Unix Commands Assignment

Slide no. 41

1. Create folder 'Test' in your home directory

Ans: mkdir Test

2. Create below files

- employeelist
- skillset

Ans: cat >employeelist

cat >skillset

3. Create folder Test2 in home directory

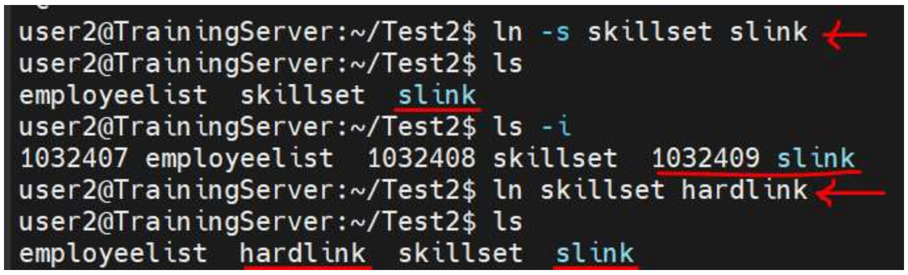
A: mkdir Test2

3.a. create symbolink and hard link for skillset file

A: command to create symbolink:- ln -s skillset slink

command to create hardlink:- ln skillset hardlink

The output will look like below image:



```
user2@TrainingServer:~/Test2$ ln -s skillset slink
user2@TrainingServer:~/Test2$ ls
employeelist  skillset  slink
user2@TrainingServer:~/Test2$ ls -i
1032407 employeelist  1032408 skillset  1032409 slink
user2@TrainingServer:~/Test2$ ln skillset hardlink
user2@TrainingServer:~/Test2$ ls
employeelist  hardlink  skillset  slink
```

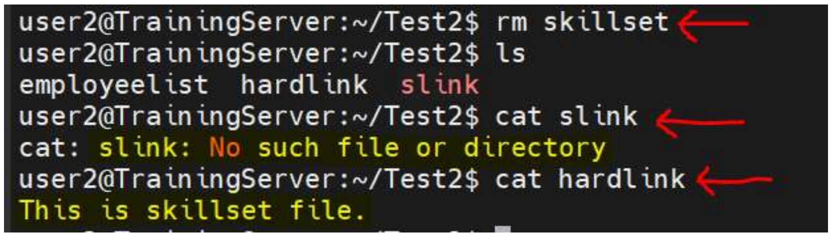
3.b delete skillset file and check if hardlink and symbolic link file exists

A: command to del skillset: rm skillset

command to check slink: cat slink

command to check hardkink: cat hardlink

The output will look like:



```
user2@TrainingServer:~/Test2$ rm skillset
user2@TrainingServer:~/Test2$ ls
employeelist  hardlink  slink
user2@TrainingServer:~/Test2$ cat slink
cat: slink: No such file or directory
user2@TrainingServer:~/Test2$ cat hardlink
This is skillset file.
```

Symbolic link will not execute when the linked file is deleted but hardlink will execute the linked file contents even after deleting the file.

3.c Give difference between symbolic link and hard link.

A: A hard link is essentially a synced carbon copy of a file that refers directly to the inode of a file.A hard link is a direct reference to a file via its inode.By using a hardlink, you can change the original file's contents or location and the hardlink will still point to the original file because its inode is still pointing to that file.

Symbolic links on the other hand refer directly to the file which refers to the inode, a shortcut. Symbolic links are essentially shortcuts that reference to a file instead of its inode value.Since the symbolic link is referring to the original file and not its inode value, then replacing the original file into a different folder will break the symbolic link.

4. Provide read and write permission to 'user2' group

A: command: sudo chgrp user2 filename

5. Revoke write permission from user2 group

A: command: chmod 744 demofile

(where demofile is the filename)

6.Grant write permission for others and test it

A: Command: chmod 746 demofile

7. Revoke write permission for others

A: Command: chmod 744 demofile

8. Change the group of employeelist file to 'user3'

A: command: sudo chgrp user3 employeelist

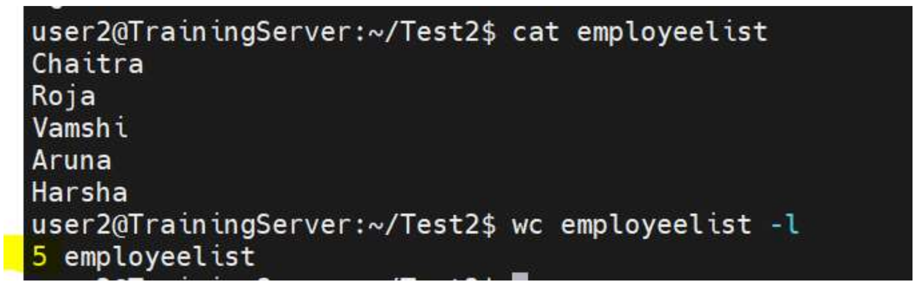
9. Change ownership of the file to user5

