## 1. awk和sed的区别

**sed的核心是正则，对于sed， 基本的两个概念是匹配和行为。**

**1. 匹配是通过区域选择加上正则表达式实现，比如“3到6行以This开头的”**

**2. 行为是增删改查。可以在某个位置新增或删除一行，可以通过正则表达式进行变量替换，可以显示满足某些条件的行。配合shell的批处理就会很强大， 比如我想把所有文件的开头添加一个注释，比如我想把所有文件的某一个变量进行替换，比如我想把所有文件满足某些条件的行进行合并和拆分。**

**3. 同时，sed提供了一个保持空间（hold space）可以实现逆序输出等操作。**

**4. 另外，sed的用法非常灵活，你可以将匹配和行为进行不同的嵌套，举个复杂的例子“将第某行到某行之间的满足A条件的里面满足B条件的行进行C操作和D操作并且将E条件的行进行F操作”这样的灵活组合方式怕是也只有sed了**

**awk，经常用于格式化输出，也就是将数据按照我们想要的方式来显示，并且可以做一些基本的统计工作。**

**1. 它的运作模式是“预处理+逐行处理+最终处理”。一般我们只用“逐行处理”比如对于满足条件的某些行，我们打印某某列。通过指定分隔符，我们很容易的对列进行操作。**

**2. 预处理来定义变量，逐行处理来修改变量，最终处理来打印变量。**

————————————————

## awk基本用法

### 选项

**-F -F ','或者 -F '正则表达式' -F选项来改变字段分隔符**

**-v -va=1赋值一个用户定义变量a的值为1**

**-f -f scripfile，从脚本文件中读取awk命令**

### linux中grep命令的用法

**常用选项：**

**-E ：开启扩展（Extend）的正则表达式。**

**-i ：忽略大小写（ignore case）。**

**-v ：反过来（invert），只打印没有匹配的，而匹配的反而不打印。**

**-n ：显示行号**

**-w ：被匹配的文本只能是单词，而不能是单词中的某一部分，如文本中有liker，而我搜寻的只是like，就可以使用-w选项来避免匹配liker**

**-c ：显示总共有多少行被匹配到了，而不是显示被匹配到的内容，注意如果同时使用-cv选项是显示有多少行没有被匹配到。**

**-o ：只显示被模式匹配到的字符串。**

**--color :将匹配到的内容以颜色高亮显示。**

**-A n：显示匹配到的字符串所在的行及其后n行，after**

**-B n：显示匹配到的字符串所在的行及其前n行，before**

**-C n：显示匹配到的字符串所在的行及其前后各n行，context**

2、使用基本正则表达式，下面谈关于基本正则表达式的使用：

　　　　匹配字符：

　　　　　　. ：任意一个字符。

　　　　　　[abc] ：表示匹配一个字符，这个字符必须是abc中的一个。

　　　　　　[a-zA-Z] ：表示匹配一个字符，这个字符必须是a-z或A-Z这52个字母中的一个。

　　　　　　[^123] ：匹配一个字符，这个字符是除了1、2、3以外的所有字符。

　　　　　　对于一些常用的字符集，系统做了定义：

　　　　　　[A-Za-z] 等价于 [[:alpha:]]

　　　　　　[0-9] 等价于 [[:digit:]]

　　　　　　[A-Za-z0-9] 等价于 [[:alnum:]]

　　　　　　tab,space 等空白字符 [[:space:]]

　　　　　　[A-Z] 等价于 [[:upper:]]

　　　　　　[a-z] 等价于 [[:lower:]]

　　　　　　标点符号 [[:punct:]]

