Министерство науки и высшего образования Российской Федерации

Рязанский государственный радиотехнический университет им. В.Ф. Уткина

Потоковый редактор sed и утилита tr

Методические указания к практическим занятиям

Рязань 2020

УДК 681.3.06

Потоковый редактор sed и утилита tr: методические указания к практическим занятиям / Рязан. гос. радиотехн. ун-т.; сост. А.А. Митрошин, В.Г. Псоянц. – Рязань, 2020. – 16 с.

Содержат описание практического занятия, используемого в курсе «Операционная система Linux». Могут использоваться при изучении других курсов, связанных с операционной системой Linux.

Предназначены для студентов очной, заочной и очно-заочной форм обучения направления подготовки «Информатика и вычислительная техника». Могут использоваться для студентов других направлений подготовки.

Могут использоваться как методические указания к лабораторным работам в курсах, связанных с изучением операционной системы Linux и свободно распространяемого программного обеспечения.

Ил. --. Библиогр.: -- назв.

Операционная система Linux, потоковый редактор sed, утилита tr

Печатается по решению редакционно-издательского совета Рязанского государственного радиотехнического университета.

Рецензент: кафедра САПР вычислительных средств Рязанского государственного радиотехнического университета (зав. кафедрой засл. деят. науки и техники РФ В.П.Корячко)

Потоковый редактор sed и утилита tr

Составители: Митрошин Александр Александрович

Псоянц Владимир Грикорович

Редактор \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Корректор \_\_\_\_\_\_\_\_\_\_\_\_\_

Подписано в печать \_\_\_\_\_\_\_\_. Формат бумаги 60×84 1/16.

Бумага газетная. Печать трафаретная. Усл. печ. л. 1,0.

Уч-изд. л. 1,0. Тираж 50 экз. Заказ

Рязанский государственный радиотехнический университет.

390005, Рязань, ул. Гагарина, 59/1.

Редакционно-издательский центр РГРТУ.

**Потоковый редактор sed**

Редактор sed (**s**tream **ed**itor – потоковый редактор) не является интерактивной программой, а относится к редакторам потокового типа. Он предназначен для обработки текстовых данных, поступающих из файла или стандартного входного потока. Редактор sed позволяет модифицировать данные в фоновом режиме.

При использовании sed пользователь определяет только команды редактирования и данные для обработки. Поскольку sed не предназначен для работы в интерактивном режиме, текстовые строки, подлежащие изменению, отбираются либо по номерам, либо на основании соответствия регулярному выражению.

Команды sed вводятся в командной строке либо размещаются в файле сценария, аналогично утилите awk.

Общая схема работы sed такова.

1) Редактор последовательно извлекает строки из файла или стандартного входного потока и копирует их *в буфер* для редактирования.

2) Затем он считывает первую команду из командной строки или из файла сценария, осуществляет поиск строки с указанным номером или строки, соответствующей шаблону, и применяет к ней эту команду.

3) Второй пункт повторяется до тех пор, пока не буден исчерпан весь список команд.

***Вызов sed***

Если sed вызывается для выполнения одиночных команд, можно использовать следующий формат вызова:

sed [опции] 'команды' входной\_файл

Если команды sed помещены в отдельный файл, sed вызывается следующим образом:

sed [опции] –f файл\_сценария входной\_файл

Если файл сценария является исполняемым, то он запускается для выполнения с помощью команды

файл\_сценария [опции] входной\_файл

Если, при любом способе вызова, входной файл не указан, то sed ,будет ожидать поступления данных из стандартного входного потока: с клавиатуры или из канала.

Основные опции sed перечислены в табл. 1.

Таблица 1. Основные опции sed

|  |  |
| --- | --- |
| **Опция** | **Описание** |
| -n | Запрет вывода на экран. При наличии этой опции sed не будет выводить обрабатываемые им строки в стандартный выходной поток; по умолчанию отображается каждая входная строка. Осуществить вывод нужной строки можно с помощью команды -p. |
| -e | Следующей командой будет команда редактирования. Опция используется, когда команд редактирования несколько |
| -f | Используется при подключении файла сценария. |

**Синтаксис команд**

Общий синтаксис команд sed:

[адрес\_1[, адрес\_2]] [!] команда [аргументы]

Команда состоит из одной буквы. Основные команды sed приведены в табл. 2.

Таблица 2. Основные команды sed

|  |  |
| --- | --- |
| **Команда** | **Описание** |
| p | Вывод адресуемых строк |
| = | Вывод номеров адресуемых строк |
| a | Добавление заданного текста *после* каждой адресуемой строки |
| i | Добавление заданного текста *перед* каждой адресуемой строкой |
| c | Замена адресуемого текста заданным текстом |
| d | Удаление адресуемых строк |
| s | Замена указанного шаблона заданным текстом в каждой адресуемой строке |
| w | Добавление адресуемых строк в указанный файл |
| r | Чтение текста из указанного файла и добавление его после каждой адресуемой строка |
| q | Завершение работы после того, как достигнута адресуемая строка |
| l | Вывод адресуемых строк с отображением непечатаемых символов в виде ASCII-кодов и переносом длинных строк |

***Вывод строк***

Команда отображения строк **p** (print - печать) имеет формат:

[адрес\_1[, адрес\_2]]p

***Примеры:***

**sed –n ‘2p’ test.txt**

Выводит вторую строку файла test.txt. Опция –n подавляет вывод на экран всех строк, что делается по умолчанию.

