Министерство науки и высшего образования Российской Федерации

Федерального государственного бюджетного образовательного учреждения высшего образования

"Сочинский государственный университет"

|  |  |
| --- | --- |
|  | УТВЕРЖДАЮПроректор по учебной работеи информатизации\_\_\_\_\_\_\_\_\_\_\_\_ А.В.Ревнивых«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г. |

Регистрационный номер \_\_\_

Факультет «Инженерно-экологический»

Кафедра ИТ «Информационные технологии»

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

**по дисциплине**

**«Операционные системы»**

**Лабораторный практикум**

для направлений (уровень бакалавриата):

09.03.03 «Прикладная информатика», профиль «Прикладная информатика в экономике»

**Автор:**

Драч В.Е.,

доцент к.т.н., доцент кафедры ИТ

vladimir@drach.pro

г. Сочи, 2022 г.

Автор:

Драч Е.В. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Рецензент:

Руководитель направления по работе с ключевыми заказчиками

Siemens Industry Software

Product Lifecycle Management

Н.Г. Тутуев

Утверждена на заседании кафедры «Информационные технологии»

Протокол №6 от «\_\_22\_\_» \_\_02\_\_ 2022 г.

Заведующий кафедрой

А.С.Копырин \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Лабораторная работа №3.
Утилиты sed и awk на примере Alma Linux.**

Сначала рассмотрим синтаксис команды sed:

$ sed опции -e команды файл

А вот её основные опции:

*-n, --quiet - не выводить содержимое буфера шаблона в конце каждой итерации;*

*Вывод из файла 1-5 страницы:*

*-e - команды, которые надо выполнить для редактирования;*

*-f - прочитать команды редактирования из файла;*

*-i - сделать резервную копию файла перед редактированием;*

*-l - указать свою длину строки;*

*-r - включить поддержку расширенного синтаксиса регулярных выражений;*

*-s - если передано несколько файлов, рассматривать их как отдельные потоки, а не как один длинный.*

У утилиты есть два буфера, это активный буфер шаблона и дополнительный буфер. Оба изначально пусты. Программа выполняет заданные условия для каждой строки в переданном ей файле.

sed читает одну строку, удаляет из неё все завершающие символы и символы новой строки и помещает её в буфер шаблона. Затем выполняются переданные в параметрах команды, с каждой командой может быть связан адрес, это своего рода условие и команда выполняется только если подходит условие.

Когда всё команды будут выполнены и не указана опция -n, содержимое буфера шаблона выводится в стандартный поток вывода перед этим добавляется обратно символ перевода строки. если он был удален. Затем запускается новая итерация цикла для следующей строки.

**АДРЕСА SED**

Каждой команде можно передать адрес, который будет указывать на строки, для которых она будет выполнена:

*номер - позволяет указать номер строки, в которой надо выполнять команду;*

*первая~шаг - команда будет выполняется для указанной в первой части сроки, а затем для всех с указанным шагом;*

*$ - последняя строка в файле;*

*/регулярное\_выражение/ - любая строка, которая подходит по регулярному выражению. Модификатор l указывает, что регулярное выражение должно быть не чувствительным к регистру;*

*номер, номер - начиная от строки из первой части и заканчивая строкой из второй части;*

*номер, /регулярное\_выражение/ - начиная от сроки из первой части и до сроки, которая будет соответствовать регулярному выражению;*

*номер, +количество - начиная от номера строки указанного в первой части и еще плюс количество строк после него;*

*номер, ~число - начиная от строки номер и до строки номер которой будет кратный числу.*

**СИНТАКСИС РЕГУЛЯРНЫХ ВЫРАЖЕНИЙ**

Вы можете использовать такие же регулярные выражения, как и для Bash и популярных языков программирования. Вот основные операторы, которые поддерживают регулярные выражения sed Linux:

*\* - любой символ, любое количество;*

*\+ - как звездочка, только один символ или больше;*

*\? - нет или один символ;*

*\{i\} - любой символ в количестве i;*

*\{i,j\} - любой символ в количестве от i до j;*

*\{i,\} - любой символ в количестве от i и больше.*

**КОМАНДЫ SED**

Чтобы пользоваться sed, рассмотрим простейшие команды, которые применяются чаще всего:

*# - комментарий, не выполняется;*

*q - завершает работу сценария;*

*d - удаляет буфер шаблона и запускает следующую итерацию цикла;*

*p - вывести содержимое буфера шаблона;*

*n - вывести содержимое буфера шаблона и прочитать в него следующую строку;*

*s/что\_заменять/на\_что\_заменять/опции - замена символов, поддерживаются регулярные выражения; Замена root на OC\_1*

