

Introduction to Open-Source Software

B. Tech Integrated /Computer / Sem IV

PART A

(PART A: TO BE REFERRED BY STUDENTS)

Experiment No.07

A.1—Aim:

To perform Stream editing operations in Linux.

A.2--- Prerequisite:

Theory:

SED command in UNIX stands for stream editor and it can perform lots of functions on file like searching, find and replace, insertion or deletion. Though most common use of SED command in UNIX is for substitution or for find and replace. By using SED you can edit files even without opening them, which is much quicker way to find and replace something in file, than first opening that file in VI Editor and then changing it.

- SED is a powerful text stream editor. Can do insertion, deletion, search and replace(substitution).
- SED command in unix supports regular expression which allows it perform complex pattern matching.

Syntax:

sed OPTIONS... [SCRIPT] [INPUTFILE...]

Example:

Consider the below text file as an input.

\$cat > happyfile.txt

unix is great os. unix is opensource. unix is free os.

learn operating system.

unix linux which one you choose.

unix is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.

Sample Commands

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.
2. **\$sed 's/unix/linux/' happyfile.txt**

Output :

linux is great os. unix is opensource. unix is free os.

learn operating system.

linux linux which one you choose.

linux is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.

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.

```
shiti@shiti:~/College$ sed 's/unix/linux/' 1
linux is great os. unix is opensource. unix is free os.
learn operating system.
linux linux which one you choose.
linux is easy to learn.unix is a multiuser os.Learn unix.unix is powerful.
```

3. **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.

4. **\$sed 's/unix/linux/2' happyfile.txt**

Output:

unix is great os. linux is opensource. unix is free os.

learn operating system.

unix linux which one you choose.

unix is easy to learn.linux is a multiuser os.Learn unix .unix is a powerful.

```
shiti@shiti:~/College$ sed 's/unix/linux/2' 1
unix is great os. linux is opensource. unix is free os.
learn operating system.
unix linux which one you choose.
unix is easy to learn.linux is a multiuser os.Learn unix.unix is powerful.
```

5. **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.

6. **\$sed 's/unix/linux/g' happyfile.txt**

Output :

linux is great os. linux is opensource. linux is free os.

learn operating system.

linux linux which one you choose.

linux is easy to learn. linux is a multiuser os. Learn linux .linux is a powerful.

```
shiti@shiti:~/College$ sed 's/unix/linux/g' 1
linux is great os. linux is opensource. linux is free os.
learn operating system.
linux linux which one you choose.
linux is easy to learn. linux is a multiuser os. Learn linux .linux is powerful.
```

7. **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.

8. **\$sed 's/unix/linux/3g' happyfile.txt**

Output:

unix is great os. unix is opensource. linux is free os.

learn operating system.

unix linux which one you choose.

unix is easy to learn. unix is a multiuser os. Learn linux .linux is a powerful.

```
shiti@shiti:~/College$ sed 's/unix/linux/3g' 1
unix is great os. unix is opensource. linux is free os.
learn operating system.
unix linux which one you choose.
unix is easy to learn. unix is a multiuser os. Learn linux .linux is powerful.
```

9. **Parenthesize first character of each word :** This sed example prints the first character of every word in parenthesis.

10. **\$ echo "Welcome To The Happy Stuff" | sed 's/(\b[A-Z])/\1)/g'**

Output:

(W)elcome (T)o (T)he (G)eek (S)tuff

```
shiti@shiti:~/College$ echo "Welcom To The Happy Stuff" | sed 's/(\b[A-Z])/\1)/g'
(W)elcom (T)o (T)he (H)appy (S)tuff
```

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

12. **\$sed '3 s/unix/linux/' happyfile.txt**

Output:

unix is great os. unix is opensource. unix is free os.

learn operating system.

linux linux which one you choose.

unix is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.

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

```
shiti@shiti:~/College$ sed '3s/unix/linux/' 1
unix is great os. unix is opensource. unix is free os.
learn operating system.
linux linux which one you choose.
unix is easy to learn.unix is a multiuser os.Learn unix.unix is powerful.
```

13. **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.

