## 第**1**章原题及讲解视频

**1.1** 原题地址**:**

[面试题]老男孩教育-三剑客笔试题集合

<https://www.jianshu.com/p/61d8883719ac>

**1.2** 讲解视频地址面试题讲解系列-三剑客面试题-三十道面试题让你玩转 sed 和 awk

<https://www.bilibili.com/video/av51101676>

## 第**2**章三剑客**-**基础部分题目

* 每道题目尽量多种方法.
* 涉及到三剑客过滤,替换,取列操作。

**2.1 Linux** 三剑客的题目如何举一反三？

* 熟悉三剑客基本操作:

grep sed awk

* sed 和 awk 的执行过程
* 只要遇到三剑客或 find 题目，多种方法。

**2.2** 选择

**2.2.1** 在给定文件中查找与条件相符字符串的命令及查找某个目录下相应文件的命令为：（多选）

**AC**

* + - * A:grep
      * B:gzip
      * C:find

**2.2.2 sort** 在给定文件中查找与设定条件相符字符串的命令为（**A**）

* + - * A：grep
      * B：gzip
      * C：find
      * D：sort #进行排序的命令

**2.2.3 \_\_\_D\_\_\_\_**命令可以从文本文件的每一行中截取指定内容的数据。

* + - * A：cp
      * B：dd
      * C：fmt **#**根据参数重新显示**/**规划每一行的内容比如每一行显示 **10** 个字符等等
      * D：cut

**2.2.4** 下面表述错误的是 **D**

* + - * ^表示开头，如^abc 表示以 abc 开头的
      * $表示结尾如 abc$表示以 abc 为结尾的
      * ?表示前一个字符出现 0 或 1 次，如 ab?c 表示 a 与 c 之间可有 0 或 1 个 b 存在
      * + 表示前一个字符出现 0 或多次。如 ab+c 表示 a 与 c 之间可有 0 或多个 b 存在

**2.3** 填空

**2.3.1 Unix** 中在当前目录下所有**.cc** 的文件中找到含有**“asiainfo”**内容的文件，命令为**\_**

find . -type f -name "\*.cc" -exec grep 'asiainfo' {} \; find . -type f -name "\*.cc"|xargs grep 'asiainfo' grep 'asiainfo' ` find . -type f -name "\*.cc"`

如果没有指定扩展名则直接使用

grep -rl 'oldboy' ./\*

* -r 递归查询
* -R 与-r 类似 只不过多了可以处理软连接
* -l （小写字母 L）只显示 grep 查找出来的文件名

**2.3.2** 使用 **AWK** 指定分隔符的参数是**\_\_\_\_-F** 或**-vFS=\_\_\_\_**

eg:

FS Field Separator 字段分隔符，每一列的结束标记.

awk -F: '{print $1,$3}' /etc/passwd awk -vFS=":" '{print $1,$3}' /etc/passwd awk 'BEGIN{FS=":"}{print $1,$3}' /etc/passwd

最后这种方法不推荐了，推荐前两种方法

**2.4** 简答

**2.4.1** 找出当前目录下包含 **127.0.0.1** 关键字的文件？

方法 1

grep -Rl '127.0.0.1' ./

方法 2 三剑客+反引号

grep "127.0.0.1" `find . -type f ` sed -n '/127.0.0.1/p' `find . -type f` awk '/127.0.0.1/' `find . -type f `

方法 3 find + |xargs +三剑客

find . -type f |xargs grep "127.0.0.1" find . -type f |xargs sed -n '/127.0.0.1/p' find . -type f |xargs awk '/127.0.0.1/'

方法 4 find -exec + 三剑客

find . -type f -exec grep "127.0.0.1" {} \; find . -type f -exec sed -n '/127.0.0.1/p' {} \; find . -type f -exec awk '/127.0.0.1/' {} \;

小结：

1.grep 过滤方法 -Rl

2.find 命令与三剑客过滤的用法

**2.4.2** 删除**/**显示 **1.txt** 文件第 **3** 行到第 **10** 行的内容？

2.4.2.1 删除

sed '3,10d' 1.txt awk 'NR<3||NR>10' 1.txt awk '!(NR>=3&&NR<=10)' 1.txt

说明:

* NR 在 awk 中内置变量表示 Number Of Field 记录号（行号）
* &&表示并且 同时成立
* || 表示或者
* ; 分隔多条命令，让你在同一行执行多条命令
* NR>=3 && NR<=10 把第 3 行到第 10 行取出来
* !表示取反
* ()被括起来的内容 整体

2.4.2.2 显示

sed -n '3,10p' 1.txt

awk 'NR==3,NR==7' 1.txt awk 'NR>=3 && NR<=10' 1.txt head -10 num.txt|tail -8

|  |
| --- |
| [root@oldboyedu-show01 ~]# head -10 num.txt|tail -8  3 4  5  6 7  8 9  10 |

2.4.2.3 玩赖的了解下:

|  |
| --- |
| [root@oldboyedu-show01 ~]# cat num.txt  1 2  3 4  5  6 7  8 9  10  [root@oldboyedu-show01 ~]# grep -n '.' num.txt|grep -A7 '^3:'  3:3  4:4 5:5  6:6 7:7  8:8 9:9  10:10 |

**2.4.3** 使用 **awk** 取出**/etc/passwd** 第一列数据也就是取出用户名。

2.4.3.1 基础必备:

awk -F":" '{print $1}' /etc/passwd cut -d: -f1 /etc/passwd sed 's#:.\*$##g' /etc/passwd egrep -o '^[0-Z\_-]+' /etc/passwd

说明:

* [0-Z] == [a-zA-Z0-9]
* cut -d 指定分隔符，-d1 次只能指定 1 个符号作为分隔符
* cut -f 取列, -f1 -f2 -f1,2,3 -f1-5

2.4.3.2 提高

egrep -o '^[^:]+' /etc/passwd sed -r 's#(^[^:]+).\*$#\1#g' /etc/passwd awk '{gsub(/:.\*$/,"");print}' /etc/passwd awk '{gsub(/:.\*$/,"")}1' /etc/passwd

说明：

* print 在 awk 中后面什么都不加的时候表示 print $0

参考视频讲解部分:

面试题讲解系列-三剑客面试题-三十道面试题让你玩转 sed 和 awk

<https://www.bilibili.com/video/av51101676>

**2.4.4** 用命令行更改 **config.txt** 文件，把里面所有的**“name”**更改为**“address”**

sed 's/name/address/g' config.txt awk '{gsub(/name/,"address")}1' config.txt

扩展: vi/vim 里面也可以进行替换

:%s#name#address#g

**2.4.5** 利用 **sed** 命令将 **test.txt** 中所有的回车替换成空格？

【吃饭的家伙】01-老男孩教育-三剑客-sed 执行流程

<https://www.jianshu.com/p/46c02da0598e>

2.4.5.1 方法 1 把每一行内容追加到 Hold Space 中，最后 1 行弄回到 Pattern space 中.然后进行替换基础版

[root@oldboyedu-show01 ~]# seq 10 |sed -n 'H;$x;s#\n# #gp'

1 2 3 4 5 6 7 8 9 10

[root@oldboyedu-show01 ~]# seq 10 |sed -n '1h;1!H;$x;s#\n# #gp'

1 2 3 4 5 6 7 8 9 10

[root@oldboyedu-show01 ~]# seq 10 |sed -n 'H;$x;s#\n# #gp'

1 2 3 4 5 6 7 8 9 10

[root@oldboyedu-show01 ~]# seq 10 |sed -n 'H;${x;s#\n# #gp}'

1 2 3 4 5 6 7 8 9 10

加强版**:**上面的方法每一行开头多个空格**.**

[root@oldboyedu-show01 ~]# seq 10 |sed -n '1h;1!H;$x;s#\n# #gp'

1 2 3 4 5 6 7 8 9 10

[root@oldboyedu-show01 ~]# seq 10 |sed -n '1h;1!H;${x;s#\n# #gp}'

1 2 3 4 5 6 7 8 9 10

备注:h,H,N 的含义参考:

【进阶】04-老男孩教育-sed 命令模式空间及保持空间命令

<https://www.jianshu.com/p/770b69cdbbb0>

2.4.5.2 方法 2 sed 命令的“循环(标签)”方法

[root@oldboyedu-show01 ~]# seq 10 |sed ':label ;N;s#\n# #g;t label'

1 2 3 4 5 6 7 8 9 10

2.4.5.3 方法 3 tr

如果不用 sed 直接用 tr 即可

[root@oldboyedu-show01 ~]# seq 10 |tr "\n" " "

1 2 3 4 5 6 7 8 9 10

2.4.5.4 方法 4 awk 修改 ORS

ORS 是 Output Record Separator 输出分隔符，用来指定 awk 在显示每一行的时候每一行的结尾是什么默认是回车这里改成了空格。正好符合题意，把回车替换成空格.

