

Bash-скрипты, часть 7: sed и обработка текстов

<https://likegeeks.com/sed-linux/>

- Блог компании RUVDS.com,
- Настройка Linux,
- Серверное администрирование
- [Перевод](#)

[Bash-скрипты: начало](#)

[Bash-скрипты, часть 2: циклы](#)

[Bash-скрипты, часть 3: параметры и ключи командной строки](#)

[Bash-скрипты, часть 4: ввод и вывод](#)

[Bash-скрипты, часть 5: сигналы, фоновые задачи, управление сценариями](#)

[Bash-скрипты, часть 6: функции и разработка библиотек](#)

[Bash-скрипты, часть 7: sed и обработка текстов](#)

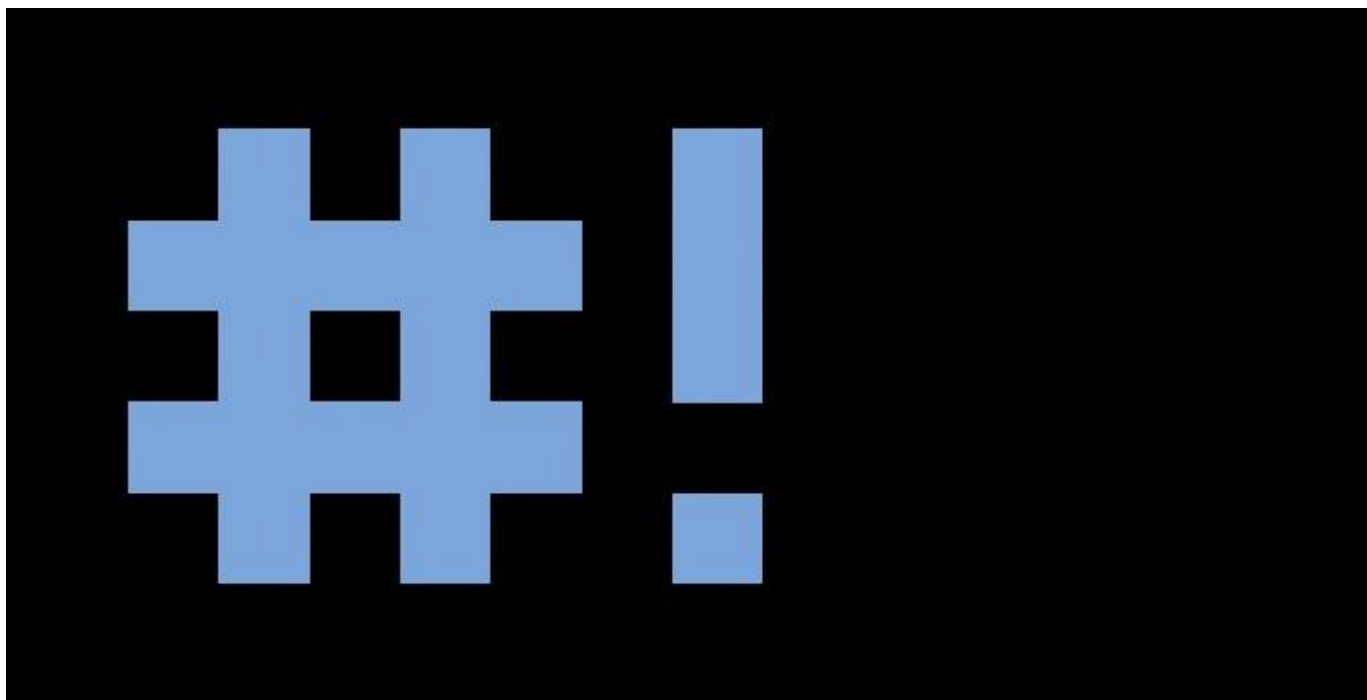
[Bash-скрипты, часть 8: язык обработки данных awk](#)

[Bash-скрипты, часть 9: регулярные выражения](#)

[Bash-скрипты, часть 10: практические примеры](#)

[Bash-скрипты, часть 11: expect и автоматизация интерактивных утилит](#)

В прошлый раз мы говорили о функциях в bash-скриптах, в частности, о том, как вызывать их из командной строки. Наша сегодняшняя тема — весьма полезный инструмент для обработки строковых данных — утилита Linux, которая называется sed. Её часто используют для работы с текстами, имеющими вид лог-файлов, конфигурационных и других файлов.



Если вы, в bash-скриптах, каким-то образом обрабатываете данные, вам не помешает знакомство с инструментами [sed](#) и [gawk](#). Тут мы сосредоточимся на sed и на работе с текстами, так как это — очень важный шаг в нашем путешествии по

бескрайним просторам разработки bash-скриптов.

Habrahabr10

Промо-код для скидки в 10% на наши виртуалы

Сейчас мы разберём основы работы с `sed`, а так же рассмотрим более трёх десятков примеров использования этого инструмента.

Основы работы с `sed`

Утилиту `sed` называют потоковым текстовым редактором. В интерактивных текстовых редакторах, наподобие `nano`, с текстами работают, используя клавиатуру, редактируя файлы, добавляя, удаляя или изменяя тексты. `Sed` позволяет редактировать потоки данных, основываясь на заданных разработчиком наборах правил. Вот как выглядит схема вызова этой команды:

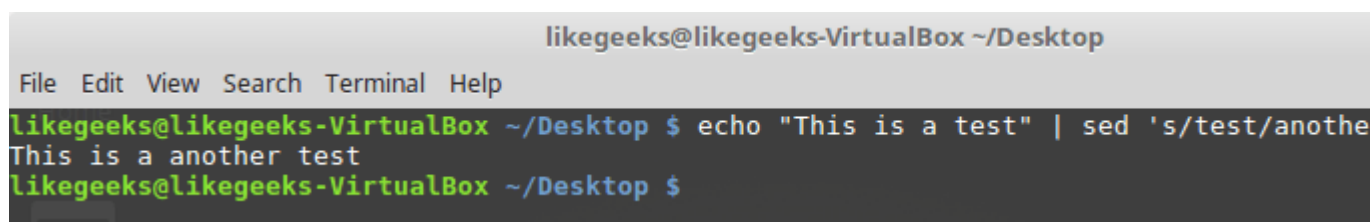
```
$ sed options file
```

По умолчанию `sed` применяет указанные при вызове правила, выраженные в виде набора команд, к `STDIN`. Это позволяет передавать данные непосредственно `sed`.