A: command: chown user5 employeelist

10. check how many employees are there

A: command: wc employeelist -l

The output looks like:



```
user2@TrainingServer:~/Test2$ cat employeelist
Chaitra
Roja
Vamshi
Aruna
Harsha
user2@TrainingServer:~/Test2$ wc employeelist -l
5 employeelist
```

Slide no. 47

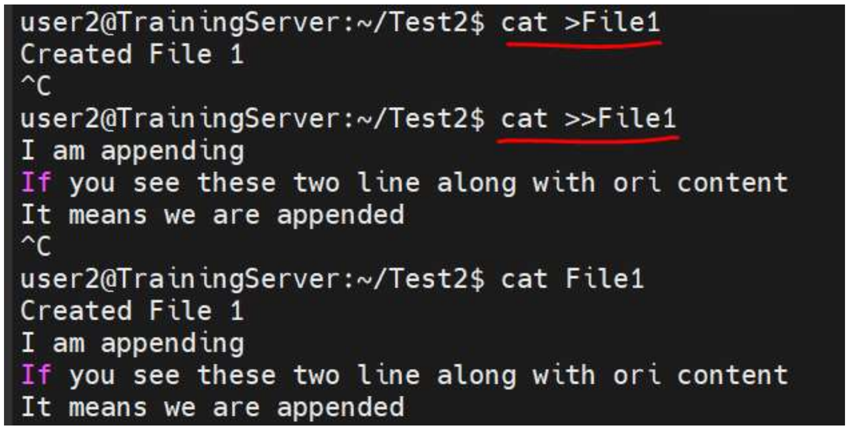
1. Create a File1

A: command: cat >File1

2. Append 2 more lines to the same file

A: command:cat >>File1

The output of above two commands is



```
user2@TrainingServer:~/Test2$ cat >File1
Created File 1
^C
user2@TrainingServer:~/Test2$ cat >>File1
I am appending
If you see these two line along with ori content
It means we are appended
^C
user2@TrainingServer:~/Test2$ cat File1
Created File 1
I am appending
If you see these two line along with ori content
It means we are appended
```

3. Create
file two with few lines

A: command: cat >File2

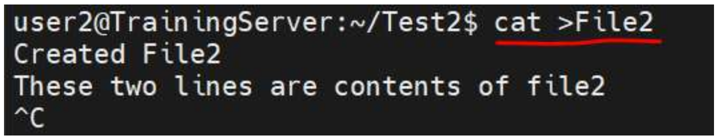
Created file2

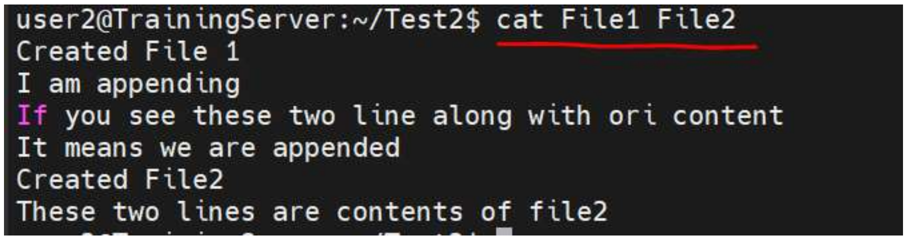
These two lines are initial contents of File2

4. Display contents of both File1 and File2

A: command: cat File1 File2

The output of above two commands:


```
user2@TrainingServer:~/Test2$ cat >File2
Created File2
These two lines are contents of file2
^C
```


```
user2@TrainingServer:~/Test2$ cat File1 File2
Created File 1
I am appending
If you see these two line along with ori content
It means we are appended
Created File2
These two lines are contents of file2
```

