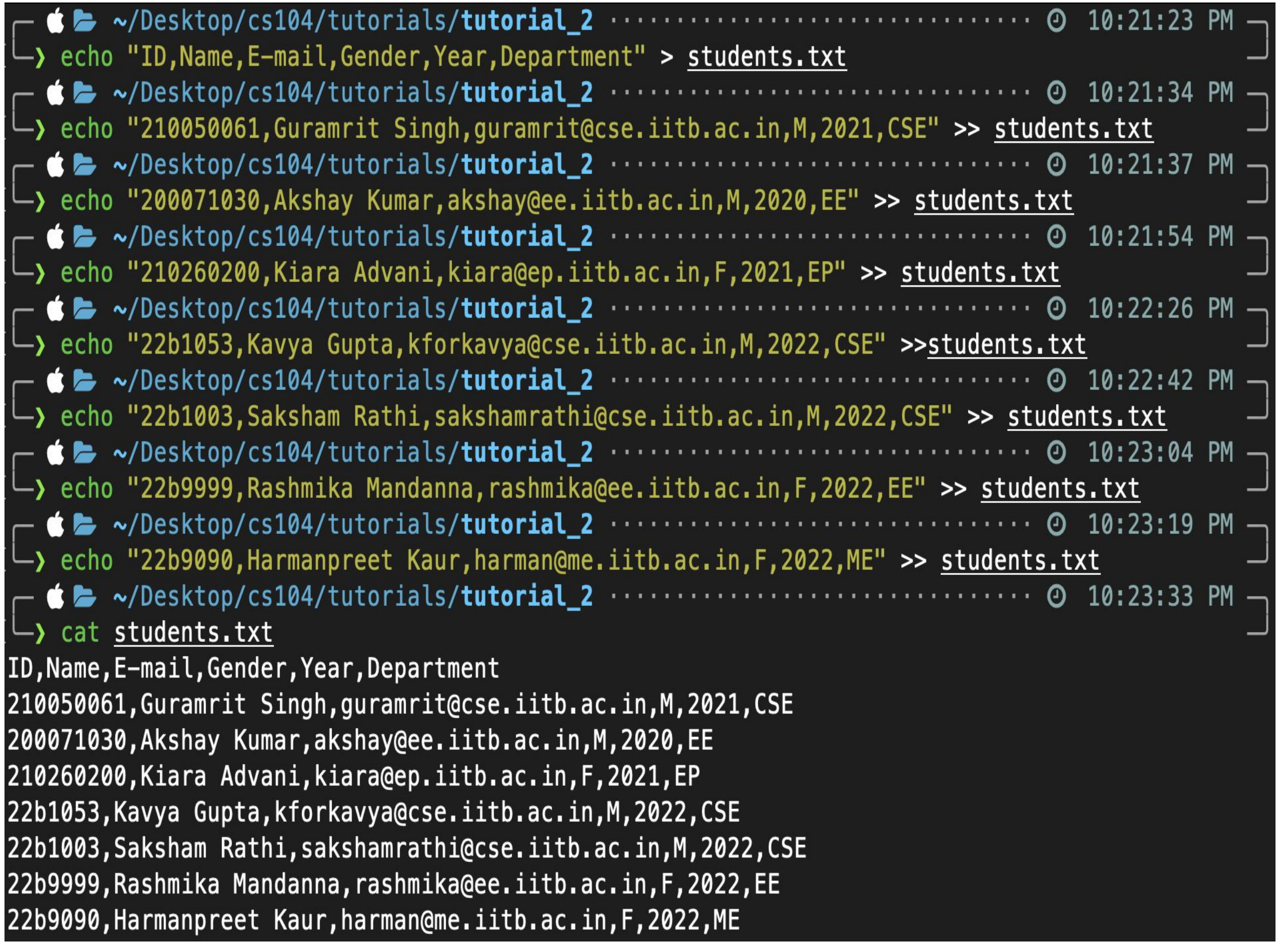
Exercise 1

Create a file students.txt with the following content using commands learnt so far.

```
ID,Name,E-mail,Gender,Year,Department
210050061,Guramrit Singh,guramrit@cse.iitb.ac.in,M,2021,CSE
200071030,Akshay Kumar,akshay@ee.iitb.ac.in,M,2020,EE
210260200,Kiara Advani,kiara@ep.iitb.ac.in,F,2021,EP
22b1053,Kavya Gupta,kforkavya@cse.iitb.ac.in,M,2022,CSE
22b1003,Saksham Rathi,sakshamrathi@cse.iitb.ac.in,M,2022,CSE
22b9999,Rashmika Mandanna,rashmika@ee.iitb.ac.in,F,2022,EE
22b9090,Harmanpreet Kaur,harman@me.iitb.ac.in,F,2022,ME
```

Solution 1

We use echo with redirection operator to accomplish the task.



```
~/Desktop/cs104/tutorials/tutorial_2 ································ 10:21:23 PM
❯ echo "ID,Name,E-mail,Gender,Year,Department" > students.txt
~/Desktop/cs104/tutorials/tutorial_2 ································ 10:21:34 PM
❯ echo "210050061,Guramrit Singh,guramrit@cse.iitb.ac.in,M,2021,CSE" >> students.txt
~/Desktop/cs104/tutorials/tutorial_2 ································ 10:21:37 PM
❯ echo "200071030,Akshay Kumar,akshay@ee.iitb.ac.in,M,2020,EE" >> students.txt
~/Desktop/cs104/tutorials/tutorial_2 ································ 10:21:54 PM