14. **\$sed 's/unix/linux/p' happyfile.txt**

Output:

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.

linux linux which one you choose.

linux linux which one you choose.

linux is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.

linux is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.

```
shiti@shiti:~/College$ sed 's/unix/linux/p' 1
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.
linux linux which one you choose.
linux linux which one you choose.
linux is easy to learn.unix is a multiuser os.Learn unix.unix is powerful.
linux is easy to learn.unix is a multiuser os.Learn unix.unix is powerful.
```

15. **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.

16. **\$sed -n 's/unix/linux/p' happyfile.txt**

Output:

linux is great os. unix is opensource. unix is free os.

linux linux which one you choose.

linux is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.

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

```
shiti@shiti:~/College$ sed -n 's/$unix/$linux/p' 1
shiti@shiti:~/College$ sed -n 's/unix/linux/p' 1
linux is great os. unix is opensource. unix is free os.
linux linux which one you choose.
linux is easy to learn.unix is a multiuser os.Learn unix.unix is powerful.
```

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

18. **\$sed '1,3 s/unix/linux/' happyfile.txt**

Output:

linux is great os. unix is opensource. unix is free os.

learn operating system.

linux linux which one you choose.

unix is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.

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

```
shiti@shiti:~/College$ sed '1,3 s/unix/linux/' 1
linux is great os. unix is opensource. unix is free os.
learn operating system.
linux linux which one you choose.
unix is easy to learn.unix is a multiuser os.Learn unix.unix is powerful.
```

\$sed '2,\$ s/unix/linux/' happyfile.txt

Output:

unix is great os. unix is opensource. unix is free os.

learn operating system.

linux linux which one you choose.

linux is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful

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

```
shitij@shitij:~/College$ sed '2,$ s/unix/linux/' 1
unix is great os. unix is opensource. unix is free os.
learn operating system.
linux linux which one you choose.
linux is easy to learn.unix is a multiuser os.Learn unix.unix is powerful.
```

19. **Deleting lines from a particular file :** SED command can also be used for deleting lines from a particular file. SED command is used for performing deletion operation without even opening the file

Examples:

1. To Delete a particular line say n in this example

20. Syntax:

21. \$ sed 'nd' filename.txt

22. Example:

23. \$ sed '5d' filename.txt

2. To Delete a last line

Syntax:

\$ sed '\$d' filename.txt

3. To Delete line from range x to y

Syntax:

\$ sed 'x,yd' filename.txt

Example:

\$ sed '3,6d' filename.txt

4. To Delete from nth to last line

Syntax:

\$ sed 'nth,\$d' filename.txt

Example:

```
$ sed '12,$d' filename.txt
```

5. To Delete pattern matching line

Syntax:

```
$ sed '/pattern/d' filename.txt
```

Example:

```
$ sed '/abc/d' filename.txt
```

We humans are certainly an intelligent species. We work with others and we depend on each other for common tasks. For example, you depend on a milkman to deliver milk in milk bottles or cartons. This logic applies to computer programs including shell scripts. When scripts get complex you need to use divide and conquer technique.

Shell functions

- Sometime shell scripts get complicated.
- To avoid large and complicated scripts use functions.
- You divide large scripts into small chunks/entities called functions.
- Functions make shell scripts modular and easy to use.
- Functions avoid repetitive code. For example, `is_root_user ()` function can be reused by various shell scripts to
- determine whether logged on user is root or not.
- Functions perform a specific task. For example, add or delete a user account.
- Functions are used like normal commands.
- In other high-level programming languages, a function is also known as procedure, method, subroutine, or routine.

Writing the hello() function

Type the following command at a shell prompt:

```
hello() { echo 'Hello world!' ; }
```

Invoking the hello() function

hello() function can be used like normal command. To execute, simply type:

```
hello
```

Passing the arguments to the hello() function

You can pass command line arguments to user defined functions. Define hello as follows:

```
hello() { echo "Hello $1, let us be a friend." ; }
```

You can hello function and pass an argument as follows:

```
hello Vivek
```

Sample outputs:

```
Hello Vivek, let us be a friend.
```

A.3--- Tasks:

1. Write a Shell script that accepts a filename, starting and ending line numbers as arguments and displays the lines between the given line numbers.
2. Write a Shell script that displays list of all the files in the current directory to which the user has read, Write and execute permissions.