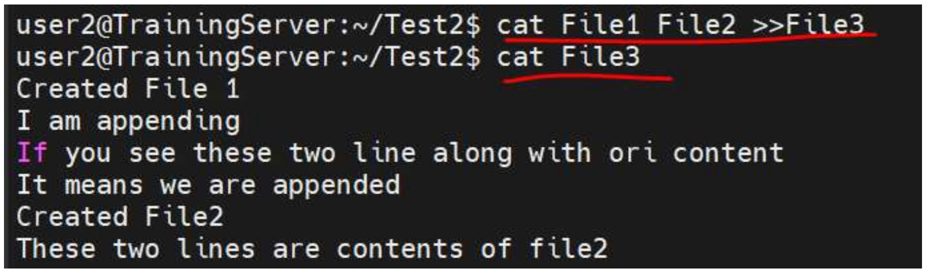
5. Concatenate both File1 & 2

A: command: cat File1 File2

6. Send the output to File3

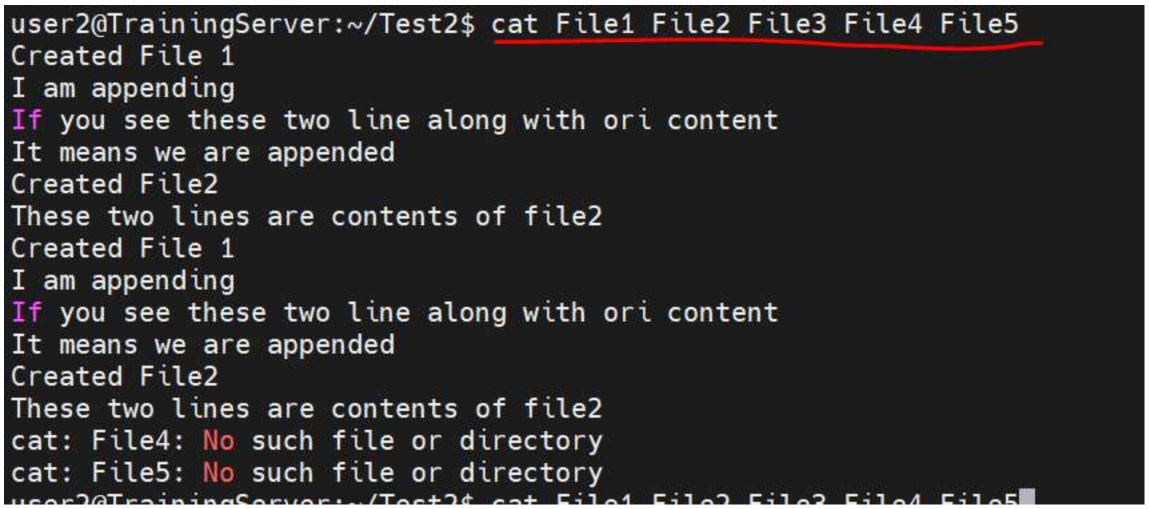
A: command: cat File1 File2 >>File3

The ouput for 6. is:


```
user2@TrainingServer:~/Test2$ cat File1 File2 >>File3
user2@TrainingServer:~/Test2$ cat File3
Created File 1
I am appending
If you see these two line along with ori content
It means we are appended
Created File2
These two lines are contents of file2
```

7. Read File1, File2, File3....File5

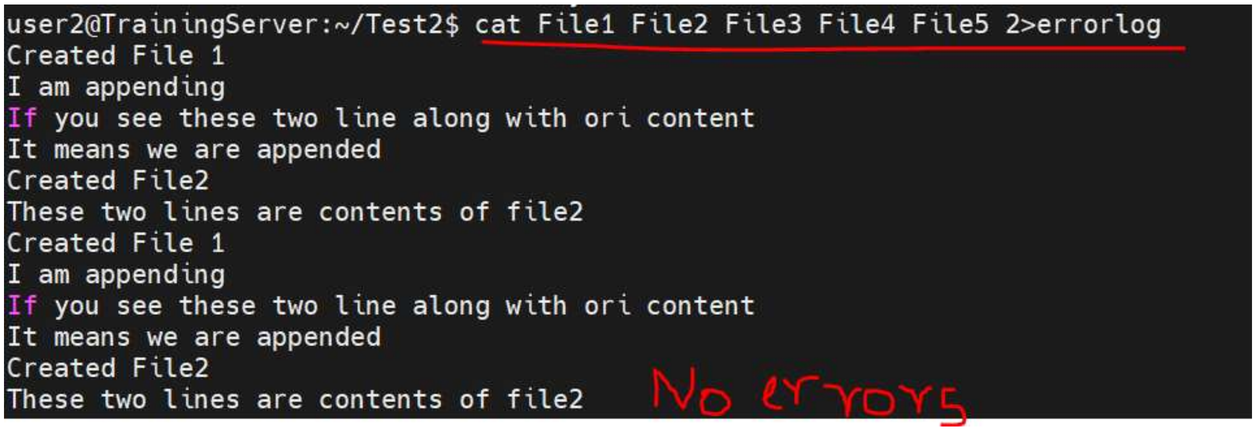
A: command: cat File1, File2, File3, File4, File5



```
user2@TrainingServer:~/Test2$ cat File1 File2 File3 File4 File5
Created File 1
I am appending
If you see these two line along with ori content
It means we are appended
Created File2
These two lines are contents of file2
Created File 1
I am appending
If you see these two line along with ori content
It means we are appended
Created File2
These two lines are contents of file2
cat: File4: No such file or directory
cat: File5: No such file or directory
user2@TrainingServer:~/Test2$ cat File1 File2 File3 File4 File5
```

8.Redirect the errors of the above command to "errorlog"

