1

a) Login to the system
b) Use the appropriate command to determine your login shell
c) Use the /etc/passwd file to verify the result of step b.
d) Use the ‘who’ command and redirect the result to a file called myfile1. Use the more command to see the contents of myfile1.
e) Use the date and who commands in sequence (in one line) such that the output of date will display on the screen and the output of who will be redirected to a file called myfile2. Use the more command to check the contents of myfile2.

2

a) Write a “sed” command that deletes the first character in each line in a file.
b) Write a “sed” command that deletes the character before the last character in each line in a file.
c) Write a “sed” command that swaps the first and second words in each line in a file.

a. Log into the system
When we return on the system one screen will appear. In this we have to type 100.0.0.9 then we enter into editor. It asks our details such as Login : krishnasai password: Then we get log into the commands.

b. use the appropriate command to determine your login shell

Syntax: $ echo $SHELL

Output: $ echo $SHELL
/bin/bash

Description:-

What is "the shell"?

Shell is a program that takes your commands from the keyboard and gives them to the operating system to perform. In the old days, it was the only user interface available on a Unix computer. Nowadays, we have graphical user interfaces (GUIs) in addition to command line interfaces (CLIs) such as the shell.

On most Linux systems a program called bash (which stands for Bourne Again SHell, an enhanced version of the original Bourne shell program, sh, written by Steve Bourne) acts as the shell program. There are several additional shell programs available on a typical Linux system. These include: ksh, tcsh and zsh.

- To find your current shell type following command
  $ echo $SHELL
- Shell is a user program or it's environment provided for user interaction. Shell is an command language interpreter that executes commands read from the standard input device (keyboard) or from a file.
- Shell is not part of system kernel, but uses the system kernel to execute programs, create files etc.
- Several shell available with Linux including:

Shell Name	Developed by	Where	Remark
BASH (Bourne-Again SHell)	Brian Fox and Chet Ramey	Free Software Foundation	Most common shell in Linux. It's Freeware shell.
CSH (C SHell)	Bill Joy	University of California (For BSD)	The C shell's syntax and usage are very similar to the C programming language.
KSH (Korn SHell)	David Korn	AT & T Bell Labs	--
TCSH	See the man page. Type $ man tcsh	--	TCSH is an enhanced but completely compatible version of the Berkeley UNIX C shell(CSH).

- To find all available shells in your system type following command:
  $ cat /etc/shells

c. use the /etc/passwd file to verify the result of step b.

$cat /etc/passwd

d. **Use the who command and redirect the result to a file called myfile1. Use the more command to see the contents of myfile1**.

Sol : $who > myfile1 | more
User1 pts/0 Apr 23 10:43
User2 pts/1 May 6 18:19

Description:

- Who>myfile1→The result of who command is stored in myfile1
- More myfile1→By using more command we print the myfile1 contents

"who" command:

The Linux "who" command lets you display the users that are currently logged into your Unix computer system.

Syntax: who

On a very busy Unix/Linux system the output of the who command may scroll off your terminal screen. To solve that, pipe the output of the who command into the Linux more command, like this: **who | more**

The -a argument of the who command lists all available output for each user on your system.

who -a

more command:

is a command to view (but not modify) the contents of a text file one screen at a time.

The syntax for the more command is:

more [options] [files]

OPTIONS

Option Description

- -c Page through the file by clearing the window. (not scrolling).
- -d Displays "Press space to continue, 'q' to quit"
- -f Count logical lines rather than screen lines (wrapping text)
- -l Ignores form feed (^L) characters.
- -r Display all control characters.
- -s Displays multiple blank lines as one blank line.
- -u Does not display underline characters and backspace (^H).
- -w Waits for a user to press a key before exiting.
- -n Displays n lines per window.
- +num Displays the file starting at line number num.
- +/pattern Displays the file starting at two lines before the pattern.

e. **Use the date and who commands in sequence (in one line) such that the output of date will display on the screen and the output of who will be redirected to a file called myfile2. Use the more command to check the contents of myfile2.**

$ date;who > myfile2
Sat Aug 23 09:38:05 IST 2008
$ more myfile2
allam tty7 2008-08-23 09:08 (:0)
allam pts/1 2008-08-23 09:09 (:0.0)

2

A) Write a sed command that deletes the first character in each line in a file.

```
$ cat  mytable
        1425    Ravi     15.65
        4320    Ramu    26.27
        6830    Sita      36.15
        1450    Raju     21.86
        7820    Anil     23.34
        9000    Allam   35.56

        $ sed  's/^./ / ' mytable
          425    Ravi       15.65
        320    Ramu     26.27
        830    Sita       36.15
        450    Raju       21.86
        820    Anil       23.34
        000   Allam     35.56
```

