**shell day 06**

================================================

**sed p d s a i c -n -r -i**

**awk -F print $1 $2 NR NF**

**BEGIN{ } 执行1次**

**{ } 逐行任务,执行n次**

**END{ } 执行1次**

**awk中的条件**

**1,使用正则作为条件**

**/root/ /^root/ ~包含 !~不包含**

awk -F: '/bin/{print}' user //输出有bin的行,无论bin在第几列

awk -F: '$6~/bin/{print}' user //输出第6列包含bin的行

awk -F: '$6!~/bin/{print}' user //输出第6列不包含bin的行

**2,使用数字和字符串**

**== != > >= < <=**

awk -F: 'NR==3{print}' /etc/passwd //输出第3行

awk -F: '$3<10{print}' /etc/passwd //输出id号是0~9的行

awk -F: '$3>=1000{print}' /etc/passwd //输出普通用户的行

awk -F: '$7=="/bin/bash"{print}' /etc/passwd //输出解释器是/bin/bash的

行,注意如果对比内容不是变量,要用双引号(数字不用)

awk -F: '$7!="/bin/bash"{print}' /etc/passwd //上述情况的取反搜索

awk -F: 'NR<11{print $1}' /etc/passwd //输出第1~10行的第1列

**3, 逻辑组合 &&并且 ||或者**

awk -F: '$3<10&&$7~/bash/' /etc/passwd //找id号小于10并且

解释器包含bash的行

awk -F: '$3>=10&&$3<=20' /etc/passwd //找id号范围是10~20的行

awk -F: 'NR>=2&&NR<=10' /etc/passwd //找2~10行

awk -F: '$3<5||$3>1000' /etc/passwd //找id号是0~4或者1001以上的行

awk 'NR<5||NR>10{print NR}' /etc/passwd //找1~4行或者11以上的行输出

行号

awk -F: 'NR>5||NR<100' /etc/passwd //找所有行

awk -F: 'NR<5&&NR>100' /etc/passwd //逻辑错误，没有输出

awk -F: '$1=="root"||$1=="bin"' /etc/passwd //找第1列是root或者bin的

**4,使用运算作为条件**

awk 'BEGIN{print 1+1}'

awk 'BEGIN{print 2-1}'

awk 'BEGIN{print 2\*2}'

awk 'BEGIN{print 2/2}'

awk 'BEGIN{print 2%2}'

seq 200 | awk '$1%7==0' //找到能被7整除的行

如何实现能被7整除或包含7的行?

seq 200 | awk '$1~/7/||$1%7==0'

awk中使用if

单分支，如果满足条件，就执行指令，不满足就不执行任何指令

if(条件){指令}

awk '{if( ){ }}' 使用时可以先写框架

awk -F: '{if($7~/bash/){x++}}END{print x}' /etc/passwd //找到

第7列包含bash的行,每找到一个就把x加1,都找完之后,使用

END任务输出最终x的值，也就是找系统中使用bash作为解释器

的用户数量

双分支，如果满足条件就执行指令，不满足就执行else后面的指令

awk '{if(条件){指令}else{指令}}'

统计/etc/passwd文件中UID小于1000、UID大于等于1000的用户个数

awk -F: '{if($3<1000){x++}else{y++}}END{print x,y}' /etc/passwd

awk -F: '{if($7~/bash/){x++}else{y++}}END{print x,y}' /etc/passwd //统

计系统中使用bash作为解释器的用户，和没有使用bash的用户数量, 判断

如果每找到一行的$7包含bash，就把x+1，否则y+1，最后使用end输出

x与y的值

awk -F: '{if($7~/bash/){x++}else{y++}}END{print "使用bash作为解释

器的用户数量是"x,"没使用bash的用户数量是"y}' /etc/passwd

多分支

awk '{if(条件){指令}else if(条件){指令}else{指令}}'

满足第一个条件就执行第一个条件后面的指令,不满足的话看第二个条件

都不满足就执行最后else后面的指令

awk -F: '{if( ){ }else if( ){ }else{ }}END{ }'

awk -F: '{if($7=="/bin/bash"){x++}else if($7=="/sbin/nologin"){y++}

else{z++}}END{print x,y,z}' /etc/passwd //找使用bash或者nologin

作为解释器的账户数量,分别用变量x与y显示,其他解释器用z显示

使用awk统计网站的访问量

yum -y remove httpd //如果httpd有问题可以先删除

yum -y install httpd //安装httpd

systemctl restart httpd //开启服务

netstat -ntulp | grep :80 //查询当前80端口是否被httpd占用

echo "web test ~~~~" > /var/www/html/index.html //定义默认页

systemctl stop firewalld

curl 192.168.4.7 //使用包括自身在内的linux系统访问该网站

tail -1 /var/log/httpd/access\_log //查看最后一条httpd访问日志

awk '{print $1}' /var/log/httpd/access\_log //找访问网站的客户的

ip信息

------------------------------

数组,相当于可以存多个值的特殊变量

格式: 数组名[下标]=该数组下标对应的值

awk 'BEGIN{a[1]=10;a[1]++;print a[1]}' //基本效果,创建数组,然后

做运算,然后输出

awk 'BEGIN{a[1]=10;a[2]=20;a[1]++;a[2]++;print a[2]}'

awk 'BEGIN{a["abc"]="abcabc";print a["abc"]}' //数组还可以使用非数字

for(变量名称 in 数组名称){指令}

awk 'BEGIN{a[1]=10;a[2]=20;a[3]=30;print a[1],a[2],a[3]}' //创建数组a,并

添加多个下标与对应的值,然后逐个输出

awk 'BEGIN{a[1]=10;a[2]=20;a[3]=30;for(i in a){print i}}' //创建数组a,并使

用for循环显示数组的所有下标

awk 'BEGIN{a[1]=10;a[2]=20;a[3]=30;for(i in a){print a[i]}}' //创建数组a,并

使用for循环显示数组的所有值

a[abc]++ a[abc]=1

a[xyz]++ a[xyz]=1

a[opq]++ a[opq]=1

a[abc]++ a[abc]=2

a[abc]++ a[abc]=3

a[xyz]++ a[xyz]=2

下课休息 16:15回