A: command: cat File1, File2, File3, File4, File5 2>errorlog

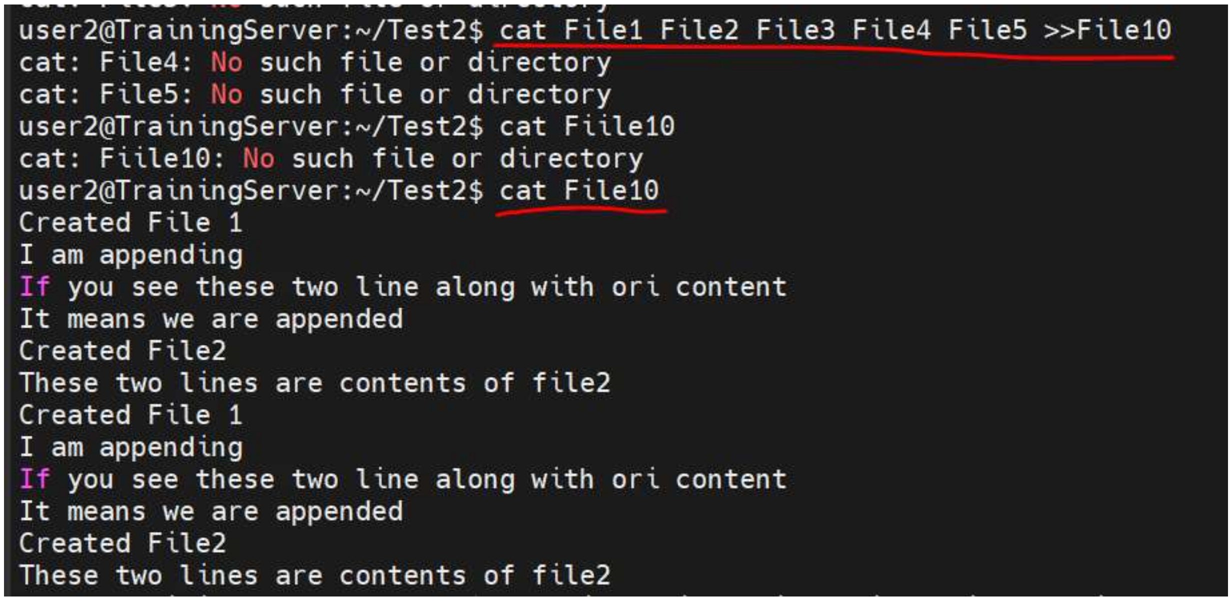


```
user2@TrainingServer:~/Test2$ cat File1 File2 File3 File4 File5 2>errorlog
Created File 1
I am appending
If you see these two line along with ori content
It means we are appended
Created File2
These two lines are contents of file2
Created File 1
I am appending
If you see these two line along with ori content
It means we are appended
Created File2
These two lines are contents of file2
```

No errors

9. Concatenate name File1 to 5 and store it in File 10. if any errors, log them in the same file

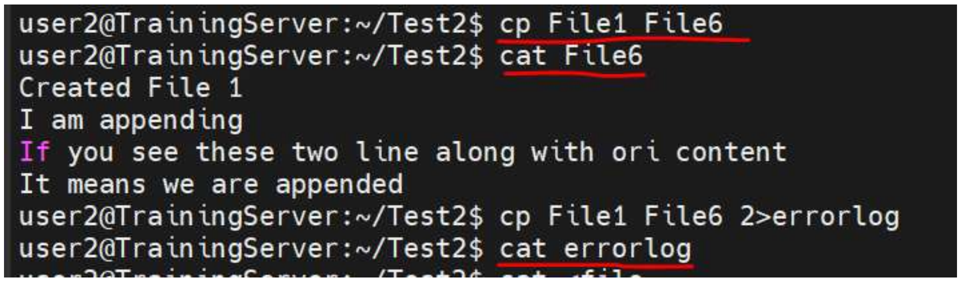
A: command: cat File1, File2, File3, File4, File5 >>File10



```
user2@TrainingServer:~/Test2$ cat File1 File2 File3 File4 File5 >>File10
cat: File4: No such file or directory
cat: File5: No such file or directory
user2@TrainingServer:~/Test2$ cat Fiile10
cat: Fiile10: No such file or directory
user2@TrainingServer:~/Test2$ cat File10
Created File 1
I am appending
If you see these two line along with ori content
It means we are appended
Created File2
These two lines are contents of file2
Created File 1
I am appending
If you see these two line along with ori content
It means we are appended
Created File2
These two lines are contents of file2
```