*y/символы/символы - позволяет заменить символы из первой части на соответствующие символы из второй части;*

*w - записать содержимое буфера шаблона в файл;*

*N - добавить перевод строки к буферу шаблона;*

*D - если буфер шаблона не содержит новую строку, удалить его содержимое и начать новую итерацию цикла, иначе удалить содержимое буфера до символа перевода строки и начать новую итерацию цикла с тем, что останется;*

*g - заменить содержимое буфера шаблона, содержимым дополнительного буфера;*

*G - добавить новую строку к содержимому буфера шаблона, затем добавить туда же содержимое дополнительного буфера.*

**Примеры модификации текстовых файлов**

Найти все трехзначные числа и заменить их строкой number:

sed -i 's/\b[0-9]\{3\}\b/number/g' file.txt

Добавить фигурные скобки {} вокруг каждого трехзначного числа, введите:

sed -i 's/\b[0-9]\{3\}\b/{&}/g' file.txt

И последнее, но не менее важное: всегда полезно сделать резервную копию при редактировании файла с помощью sed. Для этого просто используем опцию -i.

**СИНТАКСИС КОМАНДЫ AWK**

Сначала надо понять, как работает утилита. Она читает документ по одной строке за раз, выполняет указанные вами и отправляет результат на стандартный вывод. Одна из самых частых задач, для которых используется awk - это выборка одной из колонок. Все параметры awk находятся в кавычках, а действие, которое надо выполнить - в фигурных скобках. Основной её синтаксис:

$ awk опции 'условие {действие}'

$ awk опции 'условие {действие} условие {действие}'

С помощью действия можно выполнять преобразования с обрабатываемой строкой. Об этом мы поговорим позже, а сейчас давайте рассмотрим опции утилиты:

*-F, --field-separator - разделитель полей, используется для разбиения текста на колонки;*

*-f, --file - прочитать данные не из стандартного вывода, а из файла;*

*-v, --assign - присвоить значение переменной, например foo=bar;*

*-b, --characters-as-bytes - считать все символы однобайтовыми;*

*-d, --dump-variables - вывести значения всех переменных awk по умолчанию;*

*-D, --debug - режим отладки, позволяет вводить команды интерактивно с клавиатуры;*

*-e, --source - выполнить указанный код на языке awk;*

*-o, --pretty-print - вывести результат работы программы в файл;*

*-V, --version - вывести версию утилиты.*

Это далеко не все опции awk. Теперь перечислим несколько функций-действий, которые вы можете использовать:

*print(строка) - вывод чего либо в стандартный поток вывода;*

*printf(строка) - форматированный вывод в стандартный поток вывода;*

*system(команда) - выполняет команду в системе;*

*length(строка) - возвращает длину строки;*

*substr(строка, старт, количество) - обрезает строку и возвращает результат;*

*tolower(строка) - переводит строку в нижний регистр;*

*toupper(строка) - переводить строку в верхний регистр.*

Функций намного больше, но чтобы не загромождать статью я привел только те, которые мы будем использовать сегодня, а также ещё несколько для чтобы вы могли оценить масштаб возможностей утилиты.

В функциях-действиях можно использовать различные переменные и операторы, вот несколько из них:

*FNR - номер обрабатываемой строки в файле;*

*FS - разделитель полей;*

*NF - количество колонок в данной строке;*

*NR - общее количество строк в обрабатываемом тексте;*

*RS - разделитель строк, по умолчанию символ новой строки;*

*$ - ссылка на колонку по номеру.*

Задания:

Создайте текстовый файл, скопировав конфигурационный cp /etc/ssh/**sshd\_config** /home. Приветствуется использование собственного текстового файла. Выполните над ним действия.

**Для чётных вариантов**

1. Используя командную строку удалите все строки комментариев, то есть начинающиеся с символа #.

2\*. Выведите содержимое буфера шаблона файла root.

3. Написать 5 слов (можно из примеров), вывести чётные слова одним print с помощью awk.

4. Перевести слова в нижний регистр и вывести в терминал.

5. Сложить два числа с помощью переменной.

6. Замените все двузначные числа на ХХ.

**Для нечётных вариантов**

1. Используя командную строку удалите все строки, содержащие цифры.

2\*. Выведите содержимое буфера шаблона файла root.

3. Написать 5 слов (можно из примеров), вывести нечётные слова одним print с помощью awk.

4. Перевести слова в верхний регистр и вывести в терминал.

5. Вычесть два числа с помощью переменной.

6. Оберните все двузначные числа фигурными скобками.