Например, так:

```
$ echo "This is a test" | sed 's/test/another test/'
```

Вот что получится при выполнении этой команды.



```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ echo "This is a test" | sed 's/test/another test/'
This is a another test
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

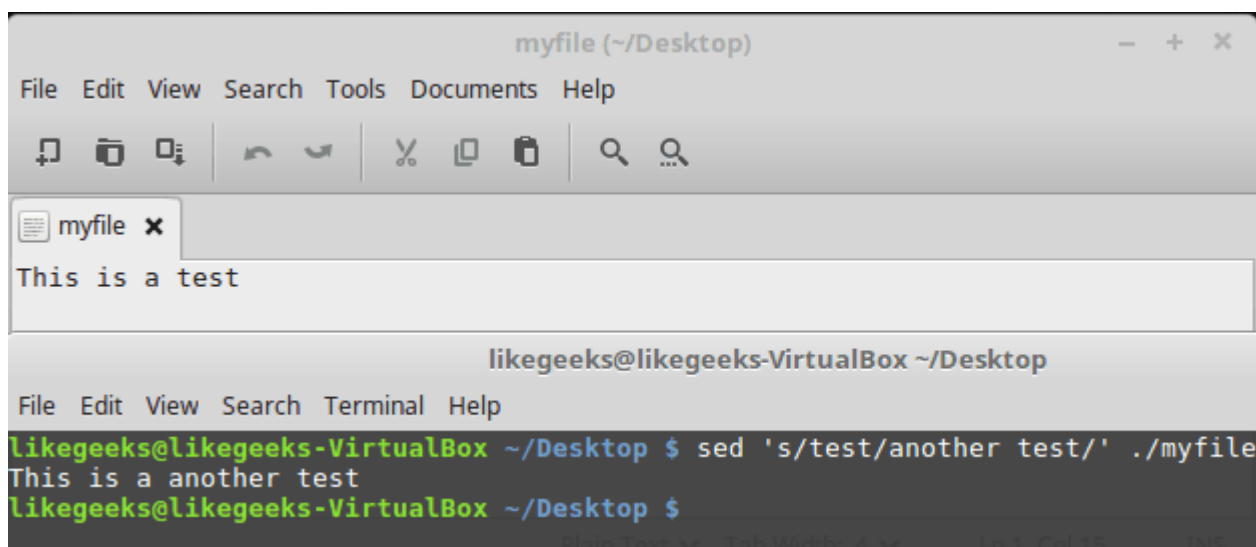
Простой пример вызова `sed`

В данном случае `sed` заменяет слово «test» в строке, переданной для обработки, словами «another test». Для оформления правила обработки текста, заключённого в кавычки, используются прямые слэши. В нашем случае применена команда вида `s/pattern1/pattern2/`. Буква «s» — это сокращение слова «substitute», то есть — перед нами команда замены. `Sed`, выполняя эту команду, просмотрит переданный текст и заменит найденные в нём фрагменты (о том — какие именно, поговорим ниже), соответствующие `pattern1`, на `pattern2`.

Выше приведён примитивный пример использования `sed`, нужный для того, чтобы ввести вас в курс дела. На самом деле, `sed` можно применять в гораздо более сложных сценариях обработки текстов, например — для работы с файлами.

Ниже показан файл, в котором содержится фрагмент текста, и результаты его обработки такой командой:

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



Текстовый файл и результаты его обработки

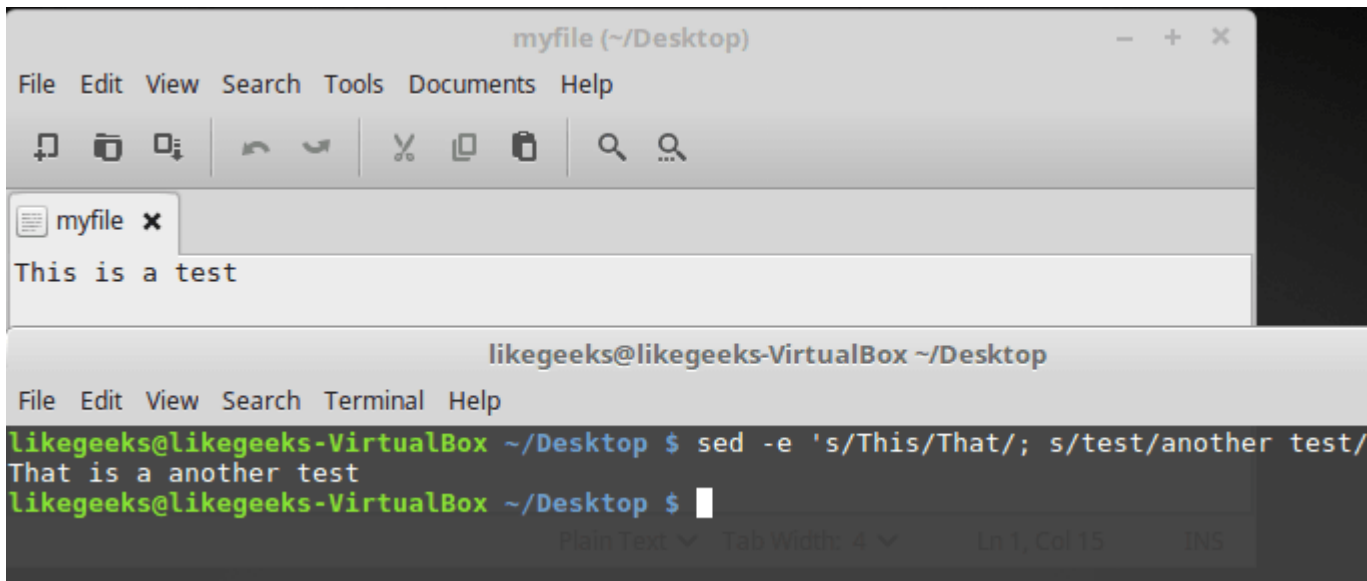
Здесь применён тот же подход, который мы использовали выше, но теперь `sed` обрабатывает текст, хранящийся в файле. При этом, если файл достаточно велик, можно заметить, что `sed` обрабатывает данные порциями и выводит то, что обработано, на экран, не дожидаясь обработки всего файла.

`Sed` не меняет данные в обрабатываемом файле. Редактор читает файл, обрабатывает прочитанное, и отправляет то, что получилось, в `STDOUT`. Для того, чтобы убедиться в том, что исходный файл не изменился, достаточно, после того, как он был передан `sed`, открыть его. При необходимости вывод `sed` можно перенаправить в файл, возможно — перезаписать старый файл. Если вы знакомы с одним из предыдущих [материалов](#) этой серии, где речь идёт о перенаправлении потоков ввода и вывода, вы вполне сможете это сделать.

Выполнение наборов команд при вызове `sed`

Для выполнения нескольких действий с данными, используйте ключ `-e` при вызове `sed`. Например, вот как организовать замену двух фрагментов текста:

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



Использование ключа -e при вызове sed

К каждой строке текста из файла применяются обе команды. Их нужно разделить точкой с запятой, при этом между окончанием команды и точкой с запятой не должно быть пробела.

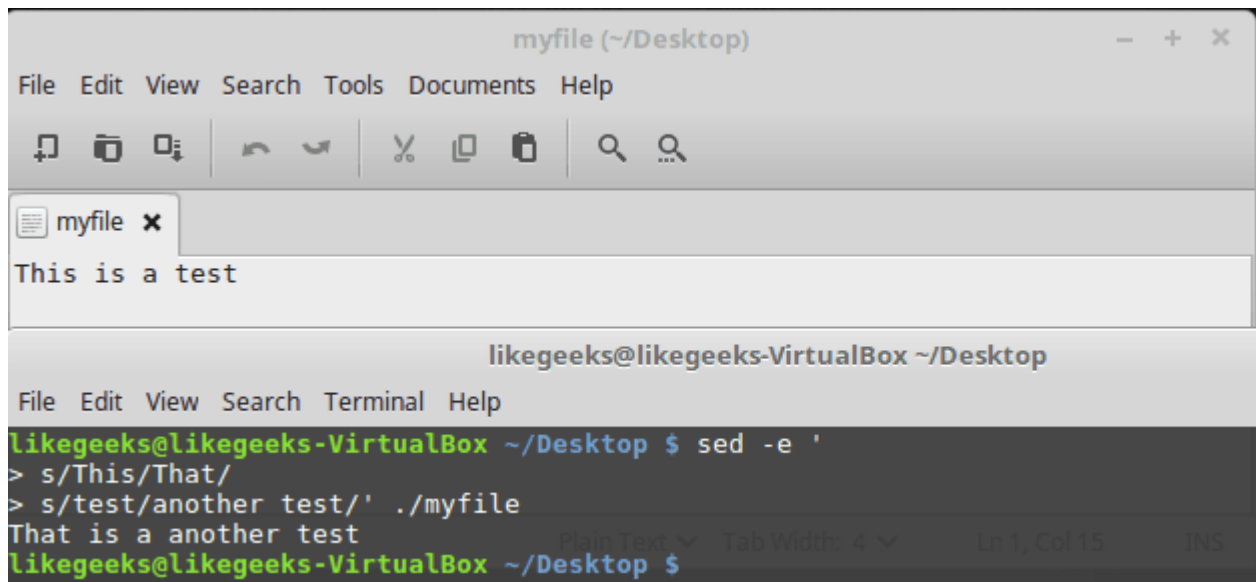
Для ввода нескольких шаблонов обработки текста при вызове sed, можно, после ввода первой одиночной кавычки, нажать Enter, после чего вводить каждое правило с новой строки, не забыв о закрывающей кавычке:

```
$ sed -e '
```

```
> s/This/That/
```

```
> s/test/another test/' ./myfile
```

Вот что получится после того, как команда, представленная в таком виде, будет выполнена.



Другой способ работы с sed

Чтение команд из файла

Если имеется множество команд `sed`, с помощью которых надо обработать текст, обычно удобнее всего предварительно записать их в файл. Для того, чтобы указать `sed` файл, содержащий команды, используют ключ `-f`:

Вот содержимое файла `mycommands`:

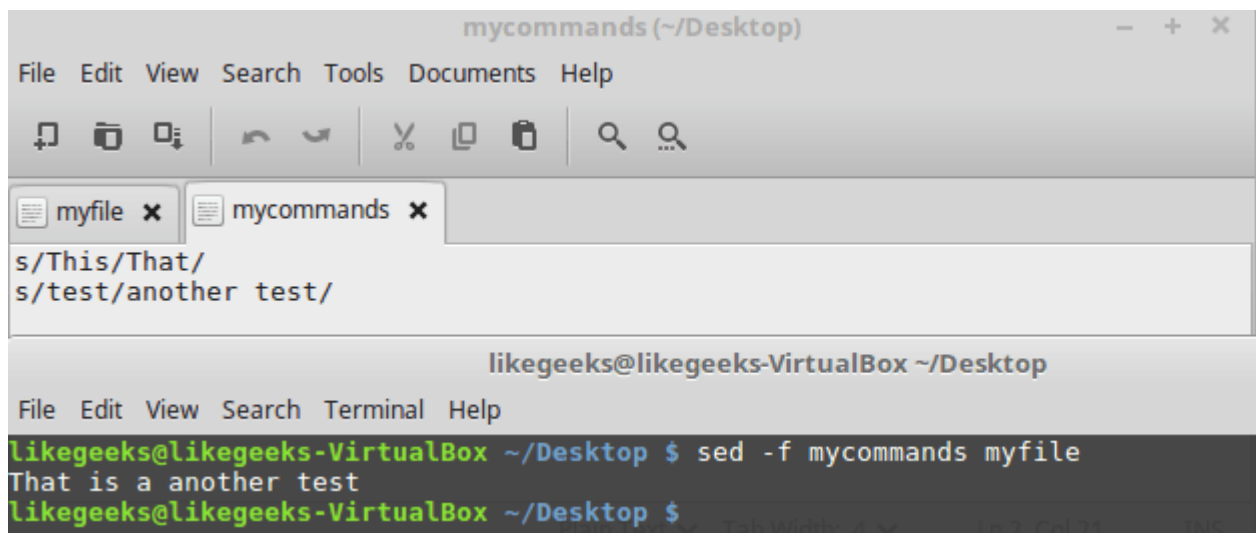
```
s/This/That/
```

```
s/test/another test/
```

Вызовем `sed`, передав редактору файл с командами и файл для обработки:

```
$ sed -f mycommands myfile
```

Результат при вызове такой команды аналогичен тому, который получался в предыдущих примерах.



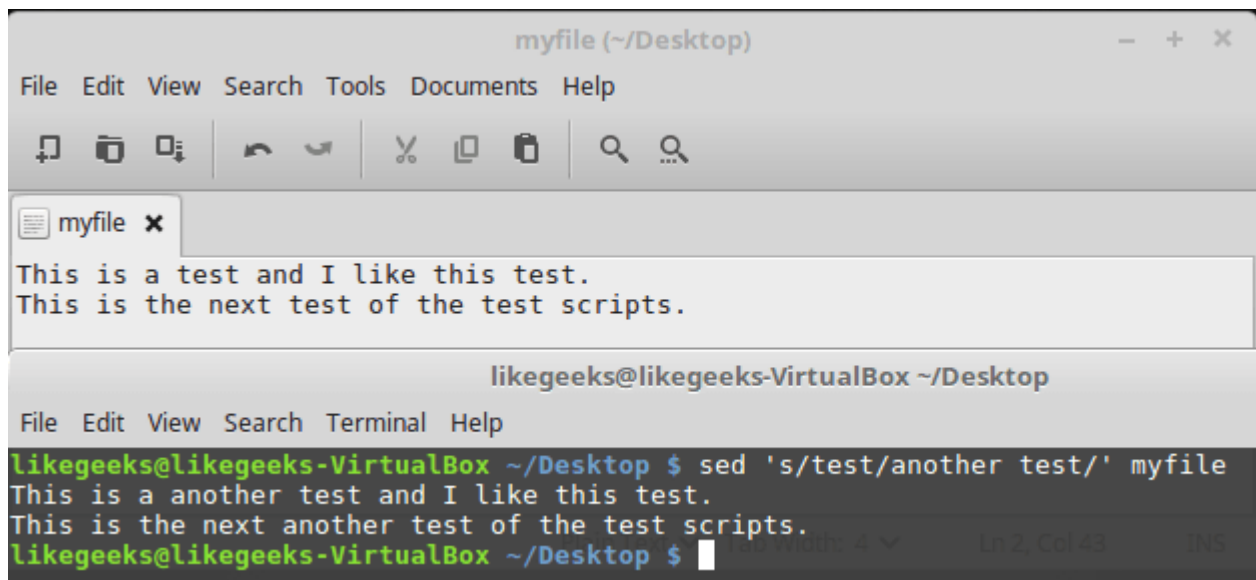
Использование файла с командами при вызове sed

Флаги команды замены

Внимательно посмотрите на следующий пример.

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

Вот что содержится в файле, и что будет получено после его обработки sed.



Исходный файл и результаты его обработки

Команда замены нормально обрабатывает файл, состоящий из нескольких строк, но заменяются только первые вхождения искомого фрагмента текста в каждой строке. Для того, чтобы заменить все вхождения шаблона, нужно использовать соответствующий флаг.

Схема записи команды замены при использовании флагов выглядит так:

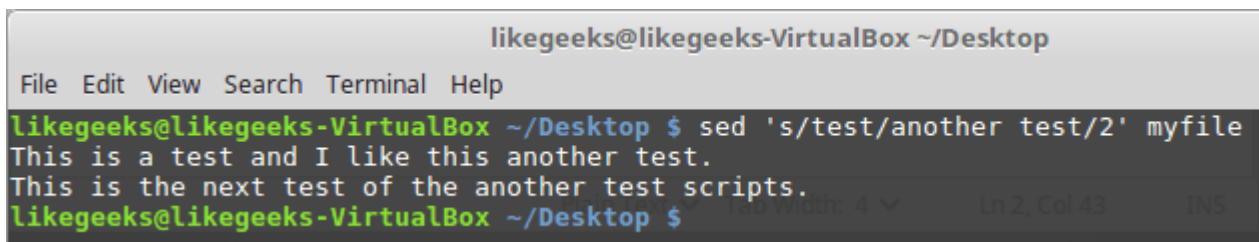
```
s/pattern/replacement/flags
```

Выполнение этой команды можно модифицировать несколькими способами.

- При передаче номера учитывается порядковый номер вхождения шаблона в строку, заменено будет именно это вхождение.
- Флаг `g` указывает на то, что нужно обработать все вхождения шаблона, имеющиеся в строке.
- Флаг `p` указывает на то, что нужно вывести содержимое исходной строки.
- Флаг вида `w file` указывает команде на то, что нужно записать результаты обработки текста в файл.

Рассмотрим использование первого варианта команды замены, с указанием позиции заменяемого вхождения искомого фрагмента:

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



```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed 's/test/another test/2' myfile
This is a test and I like this another test.
This is the next test of the another test scripts.
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Вызов команды замены с указанием позиции заменяемого фрагмента

Тут мы указали, в качестве флага замены, число 2. Это привело к тому, что было заменено лишь второе вхождение искомого шаблона в каждой строке. Теперь опробуем флаг глобальной замены — `g`:

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

Как видно из результатов вывода, такая команда заменила все вхождения шаблона в тексте.

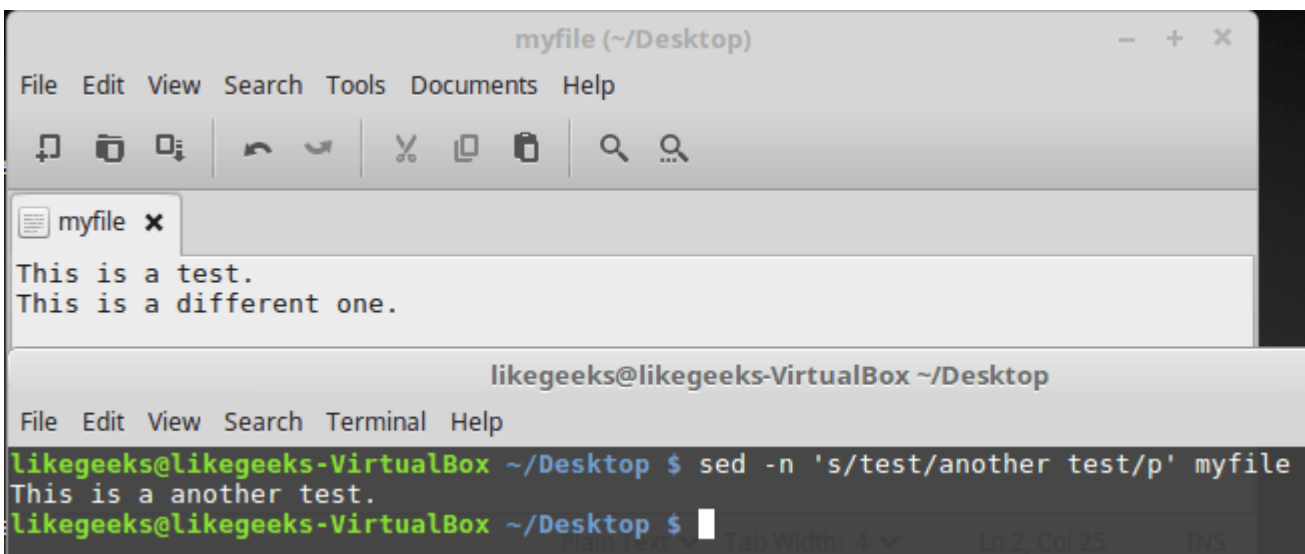
```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed 's/test/another test/g' myfile
This is a another test and I like this another test.
This is the next another test of the another test scripts.
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Глобальная замена

Флаг команды замены `p` позволяет выводить строки, в которых найдены совпадения, при этом ключ `-n`, указанный при вызове `sed`, подавляет обычный вывод:

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

Как результат, при запуске `sed` в такой конфигурации на экран выводятся лишь строки (в нашем случае — одна строка), в которых найден заданный фрагмент текста.



```
myfile (~/Desktop)
File Edit View Search Tools Documents Help
This is a test.
This is a different one.

likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed -n 's/test/another test/p' myfile
This is a another test.
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Использование флага команды замены `p`

Воспользуемся флагом `w`, который позволяет сохранить результаты обработки текста в файл:

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



```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed 's/test/another test/w output' myfile
This is a another test.
This is a different one.
likegeeks@likegeeks-VirtualBox ~/Desktop $ cat output
This is a another test.
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Сохранение результатов обработки текста в файл

Хорошо видно, что в ходе работы команды данные выводятся в [STDOUT](#), при этом обработанные строки записываются в файл, имя которого указано после `w`.

Символы-разделители

Представьте, что нужно заменить `/bin/bash` на `/bin/csh` в файле `/etc/passwd`. Задача не такая уж и сложная:

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

Однако, выглядит всё это не очень-то хорошо. Всё дело в том, что так как прямые слэши используются в роли символов-разделителей, такие же символы в передаваемых `sed` строках приходится экранировать. В результате страдает читаемость команды.

К счастью, `sed` позволяет нам самостоятельно задавать символы-разделители для использования их в команде замены. Разделителем считается первый символ, который будет встречен после `s`:

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

В данном случае в качестве разделителя использован восклицательный знак, в результате код легче читать и он выглядит куда опрятнее, чем прежде.

Выбор фрагментов текста для обработки

До сих пор мы вызывали `sed` для обработки всего переданного редактору потока данных. В некоторых случаях с помощью `sed` надо обработать лишь какую-то часть текста — некую конкретную строку или группу строк. Для достижения такой цели можно воспользоваться двумя подходами:

- Задать ограничение на номера обрабатываемых строк.

- Указать фильтр, соответствующие которому строки нужно обработать.

Рассмотрим первый подход. Тут допустимо два варианта. Первый, рассмотренный ниже, предусматривает указание номера одной строки, которую нужно обработать:

```
$ sed '2s/test/another test/' 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.
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed '2s/test/another test/' myfile
This is a test.
This is the second another test.
This is the thrid test.
This is the fourth test.
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Обработка только одной строки, номер который задан при вызове sed

Второй вариант — диапазон строк:

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

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

Обработка диапазона строк

Кроме того, можно вызвать команду замены так, чтобы файл был обработан начиная с некоей строки и до конца:

```
$ 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 $
```

Обработка файла начиная со второй строки и до конца

Для того, чтобы обрабатывать с помощью команды замены только строки, соответствующие заданному фильтру, команду надо вызвать так:

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

По аналогии с тем, что было рассмотрено выше, шаблон передаётся перед именем команды `s`.

```
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 $
```

Обработка строк, соответствующих фильтру

Тут мы использовали очень простой фильтр. Для того, чтобы в полной мере раскрыть возможности данного подхода, можно воспользоваться регулярными выражениями. О них мы поговорим в одном из следующих материалов этой серии.

Удаление строк

Утилита `sed` годится не только для замены одних последовательностей символов в строках на другие. С её помощью, а именно, используя команду `d`, можно удалять строки из текстового потока.

Вызов команды выглядит так:

```
$ sed '3d' myfile
```

Мы хотим, чтобы из текста была удалена третья строка. Обратите внимание на то,

что речь не идёт о файле. Файл останется неизменным, удаление отразится лишь на выводе, который сформирует sed.

```
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.
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed '3d' myfile
This is a test.
This is the second test.
This is the fourth test.
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Удаление третьей строки

Если при вызове команды `d` не указать номер удаляемой строки, удалены будут все строки потока.

Вот как применить команду `d` к диапазону строк:

```
$ 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 $
```

Удаление диапазона строк

А вот как удалить строки, начиная с заданной — и до конца файла:

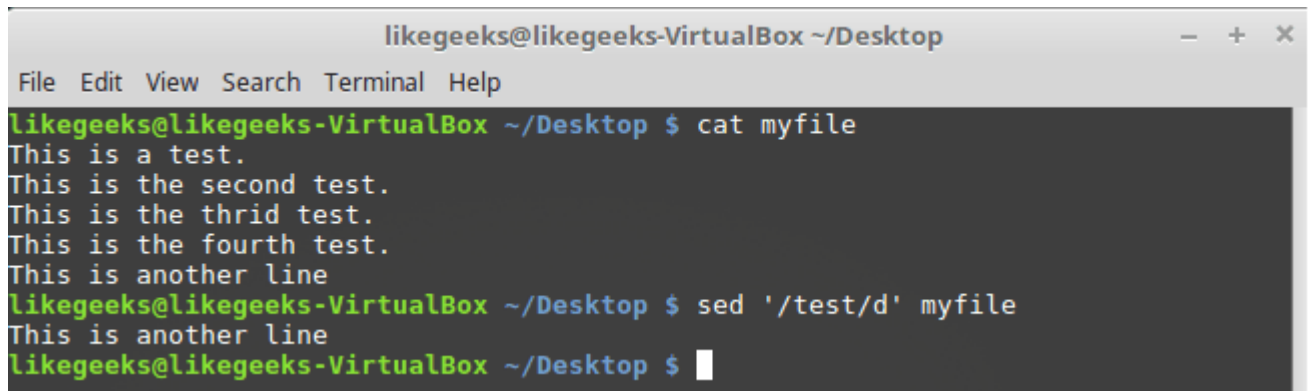
```
$ sed '3,$d' myfile
```

```
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 $
```

Удаление строк до конца файла

Строки можно удалять и по шаблону:

```
$ sed '/test/d' myfile
```

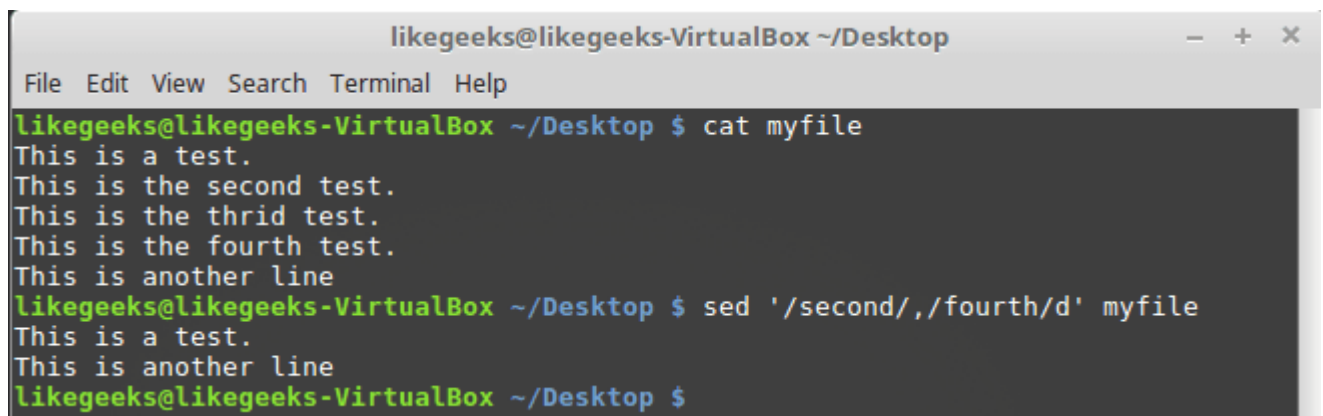


```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ cat myfile
This is a test.
This is the second test.
This is the thrid test.
This is the fourth test.
This is another line
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed '/test/d'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