10. Copy File1 to File6. Add the errors to “errorlog”

A: command: cp File1 File6 2>errorlog

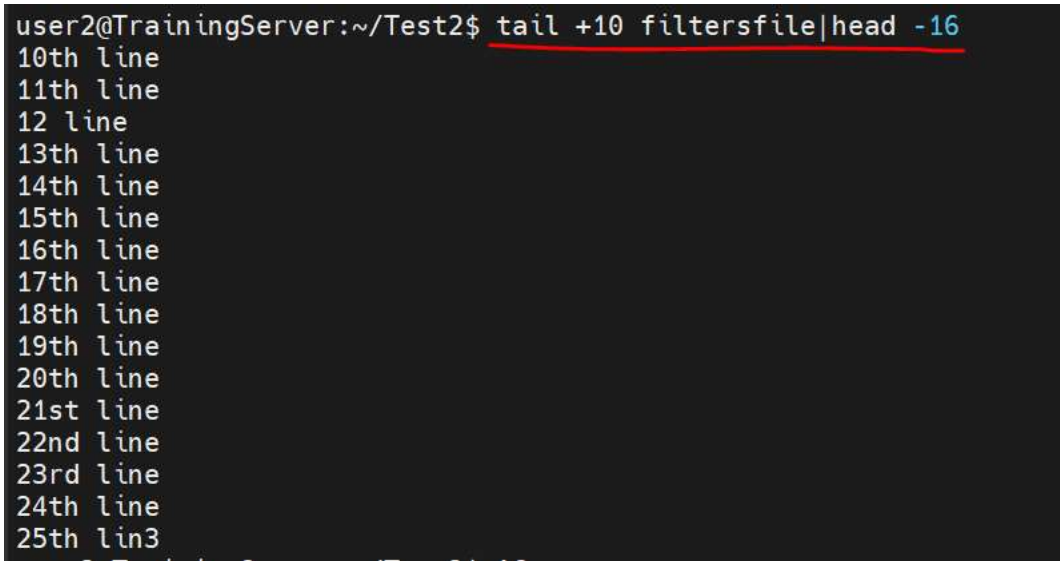


```
user2@TrainingServer:~/Test2$ cp File1 File6
user2@TrainingServer:~/Test2$ cat File6
Created File 1
I am appending
If you see these two line along with ori content
It means we are appended
user2@TrainingServer:~/Test2$ cp File1 File6 2>errorlog
user2@TrainingServer:~/Test2$ cat errorlog
```

Slide no. 55

1. Retrive The 10th to 25th lines in a file

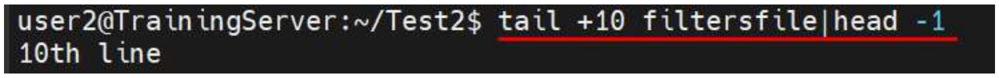
A: command: tail +10 filtersfile|head -16



```
user2@TrainingServer:~/Test2$ tail +10 filtersfile|head -16
10th line
11th line
12 line
13th line
14th line
15th line
16th line
17th line
18th line
19th line
20th line
21st line
22nd line
23rd line
24th line
25th lin3
```

2. List only 10th line in a file

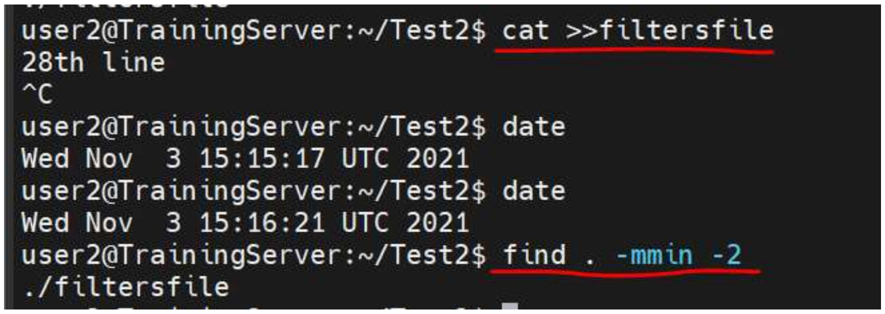
A: command: tail +10 filtersfile|head -1



```
user2@TrainingServer:~/Test2$ tail +10 filtersfile|head -1
10th line
```

3. List only the recently modified file in the current directory

A: command: find . -mmin -2 [Note to self: here, -2 refers to find the file that was modified at least 2 mins ago. Similarly, you can find files using "-mtime -2" which will give files modifies at least a day ago]