❯ echo "210260200,Kiara Advani,kiara@ep.iitb.ac.in,F,2021,EP" >> students.txt
~/Desktop/cs104/tutorials/tutorial_2 ································ 10:22:26 PM
❯ echo "22b1053,Kavya Gupta,kforkavya@cse.iitb.ac.in,M,2022,CSE" >>students.txt
~/Desktop/cs104/tutorials/tutorial_2 ································ 10:22:42 PM
❯ echo "22b1003,Saksham Rathi,sakshamrathi@cse.iitb.ac.in,M,2022,CSE" >> students.txt
~/Desktop/cs104/tutorials/tutorial_2 ································ 10:23:04 PM
❯ echo "22b9999,Rashmika Mandanna,rashmika@ee.iitb.ac.in,F,2022,EE" >> students.txt
~/Desktop/cs104/tutorials/tutorial_2 ································ 10:23:19 PM
❯ echo "22b9090,Harmanpreet Kaur,harman@me.iitb.ac.in,F,2022,ME" >> students.txt
~/Desktop/cs104/tutorials/tutorial_2 ································ 10:23:33 PM
❯ cat students.txt
ID,Name,E-mail,Gender,Year,Department
210050061,Guramrit Singh,guramrit@cse.iitb.ac.in,M,2021,CSE
200071030,Akshay Kumar,akshay@ee.iitb.ac.in,M,2020,EE
210260200,Kiara Advani,kiara@ep.iitb.ac.in,F,2021,EP
22b1053,Kavya Gupta,kforkavya@cse.iitb.ac.in,M,2022,CSE
22b1003,Saksham Rathi,sakshamrathi@cse.iitb.ac.in,M,2022,CSE
22b9999,Rashmika Mandanna,rashmika@ee.iitb.ac.in,F,2022,EE
22b9090,Harmanpreet Kaur,harman@me.iitb.ac.in,F,2022,ME
```

