Sed命令学习笔记

1. Sed简介

sed 是一种在线编辑器，它一次处理一行内容。处理时，把当前处理的行存储在临时缓冲区中，称为“模式空间”（pattern space），接着用 sed 命令处理缓冲区中的内容，处理完成后，把缓冲区的内容送往屏幕。接着处理下一行，这样不断重复，直到文件末尾。文件内容并没有改变，除非你使用重定向存储输出。Sed 主要用来自动编辑一个或多个文件；简化对文件的反复操作；编写转换程序等。以下介绍的是 Gnu 版本的 Sed 3.02。

2.定址

可以通过定址来定位你所希望编辑的行，该地址用数字构成，用逗号分隔的两个行数表示以这两行为起止的行的范围（包括行数表示的那两行）。如 1，3 表示 1，2，3 行，美元符号($)表示最后一行。范围可以通过数据，正则表达式或者二者结合的方式确定 。

3. Sed命令

调用 sed 命令有两种形式：

```
sed [options] 'command' file(s)
sed [options] -f scriptfile file(s)

a          在当前行后面加入一行文本。
b lable    分支到脚本中带有标记的地方,如果分支不存在则分支到脚本的末尾。
c          用新的文本改变本行的文本。
d          从模板块（Pattern space）位置删除行。
D          删除模板块的第一行。
i          在当前行上面插入文本。
h          拷贝模板块的内容到内存中的缓冲区。
H          追加模板块的内容到内存中的缓冲区
g          获得内存缓冲区的内容，并替代当前模板块中的文本。
G          获得内存缓冲区的内容，并追加到当前模板块文本的后面。
l          列表不能打印字符的清单。
n          读取下一个输入行，用下一个命令处理新的行而不是用第一个命令。
N          追加下一个输入行到模板块后面并在二者间嵌入一个新行，改变当前
行号码。
p          打印模板块的行。
```

```
P（大写）  打印模板块的第一行。
q          退出 Sed。
r file     从 file 中读行。
t label    if 分支，从最后一行开始，条件一旦满足或者 T，t 命令，将导致分支到带有标号的命令处，或者到脚本的末尾。
T label    错误分支，从最后一行开始，一旦发生错误或者 T，t 命令，将导致分支到带有标号的命令处，或者到脚本的末尾。
w file     写并追加模板块到 file 末尾。
W file     写并追加模板块的第一行到 file 末尾。
!          表示后面的命令对所有没有被选定的行发生作用。
s/re/string   用 string 替换正则表达式 re。
=           打印当前行号码。
#          把注释扩展到下一个换行符以前。
```

以下的是替换标记

```
g          表示行内全面替换。
p          表示打印行。
w          表示把行写入一个文件。
x          表示互换模板块中的文本和缓冲区中的文本。
y          表示把一个字符翻译为另外的字符（但是不用于正则表达式）
```

4.选项

```
-e command, --expression=command
允许多台编辑。
-h, --help
打印帮助，并显示 bug 列表的地址。
-n, --quiet, --silent
取消默认输出。
-f, --filer=script-file
引导 sed 脚本文件名。
-V, --version
打印版本和版权信息。
```

5.元字符集

```
^
锚定行的开始 如：/^sed/匹配所有以 sed 开头的行。
$
锚定行的结束 如：/sed$/匹配所有以 sed 结尾的行。
.
匹配一个非换行符的字符 如：/s.d/匹配 s 后接一个任意字符，然后是 d。
*
```

匹配零或多个字符 如：/*sed/匹配所有模板是一个或多个空格后紧跟 sed 的行。
[]
匹配一个指定范围内的字符，如/[Ss]ed/匹配 sed 和 Sed。
[^]
匹配一个不在指定范围内的字符，如：/[^A-RT-Z]ed/匹配不包含 A-R 和 T-Z 的一个字母开头，紧跟 ed 的行。
(..)
保存匹配的字符，如 s/(love)able/1rs，loveable 被替换成 lovers。
&
保存搜索字符用来替换其他字符，如 s/love/**&**/，love 这成**love**。
<
锚定单词的开始，如:/<love/匹配包含以 love 开头的单词的行。
>
锚定单词的结束，如/love>/匹配包含以 love 结尾的单词的行。
x{m}
重复字符 x，m 次，如：/0{5}/匹配包含 5 个 o 的行。
x{m,}
重复字符 x,至少 m 次，如：/o{5,}/匹配至少有 5 个 o 的行。
x{m,n}
重复字符 x，至少 m 次，不多于 n 次，如：/o{5,10}/匹配 5--10 个 o 的行。

6. 实例

1 删除：d命令

```
$ sed '2d' example-----删除 example 文件的第二行。
$ sed '2,$d' example-----删除 example 文件的第二行到末尾所有行。
$ sed '$d' example-----删除 example 文件的最后一行。
$ sed '/test/'d example-----删除 example 文件所有包含 test 的行。
```