**sed –n ‘/The/p’ test.txt**

Выводит строки, содержащие строку «The».

***Вывод номеров строк***

Для вывода номера строки, соответствующей заданному адресу, используется команда **=**.

***Пример:***

**sed –n ‘/SAPR/=’ test.txt**

Выводятся номера строк, содержащих строку «SAPR» в файле test.txt.

***Команда замены***

Для замены используется команда s (substitute - замена). Синтаксис команды **s**:

**[address]s/pattern/replacement/flags**

где:

[address] – адрес строк, в которых производится замена;

/pattern/ - шаблон замещаемого текста;

replacement – замещающий текст;

flags – флаги.

Флаги модифицируют команду замены и могут быть следующими:

- n – число от 1 до 512, определяет, что замена должна быть произведена только n соответствий шаблону;

- g – определяет, что замена должна производиться для всех соответствий шаблону. По умолчанию замена производится только для первого соответствия;

- p – печать строки, соответствующей шаблону;

- w file – записывает содержимое соответствий шаблону в файл file.

Флаги могут комбинироваться. Например, комбинация флагов gp делает подстановку глобальной в строке и печатает строку.

В отличие от адресов строк, в которых «/» используется как разделитель, регулярные выражения могут быть разделены любыми символами, за исключением символа конца строки. Например, если символ «/» входит в шаблон, в качестве разделителя можно использовать символ «!» вместо символа «/»: s**!**/usr/mail**!**/mitr/mail**!** Обратите внимание на то, что разделитель появляется три раза, в том числе и после замещающего текста.

Замещающий текст – это строка символов, на которую будет заменен текст, соответствующий шаблону pattern. При описании замещающего текста следующие символы имеют специальное значение:

- & - замещает строку, соответствующую регулярному выражению. Позволяет в шаблоне замены сослаться на фрагмент строки, соответствующий шаблону.

- \n – соответствует n-ой подстроке (n – одна цифра) определенной в шаблоне, использующем символы «\(» и «\)». То есть, с помощью операторов «\(» и «\)» можно сохранить до 9 шаблонов поиска во временном буфере, с тем, чтобы в шаблоне замены обратиться к ним с помощью оператора /n, где n – номер сохраненного шаблона.

- \ - используется для завершения & и \, а также замены разделителя команд, тогда, когда в них используются символы в секции замены. Используется также для завершения новой строки и создания замещающего текста, состоящего из нескольких строк.

В следующем примере используется специальный символ \n.

**cat test1**

первый:второй

один:два

**sed ’s/\(.\*\):\(.\*\)/\2:\1/’ test1**

второй:первый

два:один

***Команда удаления***

Для удаления подстроки, соответствующей шаблону, используется команда **d** (delete - удаление). Команда удаления изменяет поток управления в сценарии. Поскольку входной файл изменяется и потенциально могут измениться номера строк, то сценарий редактирования sed начинает просматривать файл с начала.

***Примеры:***

**sed ‘1,3d’ test.txt**

Удаляются строки с 1 по 3 в файле test.txt.

**sed ‘/SAPR/d’ test.txt**

Удаляются строки файл test.txt, содержащие «SAPR».

***Команды добавления, вставки и изменения***

Для добавления, вставки и изменения используются команды **a** (append - добавление), **i** (insert - вставка) и **c** (change - изменение). Синтаксис команд следующий.

Добавление:

[*line-address*]a\

*text\*

*text\*

*…*

*text*

Вставка:

[line-address]i\

*text\*

*text\*

*…*

*text*

Изменение:

[address]c\

*text\*

*text\*

*…*

*text*

Команда **a** вставляет одну или несколько строк текста *после* адресуемой строки. Адрес может быть представлен в виде номера строки или регулярного выражения. Во втором случае найденных строк может быть несколько. При добавлении текста отсутствует возможность задания диапазона адресов строк. Допускается указание только одного шаблона адреса. Если адрес, по которому помещается текст, не указан, то команда применяется к каждой строке входного файла.

В конце каждой добавляемой строки и после самой команды **a** необходим символ «\», который защищает от интерпретации символ новой строки. В последней строке указывать символ «\» не требуется.

Добавляемый текст записывается в стандартный выходной поток и не дублируется во входном буфере sed, поэтому не подлежит дальнейшему редактированию. Чтобы иметь возможность отредактировать полученный текст, нужно сохранить результаты работы sed в новом файле и применить команды редактирования уже к нему.

Команда **i** аналогична команде **a** за исключением того, что команда **i** вставляет текст не после, а *перед* адресуемой строкой.

Команда **c** заменяет новым текстом каждую адресуемую строку. Если выбрана группа строк, вся группа заменяется одной копией текста.