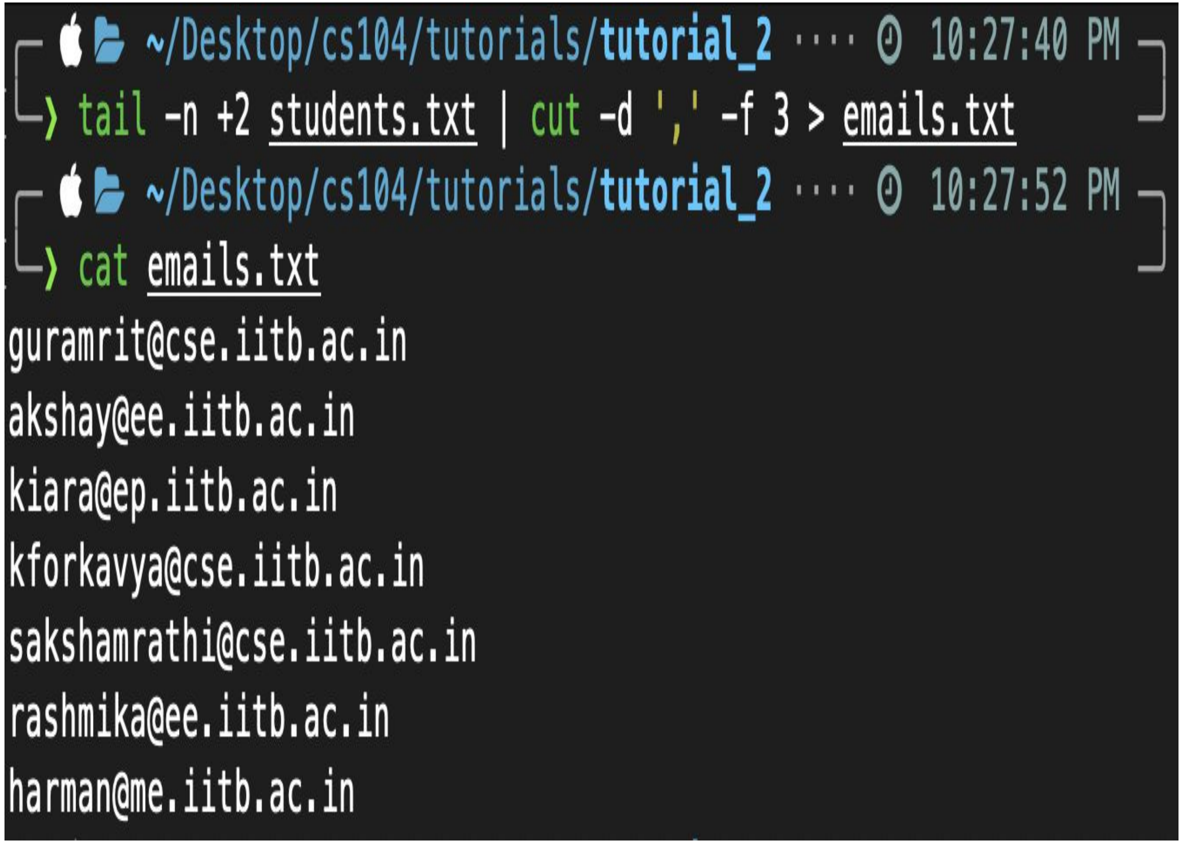
Exercise 2

Given the file students.txt, create a file named email.txt storing email ids of all the students

Solution 2

To accomplish the task, we do the following:

1. Use tail with *(-n +2)*, so as to get rid of first line.
2. Use cut to break the line into tokens with delimiter set to ',' and take the 3rd column, and redirect it to emails.txt file.



```
┌  ~/Desktop/cs104/tutorials/tutorial_2 ···· ⊙ 10:27:40 PM ┐
└❯ tail -n +2 students.txt | cut -d ',' -f 3 > emails.txt   ┘
┌  ~/Desktop/cs104/tutorials/tutorial_2 ···· ⊙ 10:27:52 PM ┐
└❯ cat emails.txt                                            ┘
guramrit@cse.iitb.ac.in
akshay@ee.iitb.ac.in
kiara@ep.iitb.ac.in
kforkavya@cse.iitb.ac.in
sakshamrathi@cse.iitb.ac.in
rashmika@ee.iitb.ac.in
harman@me.iitb.ac.in
```