[root@oldboyedu-show01 ~]# seq 10 |awk -vORS=" " '1'

1 2 3 4 5 6 7 8 9 10 [root@oldboyedu-show01 ~]#

[root@oldboyedu-show01 ~]# seq 10 |awk -vORS=" " '{print $0}'

1 2 3 4 5 6 7 8 9 10 [root@oldboyedu-show01 ~]#

**2.4.6** 写出查询 **file.txt** 以 **abc** 结尾的行

grep 'abc$' file.txt sed -n '/abc$/p' file.txt awk '/abc$/' file.txt

说明： awk 可以过滤 时候精确到某一列.

* awk '$NF~/nologin$/' /etc/passwd
* awk '$3==0' /etc/passwd

补充说明:

$0 表示一整行

* awk '/abc$/' file.txt ❑ awk '$0~/abc$/' file.txt

**2.4.7** 删除 **file.txt** 文件中的空行模拟环境:

|  |
| --- |
| echo -e 'oldboy\noldboy\n\nalex\n \nalex\lidao lidao alex\n oldboy\n\noldboyedu group |
|  |
| 598972270' >file.txt |

2.4.7.1 必会：仅删除文件中的空行

grep -v '^$' file.txt sed '/^$/d' file.txt awk '!/^$/' file.txt

2.4.7.2 提升:文件中的空行可能是只有空格的行

egrep -v '^$|^ +$' file.txt grep -v '^ \*$' file.txt sed -n '/^ \*$/p' file.txt awk '/^ \*$/' file.txt

思考:如果还可能包含 tab 键说明：

* ^ \*$ 表示两种情况:
  + \*表示前 1 个字符出现 0 次的时候:^$
  + \*表示前 1 个字符出现 1 次及 1 次以上的时候:^ +$
    1. 打印 **file.txt** 文件中的第 **10** 行

awk 'NR==10' file.txt sed -n '10p' file.txt head -10 file.txt|tail -1

* + 1. 删除**/etc/fstab** 文件中所有以**#**开头的行的行首的**#** 号及**#**后面的所有空白字符；空白字符❑ 空格
* 多个空格
* tab 键

此题可以分为两个题目:

1. 删除/etc/fstab 文件中所有以#开头的行的行首的# 号及#后面的所有空白字符
2. 删除/etc/fstab 文件中所有以#开头的行的行首的# 号及#后面的连续的空白字符

2.4.9.1 删除/etc/fstab 文件中所有以#开头的行的行首的# 号及#后面的所有空白字符

可以分为两步:哪个是条件，哪个是动作

* 第 1 步找出以#开头的行
* 第 2 步在删除这一行中的空白字符和行首的井号
* awk '/^#/{gsub(/^#|[ \t]+/,"")}1' /etc/fstab
* sed -r '/^#/s@^#|[ \t]+@@g' /etc/fstab

演示:

[root@oldboyedu59 ~]# cat /etc/fstab

#

# /etc/fstab

# Created by anaconda on Tue Mar 26 11:37:39 2019

#

# Accessible filesystems, by reference, are maintained under '/dev/disk'

# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info #

UUID=1e827748-cc43-4b1d-8bbc-9a85c239782d / xfs defaults 0 0

UUID=29a4ef75-7502-48df-b31c-c7c10264a7c8 /boot xfs defaults 0 0 UUID=60c45f5e-c78d-41fe-aa92-93d10de6def3 swap swap defaults 0 0

[root@oldboyedu59 ~]# awk '/^#/{gsub(/^#|[ \t]+/,"")}1' /etc/fstab

/etc/fstab

CreatedbyanacondaonTueMar2611:37:392019

Accessiblefilesystems,byreference,aremaintainedunder'/dev/disk' Seemanpagesfstab(5),findfs(8),mount(8)and/orblkid(8)formoreinfo