***Вывод строк в файл***

Для вывода адресуемых строк в файл используется команда **w** (write - записать). Формат команды:

[адрес\_1[, адрес\_2]]w имя\_файла

Если указанный файл не существует, он будет создан, если существует – его содержимое будет перезаписано. Если в сценарии встречается несколько команд **w**, направляющих результаты в один и тот же файл, данные всех команд будут *добавляться* в конец файла.

***Чтение строк из файла***

В процессе обработки входного файла sed позволяет читать текст из другого файла и добавлять его к текущему содержимому буфера, размещая *после каждой* строки, соответствующей шаблону адреса. Для этого используется команда r (read - читать). Формат команды:

[адрес]r имя\_файла

***Досрочное завершение работы sed***

Иногда требуется завершить работу sed сразу после нахождения первого совпадения с шаблоном. Для этого предназначена команда q (quit - выход), имеющая следующий формат:

[адрес]q

**Отображение управляющих символов**

Часто в текстовых файлах содержатся различные непечатаемые символы. При выводе таких символов на экран могут быть получены странные результаты.

В sed существует команда **l** (list - список), аналогичная команде cat –v, которая специальным образом помечает непечатаемые символы. Формат команды:

[адрес\_1[, адрес\_2]]l

***Адресация строк***

Существует два способа адресации строк.

1) По номеру строки.

2) С помощью регулярных выражений.

В каждой команде может быть:

1) не указано ни одного адреса;

2) указан один адрес;

3) указано 2 адреса.

В табл. 3 указаны все возможные правила отбора строк в зависимости от того, сколько компонентов адреса указано.

Таблица 3. Правила отбора строк

|  |  |
| --- | --- |
| **Адрес** | **Отбираемые строки** |
| нет адреса | Все строки входного файла |
| x | Строка с номером x |
| x,y | Все строки с номерами в диапазоне от x до y |
| /шаблон/ | Все строки, соответствующие шаблону |
| /шаблон\_1/,/шаблон\_2/ | Группа строк, начиная от строки, соответствующей первому шаблону, и заканчивая строкой, соответствующей второму шаблону. Подобных групп во входном файле может быть несколько. |
| /шаблон/, x | Группа строк, начиная от строки, соответствующей шаблону, и заканчивая строкой с указанным номером. |
| x, /шаблон/ | Группа строк, начиная от строки с указанным номером и заканчивая строкой, соответствующей шаблону. |
| ! | Все строки, не соответствующие заданному перед этим адресу |
| $ | Последняя строка входного файла |

***Группировка команд***

С помощью фигурных скобок можно объединить несколько команд в группу. Возможны два варианта синтаксиса группировки.

1) [адрес\_1] [, адрес\_2]{

команда\_1

команда\_2

…

команда\_N

}

2) [адрес\_1][, адрес\_2]{команда\_1; … ; команда\_N;}

В первом случае каждая команда записывается в отдельной строке, а разделителем команд является символ новой строки. Во втором случае команды записываются последовательно, отделяясь друг от друга точкой с запятой, *которая ставится также после последней команды*.

***Создание файла сценария***

Когда сценарий становится большим, удобно хранить его в файле. Имена файлов сценариев для редактора sed традиционно завершаются на .sed.

Чтобы сделать файл сценария исполняемым, разместите в его первой строке «магические» символы

#! /bin/sed -f

Не забудьте сделать файл исполняемым, например, с помощью команды chmod u+x t.sed.

***Пример создания сценария***

Создадим файл, как это показано на рис. 1.