Удаление строк по шаблону

При вызове `d` можно указывать пару шаблонов — будут удалены строки, в которых встретится шаблон, и те строки, которые находятся между ними:

```
$ sed '/second/,/fourth/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 '/second/,/fourth/d' myfile
This is a test.
This is another line
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Удаление диапазона строк с использованием шаблонов

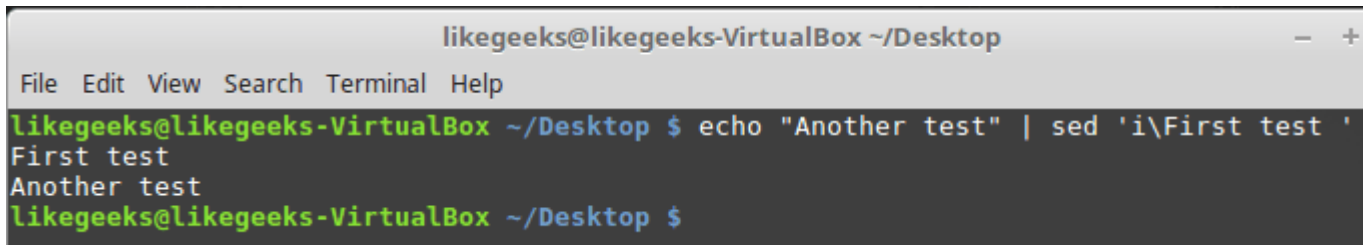
Вставка текста в поток

С помощью `sed` можно вставлять данные в текстовый поток, используя команды `i` и `a`:

- Команда `i` добавляет новую строку перед заданной.
- Команда `a` добавляет новую строку после заданной.

Рассмотрим пример использования команды `i`:

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

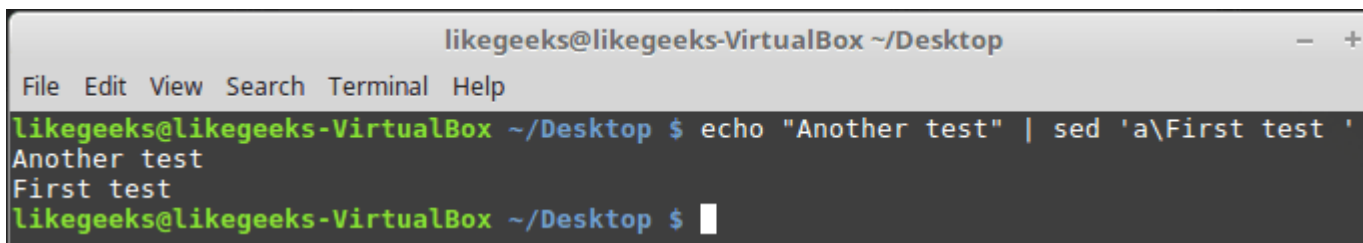


```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ echo "Another test" | sed 'i\First test '
First test
Another test
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Команда `i`

Теперь взглянем на команду `a`:

```
$ 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 $
```

Команда `a`

Как видно, эти команды добавляют текст до или после данных из потока. Что если надо добавить строку где-нибудь посередине?

Тут нам поможет указание номера опорной строки в потоке, или шаблона. Учтите, что адресация строк в виде диапазона тут не подойдёт. Вызовем команду `i`, указав номер строки, перед которой надо вставить новую строку:

```
$ 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 $
```

Команда i с указанием номера опорной строки

Проделаем то же самое с командой a:

```
$ 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 $
```

Команда a с указанием номера опорной строки

Обратите внимание на разницу в работе команд i и a. Первая вставляет новую строку до указанной, вторая — после.

Замена строк

Команда c позволяет изменить содержимое целой строки текста в потоке данных. При её вызове нужно указать номер строки, вместо которой в поток надо добавить новые данные:

```
$ 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 $
```

Замена строки целиком

Если воспользоваться при вызове команды шаблоном в виде обычного текста или регулярного выражения, заменены будут все соответствующие шаблону строки:

```
$ sed '/This is/c This is a changed line of text.' myfile
```

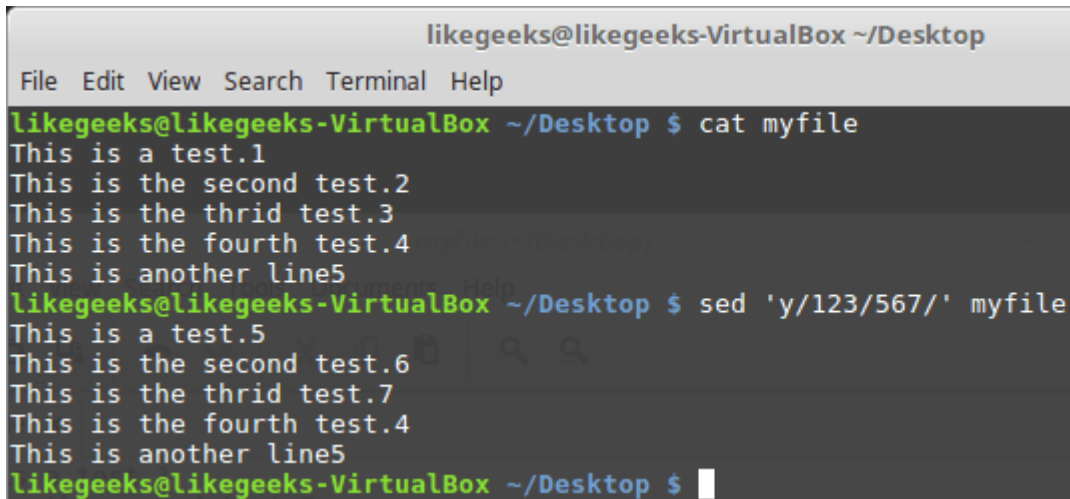
```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ cat myfile
This is a test.
This is the second test.
This is the thrid test.
This is the fourth test.
This is another line
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed '/This is/c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

Замена строк по шаблону

Замена символов

Команда `y` работает с отдельными символами, заменяя их в соответствии с переданными ей при вызове данными:


```
$ sed 'y/123/567/' myfile
```



A terminal window titled 'likegeeks@likegeeks-VirtualBox ~/Desktop' with a menu bar (File, Edit, View, Search, Terminal, Help). The prompt is 'likegeeks@likegeeks-VirtualBox ~/Desktop \$'. The user enters 'cat myfile', and the output is: 'This is a test.1', 'This is the second test.2', 'This is the thrid test.3', 'This is the fourth test.4', 'This is another line5'. Then the user enters 'sed 'y/123/567/' myfile', and the output is: 'This is a test.5', 'This is the second test.6', 'This is the thrid test.7', 'This is the fourth test.4', 'This is another line5'. The prompt is now 'likegeeks@likegeeks-VirtualBox ~/Desktop \$'.

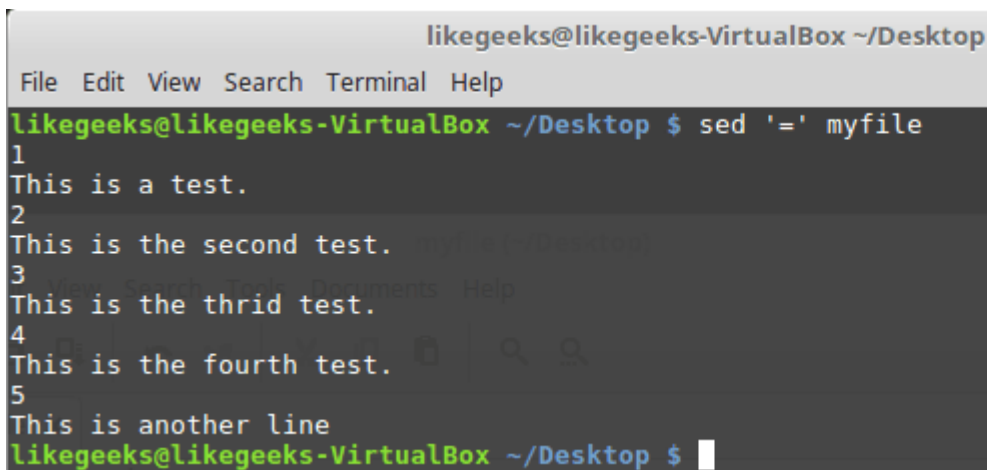
Замена символов

Используя эту команду, нужно учесть, что она применяется ко всему текстовому потоку, ограничить её конкретными вхождениями символов нельзя.

Вывод номеров строк

Если вызвать sed, используя команду =, утилита выведет номера строк в потоке данных:

```
$ sed '=' myfile
```



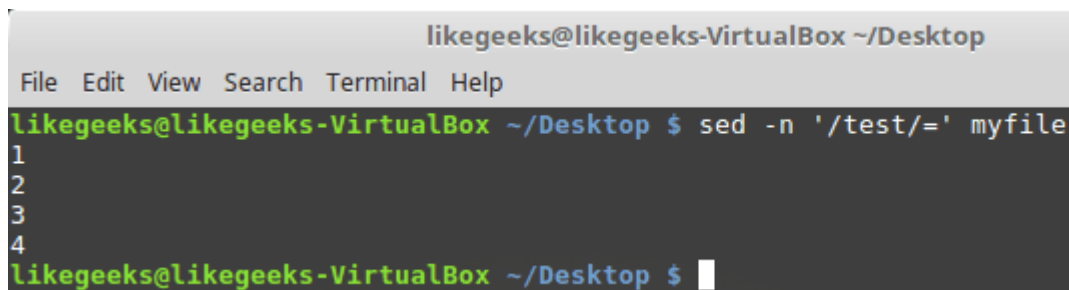
A terminal window titled 'likegeeks@likegeeks-VirtualBox ~/Desktop' with a menu bar (File, Edit, View, Search, Terminal, Help). The prompt is 'likegeeks@likegeeks-VirtualBox ~/Desktop \$'. The user enters 'sed '=' myfile', and the output is: '1', 'This is a test.', '2', 'This is the second test.', '3', 'This is the thrid test.', '4', 'This is the fourth test.', '5', 'This is another line'. The prompt is now 'likegeeks@likegeeks-VirtualBox ~/Desktop \$'.

Вывод номеров строк

Потоковый редактор вывел номера строк перед их содержимым.

Если передать этой команде шаблон и воспользоваться ключом `sed -n`, выведены будут только номера строк, соответствующих шаблону:

```
$ 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 $
```

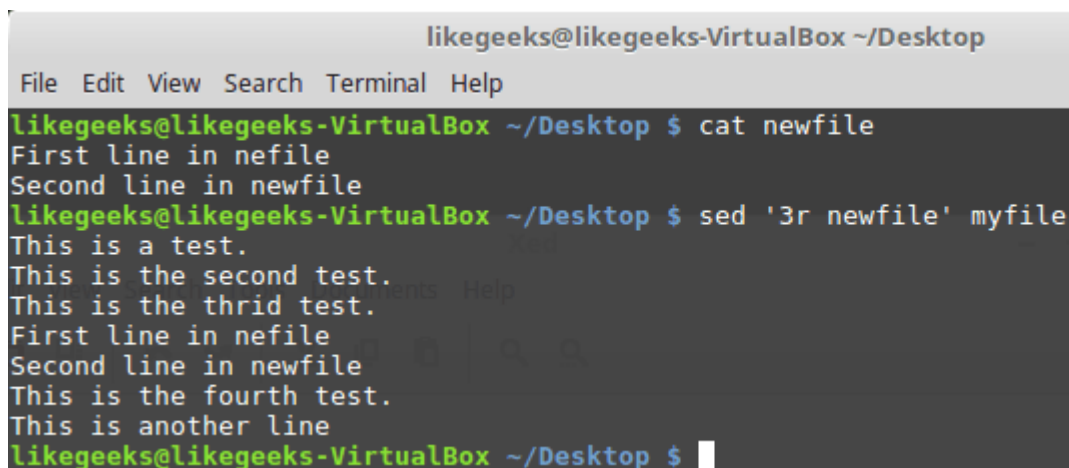
Вывод номеров строк, соответствующих шаблону

Чтение данных для вставки из файла

Выше мы рассматривали приёмы вставки данных в поток, указывая то, что надо вставить, прямо при вызове `sed`. В качестве источника данных можно воспользоваться и файлом. Для этого служит команда `r`, которая позволяет вставлять в поток данные из указанного файла. При её вызове можно указать номер строки, после которой надо вставить содержимое файла, или шаблон.

Рассмотрим пример:

```
$ sed '3r newfile' myfile
```



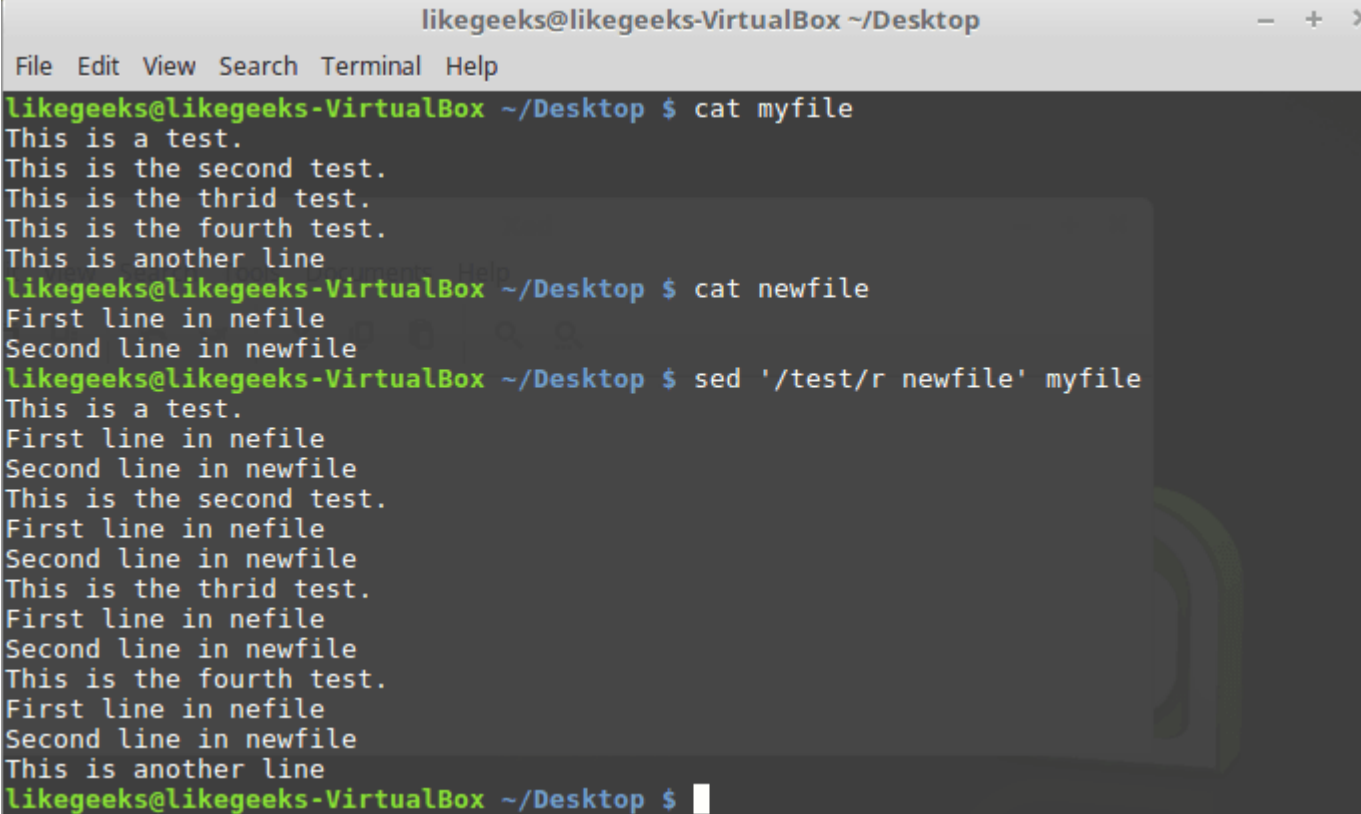
```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ cat newfile
First line in newfile
Second line in newfile
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed '3r newfile' myfile
This is a test.
This is the second test.
This is the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

