5.24总结：shell工具

shell工具

grep 、sed、awk被称为linux中的"三剑客"
grep 更适合单纯的查找或匹配文本
sed 更适合编辑匹配到的文本
awk 更适合格式化文本，对文本进行较复杂格式处理

sort

sort命令可将文件进行排序，并将排序结果标准输出

参数说明
-n 依照数值的大小排序（升序）
-r 以相反的顺序来排序
-t 设置排序时所用的分隔字符
-k 指定需要排序的列

例如，文件夹sh下有如下sort.txt文件：
bb:40:5.4
bd:20:4.2
xz:50:2.3
cls:10:3.5
ss:30:1.6

按照":"分割后的第三列倒序排序：

```
sort -t ":" -nrk 3 /root/sh/sort.txt

#输出结果
bb:40:5.4
bd:20:4.2
cls:10:3.5
xz:50:2.3
ss:30:1.6
```

sed

stream editor（流编辑器）的简称
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（“模式空间“），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。

参数说明
p 打印，亦即将某个选择的数据印出。通常 p 会与参数 sed -n 一起运行
i 插入， i 的后面可以接字串，而这些字串会在新的一行出现(目前的上一行)
a 新增， a 的后面可以接字串，而这些字串会在新的一行出现(目前的下一行)
s 取代，可以直接进行取代的工作，通常这个 s 的动作可以搭配正则表示法

```
#1.显示文件的第 2 行的内容：
sed -n '2 p' /root/sh/sort.txt

#2.显示文件的第 2 行到第 4 行的内容：
sed -n '2,4 p' /root/sh/sort.txt

#3.将文件中的bb全部替换为BB
sed 's/bb/BB/g' /root/sh/sort.txt

#4.以文件bb开头的上一行添加
sed '/^bb/i hello' /root/sh/sort.txt

#5.将文件中的d全部删除
sed 's/d//g' /root/sh/sort.txt

# 可以使用管道符 | 连续处理，接着重定向保存
# 使用 | 连续处理时只需在第一个语句里指定文件路径，后续的处理不能写！
# 命令过长需要换行时可以使用\拼接
# 使用 | 换行时可不使用 \
sed 's/bd/BB/g' /root/sh/sort.txt \
| sed 's/ss/SS/g' | sed 's/cls/AA/g' \
| sort -t ":"  -nrk 3 | sed '/^bb/i one,two,three' \
| sed 's/:/./g' > /root/sh/sort.csv

sed 's/bd/BB/g' /root/sh/sort.txt |
sed 's/ss/SS/g' | sed 's/cls/AA/g' |
sort -t ":"  -nrk 3 | sed '/^bb/i one,two,three' |
sed 's/:/./g' > /root/sh/sort.csv
```

awk

一个完整的awk命令格式：

```
awk [options] 'BEGIN{command1} {command2} END{command3}' file
#[options]设置分隔字段
#BEGIN可设置列名
#command2设置打印的列
#END可显示行列总数量
#file表示文件路径
```

-v FS 指定以什么对字段进行分隔
-v OFS 指定输出的字段以什么分隔
不写导出后的分隔符默认以空格划分

内置参数
NF 分割完字段的数量
$1 代表文本行中的第 1 个数据字段
$2 代表文本行中的第 2 个数据字段
输出指定列：{print $1,$2}
分隔符相同的情况输出一整行：{print}

```
#1-3重要
#1.以:为分隔符，打印第2列和第1列
awk -v FS=":" '{print $2,$1}' /root/shell/sort.txt

#2.以:为分隔符，打印第2列和第1列，列之间用,分割
awk -v FS=":" -v OFS="," '{print $2,$1}' /root/shell/sort.txt

#3.添加列保存为csv，下载，使用excel查看
awk -v FS=":" -v OFS="," 'BEGIN{print "one,two,three"}{print $2,$1,$3}'
/root/shell/sort.txt > /root/shell/sort.csv

#4-5扩展
#4.第二列大于30
awk -v FS=":" '{ if($2>30){print $2}}' /root/shell/sort.txt
#5.行列总数量
awk -v FS=":" 'BEGIN{n=0}{for(i=1;i<=NF;i++){n++} }END{print n}'
/root/shell/sort.txt
```

作业练习

```
#1.复制网卡文件/etc/sysconfig/network-scripts/ifcfg-ens33 到家目录，并且改名为
wangka.txt
cp /etc/sysconfig/network-scripts/ifcfg-ens33 $HOME/wangka.txt

#2.找到含有 IP 的行输出
```

```
cat wangka.txt | grep "IPADDR"

#3.显示文件的第 3 行到第 5 行的内容
sed -n '3,5 p' $HOME/wangka.txt

#4.将文件中的255全部替换为250
sed 's/255/250/g' $HOME/wangka.txt

#5.以文件IPADDR开头的上一行添加hello
sed '/^IPADDR/i hello' $HOME/wangka.txt

#6.找到所有DNS，并且删掉
sed 's/DNS//g' $HOME/wangka.txt

#7.使用管道符连续处理4. 5. 6. 题，并且重定向结果保存为wangka.csv
sed 's/255/250/g' $HOME/wangka.txt |
sed '/^IPADDR/i hello' |
sed 's/DNS//g' > $HOME/wangka.csv

#8. 根据wangka.csv文件自己灵活处理重定向为ip.txt,内容如下：
# 192.168.145.151
# 250.250.250.0
# 192.168.145.2
# 8.8.8.8
# 114.114.114.114
sed -n '20,24 p' $HOME/wangka.csv |
awk -v FS="\"" '{print $2}' > $HOME/ip.txt

#9.ip.txt中以.分割，按照第一列进行降序排序
sort -t "." -nrk 1 $HOME/ip.txt

#10.ip.txt中以.为分隔符，打印第1列和第2列
awk -v FS="." '{print $1,$2}' $HOME/ip.txt

#11.ip.txt中以.为分隔符，打印第3列和第4列，列之间用,分割
awk -v FS="." -v OFS="," '{print $1,$2}' $HOME/ip.txt

#12.ip.txt中以.为分隔符，列之间用,分割，且加一行，保存为ip.csv,格式如下：
# one,two,three,four
# 192,168,145,151
# 250,250,250,0
# 192,168,145,2
# 8,8,8,8
# 114,114,114,114
awk -v FS="." -v OFS="," 'BEGIN{print "one,two,three,four"}{print
$1,$2,$3,$4}' $HOME/ip.txt > $HOME/ip.csv
```

```
#13.在家目录下创建一个names.txt的文件，写入班级的所有同学的姓名，每个一行，每次随机产生一个姓名
n=$[RANDOM%13+1]
sed -n "$n p" $HOME/names.txt

#14.批量修改家目录下的文件扩展名，使用位置参数传递两种扩展名，例如txt文件为csv 文件。（注：碰到特殊符号使用\进行转义）
cd
f=`ls *.$1`
for i in $f
    do
        new=`echo "$i" | sed "s/\.$1$/.$2/"`
        mv $i $new
    done
```