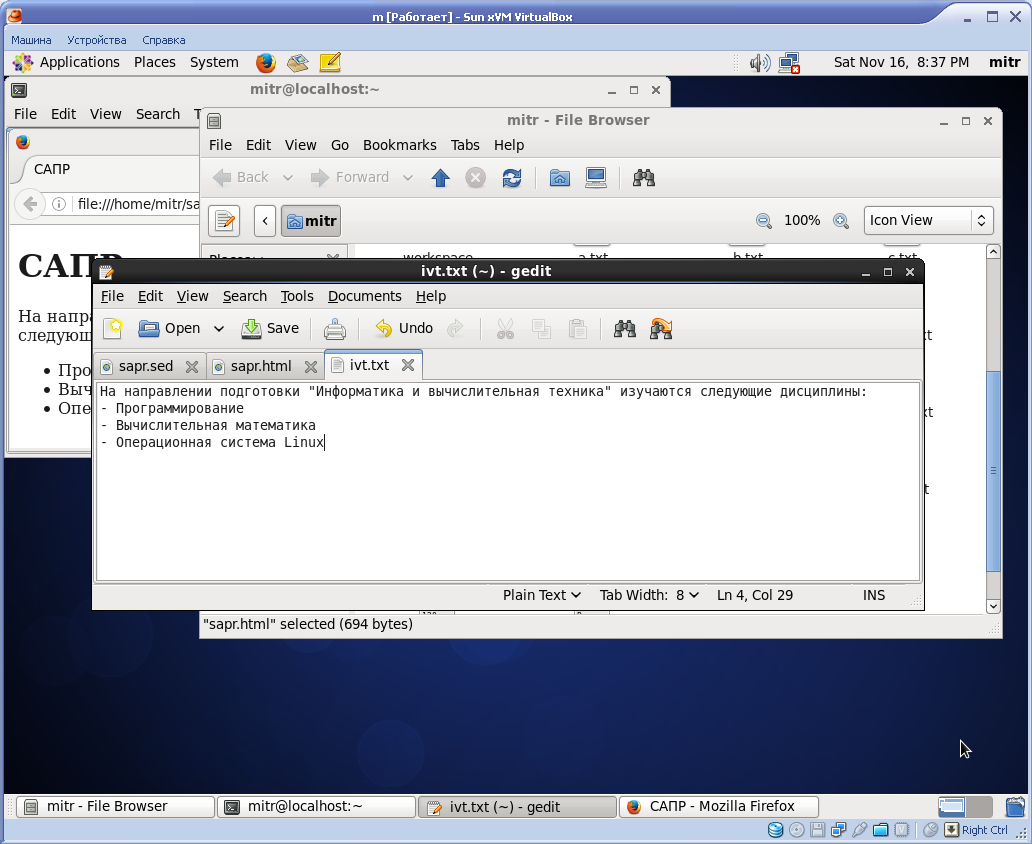


Рисунок 1. Файл inv.txt в редакторе gedit

Создадим сценарий sed, преобразующий этот текстовый файл в файл html. Сценарий sapr.sed показан на рис. 2.

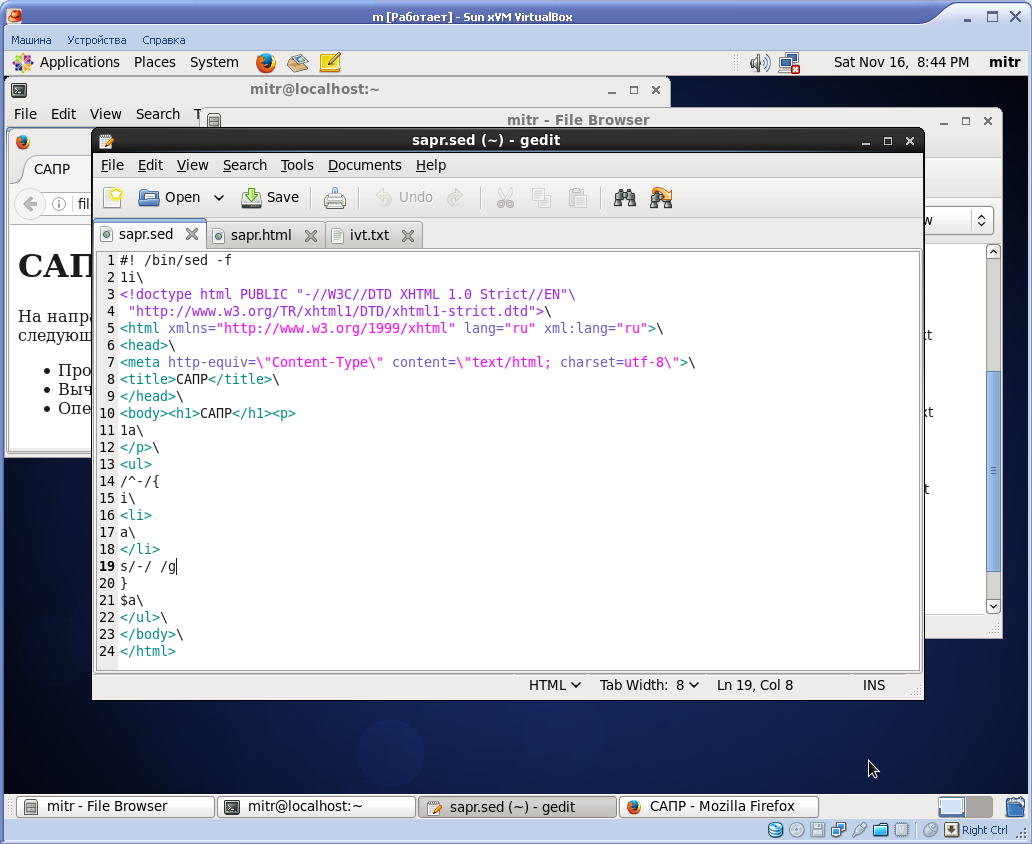


Рисунок 2. Сценарий sapr.sed

Первая строка сценария – «магические» символы, определяющие, что именно sed должен выполнять этот сценарий.

Вторая строка определяет, что *перед* первой строкой входного файла нужно поместить строки 3-10. Обратите внимание на то, и после адреса 1i и после каждой строки, кроме строки 10 помещен символ «\». Этот символ маскирует интерпретацию символ новой строки. В строке, предшествующей следующему адресу, указывать символ «\» не требуется.

Строка 11 определяет, что *после* первой строки входного файла нужно добавить строки 12 и 13.

Строка 14 адресует строки, которые начинаются с символа «-». Для каждой такой строки выполняются следующие команды:

- *перед* строкой помещается строка 16;

- после строки помещается строка 18;

- символ «-» в каждой строке удаляется.

