likegeeks.com

31+ Examples for sed Linux command in text manipulation

admin https://likegeeks.com

7-9 minutes

In the previous post, we talked about bash functions and how to use them from the command line directly and we saw some other cool stuff. Today we will talk about a very useful tool for string manipulation called sed or **sed Linux command**.

Sed is used to work with text files like log files, configuration files, and other text files.

In this post, we are going to focus on sed Linux command which is used for text manipulation, which is a very important step in our

bash scripting journey.

Linux system provides some tools for text processing, one of those tools is sed.

We will discuss the 31+ examples with pictures to show the output of every example.

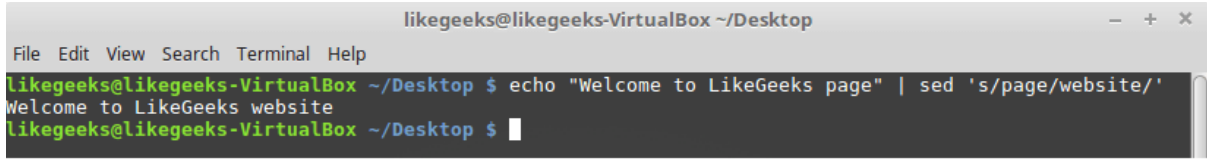
Understand sed Linux Command

The sed command is an interactive text editor like nano. Sed Linux command edits data based on the rules you provide, you can use it like this:

$ sed options file

You are not limited to use sed to manipulate files, you apply it to the STDIN directly like this:

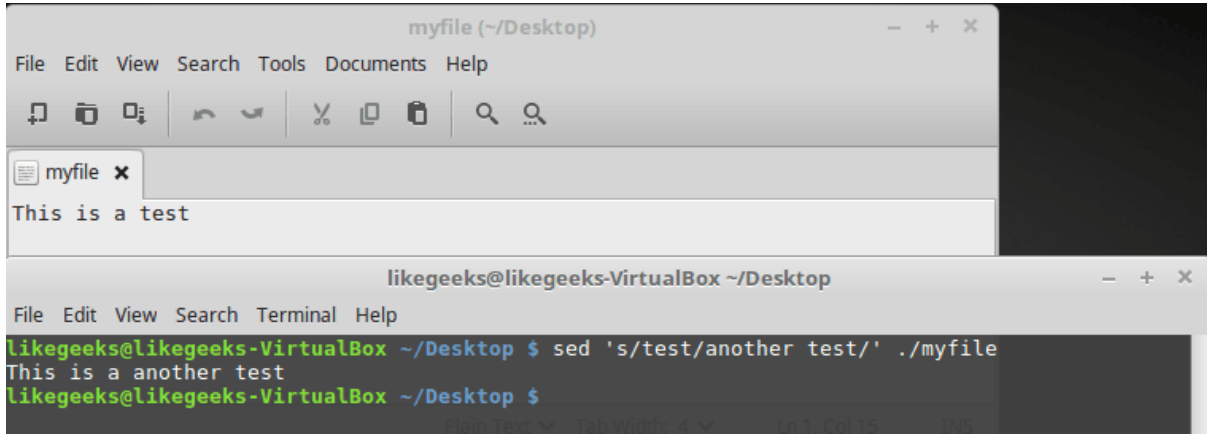
$ echo "Welcome to LikeGeeks page" | sed 's/page/website/'



The s command replaces the first text with the second text pattern. In this case, the string “website” was replaced with the word “page”, so the result will be as shown.

The above example was a very basic example to demonstrate the tool. We can use sed Linux command to manipulate files as well.

This is our file:



$ sed 's/test/another test' ./myfile

The results are printed to the screen instantaneously, you don’t have to wait for processing the file to the end.

