一、awk 基础语法

P1 基础知识

作用：awk 在 shell 脚本中主要是做数据分析和数据过滤

使用方法

命令 | awk [选项] ‘[条件]{指令}’ #[条件]{指令}多条指令可以以分号分隔

awk 选项 ‘[条件]{指令}’ 文本

awk 文本过滤的基本用法

格式：awk [选项] '[条件]{指令}' 文件

直接过滤文件内容：

```
[root@svr7 ~]# vim test.txt
hello the world
welcome to beijing
[root@svr7 ~]# awk '{print $1}' test.txt        #打印第 1 列
hello
welcome
[root@svr7 ~]# awk '{print $1,$3}' test.txt     #打印第 1 列和第 3 列
[root@svr7 ~]# awk '{print $3,$1}' test.txt     #打印第 3 列和第 1 列
```

语法格式： -F 指定分隔符，默认分隔符为（空格或者 tab 键）

```
[root@svr7 ~]# awk '{print $1}' /etc/passwd       #默认以空格或 tab 键打印
[root@svr7 ~]# awk -F: '{print $1}' /etc/passwd   #以：为分隔打印
```

awk 常用内置变量：

FS 保存或设置字段分隔符，例如 FS=':'，与-F 功能一样

$0 文本当前行的全部内容

$1 文本的第 1 列

$2 文件的第 2 列

$3 文件的第 3 列，依此类推

NR 文件当前行的行数

NF 文件当前行的列数（有几列）

```
[root@svr7 ~]# awk -F: '{print NF}' /etc/passwd          #以冒号为分隔，每行有几列
[root@svr7 ~]# awk -F: '{print $NF}' /etc/passwd         #以冒号为分隔，每行有几列，$NF 为这列的内容是什么
[root@svr7 ~]# awk -F: '{print $(NF-1)}' /etc/passwd     #打印倒数第 2 列内容
[root@svr7 ~]# awk '{print NR}' /etc/passwd              #打印行数
```

awk 不仅可以打印变量，还可以打印常量

```
[root@svr7 ~]# awk -F: '{print "用户名是："$1,"UID 是："$3}' /etc/passwd
```

awk 过滤的时机

awk 会逐行处理文本，支持在处理第一行之前做一些准备工作，以及在处理完最后一行之后做一些总结性质的工作。在命令格式上分别体现如下：

awk [选项] '[条件]{指令}' 文件
awk [选项] ' BEGIN{指令} {指令} END{指令}' 文件
BEGIN{ } 行前处理，读取文件内容前执行，指令执行 1 次
{ } 逐行处理，读取文件过程中执行，指令执行 n 次
END{ } 行后处理，读取文件结束后执行，指令执行 1 次

awk 做计算器
[root@svr7 ~]# awk 'BEGIN{a=34;print a+12}'
[root@svr7 ~]# awk 'BEGIN{a=34;b=33;print a+b}'
[root@svr7 ~]# awk 'BEGIN{a=3.4;b=3.3;print a+b}'
awk 统计/ect/passwd 里面有多少个可以登录的用户
[root@svr7 ~]# awk 'BEGIN{x=0} /bash$/{x++} END{print x}' /etc/passwd
输出打印之前的行数，以及读取文件之后的行数
[root@svr7 ~]# awk -F: 'BEGIN {print NR} END{print NR}' /etc/passwd

二、awk 基础应用案例

P1 监控操作系统信息

过滤内存信息（查看内存的可用剩余空间）
[root@svr7 ~]# free
[root@svr7 ~]# free | awk '{print $7}'　　#会只有内存那一列显示出来，因为第 1 行和第 7 行没有第 7 列，所以为空
[root@svr7 ~]# free | awk '{print $NF}'　　#打印最后一列
[root@svr7 ~]# free | awk '/Mem/{print $NF}'　　#值打印内存剩余容量

过滤磁盘信息（查看根分区可用的空间）
[root@svr7 ~]# df -h | grep "/$" | awk '{print $4}'