2 替换：s命令

```
$ sed 's/test/mytest/g' example-----在整行范围内把 test 替换为 mytest。如果没有 g 标记，则只有每行第一个匹配的 test 被替换成 mytest。
$ sed -n 's/^test/mytest/p' example-----(-n)选项和 p 标志一起使用表示只打印那些发生替换的行。也就是说，如果某一行开头的 test 被替换成 mytest，就打印它。
$ sed 's/^192.168.0.1/&localhost/' example-----&符号表示替换换字符串中被找到的部份。所有以 192.168.0.1 开头的行都会被替换成它自己加 localhost，变成 192.168.0.1localhost。
$ sed -n 's/(love)able/1rs/p' example-----love 被标记为 1，所有 loveable 会被替换成 lovers，而且替换的行会被打印出来。
$ sed 's#10#100#g' example-----不论什么字符，紧跟着 s 命令的都被认为是新的分隔符，所以，“#”在这里是分隔符，代替了默认的“/”分隔符。表示把
```

所有 10 替换成 100。

3 选定行的范围：逗号

```
$ sed -n '/test/,/check/p' example-----
```
所有在模板 test 和 check 所确定的范围内的行都被打印。
```
$ sed -n '5,/^test/p' example-----
```
打印从第五行开始到第一个包含以 test 开始的行之间的所有行。
```
$ sed '/test/,/check/s/$/sed test/' example-----
```
对于模板 test 和 west 之间的行，每行的末尾用字符串 sed test 替换。

4 多点编辑：e命令

```
$ sed -e '1,5d' -e 's/test/check/' example-----
```
(-e)选项允许在同一行里执行多条命令。如例子所示，第一条命令删除 1 至 5 行，第二条命令用 check 替换 test。命令的执行顺序对结果有影响。如果两个命令都是替换命令，那么第一个替换命令将影响第二个替换命令的结果。
```
$ sed --expression='s/test/check/' --expression='/love/d' example-----
```
一个比-e 更好的命令是--expression。它能给 sed 表达式赋值。

5 从文件读入：r命令

```
$ sed '/test/r file' example-----
```
file 里的内容被读进来，显示在与 test 匹配的行后面，如果匹配多行，则 file 的内容将显示在所有匹配行的下面。
写入文件：w 命令
```
$ sed -n '/test/w file' example-----
```
在 example 中所有包含 test 的行都被写入 file 里。

6 追加命令：a命令

```
$ sed '/^test/a\--->this is a example' example<-----'this is a example'
```
被追加到以 test 开头的行后面，sed 要求命令 a 后面有一个反斜杠。

7 插入：i命令

```
$ sed '/test/i\
new line
------------------------' example
```
如果 test 被匹配，则把反斜杠后面的文本插入到匹配行的前面。

8 下一行：n命令

```
$ sed '/test/{ n; s/aa/bb/; }' example-----
```
如果 test 被匹配，则移动到匹配行的下一行，替换这一行的 aa，变为 bb，并打印该行，然后继续。

9 变形：y命令

```
$ sed '1,10y/abcde/ABCDE/' example-----
```
把 1--10 行内所有 abcde

转变为大写，注意，正则表达式元字符不能使用这个命令。

10 退出：q命令

`$ sed '10q' example-----`打印完第 `10` 行后，退出 `sed`。

11 保持和获取：h命令和G命令

`$ sed -e '/test/h' -e '$G example-----`在 `sed` 处理文件的时候，每一行都被保存在一个叫模式空间的临时缓冲区中，除非行被删除或者输出被取消，否则所有被处理的行都将打印在屏幕上。接着模式空间被清空，并存入新的一行等待处理。在这个例子里，匹配 `test` 的行被找到后，将存入模式空间，`h` 命令将其复制并存入一个称为保持缓存区的特殊缓冲区内。第二条语句的意思是，当到达最后一行后，`G` 命令取出保持缓冲区的行，然后把它放回模式空间中，且追加到现在已经存在于模式空间中的行的末尾。在这个例子中就是追加到最后一行。简单来说，任何包含 `test` 的行都被复制并追加到该文件的末尾。

保持和互换：`h` 命令和 `x` 命令

`$ sed -e '/test/h' -e '/check/x' example -----`互换模式空间和保持缓冲区的内容。也就是把包含 `test` 与 `check` 的行互换。

7. 脚本

Sed 脚本是一个 sed 的命令清单，启动 Sed 时以-f 选项引导脚本文件名。Sed 对于脚本中输入的命令非常挑剔，在命令的末尾不能有任何空白或文本，如果在一行中有多个命令，要用分号分隔。以#开头的行为注释行，且不能跨行。