If your file is huge enough, you will see the result before the

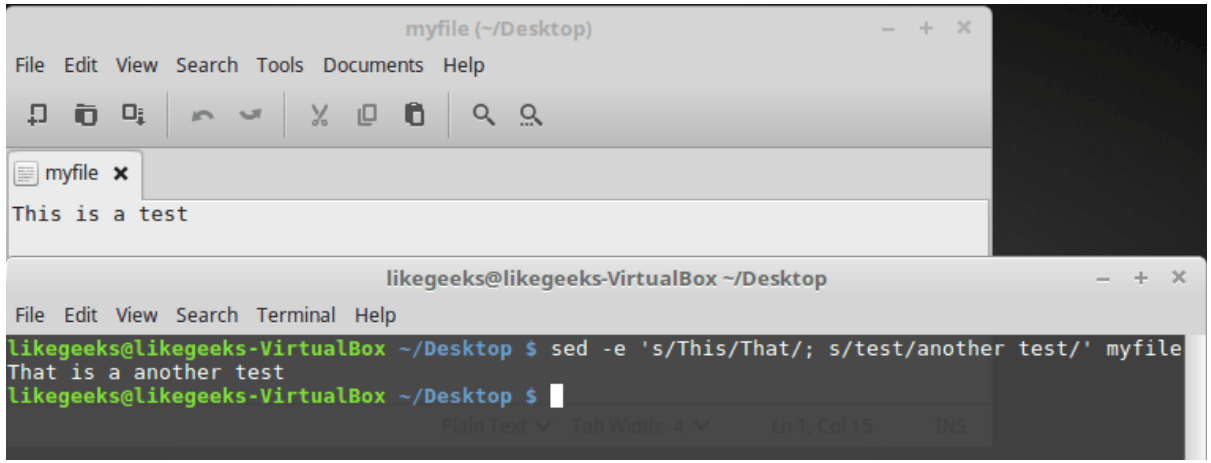
processing is finished.

Sed Linux command doesn't update your data. It only sends the changed text to STDOUT. The file still untouched. If you need to overwrite the existing content, you can check our previous post which was talking about redirections.

Using Multiple sed Linux Commands in The Command Line

To run multiple sed commands, you can use the -e option like this:

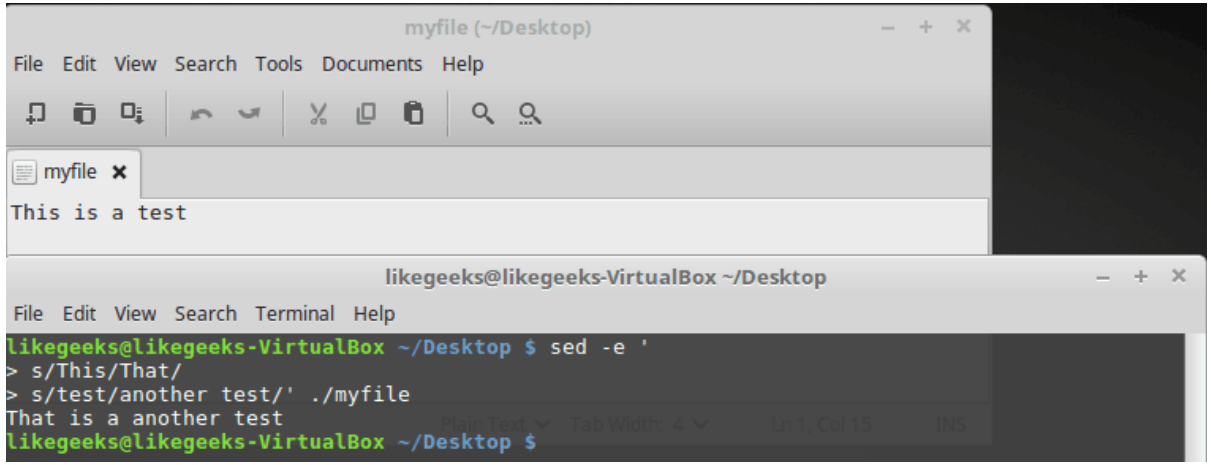
$ sed -e 's/This/That/; s/test/another test/' ./myfile



Sed command must be separated by a semi colon without any spaces.

Also, you can use a single quotation to separate commands like this:

```
$ sed -e '
> s/This/That/
> s/test/another test/' myfile
```



The same result, no big deal.