过滤 cpu 的信息（过滤 CPU 的型号和显示 CPU 核数）
[root@svr7 ~]# LANG=C lscpu　　#LANG=C 英文的方式显示
CPU(s):1
...
Model name:Intel(R) Core(TM) i3-4170 CPU @ 3.70GHz
...
[root@svr7 ~]# LANG=C lscpu | grep "Model name"| awk -F: '{print $2}'
[root@svr7 ~]# LANG=C lscpu | grep "^CPU(s)" | awk -F: '{print $2}'

显示 15 分钟的负载
[root@svr7 ~]# uptime
[root@svr7 ~]# uptime | awk '{print $NF}'

过滤网卡的信息
[root@svr7 ~]# ifconfig eth0

```
[root@svr7 ~]# ifconfig eth0 | grep "RX p" | awk '{print "进站流量为："$5"字节"}'
[root@svr7 ~]# ifconfig eth0 | grep "TX p" | awk '{print "出站流量为："$5"字节"}'
```

监控暴力破解的 IP 地址

```
[root@svr7 ~]# ssh 192.168.4.7          #自己远程自己，故意输错密码
[root@svr7 ~]# grep "Failed" /var/log/secure
[root@svr7 ~]# grep "Failed" /var/log/secure | awk '{print $11}'
```

三、awk 条件判断

P1 条件判断概述

条件表达式

- 正则表达式
- 数值/字符比较
- 逻辑比较

P2 条件

使用正则设置条件

/正则表达式/　　~匹配　　!~不匹配

```
[root@svr7 ~]# awk -F: '/root/{print}' /etc/passwd          #打印包含 root 的行
[root@svr7 ~]# awk -F: '$1~/root/{print}' /etc/passwd       #打印第一列是 root 的行
[root@svr7 ~]# awk -F: '$7!~/bash$/{print}' /etc/passwd     #打印第 7 列不是 bash 结尾的行
```

使用数值/字符串比较设置条件

比较符号：==(等于)　!=（不等于）　>（大于）　>=（大于等于）　<（小于）　<=（小于等于）

```
[root@svr7 ~]# awk -F: 'NR==2' /etc/passwd                  #打印第二行
[root@svr7 ~]# awk -F: 'NR==2{print}' /etc/passwd           #打印第二行
```

字符串的比较

```
[root@svr7 ~]# awk -F: '$1=="root"{print}' /etc/passwd      #打印第 1 列等于 root 的行
[root@svr7 ~]# awk -F: '$1=="root"{print $1,$3}' /etc/passwd  #打印第 1 列等于 root 的字符串，显示第 1 列和第 3 列
[root@svr7 ~]#    awk -F: '$3>=1000{print $1,$3}' /etc/passwd  #打印用户 UID 大于 1000 的用户名称和 UID 信息
```

逻辑比较　　&&逻辑与：期望多个条件成立　　||逻辑或：只要一个条件成立既满足要求

```
[root@svr7 ~]# awk -F: '$3>=0 && $3<2{print $1,$3}' /etc/passwd   #打印用户 UID 大于等于 0 并且小于 2 的用户信息
[root@svr7 ~]# awk -F: '$3==1 || $3==7{print $1,$3}' /etc/passwd  #输出用户 UID 等于 1 或用户 UID 等于 7 的用户信息
```

运算符

```
[root@svr7 ~]# awk 'NR%2==0' /etc/passwd       #打印偶数行，即 NR 行数对 2 取余得 0
[root@svr7 ~]# awk 'NR%2==1{print}' /etc/passwd    #打印奇数行，即 NR 行数对 2 取余得 1
```

[root@svr7 ~]# seq 200　　#产生 1-200 之间的整数

判断这些数（1-200）能被 3 和 13 整除的数有多少个

[root@svr7 ~]# seq 200 | awk 'BEGIN{i=0} $1%3==0 && $1%13==0{i++} END{print i}'　　#满足条件执行 i++，不满足执行下一个，最后打印 i