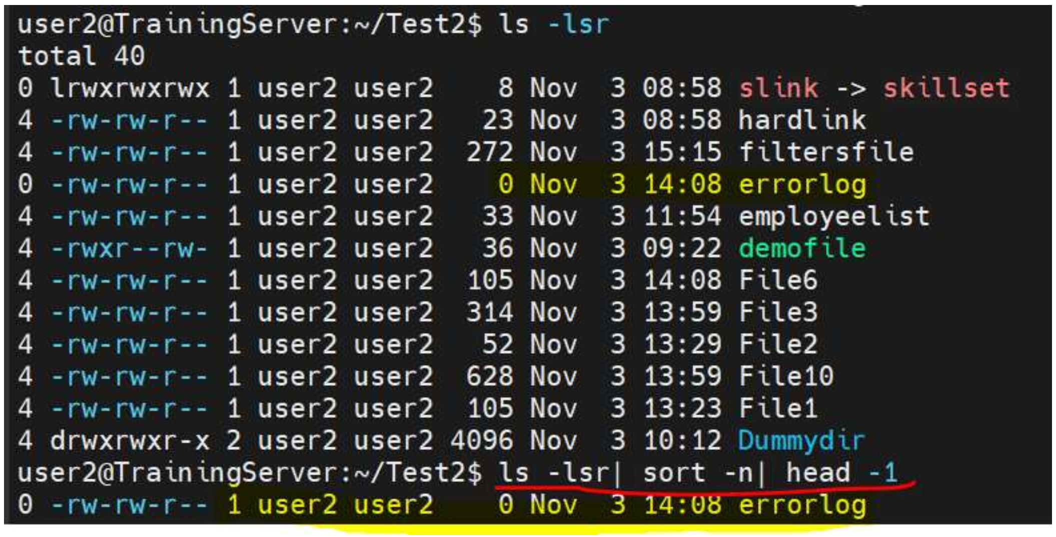


```
user2@TrainingServer:~/Test2$ cat >>filtersfile
28th line
^C
user2@TrainingServer:~/Test2$ date
Wed Nov  3 15:15:17 UTC 2021
user2@TrainingServer:~/Test2$ date
Wed Nov  3 15:16:21 UTC 2021
user2@TrainingServer:~/Test2$ find . -mmin -2
./filtersfile
```

4. List only the smallest file in the current directory

A: command: ls -lsr| sort -n| head -1

To check the file sizes command: ls -lsr

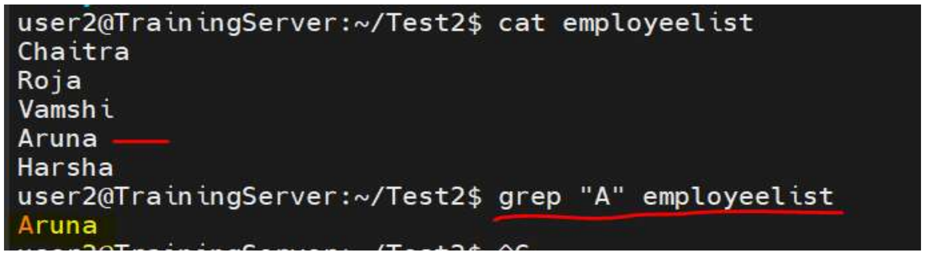


```
user2@TrainingServer:~/Test2$ ls -lsr
total 40
0 lrwxrwxrwx 1 user2 user2    8 Nov  3 08:58 slink -> skillset
4 -rw-rw-r-- 1 user2 user2   23 Nov  3 08:58 hardlink
4 -rw-rw-r-- 1 user2 user2  272 Nov  3 15:15 filtersfile
0 -rw-rw-r-- 1 user2 user2    0 Nov  3 14:08 errorlog
4 -rw-rw-r-- 1 user2 user2   33 Nov  3 11:54 employeelist
4 -rwxr--rw- 1 user2 user2   36 Nov  3 09:22 demofile
4 -rw-rw-r-- 1 user2 user2  105 Nov  3 14:08 File6
4 -rw-rw-r-- 1 user2 user2  314 Nov  3 13:59 File3
4 -rw-rw-r-- 1 user2 user2   52 Nov  3 13:29 File2
4 -rw-rw-r-- 1 user2 user2  628 Nov  3 13:59 File10
4 -rw-rw-r-- 1 user2 user2  105 Nov  3 13:23 File1
4 drwxrwxr-x 2 user2 user2 4096 Nov  3 10:12 Dummydir
user2@TrainingServer:~/Test2$ ls -lsr| sort -n| head -1
0 -rw-rw-r-- 1 user2 user2    0 Nov  3 14:08 errorlog
```

Slide no. 61

1. Lines containing A

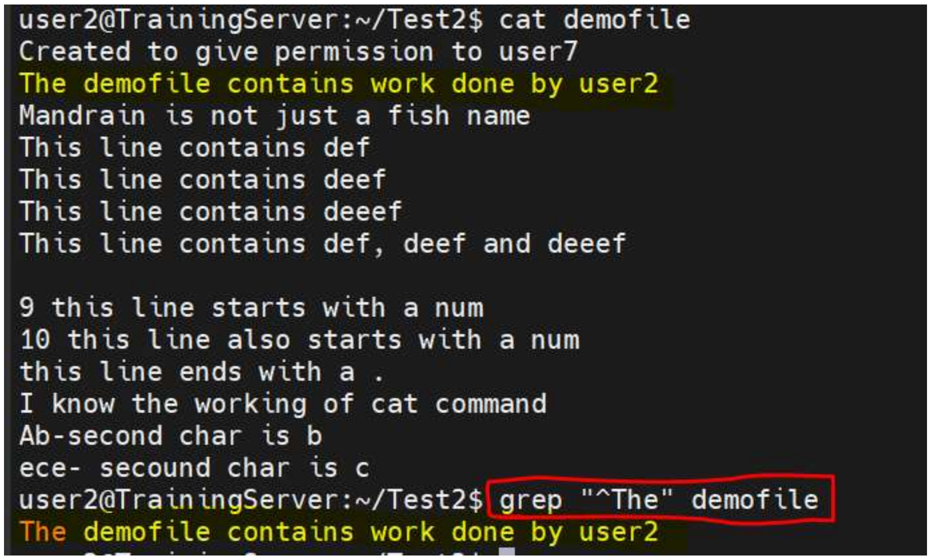
A: command: grep "A" employeelist



```
user2@TrainingServer:~/Test2$ cat employeelist
Chaitra
Roja
Vamshi
Aruna
Harsha
user2@TrainingServer:~/Test2$ grep "A" employeelist
Aruna
```