Reading Commands From a File

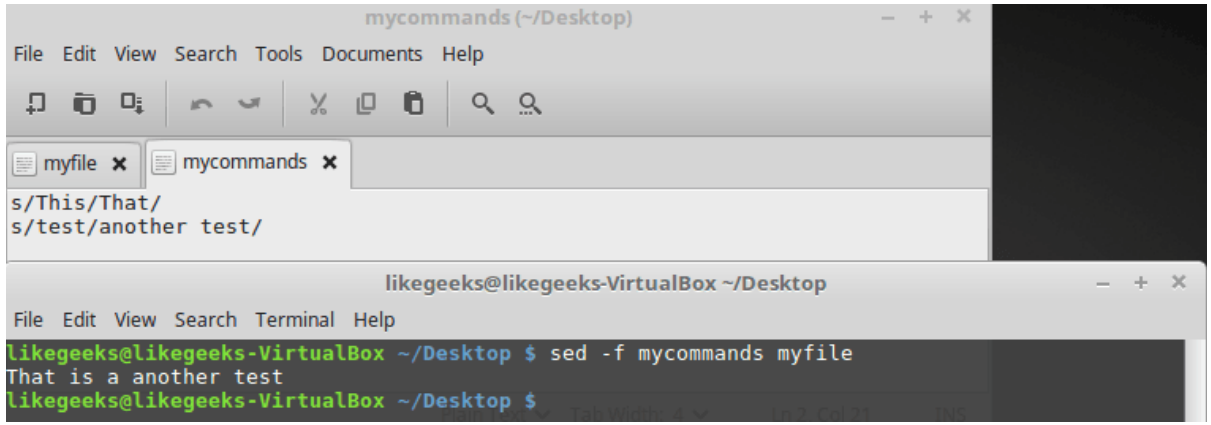
You can save your sed commands in a file and use them by specifying the file using -f option.

$ cat mycommands

s/This/That/

s/test/another test/

$ sed -f mycommands myfile

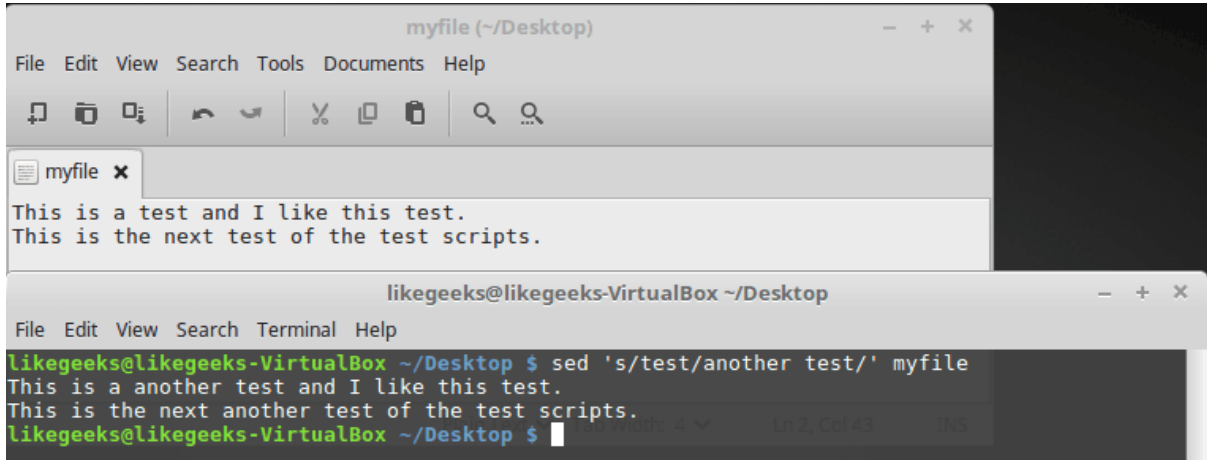


Substituting Flags

Look at the following example carefully:

$ cat myfile

$ sed 's/test/another test/' myfile



The above result shows the first occurrence in each line is only replaced. To substitute all occurrences of a pattern, use one of the following substitution flags.

The flags are written like this:

s/pattern/replacement/flags

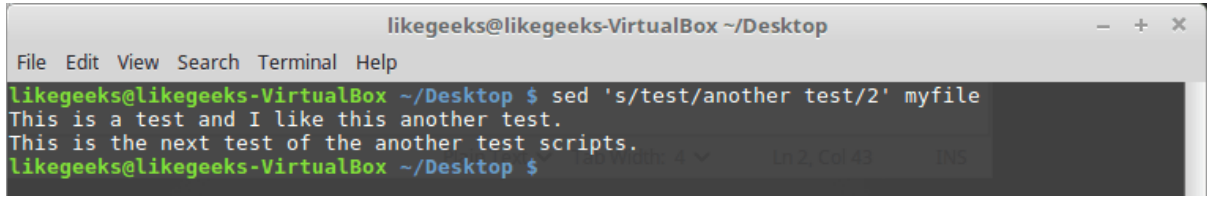
There are four types of substitutions:

- g, replace all occurrences.
- A number, the occurrence number for the new text that you want to substitute.

