**SED command in Linux | Set 2**

We have discussed some of the SED command options in [Sed Command in Linux/Unix with examples](https://www.geeksforgeeks.org/sed-command-in-unix/)

SED is used for finding, filtering, text substitution, replacement and text manipulations like insertion, deletion search, etc. It’s a one of the powerful utilities offered by Linux/Unix systems. We can use sed with regular expressions. I hope at least you have the basic knowledge about Linux regular expressions.

It provides non-interactive editing of text files that’s why it’s used to automate editing and has two buffers – **pattern buffer** and **hold buffer**. Sed use Pattern buffer when it read files, line by line and that currently read line is inserted into pattern buffer whereas hold buffer is a long-term storage, it catches the information, store it and reuse it when it is needed. Initially, both are empty. SED command is used for performing different operations without even opening the file.

**sed general syntax** – **sed OPTIONS… [SCRIPT] [INPUTFILE…]**

First create **a.txt** file on which I am going to perform operations for SED commands. In this blog, I used “a.txt” file to explain all the examples. Blog will become too long if i write the output of each sed command. So, you may refer the same file to practice all the commands initially.

Syntax: cat a.txt

life isn't meant to be easy; life is meant to be lived.

Try to learn & understand something new every day in life.

Respect everyone & most important love everyone.

Don’t hesitate to ask for love & don’t hesitate to show love too.

Life is too short to be shy.

In life, experience will help you differentiating right from wrong.

**# Let’s start with File Spacing**

**1 –**Insert one blank line after each line – sed G a.txt

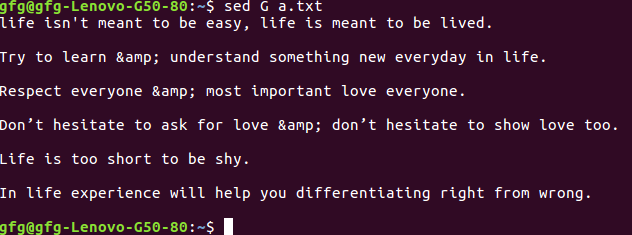
**2 –** To insert two blank lines – sed 'G;G' a.txt

**3 –** Delete blank lines and insert one blank line after each line – sed '/^$/d;G' a.txt

**4 –** Insert a black line above every line which matches “love” – sed '/love/{x;p;x;}' a.txt

**5 –** Insert a blank line below every line which matches “love” – sed '/love/G' a.txt

**6 –** Insert 5 spaces to the left of every lines – sed 's/^/ /' a.txt



**# Numbering lines**

**1 –** Number each line of a file (left alignment). \*\*=\*\* is used to number the line. \t is used for tab between number and sentence –

Syntax: sed = a.txt | sed 'N;s/\n/\t/'

**2 –** Number each line of a file (number on left, right-aligned). This command is similar to `cat -n filename`.

Syntax: sed = a.txt | sed 'N; s/^/ /; s/ \*\(.\{4,\}\)\n/\1 /'

**3 –** Number each line of file, only if line is not blank –

Syntax: sed '/./=' a.txt | sed '/./N; s/\n/ /'

**# Deleting lines**

**1 –** Delete a particular line –

Syntax: sed ‘nd’ filename

Example: sed '5d' a.txt

**2 –** Delete the last line

Syntax: sed ‘$d’ filename

**3 –** Delete line from range x to y

Syntax: sed ‘x,yd’ filename

Example: sed '3,5d' a.txt

**4 –** Delete from nth to last line

Syntax: sed ‘nth,$d’ filename

Example: sed '2,$d' a.txt

**5 –** Delete the pattern matching line –

Syntax: sed ‘/pattern/d’ filename

Example: sed '/life/d' a.txt

**6 –** Delete lines starting from nth line and every 2nd line from there –

Syntax: sed ‘n~2d’ filename

Example: sed '3~2d' a.txt

**7 –** Delete the lines which matches the pattern and 2 lines after to that –

Syntax: sed ‘/pattern/,+2d’ filename

Example: sed '/easy/,+2d' a.txt

**8 –** Delete blank Lines

Syntax: sed '/^$/d' a.txt

**9 –** Delete empty lines or those begins with “#” –

Syntax: sed -i '/^#/d;/^$/d' a.txt

**# View/Print the files**

If we want to view content of file, then we use **cat** command and if we want to view the bottom and the top content of any file, we use tools such as **head** and **tail.** But what if we need to view a particular section in the middle of any file? Here we’ll discuss, how to use SED command to view a section of any file.