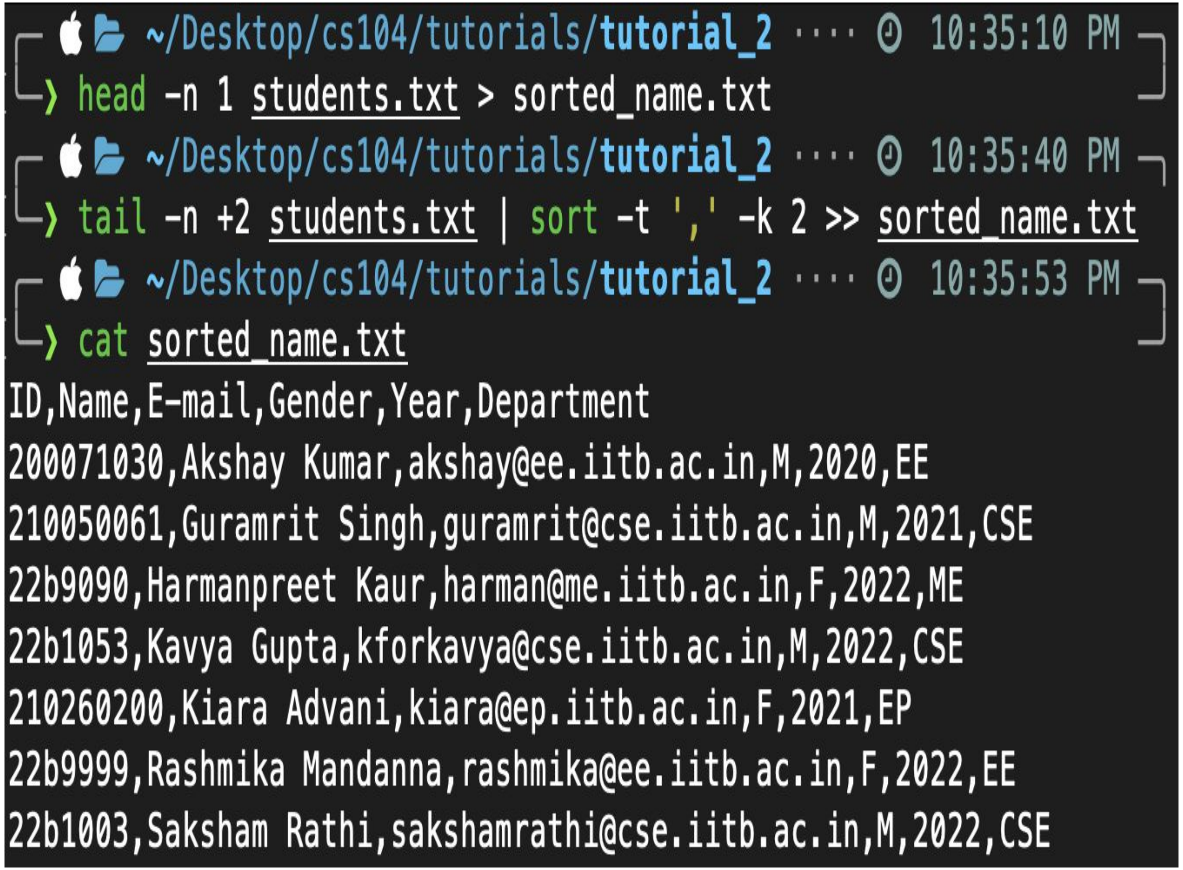
Exercise 3

Create a file named sorted_name.txt from the contents of students.txt, arranging student information with the header intact. Ensure that the names of the students are organized in lexicographical order.

Solution 3

To accomplish the task, we do the following:

1. Firstly, using head, we copy the header to sorted_name.txt
2. Then using tail with *(-n +2)*, we get rid of header, then sort these lines based on 2nd column delimited by ‘,’ and finally redirecting output to sorted_name.txt