- p, print the original content.
- w file: means write the results to a file.

You can limit your replacement by specifying the occurrence number that should be replaced like this:

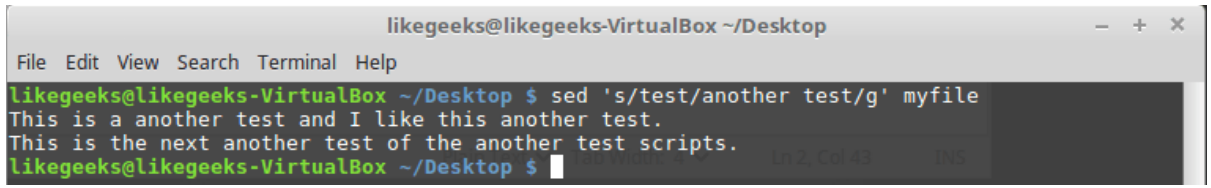
$ sed 's/test/another test/2' myfile



As you can see, only the second occurrence on each line was replaced.

The g flag means global, which means a global replacement for all occurrences:

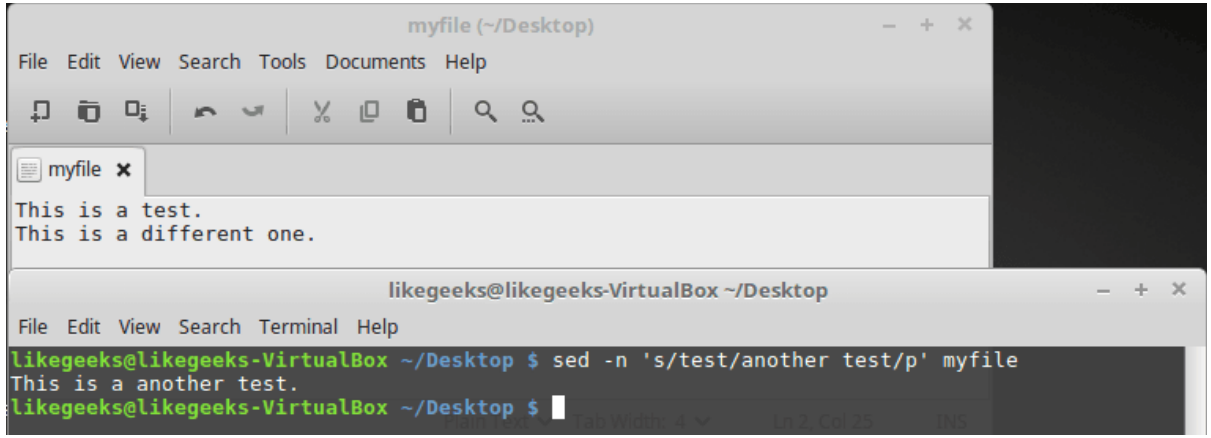
$ sed 's/test/another test/g' myfile



The p flag prints each line contains a pattern match, you can use the -n option to print the modified lines only.

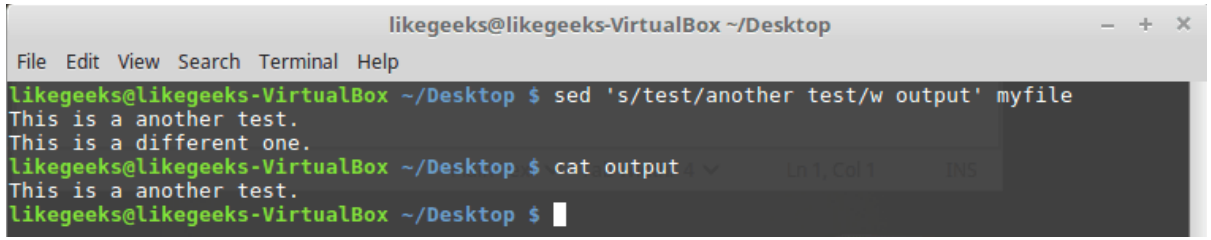
$ cat myfile

$ sed -n 's/test/another test/p' myfile



The w flag saves the output to a specified file:

$ sed 's/test/another test/w output' myfile



The output is printed on the screen, but the matching lines are saved to the output file.

