Операционные системы (6 семестр)

Лекция 2.4. Потоковые редакторы.

Ассистент, к.т.н. Митричев Иван Игоревич

Москва 2018

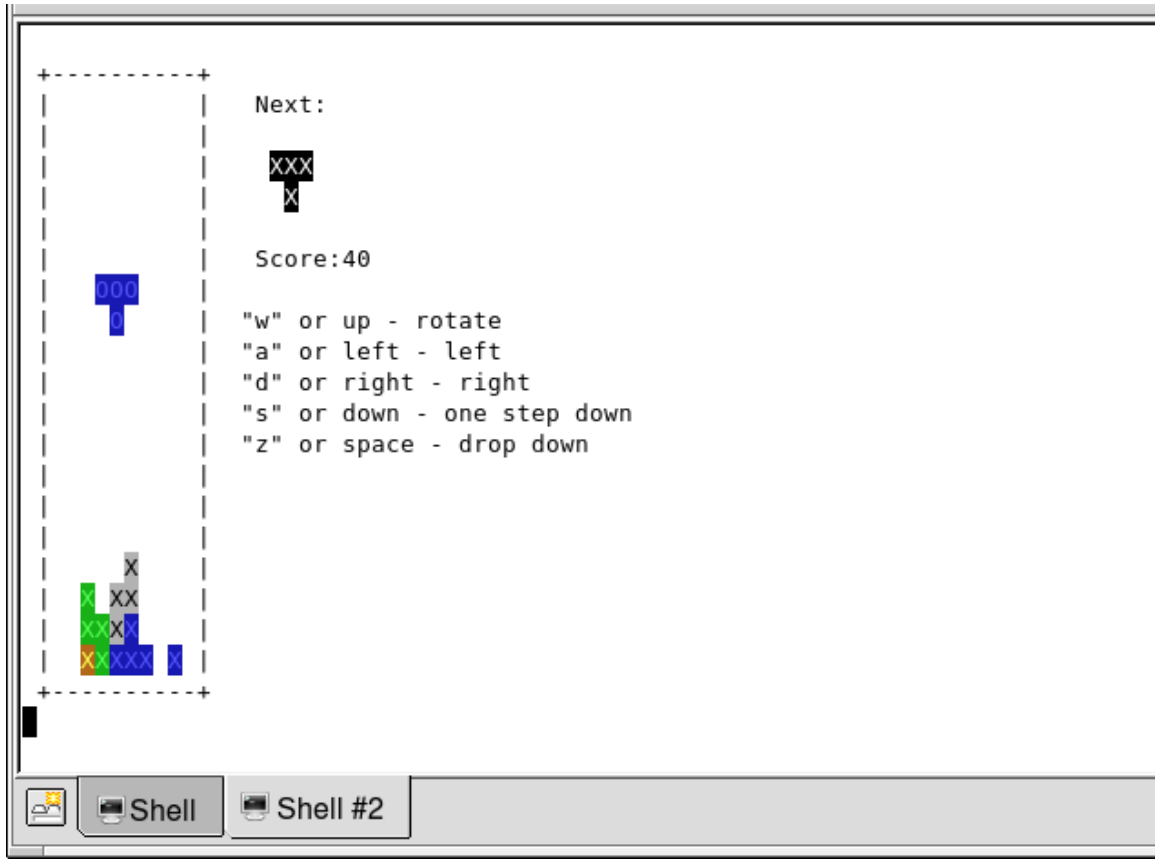
План лекции

- Потоковый редактор Sed: буферы, замена, обратные ссылки, модификаторы, опции, удаление, печать, инвертирование выбора, запись, чтение.
- Потоковый редактор awk: шаблон, команды, встроенные переменные.
- Вычисления на awk.
- Написание скриптов awk.
- Предопределенные функции awk.
- Передача переменных из командной оболочки в awk и обратно.

Потоковый редактор Sed

Sed — многофункциональный редактор потока данных.

Применяется в случаях:
-редактирования больших текстов;
-редактирования файлов любой величины;
-редактирования данных по мере поступления (в режиме реального времени).



Тетрис на sed
https://github.com/uuner/sedtris

Буферы sed

Область редактирования (pattern space)

В этот буфер помещается очередная строка для редактирования

Область удержания (hold space)

В этом буфере можно запомнить промежуточный результат

Замена текста

Часто используемые команды языка sed:

d – удаление;

p – вывод строки;

s – замена подстроки на заданную;

i – вставка строки;

a – добавление строки;

c – изменение всей строки на заданную;

q – выход без дальнейшей обработки строк;

= – команда печати номера строки.

sed s/образец/замена/ ← синтаксис

можно использовать другие разделители:

echo день | sed s_день_ночь_

Использование условных знаков \(, \), \1 для запоминания части регулярного выражения (механизм обратной ссылки)

\1 — первая запомненная часть

\2 — вторая запомненная часть

...