После последней строки входного файла помещаются строки 22-24.

Если сценарий рис. 2 запустить на выполнение с помощью команды

**sapr.sed tvt.txt > sapr.html**

то получим файл sapr.html, показанный на рис. 3.

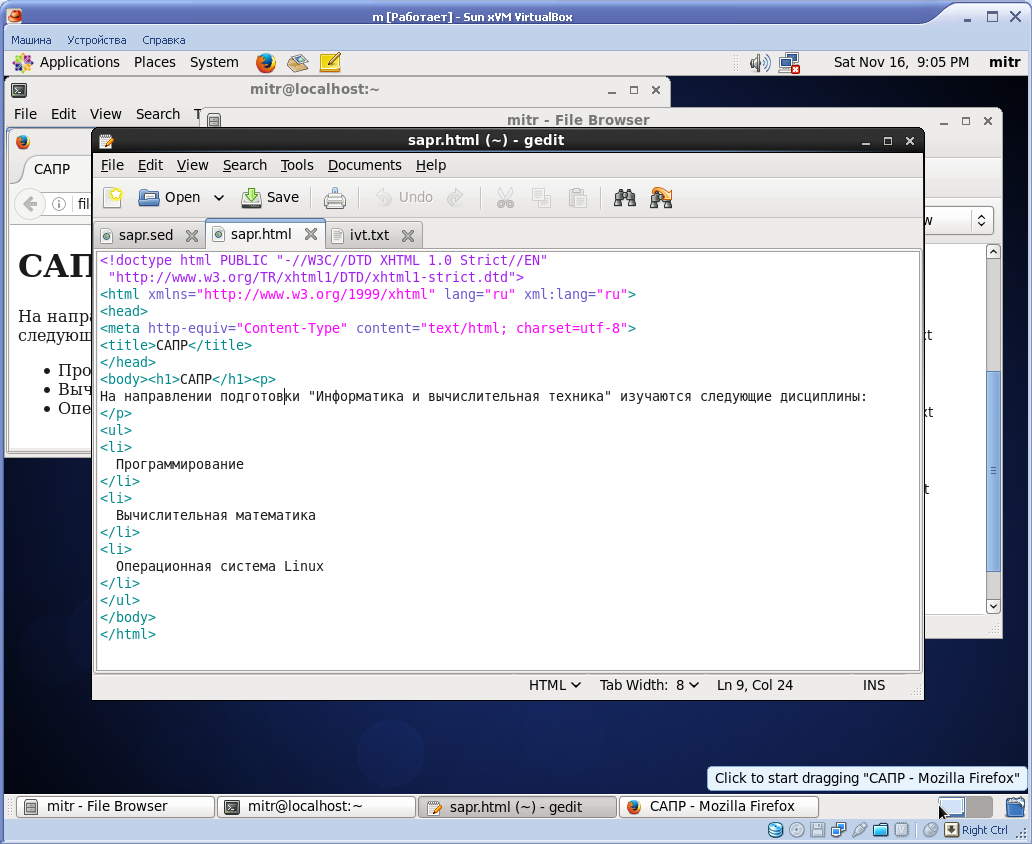


Рисунок 3. Файл sapr.html

В браузере этот файл выглядит так, как показано на рис. 4.