**1 –** Viewing a file from x to y range –

Syntax: sed -n ‘x,yp’ filename

Example: sed -n '2,5p' a.txt

**2 –** View the entire file except the given range –

Syntax: sed ‘x,yd’ filename

Example: sed '2,4d' a.txt

**3 –** Print nth line of the file –

Syntax: sed -n ‘address’p filename

Example: sed -n '4'p a.txt

**4 –** Print lines from xth line to yth line.

Syntax: sed -n ‘x,y’p filename

Example: sed -n '4,6'p a.txt

**5 –** Print only the last line –

Syntax: sed -n ‘$’p filename

**6 –** Print from nth line to end of file –

Syntax: sed -n ‘n,$p’ filename

Example: sed -n '3,$'p a.txt

**Pattern Printing**

**7 –** Print the line only which matches the pattern –

Syntax: sed -n /pattern/p filename

Example: sed -n /every/p a.txt

**8 –** Print lines which matches the pattern i.e from input to xth line.

Syntax: sed -n ‘/pattern/,xp’ filename 

Example: sed -n '/everyone/,5p' a.txt

Following prints lines which matches the pattern, 3rd line matches the pattern “everyone”, so it prints from 3rd line to 5th line. Use **$** in place of 5, if want to print the file till end.

**9 –** Prints lines from the xth line of the input, up-to the line which matches the pattern. If the pattern doesn’t find then it prints up-to end of the file.

Syntax: sed -n ‘x,/pattern/p’ filename

Example: sed -n '1,/everyone/p' a.txt

**10 –** Print the lines which matches the pattern up-to the next xth lines –

Syntax: sed -n ‘/pattern/,+xp’ filename

Example:  sed -n '/learn/,+2p' a.txt

**# Replacement with the sed command**

**1 –** Change the first occurrence of the pattern –

Syntax: sed 's/life/leaves/' a.txt

**2 –** Replacing the nth occurrence of a pattern in a line –

Syntax: sed ‘s/old\_pattern/new\_pattern/n’ filename

Example: sed 's/to/two/2' a.txt

We wrote “2” because we replace the second occurrence. Likewise, you can use 3, 4 etc according to need.

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

Syntax: sed 's/life/learn/g' a.txt

**4 –** Replace pattern from nth occurrence to all occurrences in a line.

Syntax: sed ‘s/old\_pattern/new\_pattern/ng’ filename 

Example: sed 's/to/TWO/2g' a.txt

**Note –** This sed command replaces the second, third, etc occurrences of pattern “to” with “TWO” in a line.

If you wish to print only the replaced lines, then use “-n” option along with “/p” print flag to display only the replaced lines –

Syntax: sed -n 's/to/TWO/p' a.txt

And if you wish to print the replaced lines twice, then only use “/p” print flag without “-n” option-

Syntax: sed 's/to/TWO/p' a.txt

**5 –** Replacing pattern on a specific line number. Here, “m” is the line number.

Syntax: sed ‘m s/old\_pattern/new\_pattern/’ filename 

Example: sed '3 s/every/each/' a.txt

If you wish to print only the replaced lines –

sed -n '3 s/every/each/p' a.txt

**6 –** Replace string on a defined range of lines – 

Syntax: sed ‘x,y s/old\_pattern/new\_pattern/’ filename 

where, x = starting line number and y = ending line number

Example: sed '2,5 s/to/TWO/' a.txt

**Note –** **$** can be used in place of “y” if we wish to change the pattern up-to last line in the file. 

Example: sed '2,$ s/to/TWO/' a.txt

**7 –** If you wish to replace pattern in order to ignore character case (beginning with uppercase or lowercase), then there are two ways to replace such patterns – 

First, By using “/i” print flag – 

Syntax: sed ‘s/old\_pattern/new\_pattern/i’ filename 

Example: sed 's/life/Love/i' a.txt

Second, By using regular expressions –

Syntax: sed 's/[Ll]ife/Love/g' a.txt

**8 –** To replace multiple spaces with a single space –

Syntax: sed 's/ \*/ /g' filename

**9 –** Replace one pattern followed by another pattern –

Syntax: sed ‘/followed\_pattern/ s/old\_pattern/new\_pattern/’ filename

Example: sed '/is/ s/live/love/' a.txt

**10 –** Replace a pattern with other except in the nth line.

Syntax: sed ‘n!s/old\_pattern/new\_pattern/’ filename

Example: sed -i '5!s/life/love/' a.txt