\9 — девятая запомненная часть

Удаление повторов — использование обратной ссылки внутри замены

echo первый первый второй | sed 's_\([ая]*\) \1_\1_'

Пример:

echo "abccbd" | sed "s/a\([bc]*\)d/\1/g"

- заменяются все вхождения буквы «a», за которой следуют b либо c, а далее произвольное количество букв, завершавшиеся буквой «d» на захваченную группу символов с номером 1 (\1). Для захвата группы символов используются круглые скобки, скобки экранированы обратным слэшем. Чтобы не экранировать скобки \(\) добавьте опцию -r (регулярные выражения).

Модификаторы команды замены текста

Модификатор /g команды замены позволяет заменить все вхождения (глобальная замена):
echo ‘цветы ландыша' sed 's/[а-я]*/(&)/g'
Результат:
(цветы) (ландыша)

echo ‘цветы ландыша' sed 's/[а-я]*/(&)/'
Результат:
(цветы) ландыша

Модификатор /i команды замены позволяет игнорировать регистр букв:
echo 'Цветы ландыша' sed 's/[а-я]*/(&)/gi'
Результат:
(Цветы) (ландыша)

Для захвата всего шаблона используют «&»:

echo 'цветы ландыша' sed 's/[а-я]*/(&)/g'

Результат:
(цветы) (ландыша)

Опция **-e** позволит выполнить несколько операций над одним текстом:
echo 'Цветы ландыша' sed -e 's/[а-я]*/(&)/g' -e 's/ы/ок/'
Результат:
(Цветок) (ландыша)

Удаление текста

sed '2,4d' - удалить вторую и четвертую строки текста.

Модификатор «!» инвертирует выбор:
sed '2,4 !d' - удалить все строки, кроме второй и четвертой.

Для использования переменной оболочки в sed экранируйте команды двойными кавычками, а не одинарными:

pattern=TO_DELETE
sed "/^$pattern/d" file.txt

Предварительная фильтрация текста по шаблону

sed '2,$ s/кот/бык/i' file.txt - редактировать от второй строки до конца файл file.txt
sed '/Привет/,/Пока/ s/кот/бык/i' - редактировать от слова «Привет» до слова «Пока» файл file.txt.
sed '5,/^$/ d' - удалить текст от пятой до первой пустой строки.

Другие команды sed

Вставка перед второй строкой

```
who | sed '2i новая строка'
```

Результат:
root tty1 Mar 13 17:23
новая строка
user tty7 Mar 13 17:04
новая строка

Вывод номеров строк, содержащих «Привет»

```
sed -n '/Привет/=' file.txt
```

Результат:
1
Опция **–n** подавляет вывод строк, не совпавших с шаблоном

Вставка после второй строки

```
who | sed '2a\
новая строка'
```

Результат:
root tty1 Mar 13 17:23
user tty7 Mar 13 17:04

Прекращение работы

```
sed '/Конец/q' file.txt
```

Результат:
Привет
Конец

file.txt

Привет
Конец
Файла
тут

awk

awk — командный интерпретатор, главным образом, предназначенный для обработки **структурированных записей**, содержащих текст.

Преимущества и недостатки awk

+ прост в использовании, удобно использовать в командной строке для обработки текста/данных, имеющих структуру (разделенных на записи и поля);
- исполнение программ интерпретируемого языка (awk) медленнее, чем компилируемого (C);
- не является языком глобального назначения, не предназначен для интенсивных вычислений , т. е., язык - специализированный;
+/- все переменные находятся в глобальной области видимости за исключением параметров, переданных в функцию.

Шаблоны вызова awk

Команды | awk '{команды}' – в цепочке конвейера

awk '{команды}' file.txt — для обработки file.txt

awk '{команды}' < file.txt — аналог предыдущей команды

awk -f file.awk file.txt — выполнить awk-скрипт из файла file.awk для обработки file.txt

./file.awk - выполнить исполняемый файл, содержащий awk-скрипт (файл file.awk)

Встроенные переменные awk

FS - разделитель полей ввода, значение по умолчанию - пробел. Вы можете изменить это, используя опцию командной строки -F.

NF - количество полей в текущей записи.

NR - номер текущей записи. **Запомните: awk работает не со строками, а с записями, разделенными RS.**

FNR - похож на NR, но в текущем файле.

OFMT - шаблон формата вывода (для форматирования чисел), а его значение по умолчанию -% .6g.

OFS - разделитель полей в выводе, значение по умолчанию - пробел.

ORS - разделитель записей в выводе, значение по умолчанию - новая строка.

RS - разделитель записей, значение по умолчанию - новая строка.

$0 – текущая запись (вся строка).

$n – поле с номером n в текущей записи, поля разделяются с помощью FS.

awk -F':' 'BEGIN{OFS="=";} {print $3,$4;}' /etc/passwd

- выводит поля 3 и 4 с разделителем «=», использовав файл **/etc/passwd**, хранящего информацию о пользователях и аутентификации, где разделителем является знак «:»

Результат:

0=0
1=1
2=2
...

print – самая часто используемая встроенная команда awk, выводит строки.