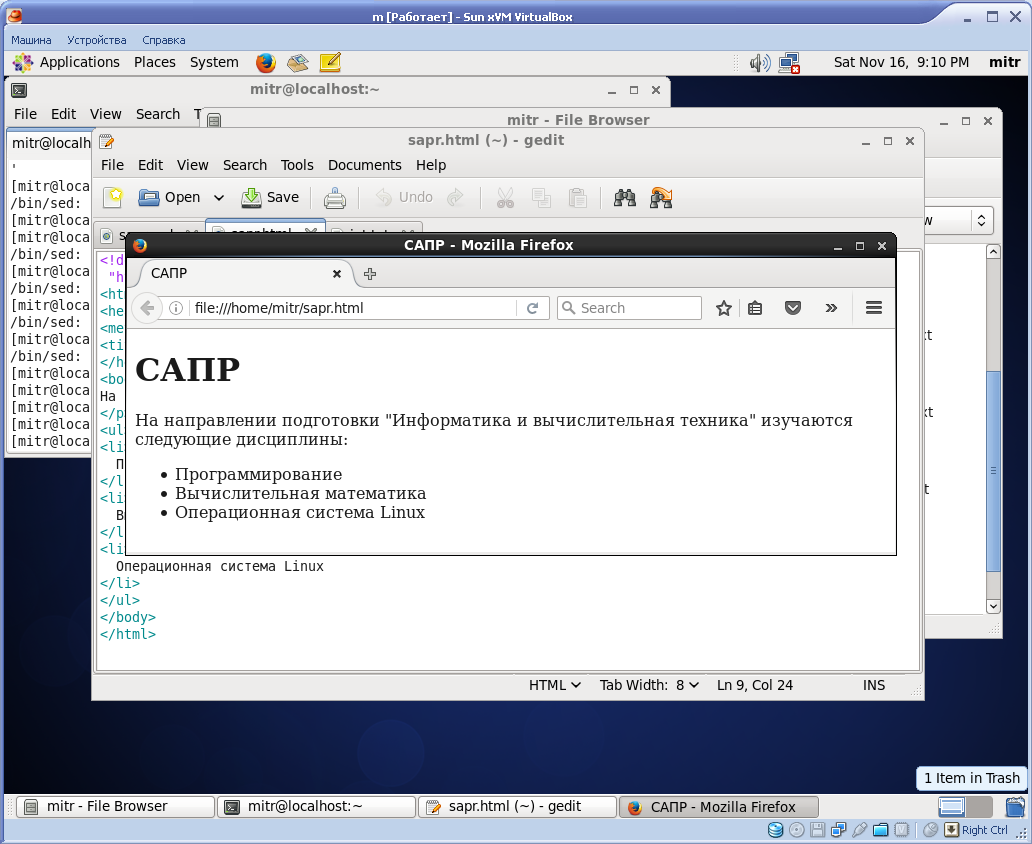


Рисунок 4. Файл sapr.html в браузере

**Утилита tr**

Утилита tr выполняет символьное преобразование путем подстановки или удаления символов из стандартного входного потока. Как правило, утилите tr передаются две строки: первая строка содержит искомые символы, а вторая – символы, на которые их требуется заменить. При выполнении команды устанавливается соответствие между символами обеих строк, а затем выполняется преобразование.

Формат утилиты tr с наиболее часто применяемыми параметрами:

**tr –c –d –s [“строка\_1”] [“строка\_2”] входной\_файл**

где:

- c – задает замену символов, указанных в «строка\_1», их собственным дополнением при условии, что значения этих символов находятся в диапазоне значений кодов ASCII;

- d – задает удаление во входном файле всех символов, указанных в «строка\_1»;

- s – задает удаление последовательности повторяющихся символов всех символов, кроме первого, благодаря чему удаляются повторяющиеся символы.

Параметр входной\_файл определяет имя файла, содержимое которого необходимо преобразовать.

При использовании tr можно указать диапазоны или списки символов в виде шаблонов, которые образованы строками. Эти шаблоны похожи на регулярные выражения, однако таковыми не являются. При указании содержимого строк строка\_1 и строка\_2 используются только диапазоны и последовательности символов, либо отдельные символы:

- [a-z] –строка символов, находящихся в диапазоне a-z;

- [A-Z] –строка символов, находящихся в диапазоне A-Z;

- [0-9] – строка чисел;

- /octal – восьмеричное число, состоящее из трех восьмеричных цифр;

- [o\*n] – означает символ «o», встречающийся столько раз, сколько указывает значение n.

В большинстве вариантов утилиты tr поддерживаются классы символов и сокращенная запись управляющих символов. Часто используются следующие классы:

[:alnum:] – буквенно-цифровые символы;

[:alpha:] – буквы;

[:blank:] – пропуски;

[:upper:] – прописные буквы;

[:lower:] – строчные буквы;

[:cntrl:] – управляющие символы;

[:space:] – пробелы;

[:digit:] – цифры;

[:graph:] – графические символы.

В таблице 4 представлен способ сокращения некоторых наиболее распространенных управляющих символов, используемый вместо их восьмеричного представления.

Табл. 4. Способы указания управляющих символов в утилите tr

|  |  |  |
| --- | --- | --- |
| **Сокращение** | **Значение** | **Восьмеричное значение** |
| \a | Control-G – звуковой сигнал | /007 |
| \b | Control-H – клавиша возврата на одну позицию | /010 |
| \f | Control-L – прокрутка страницы | /014 |
| \n | Control-J – новая строка | /012 |
| \r | Control-M – возврат каретки | /015 |
| \t | Control-I – табуляция | /011 |
| \v | Control-X | /030 |

На рис. 5 утилита tr использована для замены повторяющихся «1» на «2». Вначале утилита удаляет повторяющиеся «1», а затем заменяет оставшиеся на «2». Использована опция -s.