B) Write a sed command that deletes the character before the last character in each line in a file.

```
# Delete the last character in each line in a file.

        $ sed  's/.$/  /'  mytable
        1425    Ravi     15.6
        4320    Ramu    26.2
        6830    Sita      36.1
        1450    Raju     21.8
        7820    Anil     23.3
        9000    Allam   35.5
```

C) Write a sed command that swaps the first and second words in a file.

```
# Swaps the first and second words in each line in a file.

        $ sed   ‘s/ \([^ ]*\) *\([^ ]*\)/  \2  \1 /g’ mytable
        Ravi            1425    15.65
        Ramu            4320    26.27
        Sita            6830    36.15
        Raju            1450    21.86
        Anil            7820    23.34
        Allam                   9000          35.56
```

DESCRIPITON

Sed is a Stream Editor used for modifying the files in unix (or linux). Whenever you want to make changes to the file automatically, sed comes in handy to do this.

>cat file.txt

unix is great os. unix is opensource. unix is free os.
learn operating system.
unixlinux which one you choose.

SED COMMAND EXAMPLES

1. Replacing or substituting string

Sed command is mostly used to replace the text in a file. The below simple sed command replaces the word "unix" with "linux" in the file.

```
>sed 's/unix/linux/' file.txt
linux is great os. unix is opensource. unix is free os.
learn operating system.
linuxlinux which one you choose.
```

Here the "s" specifies the substitution operation. The "/" are delimiters. The "unix" is the search pattern and the "linux" is the replacement string.

By default, the sed command replaces the first occurrence of the pattern in each line and it won't replace the second, third...occurrence in the line.

2. Replacing the nth occurrence of a pattern in a line.

Use the /1, /2 etc flags to replace the first, second occurrence of a pattern in a line. The below command replaces the second occurrence of the word "unix" with "linux" in a line.

```
>sed 's/unix/linux/2' file.txt
unix is great os. linux is opensource. unix is free os.
learn operating system.
unixlinux which one you choose.
```

3. Replacing all the occurrence of the pattern in a line.

The substitute flag /g (global replacement) specifies the sed command to replace all the occurrences of the string in the line.

```
>sed 's/unix/linux/g' file.txt
linux is great os. linux is opensource. linux is free os.
learn operating system.
linuxlinux which one you choose.
```

4. Replacing from nth occurrence to all occurrences in a line.
Use the combination of /1, /2 etc and /g to replace all the patterns from the nth occurrence of a pattern in a line. The following sed command replaces the third, fourth, fifth... "unix" word with "linux" word in a line.

```
>sed 's/unix/linux/3g' file.txt
unix is great os. unix is opensource. linux is free os.
learn operating system.
unixlinux which one you choose.
```

5. Changing the slash (/) delimiter
You can use any delimiter other than the slash. As an example if you want to change the web url to another url as

```
>sed 's/http:\/\/www/' file.txt
```

In this case the url consists the delimiter character which we used. In that case you have to escape the slash with backslash character, otherwise the substitution won't work.
Using too many backslashes makes the sed command look awkward. In this case we can change the delimiter to another character as shown in the below example.

```
>sed 's_http://_www_' file.txt
>sed 's|http://|www|' file.txt
```

6. Using & as the matched string

There might be cases where you want to search for the pattern and replace that pattern by adding some extra characters to it. In such cases & comes in handy. The & represents the matched string.

```
>sed 's/unix/{&}/' file.txt
{unix} is great os. unix is opensource. unix is free os.
learn operating system.
{unix}linux which one you choose.

>sed 's/unix/{&&}/' file.txt
{unixunix} is great os. unix is opensource. unix is free os.
learn operating system.
{unixunix}linux which one you choose.
```

7. Using \1,\2 and so on to \9

The first pair of parenthesis specified in the pattern represents the \1, the second represents the \2 and so on. The \1,\2 can be used in the replacement string to make changes to the source string. As an example, if you want to replace the word "unix" in a line with twice as the word like "unixunix" use the sed command as below.

```
>sed 's/\(unix\)/\1\1/' file.txt
unixunix is great os. unix is opensource. unix is free os.
learn operating system.
unixunixlinux which one you choose.
```

The parenthesis needs to be escaped with the backslash character. Another example is if you want to switch the words "unixlinux" as "linuxunix", the sed command is

```
>sed 's/\(unix\)\(linux\)/\2\1/' file.txt
unix is great os. unix is opensource. unix is free os.
learn operating system.
linuxunix which one you choose.
```

Another example is switching the first three characters in a line

```
>sed 's/^\(.\)\(.\)\(.\)/\3\2\1/' file.txt
inux is great os. unix is opensource. unix is free os.
aelrn operating system.
inuxlinux which one you choose.
```