Replace Characters

Suppose that you want to search for bash shell and replace it with csh shell in the /etc/passwd file, you can do it like this:

```
$ sed 's/\/bin\/bash/\/bin\/csh/' /etc/passwd
```

Oh!! that looks terrible.

Luckily, there is another way to achieve that. You can use the exclamation mark (!) as string delimiter like this:

```
$ sed 's!/bin/bash!/bin/csh!' /etc/passwd
```

Now it’s easier to read.

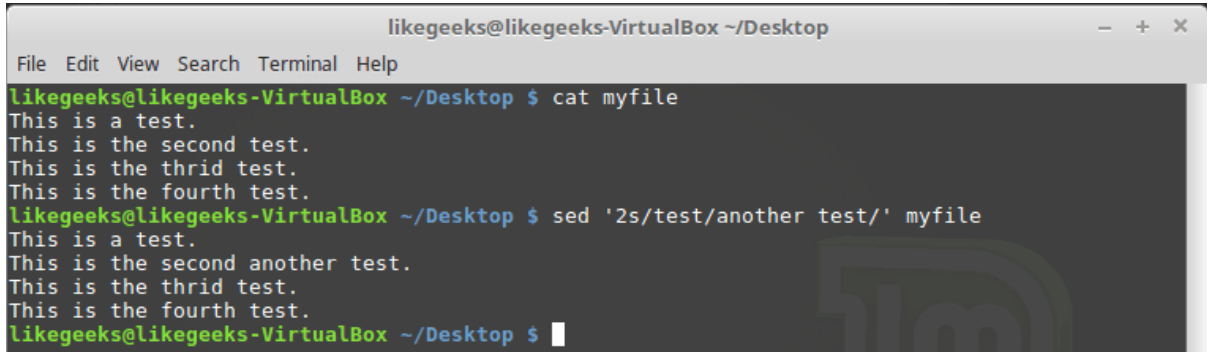
Limiting sed

Sed command processes your entire file. However, you can limit the sed command to process specific lines, there are two ways:

- A range of lines.
- A pattern that matches a specific line.

  You can type one number to limit it to a specific line:

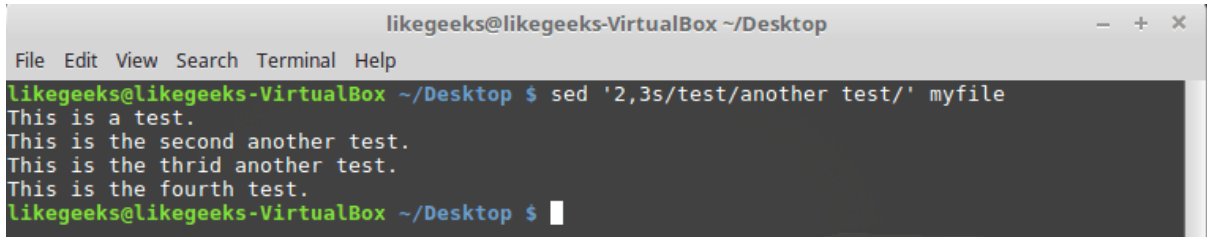
  $ sed '2s/test/another test/' myfile



  Only line two is modified.

  What about using a range of lines:

  $ sed '2,3s/test/another test/' myfile



Also, we can start from a line to the end of the file:

$ sed '2,$s/test/another test/' myfile

```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed '2,$s/test/another test/' myfile
This is a test.
This is the second another test.
This is the thrid another test.
This is the fourth another test.
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Or you can use a pattern like this:

$ sed '/likegeeks/s/bash/csh/' /etc/passwd

```
rtkit:x:119:127:RealtimeKit,,,:/proc:/bin/false
saned:x:120:128::/var/lib/saned:/bin/false
usbmux:x:121:46:usbmux daemon,,,:/var/lib/usbmux:/bin/false
likegeeks:x:1000:1000:likegeeks,,,:/home/likegeeks:/bin/csh
vboxadd:x:999:1::/var/run/vboxadd:/bin/false
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Awesome!!

You can use regular expressions to write this pattern to be more generic and useful.

Delete Lines

To delete lines, the delete (d) flag is your friend.

The delete flag deletes the text from the stream, not the original file.