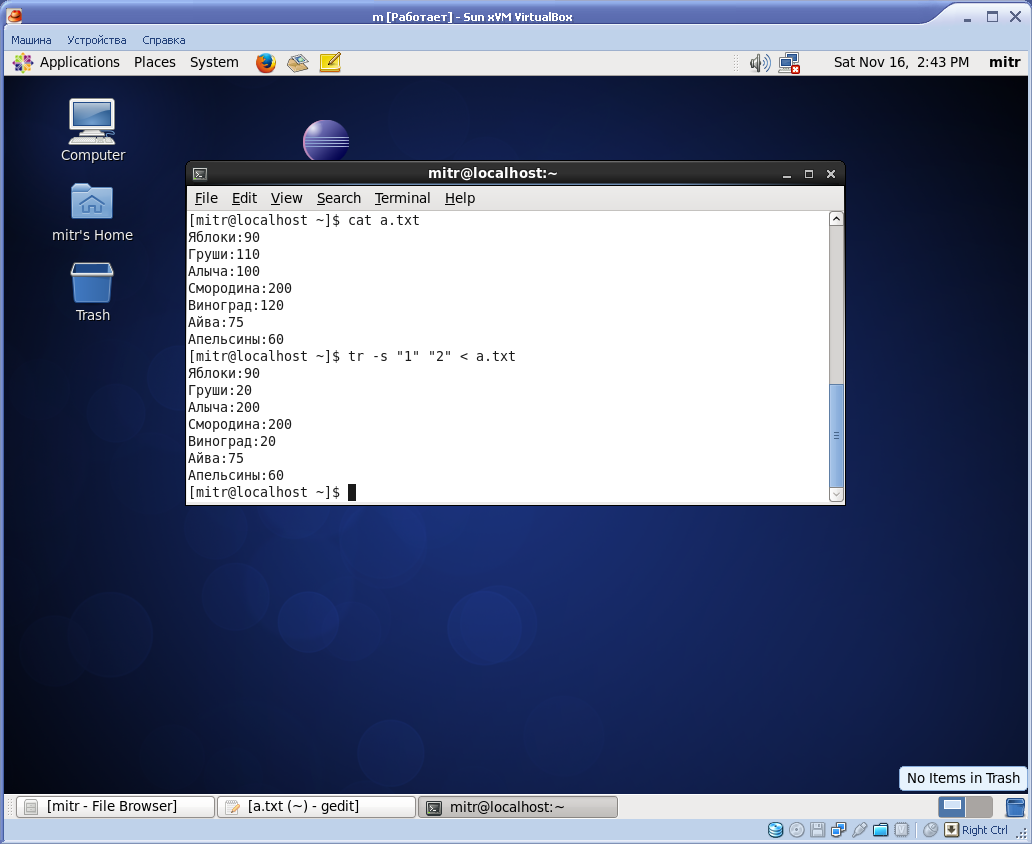


Рисунок 5. Использование tr для замены

На рис. 6 с помощью утилиты tr удаляются цифры. Использована опция –d.

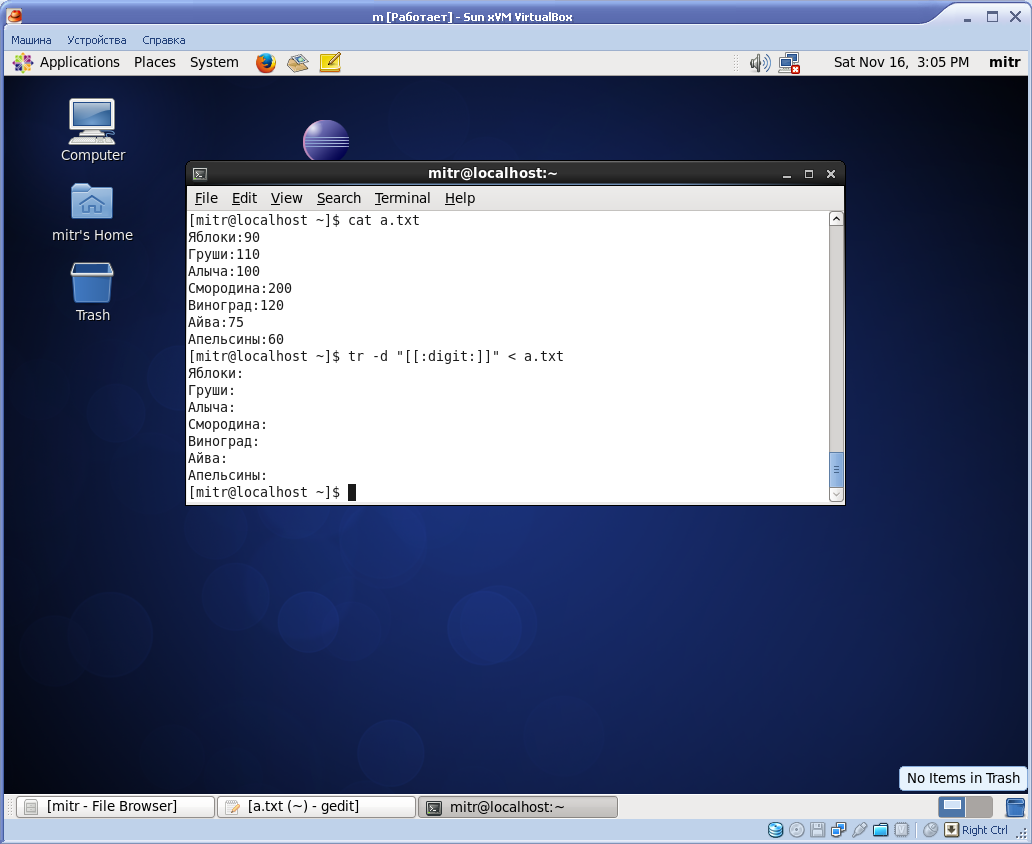


Рисунок 6. Удаление всех цифр

На рис. 7 показан процесс замены символа табуляции на символ «s». Сначала приводится вывод содержимого файла st.txt с помощью команды cat. Затем команда cat вызывается с опцией –T, что позволяет при выводе заменить символы табуляции комбинацией символов ^I, то есть увидеть, что символы табуляции в файле есть и где они расположены. В конце с помощью tr символы табуляции заменяются на «s».