Вопрос: что выведет на экран команда **print $NF**?

Специальные блоки в awk

Общая структура скрипта:

awk 'BEGIN {команды, выполняемые перед чтением первой записи}; {команды, выполняемые для каждой записи}; END {команды, выполняемые после чтения последней записи}'

Пример:

```
$ awk '
> BEGIN {print "Ищем студента…"}
> /Petrov/ {++n}
> END {print "\"Petrov\" появляется в ",n, " записях." }' my_students.txt
```

Результат:

"Petrov" появляется в 3 записях.

Циклы и ветвление в awk

if (условие) {операторы} [else {операторы}]

while (условие) {операторы}

for (выражение; условие; выражение) {операторы}

for (индекс in имя_массива) {операторы}

Операторы:
exit — завершить исполнение скрипта;
next — перейти к обработке следующей записи;
break — завершение цикла;
continue — переход к следующей итерации цикла.

Пример: вывод полей записи через одно

awk '{ for(k=1; k<=NF; k+=2); {print $k}}'

Пример: отбор записей, где первое поле больше 10:

awk '{ if ($1>10) {print}}'

Массивы в awk

Массивы используют строковые индексы

echo "" | awk '{a["k"]=10; print a["k"]}'

 # выводит 10

Многомерный массив:

#объявление массива размерностью 4x4

```
for (i=1; i<5; i++)
 for (j=1; j<5; j++)
       vec[i,j]=i+j;
```

Следующая функция транспонирует двумерный массив:

```
{
    if (max_nf < NF)
        max_nf = NF
    max_nr = NR
    for (x = 1; x <= NF; x++)
        vector[x, NR] = $x
}
END {
    for (x = 1; x <= max_nf; x++) {
        for (y = 1; y <= max_nr; y++)
            printf("%s ", vector[x, y])
        printf("\n")
    }
}
```

Фильтрация в awk

Фильтрация по выражению
~ означает «содержит»

$ awk '$3~0 {print}‘ file.txt
1 2 0 5
6 6 0 7

$ awk '$3!~0 {print}' file.txt
Петров Иван Иванович
1 2 1 5

Регулярные выражение внутри шаблонов
- **/^a/** - поле начинается с **a**
- **/a$/** - поле кончается **a**
- **[abc]** - любой из символов **a**, **b** и **c**
- **[a-p]** - любой символ диапазона
- ***** - 0 или больше вхождений регулярного выражения
- **+** - 1 или больше вхождений регулярного выражения
- **?** - 0 или 1 вхождение регулярного выражения
- **ab|cd** - **ab** или **cd**

- **Чтобы искать текст, а не использовать регулярное выражение, применяйте экранирование:**
- \+ - экранирует +

Фильтрация по шаблону в awk

One-liner для выбора всех строк (записей), содержащих «honey»
awk '/honey/ {print}'

Встроенные функции awk

sin (expr)	синус expr
cos (expr)	косинус expr
exp (expr)	возведение в степень expr
log (expr)	натуральный логорифм expr
sqrt (expr)	извлечение корня expr
int (expr)	целая часть числа
length (s)	длина строки s
printf (fmt, ...)	форматированный вывод по спецификации fmt (аналог функции языка C).
substr (s, m, n)	подстрока в n символов строки s, начинающаяся с m.
getline ()	чтение следующей строки.
index (s1, s2)	номер позиции, с которой s1 совпадает с s2, иначе 0.
split (s, M, c)	строка s разбивается элементы массива M по разделителю c (по умолчанию FS=" "); функция возвращает число полей.

count = split("my sweet home", my_arr, / /);

В переменной count сохранится число элементов (count = 3), в массив my_arr будет записано три элемента (последний - «home»).

Системные вызовы в awk

Возможно использование системных вызовов для вызова сторонних утилит из awk.

Данный скрипт выводит количеством символов в первом и третьем столбце введенного текста с учетом пробела между ними:

echo "22 23 24" | awk '{system("echo -n "$1" "$3"| wc -m")}'

Результат:

5

Примеры регулярных выражений awk

Данная команда осуществляет фильтрацию строк, являющихся действительными e-mail адресами

```
awk '/^([a-zA-Z0-9_\-\.\+]+)@([a-zA-Z0-9_\-\.]+)\.([a-zA-Z]{2,5})$/{print $0}'
```

a-zA-Z0-9_\-\.\+ - допустимые символы. Они перечисляются внутри квадратных скобок (любой из…),
[….]+ - означает 1 или более вхождений допустимого символа,
e-mail должен содержать три части, разделенные «@» и «.».

Данная команда осуществляет печать строк между строками, содержащими «PT1» и «PT2», включая эти строки:

```
awk '/PT1/{f=1} f; /PT2/{f=0}' file
```

Вне зависимости от совпадения с шаблонами, основное тело данной программы содержит просто указание «f». Если значение этой переменной будет отлично от нуля, то вызовется действие по умолчанию – «print $0» - печать строки. Это обеспечит печать нужных строк.