Вставка в поток содержимого файла

Тут содержимое файла `newfile` было вставлено после третьей строки файла `myfile`.

Вот что произойдёт, если применить при вызове команды `r` шаблон:

```
$ 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 $
```

Использование шаблона при вызове команды `r`

Содержимое файла будет вставлено после каждой строки, соответствующей шаблону.

Пример

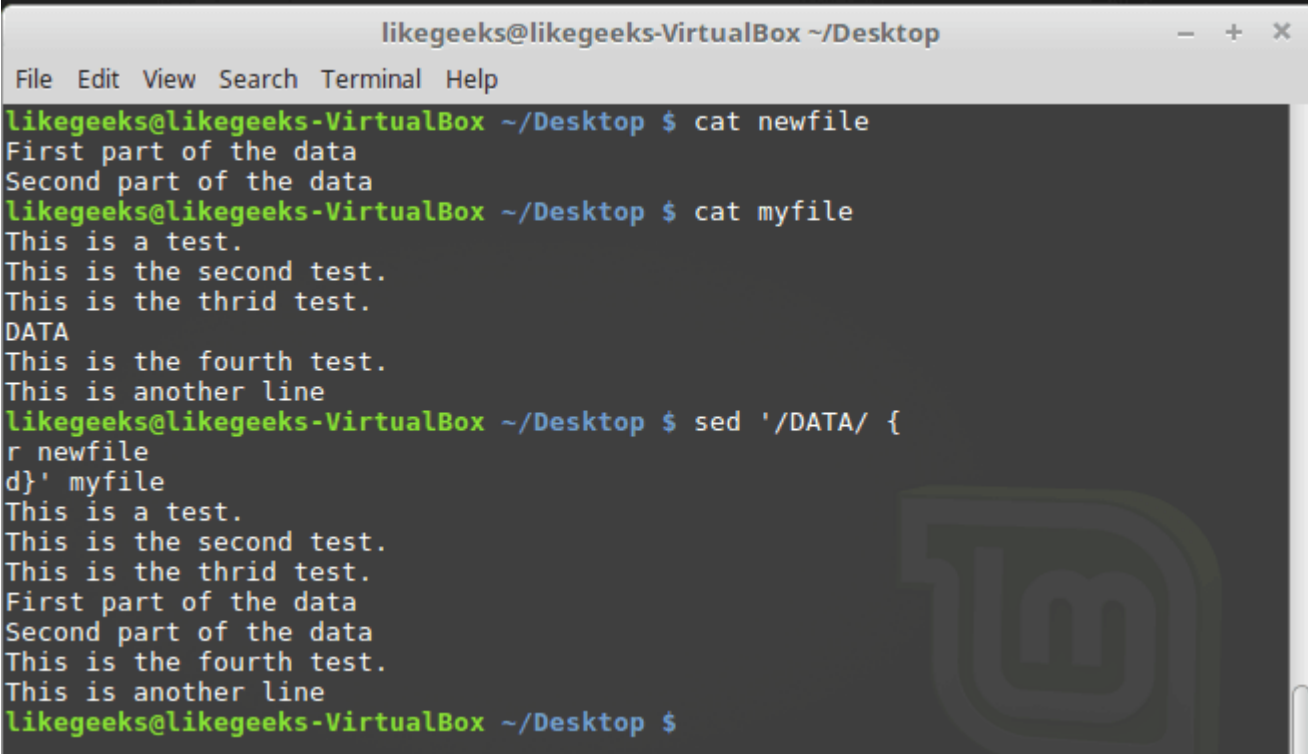
Представим себе такую задачу. Есть файл, в котором имеется некая последовательность символов, сама по себе бессмысленная, которую надо заменить на данные, взятые из другого файла. А именно, пусть это будет файл `newfile`, в котором роль указателя места заполнения играет последовательность символов `DATA`. Данные, которые нужно подставить вместо `DATA`, хранятся в файле `data`.

Решить эту задачу можно, воспользовавшись командами `r` и `d` потокового редактора `sed`:

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

```
r newfile
```

```
d}' myfile
```



```
likegeeks@likegeeks-VirtualBox ~/Desktop
File Edit View Search Terminal Help
likegeeks@likegeeks-VirtualBox ~/Desktop $ cat newfile
First part of the data
Second part of the data
likegeeks@likegeeks-VirtualBox ~/Desktop $ cat myfile
This is a test.
This is the second test.
This is the thrid test.
DATA
This is the fourth test.
This is another line
likegeeks@likegeeks-VirtualBox ~/Desktop $ sed '/DATA/ {
r newfile
d}' myfile
This is a test.
This is the second test.
This is the thrid test.
First part of the data
Second part of the data
This is the fourth test.
This is another line
likegeeks@likegeeks-VirtualBox ~/Desktop $
```

Замена указателя места заполнения на реальные данные

Как видите, вместо заполнителя `DATA` `sed` добавил в выходной поток две строки из файла `data`.

Итоги

Сегодня мы рассмотрели основы работы с потоковым редактором `sed`. На самом деле, `sed` — это огромнейшая тема. Его изучение вполне можно сравнить с изучением нового языка программирования, однако, поняв основы, вы сможете освоить `sed` на любом необходимом вам уровне. В результате ваши возможности по обработке с его помощью текстов будет ограничивать лишь воображение.

На сегодня это всё. В следующий раз поговорим о языке обработки данных `awk`.