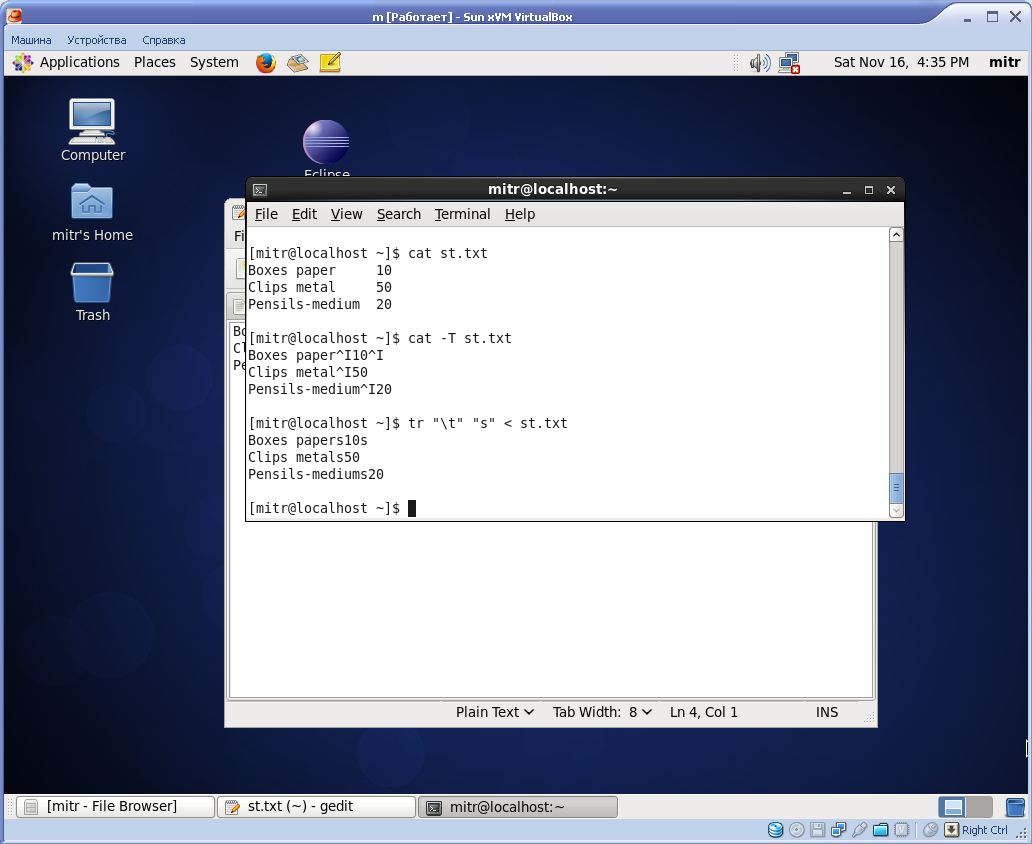


Рисунок 7. Замена символа табуляции на символ «s»

Сначала приводится вывод содержимого файла st.txt с помощью команды cat. Затем команда cat вызывается с опцией –T, что позволяет при выводе заменить символы табуляции комбинацией символов ^I, то есть увидеть, что символы табуляции в файле есть и где они расположены. В конце с помощью tr символы табуляции заменяются на «s».

***Порядок выполнения работы***

1) Изучите теоретический материал.

2) Выполните практическое задание.

3) Ответьте на контрольные вопросы.

***Практическое задание***

1) Напишите сценарий sed, который из файла со следующим содержимым:

Фрукты:

- яблоки;

- груши.

Овощи:

- огурцы;

- помидоры.

Делает файл с таким содержимым:

Магазин «Теремок» предлагает всегда свежие

!! Яблоки

!! Груши

!! Огурцы

!! Помидоры

2) В полученном файле с помощью утилиты tr замените «!!» на «?».

3) Продемонстрируйте выполненную работу преподавателю.

***Контрольные вопросы***

1) Почему sed называется «потоковым» редактором.

2) Какие способы адресации строк существуют в sed.

3) Какова общая схема работы sed?

4) Назовите команды sed и объясните их назначение.

5) Объясните работу сценария sapr.sed.

6) Почему в сценариях sed строки часто завершаются символом «\»? В каких случаях он не нужен?

7) Для чего предназначена утилита tr?

8) Назовите опции утилиты tr и поясните их назначение.

9) Что такое «классы символов»? Назовите некоторые из них.

10) Каким образом могут быть определены управляющие символы в утилите tr?

***Библиографический список***

1. Пик Д., О’Райли Т., Лукидис М. UNIX: инструментальные средства. – К.: Издательская группа BHV, 2002.

2. Тейнсли Д. Linux и UNIX: программирование в shell. Руководство разработчика. – К.: Издательская группа BHV,2001.