UUID=1e827748-cc43-4b1d-8bbc-9a85c239782d / xfs defaults 0 0 UUID=29a4ef75-7502-48df-b31c-c7c10264a7c8 /boot xfs defaults 0 0 UUID=60c45f5e-c78d-41fe-aa92-93d10de6def3 swap swap defaults 0 0

[root@oldboyedu59 ~]# sed -r '/^#/s@^#|[ \t]+@@g' /etc/fstab

/etc/fstab

CreatedbyanacondaonTueMar2611:37:392019

Accessiblefilesystems,byreference,aremaintainedunder'/dev/disk' Seemanpagesfstab(5),findfs(8),mount(8)and/orblkid(8)formoreinfo

UUID=1e827748-cc43-4b1d-8bbc-9a85c239782d / xfs defaults 0 0

UUID=29a4ef75-7502-48df-b31c-c7c10264a7c8 /boot xfs defaults 0 0

UUID=60c45f5e-c78d-41fe-aa92-93d10de6def3 swap swap defaults 0 0

2.4.9.2 删除/etc/fstab 文件中所有以#开头的行的行首的# 号及#后面的连续的空白字符

|  |  |
| --- | --- |
| ❑ | awk '{gsub(/^#[ \t]+/,"")}1' /etc/fstab |
| ❑ | sed -r 's@^#[ \t]+@@g' /etc/fstab |

[root@oldboyedu59 ~]# cat /etc/fstab

#

# /etc/fstab

# Created by anaconda on Tue Mar 26 11:37:39 2019

#

# Accessible filesystems, by reference, are maintained under '/dev/disk'

# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info #

UUID=1e827748-cc43-4b1d-8bbc-9a85c239782d / xfs defaults 0 0

UUID=29a4ef75-7502-48df-b31c-c7c10264a7c8 /boot xfs defaults 0 0 UUID=60c45f5e-c78d-41fe-aa92-93d10de6def3 swap swap defaults 0 0

[root@oldboyedu59 ~]# awk '{gsub(/^#[ \t]+/,"")}1' /etc/fstab

#

/etc/fstab

Created by anaconda on Tue Mar 26 11:37:39 2019

#

Accessible filesystems, by reference, are maintained under '/dev/disk'

See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info #

UUID=1e827748-cc43-4b1d-8bbc-9a85c239782d / xfs defaults 0 0

UUID=29a4ef75-7502-48df-b31c-c7c10264a7c8 /boot xfs defaults 0 0 UUID=60c45f5e-c78d-41fe-aa92-93d10de6def3 swap swap defaults 0 0

[root@oldboyedu59 ~]# sed -r 's@^#[ \t]+@@g' /etc/fstab

#

/etc/fstab

Created by anaconda on Tue Mar 26 11:37:39 2019

#

Accessible filesystems, by reference, are maintained under '/dev/disk'

See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info #

UUID=1e827748-cc43-4b1d-8bbc-9a85c239782d / xfs defaults 0 0

UUID=29a4ef75-7502-48df-b31c-c7c10264a7c8 /boot xfs defaults 0 0 UUID=60c45f5e-c78d-41fe-aa92-93d10de6def3 swap swap defaults 0 0

**2.4.10** 查找最后创建时间是 **3** 天前，后缀是**\*.log** 的文件并删除

假设目录是/oldboy

find /oldboy -mtime +3 -type f -name "\*.log"

2.4.10.1 方法 1 find +xargs

find /oldboy -mtime +3 -type f -name '\*.log' | xargs rm -f

2.4.10.2 方法 2 find +$()或``

\rm -f `find /oldboy -mtime +3 -type f -name '\*.log'`

\rm -f $(find /oldboy -mtime +3 -type f -name '\*.log')

2.4.10.3 方法 3 find -exec

find /oldboy -mtime +3 -type f -name '\*.log' -exec rm -f {} \;

说明:

* \;形式 -exec find 命令找出 1 个文件就把文件放在{}的位置.

2.4.10.4 方法 4

find /oldboy -mtime +3 -type f -name '\*.log' -exec rm {} +

说明:

* 如果-exec 后面使用+find 会把前面找到的所有文件 1 次性的放在{}所在位置.