(PART - B)

(TO BE COMPLETED BY STUDENTS)

(Students must submit the soft copy as per following segments within two hours of the practical.
The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge
faculties at the end of the practical in case there is no Black board access available)

Roll.No.: C020	Name: Shitij Agrawal
Sem/Year: Sem IV	Batch: B1
Date of Experiment: 08/02/2022	Date of Submission: 08/02/2022
Grade --	

B.1: Procedure of performed experiment

1. Write a Shell script that accepts a filename, starting and ending line numbers as arguments and displays the lines between the given line numbers.

```
2.1 ~/College
Open [v] [f]
1 #!/bin/sh
2 echo "Enter filename"
3 read filename
4 echo "Start line"
5 read start
6 echo "End line"
7 read end
8 sed -n $start,$end\p $filename | cat > newline
9 cat newline
```

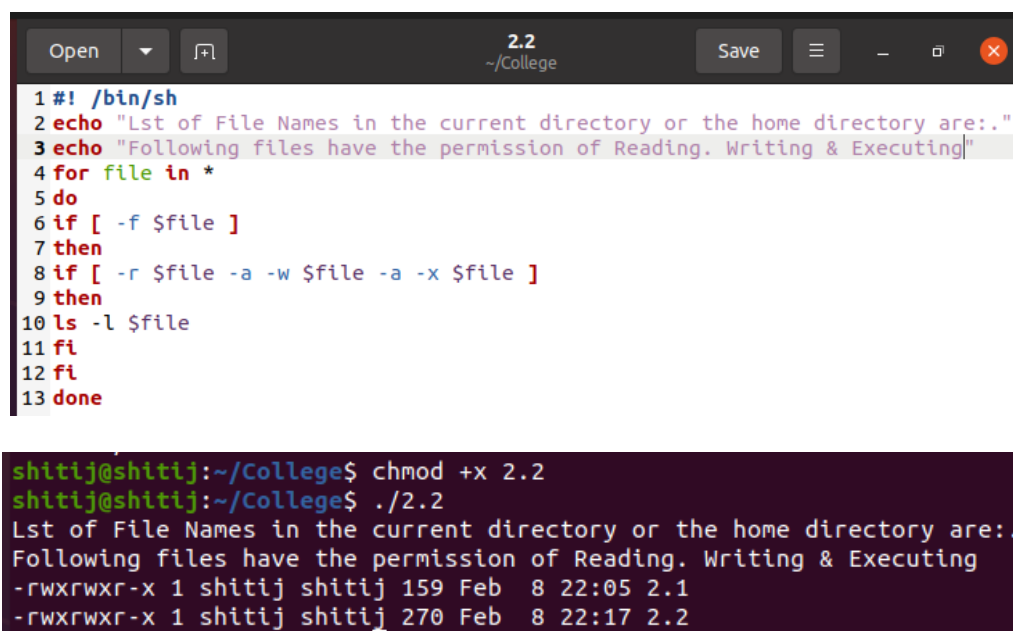
```
shitij ~/College
Open [v] [f]
1 Shitij
2 Agrawal
3 C020
4 B1
5 Semester IV
6 6 year 2|
```

```

shitij@shitij:~/College$ ./2.1
Enter filename
shitij
Start line
1
End line
4
Shitij
Agrawal
C020
B1

```

2. Write a Shell script that displays list of all the files in the current directory to which the user has read, Write and execute permissions.



The screenshot shows a code editor window titled '2.2' with a file icon and 'Save' button. The script content is as follows:

```

1 #!/bin/sh
2 echo "Lst of File Names in the current directory or the home directory are:."
3 echo "Following files have the permission of Reading. Writing & Executing"
4 for file in *
5 do
6 if [ -f $file ]
7 then
8 if [ -r $file -a -w $file -a -x $file ]
9 then
10 ls -l $file
11 fi
12 fi
13 done

```

Below the editor, a terminal window shows the execution of the script:

```

shitij@shitij:~/College$ chmod +x 2.2
shitij@shitij:~/College$ ./2.2
Lst of File Names in the current directory or the home directory are:.
Following files have the permission of Reading. Writing & Executing
-rwxrwxr-x 1 shitij shitij 159 Feb  8 22:05 2.1
-rwxrwxr-x 1 shitij shitij 270 Feb  8 22:17 2.2

```

B.2: Observations and Learning's:

Learned how to use different string functions in shell (i.e how to change one string from one to another, deleting strings, etc). Also learned how to access files in the directory from the terminal.

B.3: Conclusion:

Performing Stream editing operations in Linux.