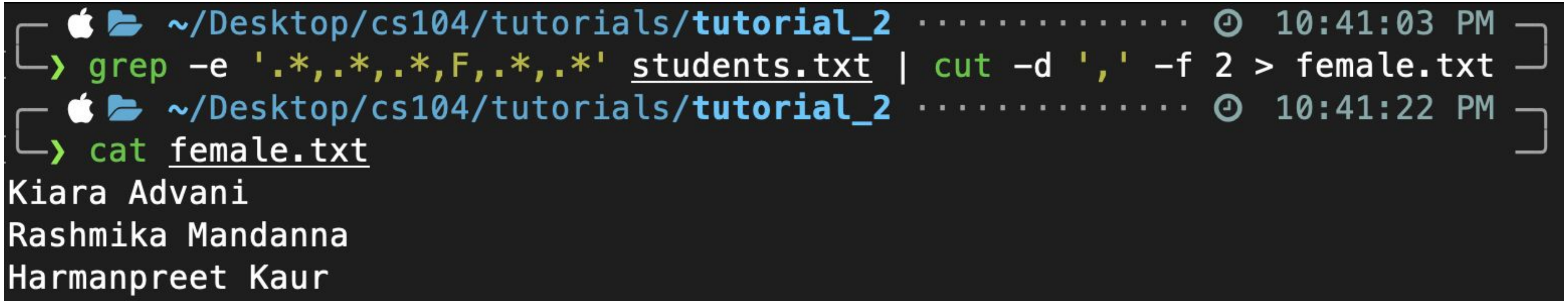


```
╭─  ~/Desktop/cs104/tutorials/tutorial_2 ···· 10:35:10 PM ─╮
╰─❯ head -n 1 students.txt > sorted_name.txt                ─╯
╭─  ~/Desktop/cs104/tutorials/tutorial_2 ···· 10:35:40 PM ─╮
╰─❯ tail -n +2 students.txt | sort -t ',' -k 2 >> sorted_name.txt
╭─  ~/Desktop/cs104/tutorials/tutorial_2 ···· 10:35:53 PM ─╮
╰─❯ cat sorted_name.txt                                     ─╯
ID,Name,E-mail,Gender,Year,Department
200071030,Akshay Kumar,akshay@ee.iitb.ac.in,M,2020,EE
210050061,Guramrit Singh,guramrit@cse.iitb.ac.in,M,2021,CSE
22b9090,Harmanpreet Kaur,harman@me.iitb.ac.in,F,2022,ME
22b1053,Kavya Gupta,kforkavya@cse.iitb.ac.in,M,2022,CSE
210260200,Kiara Advani,kiara@ep.iitb.ac.in,F,2021,EP
22b9999,Rashmika Mandanna,rashmika@ee.iitb.ac.in,F,2022,EE
22b1003,Saksham Rathi,sakshamrathi@cse.iitb.ac.in,M,2022,CSE
```

Exercise 4

Given the file students.txt, create a file named female.txt storing name of all the female students.

Solution 4



```
╭─  ~/Desktop/cs104/tutorials/tutorial_2 ··············· 10:41:03 PM ─╮
╰─❯ grep -e '.*,.*,.*,F,.*,.*' students.txt | cut -d ',' -f 2 > female.txt ─╯
╭─  ~/Desktop/cs104/tutorials/tutorial_2 ··············· 10:41:22 PM ─╮
╰─❯ cat female.txt ─╯
Kiara Advani
Rashmika Mandanna
Harmanpreet Kaur
```

To accomplish the task, we do the following:

1. We use grep (with -e) to find all lines where 4th field is ‘F’.
2. The regex `.*,.*,.*,F,.*,.*` is matched by all lines where 4th field is ‘F’ and other fields can be anything denoted by `.*`
3. Finally, we get the 2nd column (i.e. the name) of all female students and redirect it to female.txt file.