$ sed '2d' myfile

```
likegeeks@likegeeks-VirtualBox ~/Desktop
File  Edit  View  Search  Terminal  Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ cat ./myfile
First line
Second line
Third line
Fourth line
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed '2d' myfile
First line
Third line
Fourth line
likegeeks@likegeeks-VirtualBox ~/Desktop $ cat ./myfile
First line
Second line
Third line
Fourth line
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Here we delete the second line only from myfile.

What about deleting a range of lines?

$ sed '2,3d' myfile

```
likegeeks@likegeeks-VirtualBox ~/Desktop
File  Edit  View  Search  Terminal  Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed '2,3d' myfile
This is a test.
This is the fourth test.
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Here we delete a range of lines, the second and the third.

Another type of ranges:

$ sed '3,$d' myfile

```
likegeeks@likegeeks-VirtualBox ~/Desktop
File  Edit  View  Search  Terminal  Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed '3,$d' myfile
This is a test.
This is the second test.
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Here we delete from the third line to the end of the file.

All these examples never modify your original file.

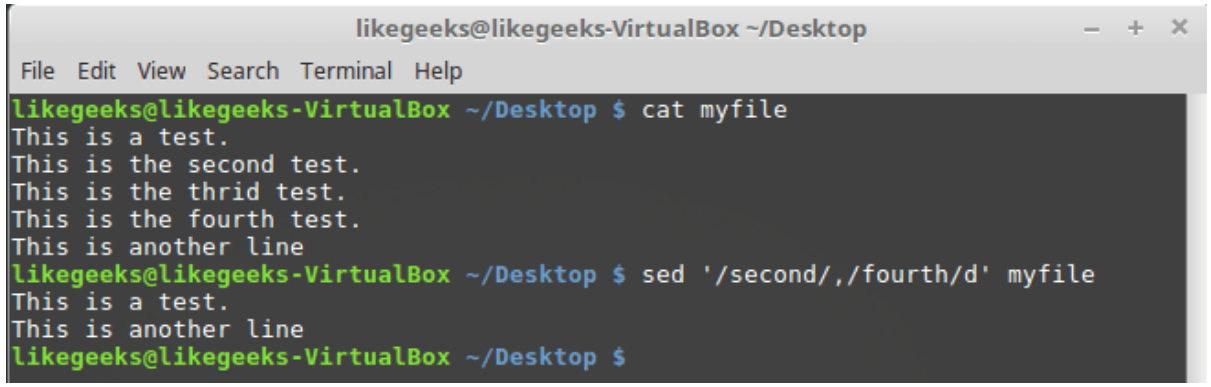
$ sed '/test 1/d' myfile

```
likegeeks@likegeeks-VirtualBox ~/Desktop
File  Edit  View  Search  Terminal  Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ cat myfile
This is a test.
This is the second test.
This is the thrid test.
This is the fourth test.
This is another line
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed '/test/d' myfile
This is another line
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Here we use a pattern to delete the line if matched on the first line.

If you need to delete a range of lines, you can use two text patterns like this:

$ sed '/second/,/fourth/d' myfile



The first to the third line deleted.

Insert and Append Text

You can insert or append text lines using the following flags:

- The (i) flag.
- The (a) flag.

$ echo "Another test" | sed 'i\First test '

```
File  Edit  View  Search  Terminal  Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ echo "Another test" | sed 'i\First test '
First test
Another test
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Here the text is added before the specified line.

$ echo "Another test" | sed 'a\First test '

```
likegeeks@likegeeks-VirtualBox ~/Desktop
File  Edit  View  Search  Terminal  Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ echo "Another test" | sed 'a\First test '
Another test
First test
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Here the text is added after the specified line.

Well, what about adding text in the middle?

Easy, look at the following example:

$ sed '2i\This is the inserted line.' myfile

```
likegeeks@likegeeks-VirtualBox ~/Desktop
File  Edit  View  Search  Terminal  Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ cat myfile
This is a test.
This is the second test.
This is the thrid test.
This is the fourth test.
This is another line
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed '2i\This is the inserted line.' myfile
This is a test.
This is the inserted line.
This is the second test.
```

```
This is the thrid test.
This is the fourth test.
This is another line
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

And the appending works the same way, but look at the position of the appended text:

$ sed '2a\This is the appended line.' myfile

```
likegeeks@likegeeks-VirtualBox ~/Desktop
File  Edit  View  Search  Terminal  Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ cat myfile
This is a test.
This is the second test.
This is the thrid test.
This is the fourth test.
This is another line
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed '2a\This is the appended line.' myfile
This is a test.
This is the second test.
This is the appended line.
This is the thrid test.
This is the fourth test.
This is another line
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

The same flags are used but with a location of insertion or appending.