2. lines containing “The”at the beginning

A: command: grep “^The” demofile



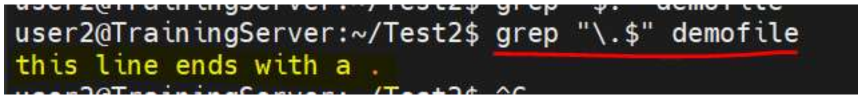
```
user2@TrainingServer:~/Test2$ cat demofile
Created to give permission to user7
The demofile contains work done by user2
Mandrain is not just a fish name
This line contains def
This line contains deef
This line contains deeef
This line contains def, deef and deeef

9 this line starts with a num
10 this line also starts with a num
this line ends with a .
I know the working of cat command
Ab-second char is b
ece- secound char is c
user2@TrainingServer:~/Test2$ grep "^The" demofile
The demofile contains work done by user2
```

3. lines

ending with .

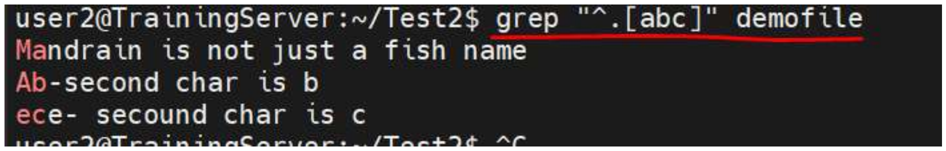
A: command: grep "\.$" demofile [Note to self: grep can match the beginning of a line with the '^' character and the end of a line with the '$' character. Any character is matched by the '.' character, so to match a literal "." you will need to use the escaped '\.' sequence]



```
user2@TrainingServer:~/Test2$ grep "\.$" demofile
this line ends with a .
```

4. lines with a,b or c as the second letter

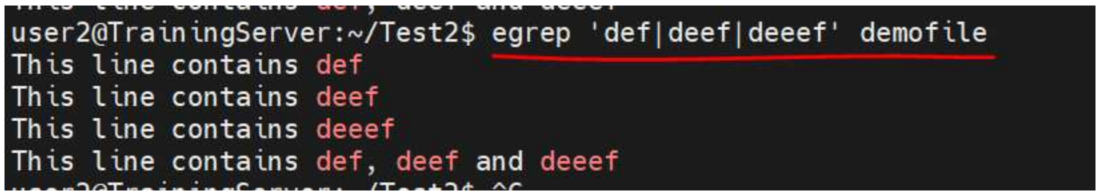
A: command: grep "^.[abc]" demofile [Note to self: [...] any character inside '[]' wil be considered. Opposite of this is [^..]]



```
user2@TrainingServer:~/Test2$ grep "^.[abc]" demofile
Mandrain is not just a fish name
Ab-second char is b
ece- secound char is c
```

5. lines which contain def , deef or deeeef

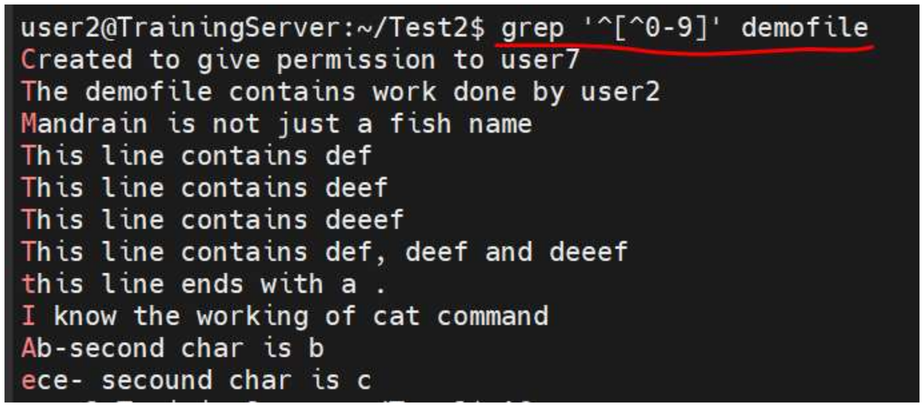
A: command: egrep 'def|deef|deeef' demofile



```
user2@TrainingServer:~/Test2$ egrep 'def|deef|deeef' demofile
This line contains def
This line contains deef
This line contains deeef
This line contains def, deef and deeef
```