8. Duplicating the replaced line with /p flag

The /p print flag prints the replaced line twice on the terminal. If a line does not have the search pattern and is not replaced, then the /p prints that line only once.

```
>sed 's/unix/linux/p' file.txt
linux is great os. unix is opensource. unix is free os.
linux is great os. unix is opensource. unix is free os.
learn operating system.
linuxlinux which one you choose.
linuxlinux which one you choose.
```

9. Printing only the replaced lines

Use the -n option along with the /p print flag to display only the replaced lines. Here the -n option suppresses the duplicate rows generated by the /p flag and prints the replaced lines only one time.

```
>sed -n 's/unix/linux/p' file.txt
linux is great os. unix is opensource. unix is free os.
linuxlinux which one you choose.
```

If you use -n alone without /p, then the sed does not print anything.

10. Running multiple sed commands.
You can run multiple sed commands by piping the output of one sed command as input to another sed command.

```
>sed 's/unix/linux/' file.txt| sed 's/os/system/'
linux is great system. unix is opensource. unix is free os.
learn operating system.
linuxlinux which one you chosysteme.
```

Sed provides -e option to run multiple sed commands in a single sed command. The above output can be achieved in a single sed command as shown below.

```
>sed -e 's/unix/linux/' -e 's/os/system/' file.txt
linux is great system. unix is opensource. unix is free os.
learn operating system.
linuxlinux which one you chosysteme.
```

11. Replacing string on a specific line number.
You can restrict the sed command to replace the string on a specific line number. An example is

```
>sed '3 s/unix/linux/' file.txt
unix is great os. unix is opensource. unix is free os.
learn operating system.
linuxlinux which one you choose.
```

The above sed command replaces the string only on the third line.

12. Replacing string on a range of lines.
You can specify a range of line numbers to the sed command for replacing a string.

```
>sed '1,3 s/unix/linux/' file.txt
linux is great os. unix is opensource. unix is free os.
learn operating system.
linuxlinux which one you choose.
```

Here the sed command replaces the lines with range from 1 to 3. Another example is

```
>sed '2,$ s/unix/linux/' file.txt
linux is great os. unix is opensource. unix is free os.
learn operating system.
linuxlinux which one you choose.
```

Here $ indicates the last line in the file. So the sed command replaces the text from second line to last line in the file.

13. Replace on a lines which matches a pattern.
You can specify a pattern to the sed command to match in a line. If the pattern match occurs, then only the sed command looks for the string to be replaced and if it finds, then the sed command replaces the string.

```
>sed '/linux/ s/unix/centos/' file.txt
unix is great os. unix is opensource. unix is free os.
learn operating system.
centoslinux which one you choose.
```

Here the sed command first looks for the lines which has the pattern "linux" and then replaces the word "unix" with "centos".

14. Deleting lines.
You can delete the lines a file by specifying the line number or a range or numbers.

```
>sed '2 d' file.txt
>sed '5,$ d' file.txt
```

15. Duplicating lines
You can make the sed command to print each line of a file two times.

```
>sed 'p' file.txt
```

16. Sed as grep command
You can make sed command to work as similar to grep command.

```
>grep 'unix' file.txt
>sed -n '/unix/ p' file.txt
```

Here the sed command looks for the pattern "unix" in each line of a file and prints those lines that has the pattern.
You can also make the sed command to work as grep -v, just by using the reversing the sed with NOT (!).

```
>grep -v 'unix' file.txt
>sed -n '/unix/ !p' file.txt
```

The ! here inverts the pattern match.

17. Add a line after a match.

The sed command can add a new line after a pattern match is found. The "a" command to sed tells it to add a new line after a match is found.

```
>sed '/unix/ a "Add a new line"' file.txt
unix is great os. unix is opensource. unix is free os.
"Add a new line"
learn operating system.
unixlinux which one you choose.
"Add a new line"
```

18. Add a line before a match

The sed command can add a new line before a pattern match is found. The "i" command to sed tells it to add a new line before a match is found.

```
>sed '/unix/ i "Add a new line"' file.txt
"Add a new line"
unix is great os. unix is opensource. unix is free os.
learn operating system.
"Add a new line"
unixlinux which one you choose.
```

19. Change a line

The sed command can be used to replace an entire line with a new line. The "c" command to sed tells it to change the line.

```
>sed '/unix/ c "Change line"' file.txt
"Change line"
learn operating system.
"Change line"
```

20. Transform like tr command

The sed command can be used to convert the lower case letters to upper case letters by using the transform "y" option.

```
>sed 'y/ul/UL/' file.txt
Unix is great os. Unix is opensoUrce. Unix is free os.
Learn operating system.
UnixLinUx which one yoU choose.
```

Here the sed command transforms the alphabets "ul" into their uppercase format "UL"