Modifying Lines

To modify a specific line, you can use the (c) flag like this:

$ sed '3c\This is a modified line.' myfile

```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ cat myfile
This is a test.
This is the second test.
This is the thrid test.
This is the fourth test.
This is another line
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed '3c\This is a modified line.' myfile
This is a test.
This is the second test.
This is a modified line.
This is the fourth test.
This is another line
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

You can use a regular expression pattern and all lines match that pattern will be modified.

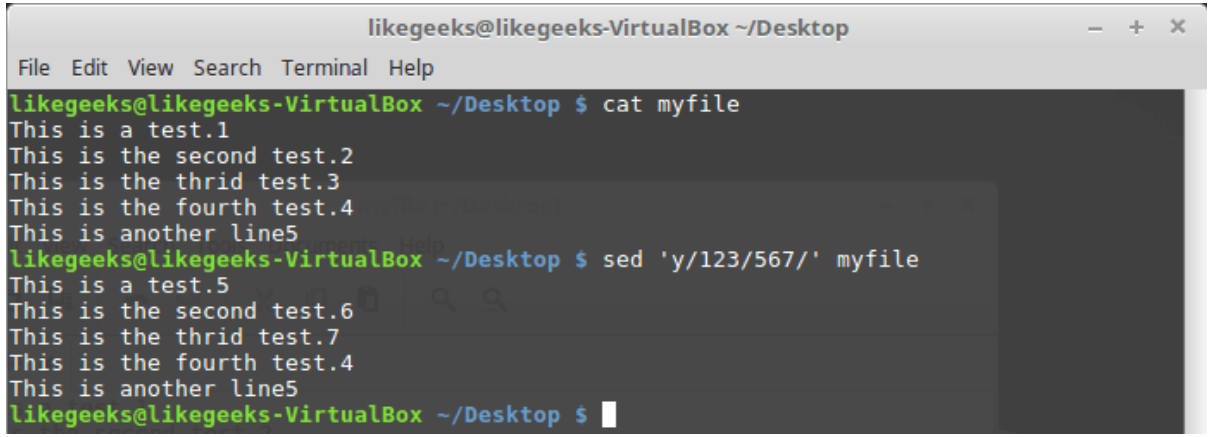
$ sed '/This is/c Line updated.' myfile

```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ cat myfile
This is First line
This is Second line
This is Third line
This is Fourth line
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed '/This is/c Line updated.' myfile
Line updated.
Line updated.
Line updated.
Line updated.
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Transform Characters

The transform flag (y) works on characters like this:

$ sed 'y/123/567/' myfile

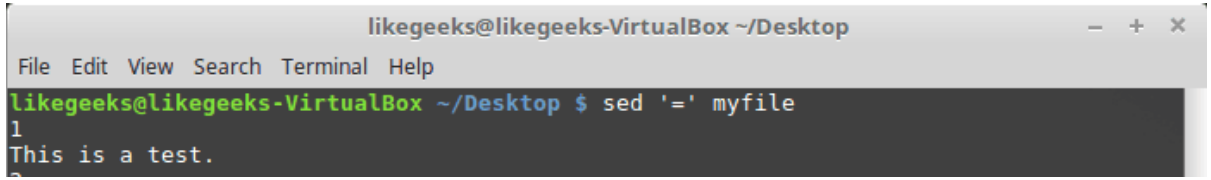


The transformation is applied to all data and cannot be limited to a specific occurrence.

Print Line Numbers

You can print line number using the (=) sign like this:

$ sed '=' myfile



```
2
This is the second test.
3
This is the thrid test.
4
This is the fourth test.
5
This is another line
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

However, by using -n combined with the equal sign, the sed command displays the line number that contains matching.

$ sed -n '/test/=' myfile

```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed -n '/test/=' myfile
1
2
3
4
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Read Data From a File

You can use the (r) flag to read data from a file.

You can define a line number or a text pattern for the text that you want to read.