位置锚定：

　　　　　　^ ：锚定行首

　　　　　　$ ：锚定行尾。技巧："^$"用于匹配空白行。

　　　　　　\b或\<：锚定单词的词首。如"\blike"不会匹配alike，但是会匹配liker

　　　　　　\b或\>：锚定单词的词尾。如"\blike\b"不会匹配alike和liker，只会匹配like

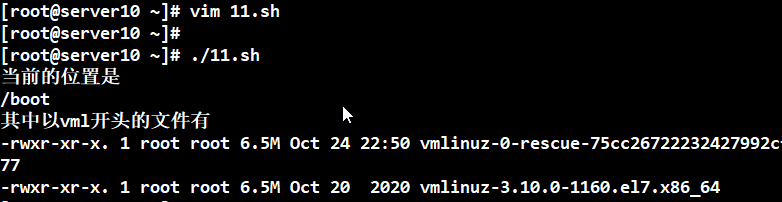
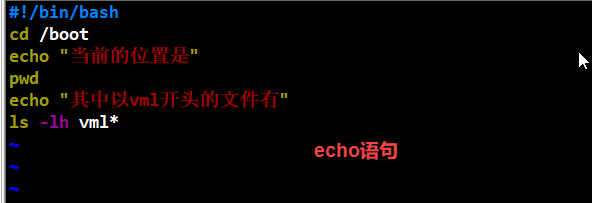
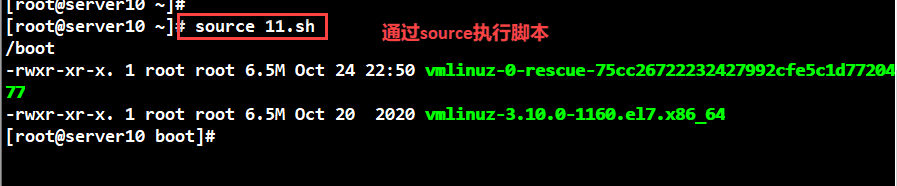
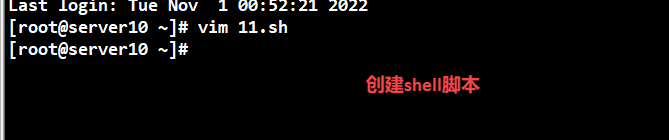
　　　　　　\B ：与\b作用相反。

&>/dev/null 的作用

就是让 目前的信息 不显示

把 stdout 送到 /dev/null 里面

那什麼是 /dev/null 呢，/dev/null 是 Unix/Linux 里的【无底洞】



**2.用shell脚本源码安装apache**

**#!/bin/bash**

**DATE1=`date +%s%N|cut -c1-13`**

**tar -zxf apr-1.4.6.tar.gz**

**cd apr-1.4.6**

**./configure --prefix=/usr/local/apr &>/dev/null**

**make &>/dev/null**

**make install &>/dev/null**

**cd**

**tar -zxf apr-util-1.4.1.tar.gz**

**cd apr-util-1.4.1**

**./configure --prefix=/usr/local/apr-util --with-apr=/usr/local/apr/bin/apr-1-config &>/dev/null**

**make &>/dev/null**

**make install &>/dev/null**

**cd**

**tar -zxf pcre-8.30.tar.gz**

**cd pcre-8.30**

**./configure --prefix=/usr/local/pcre &>/dev/null**

**make &>/dev/null**

**make install &>/dev/null**

**cd**

**yum install -y pcre-devel**

**yum install -y openssl-devel**

**tar zxf httpd-2.4.39.tar.gz**

**cd httpd-2.4.39**

**./configure --prefix=/usr/local/apache2 -enable-so -enable-rewrite --with-mpm=worker --with-apr=/usr/local/apr --with-apr-util=/usr/local/apr-util --with-pcre=/usr/local/pcre &>/dev/null**

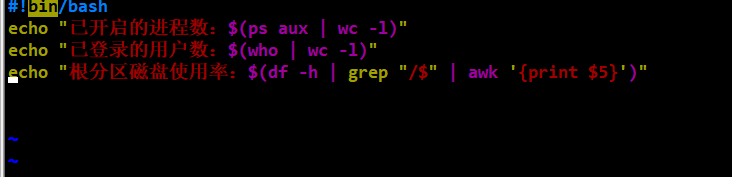
**make &>/dev/null**

**make install &>/dev/null**

**DATE2=`date +%s%N|cut -c1-13`**

**echo "Running time:: $((${DATE2}-${DATE1}))"**

**echo "Done!"**



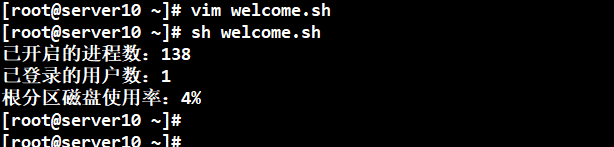
**a 显示现行终端机下的所有程序，包括其他用户的程序。**

**u 　 以用户为主的格式来显示程序状况。**

**x 　 显示所有程序，不以终端机来区分。**

**awk语言的最基本功能是在文件或者字符串中基于指定规则浏览和抽取信息，awk抽取信息后，才能进行其他文本操作。完整的awk脚本通常用来格式化文本文件中的信息。**

**通常，awk是以文件的一行为处理单位的。awk每接收文件的一行，然后执行相应的命令，来处理文本**



1.新建脚本文件welcome.sh用来输出各种监控信息





GREP -O选项

-o代表的是只输出匹配的选项。

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

FS 输入域分隔符，默认为一个空格

3、面试题 根据以下要求截取出字符串中的字符：字符串 var=http://www.aaa.com/root/123.htm

1.取出www.aaa.com/root/123.htm

2.取出123.htm

3.取出http://www.aaa.com/root

4.取出http:

5.取出http://

6.取出www.aaa.com/root/123.htm

7.取出123

8.取出123.htm