2.4.10.5 方法 5

find /oldboy -mtime +3 -type f -name "\*.log" -delete

-delete 删除 2.4.10.6 方法 6

find /oldboy -mtime +3 -type f -name '\*.log' | xargs -i rm -f {}

**2.4.11** 查找文件中 **1** 到 **20** 行之间，同时将**"aaa"**替换为**"AAA","ddd"**替换**"DDD"**

[root@oldboyedu59 ~]# cat /oldboy/lidao.txt a b c d e f g h i j k l m n o p q r s t u v w x y z aaadddaaaddd aaadddaaaddd

aaadddaaaddd

思路 1:简单的替换 aaa 替换为 AAA，ddd 替换为 DDD

* awk 'NR==1,NR==20{gsub(/aaa/,"AAA");gsub(/ddd/,"DDD")}1' lidao.txt
* sed '1,20{s#aaa#AAA#g;s#ddd#DDD#g}' lidao.txt

[root@oldboyedu-show01 ~]# awk 'NR==1,NR==20{gsub(/aaa/,"AAA");gsub(/ddd/,"DDD")}1' lidao.txt a b c d e f g h i j k l m n o p q r s t u v w x y z

AAADDDAAADDD

AAADDDAAADDD

AAADDDAAADDD

[root@oldboyedu-show01 ~]# sed '1,20{s#aaa#AAA#g;s#ddd#DDD#g}' lidao.txt a b c d e f g h i j k l m n o p q r s t u v w x y z

AAADDDAAADDD

AAADDDAAADDD

AAADDDAAADDD

思路 2:大小写转换，把小写字母转换为大写字母

* tr 'ad' 'AD' <lidao.txt
* sed 'y/ad/AD/' lidao.txt
* awk '{print toupper($0)}' lidao.txt
* sed 's#[ad]#\U&#g' lidao.txt

|  |
| --- |
| [root@oldboyedu-show01 ~]# tr 'ad' 'AD' <lidao.txt A b c D e f g h i j k l m n o p q r s t u v w x y z  AAADDDAAADDD AAADDDAAADDD  AAADDDAAADDD |

|  |
| --- |
| [root@oldboyedu-show01 ~]# sed 'y/ad/AD/' lidao.txt A b c D e f g h i j k l m n o p q r s t u v w x y z  AAADDDAAADDD  AAADDDAAADDD AAADDDAAADDD |

说明:

sed 命令中 y/// 类似于 tr1 对 1 替换，但是不能使用 a-z A-Z 范围

|  |
| --- |
| [root@oldboyedu-show01 ~]# awk '{print toupper($0)}' lidao.txt  A B C D E F G H I J K L M N O P Q R S T U V W X Y Z  AAADDDAAADDD AAADDDAAADDD  AAADDDAAADDD |

[root@oldboyedu-show01 ~]# sed 's#[ad]#<&>#g' lidao.txt

<a> b c <d> e f g h i j k l m n o p q r s t u v w x y z

<a><a><a><d><d><d><a><a><a><d><d><d> <a><a><a><d><d><d><a><a><a><d><d><d>

<a><a><a><d><d><d><a><a><a><d><d><d>

[root@oldboyedu-show01 ~]# sed 's#[ad]#\U&#g' lidao.txt

A b c D e f g h i j k l m n o p q r s t u v w x y z

AAADDDAAADDD

AAADDDAAADDD

AAADDDAAADDD

思路 3:**1** 到 **20** 行大小写转换，把小写字母转换为大写字母

* sed -n '1,20p' lidao.txt| tr 'ad' 'AD'
* awk 'NR==1,NR==20{print toupper($0)}' lidao.txt
* awk 'NR>=1&&NR<=20{print toupper($0)}' lidao.txt
* sed '1,20s#[ad]#\U&#g' lidao.txt
* sed '1,20y/ad/AD/' lidao.txt

**2.4.12** 使用 **sed** 命令打印出文件的第一行到第三行

* head -3 file.txt
* sed -n '1,3p' file.txt
* awk 'NR==1,NR==3' oldboy.txt
* awk 'NR>=1&&NR<=3' oldboy.txt
* awk 'NR<=3' oldboy.txt

**2.4.13** 在**/var/sync/log/cef\_watchd-20080424.1og** 文件中查找大小写不敏感**“mysql"**字符串中的命令是

|  |  |
| --- | --- |
| ❑ | grep -i "mysql" /var/sync/log/cef\_watchd-20080424.log |
| ❑ | sed -n '/[a-z]/Ip' /var/sync/log/cef\_watchd-20080424.log |
| ❑ | awk -vIGNORECASE=1 '/[a-z]/' /var/sync/log/cef\_watchd-20080424.log |