6. lines not having numbers at the beginning

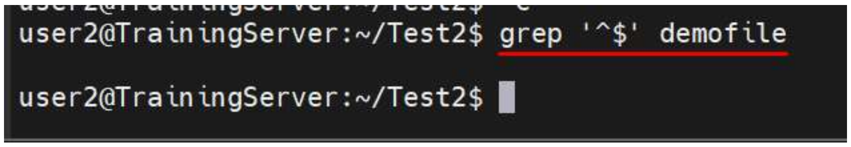
A: command : grep '^[^0-9]' demofile [Note: '^'-The beginning of a text line and [^..]- Any character not in the list or range]



```
user2@TrainingServer:~/Test2$ grep '^[^0-9]' demofile
Created to give permission to user7
The demofile contains work done by user2
Mandrain is not just a fish name
This line contains def
This line contains deef
This line contains deeef
This line contains def, deef and deeef
this line ends with a .
I know the working of cat command
Ab-second char is b
ece- secound char is c
```

7. empty lines

A: command: grep '^$' demofile

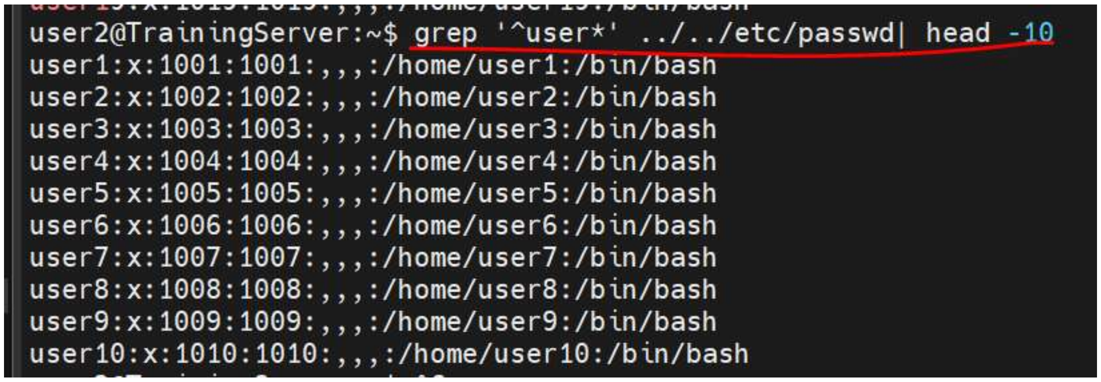


```
user2@TrainingServer:~/Test2$ grep '^$' demofile

user2@TrainingServer:~/Test2$ 
```

8. list the password file details of user1 – user10

A: command: grep '^user*' ../../etc/passwd| head -10



```
user2@TrainingServer:~$ grep '^user*' ../../etc/passwd| head -10
user1:x:1001:1001:,,,:/home/user1:/bin/bash
user2:x:1002:1002:,,,:/home/user2:/bin/bash
user3:x:1003:1003:,,,:/home/user3:/bin/bash
user4:x:1004:1004:,,,:/home/user4:/bin/bash
user5:x:1005:1005:,,,:/home/user5:/bin/bash
user6:x:1006:1006:,,,:/home/user6:/bin/bash
user7:x:1007:1007:,,,:/home/user7:/bin/bash
user8:x:1008:1008:,,,:/home/user8:/bin/bash
user9:x:1009:1009:,,,:/home/user9:/bin/bash
user10:x:1010:1010:,,,:/home/user10:/bin/bash
```

9. list only directories

A: command: ls -d */   (or)

ls -l | grep '^d'   (or)

ls -l | egrep '^d'

Slide no. 64

1. Create a file which contains long listing of the files in root directory

A: command: ls -l ../../root >longlistingfile

2. Sort the file based on the filename descending order (last field)

A:command: sort -r +8 longlistingfile (or) sort -r -k 9 longlistingfile

3.Sort the file based only on the inode count

A: command: sort -n +1 Longlist

4. Sort the file based on the user name and group name in reverse order

A: command: sort -k 3,4 Longlist

5. Sort the file based on the file size and store the output in a file called sortedfile

A: command: sort +4 -o sortedfile Longlist

cat sortedfile

Slide no. 64

1. Convert all capital letters into small letters in file

A: command: tr [:upper:] [:lower:] < demofile

2. Remove all occurrences of numbers in file

A: command: tr -d [0-9] < demofile

3. Remove all occurrences of spaces in a file

A: command: tr -d "[:space:]" < demofile

4. suppress multiple occurrence of spaces in long listing of filenames

A: command: ls -l | tr -s "[:space:]"

5. count the number of alphabets in the file

A: command: `grep -o [[:alpha:]] trialfile | wc -l`

6.count the occurrences of ovals in the given file

A: command: `grep -o [aeiouAEIOU] trialfile | wc -l`