$ cat newfile

$ sed '3r newfile' myfile

```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ cat newfile
First line in nefile
Second line in newfile
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed '3r newfile' myfile
This is a test.
This is the second test.
This is the thrid test.
First line in nefile
Second line in newfile
This is the fourth test.
This is another line
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

The content is just inserted after the third line as expected.

And this is using a text pattern:

$ sed '/test/r newfile' myfile

```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ cat myfile
This is a test.
This is the second test.
This is the thrid test.
This is the fourth test.
This is another line
likegeeks@likegeeks-VirtualBox ~/Desktop $ cat newfile
First line in nefile
Second line in newfile
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed '/test/r newfile' myfile
This is a test.
First line in nefile
Second line in newfile
This is the second test.
First line in nefile
```

```
Second line in newfile
This is the thrid test.
First line in nefile
Second line in newfile
This is the fourth test.
First line in nefile
Second line in newfile
This is another line
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Cool right?

Useful Examples

We have a file that contains a text with a placeholder and we have another file that contains the data that will be filled in that placeholder.

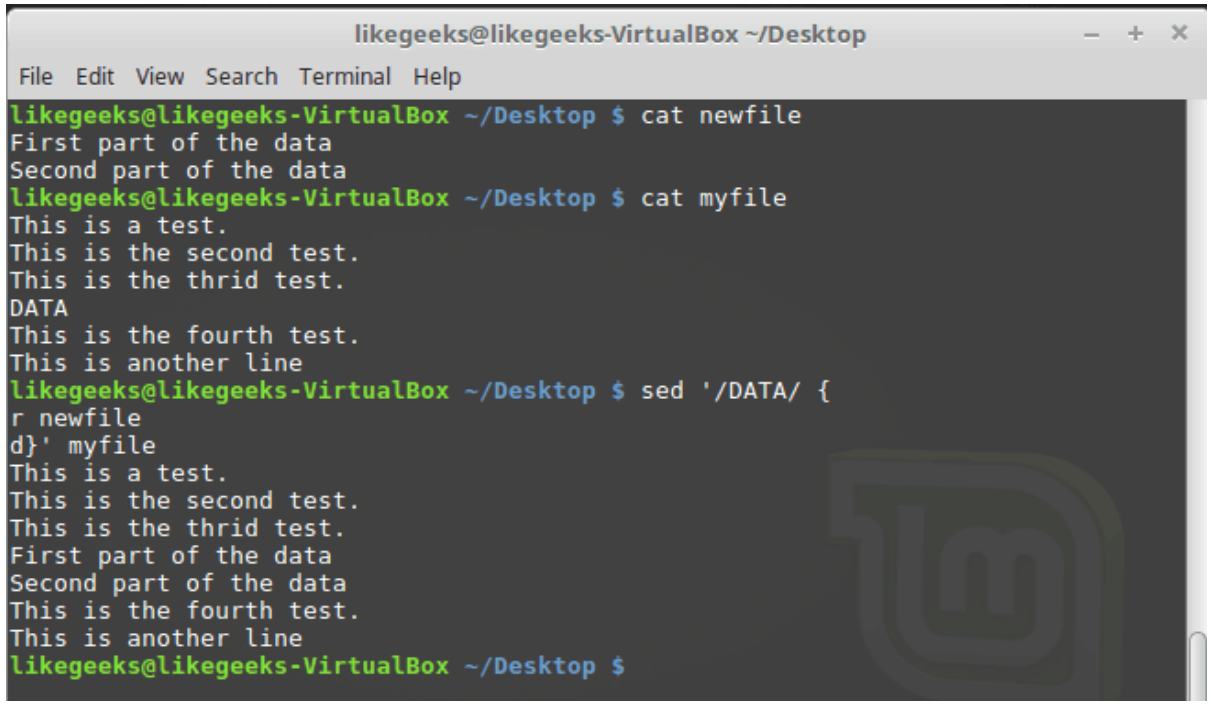
We will use the (r) and (d) flags to do the job.

The word DATA in that file is a placeholder for a real content which is stored in another file called data.

We will replace it with the actual content:

```
$ Sed '/DATA>/ {
r newfile
```

d}' myfile



Awesome!! as you can see, the placeholder location is filled with the data from the other file.

This is just a very small intro about sed command. Actually, sed Linux command is another world by itself.

The only limitation is your imagination.

I hope you enjoy what've introduced today about the string manipulation using sed Linux command.

Thank you.