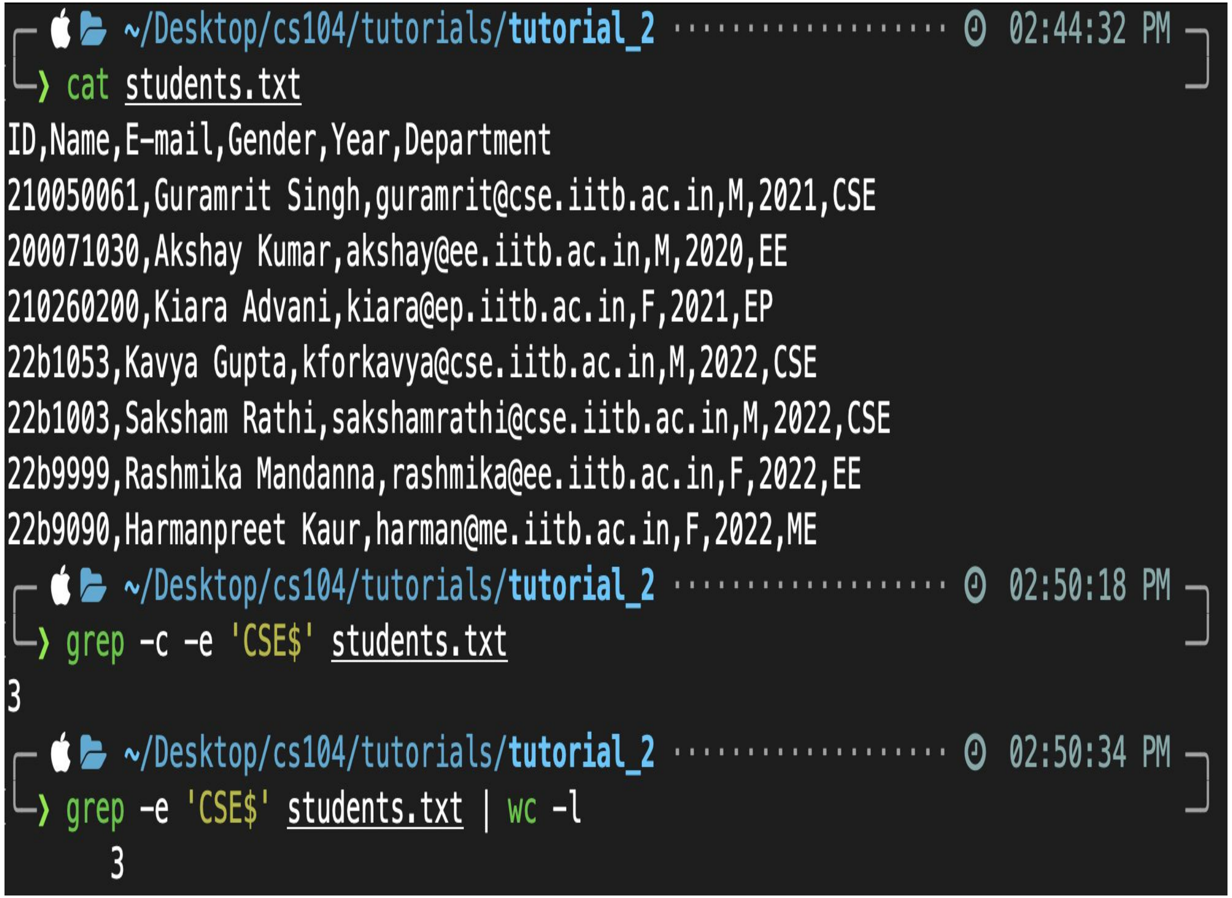
Exercise 5

Given the file students.txt, find the number of students in the CSE department.

Solution 5

To accomplish the task, we have two ways:

1. You can use grep with -c option.
2. You can find all CSE students using grep and then use wc -l to find the count.

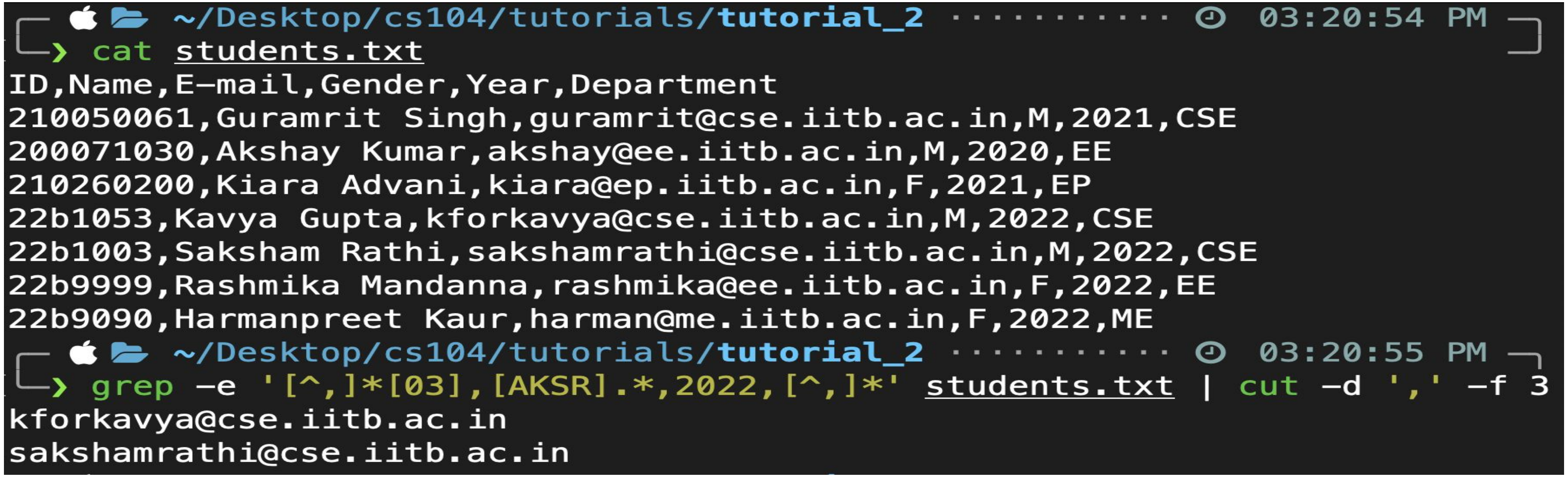


```
┌  ~/Desktop/cs104/tutorials/tutorial_2 ·················· 02:44:32 PM ┐
└─❯ cat students.txt                                                    ┘
ID,Name,E-mail,Gender,Year,Department
210050061,Guramrit Singh,guramrit@cse.iitb.ac.in,M,2021,CSE
200071030,Akshay Kumar,akshay@ee.iitb.ac.in,M,2020,EE
210260200,Kiara Advani,kiara@ep.iitb.ac.in,F,2021,EP
22b1053,Kavya Gupta,kforkavya@cse.iitb.ac.in,M,2022,CSE
22b1003,Saksham Rathi,sakshamrathi@cse.iitb.ac.in,M,2022,CSE
22b9999,Rashmika Mandanna,rashmika@ee.iitb.ac.in,F,2022,EE
22b9090,Harmanpreet Kaur,harman@me.iitb.ac.in,F,2022,ME
┌  ~/Desktop/cs104/tutorials/tutorial_2 ·················· 02:50:18 PM ┐
└─❯ grep -c -e 'CSE$' students.txt                                      ┘
3
┌  ~/Desktop/cs104/tutorials/tutorial_2 ·················· 02:50:34 PM ┐
└─❯ grep -e 'CSE$' students.txt | wc -l                                 ┘
       3
```

Exercise 6

Given the file students.txt, find the email id of all students whose name starts with either of A/K/S/R, belonging to 2022 batch and have their ID ending with either 0 or 3.

Solution 6



```
╭─  ~/Desktop/cs104/tutorials/tutorial_2 ············· ◷ 03:20:54 PM ─╮
╰─❯ cat students.txt                                                  ─╯
ID,Name,E-mail,Gender,Year,Department
210050061,Guramrit Singh,guramrit@cse.iitb.ac.in,M,2021,CSE
200071030,Akshay Kumar,akshay@ee.iitb.ac.in,M,2020,EE
210260200,Kiara Advani,kiara@ep.iitb.ac.in,F,2021,EP
22b1053,Kavya Gupta,kforkavya@cse.iitb.ac.in,M,2022,CSE
22b1003,Saksham Rathi,sakshamrathi@cse.iitb.ac.in,M,2022,CSE
22b9999,Rashmika Mandanna,rashmika@ee.iitb.ac.in,F,2022,EE
22b9090,Harmanpreet Kaur,harman@me.iitb.ac.in,F,2022,ME
╭─  ~/Desktop/cs104/tutorials/tutorial_2 ············· ◷ 03:20:55 PM ─╮
╰─❯ grep -e '[^,]*[03],[AKSR].*,2022,[^,]*' students.txt | cut -d ',' -f 3
kforkavya@cse.iitb.ac.in
sakshamrathi@cse.iitb.ac.in
```

To accomplish the task, we do the following:

1. First we use grep to find all such matches. Carefully look at the regex used for pattern matching.
2. Then we use cut to get the email ids.

Thank You !!!