[root@oldboyedu-show01 ~]# sed -n '/[a-z]/Ip' lidao.txt a b c d e f g h i j k l m n o p q r s t u v w x y z aaadddaaaddd aaadddaaaddd aaadddaaaddd

A B C D E

F G H I J K L M N O

P Q R S T

U V W X Y

Z

|  |
| --- |
| [root@oldboyedu-show01 ~]# awk '/[a-z]/' lidao.txt a b c d e f g h i j k l m n o p q r s t u v w x y z aaadddaaaddd aaadddaaaddd aaadddaaaddd  [root@oldboyedu-show01 ~]# awk -vIGNORECASE=1 '/[a-z]/' lidao.txt a b c d e f g h i j k l m n o p q r s t u v w x y z aaadddaaaddd aaadddaaaddd aaadddaaaddd A B C D E F G H I J  K L M N O  P Q R S T  U V W X Y  Z  [root@oldboyedu-show01 ~]# #ignore case 忽略大小写 |

**2.4.14** 为了在**/var/sync/log** 目录中查找更改时间在 **5** 日以前的文件井删除它们，**linux** 命令是同上题目**: 1.4.10** 查找最后创建时间是 **3** 天前，后缀是**\*.log** 的文件并删除

**2.4.15** 把 **data** 目录及其子目录下所有以扩展名**.txt** 结尾的文件中包含 **oldgirl** 的字符串全部替换为 **oldboy.**

同此题:查找最后创建时间是 3 天前，后缀是\*.log 的文件并删除把删除改为 sed 或 awk 替换.

**2.4.16** 写出删除 **abc.txt** 文件中的空行

同此题: 1.4.7 删除 file.txt 文件中的空行

**2.4.17** 查看 **linux** 服务器 **ip** 的命令，同时只显示包含 **ip** 所在的行打印出来

hostname -I | awk 'NR==1{print $2}' ip a s eth0|awk –F ‘[ /]+’ ‘NR==3{print $3}’ ip a s eth0|sed –rn ‘3s#^.\*t(.\*)/.\*$#\1#gp’

[root@web01 ~]# ip a s eth0 |awk -F"inet |/24" 'NR==3{print $2,$0}'

10.0.0.7 inet 10.0.0.7/24 brd 10.0.0.255 scope global eth0

**2.4.18** 用 **awk** 获取文件中第三行的倒数第二列字段awk 默认的分隔符就是空格，连续的空格，tab 键

awk 'NR==3{print $(NF-1)}' oldboy.txt

**2.4.19** 有文件 **file1**

* 请用 shell 查询 file1 里面空行的所在行号
* 查询 file1 以 abc 结尾的行
* 打印 file 文件第一行到第三行

* 请用 shell 查询 file1 里面空行的所在行号

grep '^$' -n file1 awk '/^$/{print NR,$0}' file.txt sed -n '/^$/{=;p}' file.txt|sed 'N;s#\n# #g'

[root@oldboyedu-show01 ~]# awk '/^$/{print NR,$0}' file.txt

3

8

|  |
| --- |
| [root@oldboyedu-show01 ~]# sed -n '/^$/{=;p}' file.txt|sed 'N;s#\n# #g'  3 8  [root@oldboyedu-show01 ~]# seq 10  1 |
| 2  3 4  5  6  7 8  9  10  [root@oldboyedu-show01 ~]# seq 10 |sed 'N;s#\n# #g'  1 2 3 4  5 6 7 8  9 10 |

* 查询 file1 以 abc 结尾的行
* 打印 file 文件第一行到第三行参考上面的题目:

**2.4.20** 显示 **file** 文件里匹配 **foo** 字串那行以及上下 **5** 行、显示 **foo** 及前 **5** 行、显示 **foo** 及后 **5** 行

grep "foo" -B 5 file grep "foo" -A 5 file grep "foo" -C 5 file

**2.4.21 echo "ABCDEF abtdefg" |sed -i s/c/t/g** 这条命令是否正确？如若正确可能会显示什么样的结果？不正确

-i 修改文件内容

no input files

sed无法找到要修改的文件

必须熟练：单引号 双引号 不加引号区别

<https://www.jianshu.com/p/60a810af2656>

**2.4.22** 当前系统中没有任何文本编辑器（**vi,emacs,vim,edit** 等），如何过滤掉注释行和空行查看

**/etc/ssh/sshd\_config** 文件

awk '!/^#|^$/' /etc/ssh/sshd\_config egrep -v '^#|^$' /etc/ssh/sshd\_config sed -r '/^#|^$/d' /etc/ssh/sshd\_config sed -nr '/^#|^$/!p' /etc/ssh/sshd\_config egrep '^[a-Z]' /etc/ssh/sshd\_config

参考: 删除 file.txt 文件中的空行

例如：文件名为“filename”，文件内容如下：

#This is a comments-1

#This is a comments-2

This is a comments-1 This is a comments-2

This is a comments-3

This is a comments-4

**2.4.23** 查找 **file.log** 文件中的包含关键字**“helloworld”**的内容，及其上下两行的内容重定向保存到

**1.txt**。请写出命令

grep "helloword" -C 2 file.log > 1.txt

第**3**章进阶题目**:**

需要用到awk数组和函数知识点

**3.1** 找出 **ifconfig** 命令结果中的 **1-255** 之间的数值；

net-tools

ifconfig| egrep -o '[0-9]+'|awk '$0>=1 && $0<=255'

ifconfig |awk -vRS="[^0-9]+" '{print NR,$0}'

ifconfig |awk -vRS="[^0-9]+" '$0>=1 && $0<=255'

**3.2** 文件内容排序、文件行取唯一值的命令分别是？

sort uniq

**3.3** 用 **awk** 获取文件中第三行的倒数第二列字段

awk 'NR==3{print $(NF-1)}' file.txt

**3.4 awk** 是一个很强大的文本处理工具，请使用 **awk** 统计当前主机的并发访问量

netstat -ant | awk '/^tcp/ {++state[$NF]} END {for(key in state) print key,state[key]}'

ss -ant | awk ' {++state[$1]} END {for(key in state) print key,state[key]}'

**3.5** 有一个文件 **nginx.log**，请用 **awk** 提取时间（**HH:MM**）**,**文件内容如下：

|  |
| --- |
| 第1行内容:183.250.220.178|-l[20/jul/2017:10:35:14 +0800]|POST /audiosearch/searchHTTP/1.1|200|54|- |
|  |
| lDalvik/1.6.0（linux;U;Android 4,4,4;Konka Android TV 638Build/KTU84P）|-l- |
|  |
| [5.069|5.001,0.005|www.kuyun.com|8771|172.21.19.67:8084,172.21.19.66:8084] 第2行内容:183.250.220.178|-l[20/jul/2017:10:35:14 +0800]|POST /audiosearch/searchHTTP/1.1|200|54|lDalvik/1.6.0（linux;U;Android 4,4,4;Konka Android TV 638Build/KTU84P）|-l[5.069|5.001,0.005|www.kuyun.com|8771|172.21.19.67:8084,172.21.19.66:8084] |

[root@web01 ~]# awk -F'2017:|:14' '{print $2}' time.log

10:35

10:35

[root@web01 ~]# awk -F'/....:|:.. ' '{print $2}' time.log

10:35

10:35

**3.6** 统计 **apache** 访问日志流量排名前 **10** 个 **IP**

awk '{h[$1]+=$10}END{for(p in h)print p,h[p]}' access.log |sort -rnk2|head

**3.7** 统计出 **apache** 的 **access.log** 中访问最多的 **5** 个 **ip**

|  |
| --- |
| awk '{h[$1]++}END{for(p in h)print p,h[p]}' access.log |sort -rnk2|head -5  awk '{print $1}' access.log|sort |uniq -c|sort -rnk2|head -5 |

**3.8** 这条 **linux** 命令的含义：**netstat -n |awk '/^tcp/ {++h[$NF]} END {for(a in h) print a,h[a]}'**

tcp11 状态 每种状态的数量。

**3.9** 使用 **awk** 分析日志（最后一个字段是 **IP**，统计访问量前十的 **IP**，并进行排序）

|  |
| --- |
| awk '{h[$NF]++}END{for(p in h)print p,h[p]}' access.log |sort -rnk2|head  awk '{print $NF}' access.log|sort |uniq -c|sort -rnk2|head |

**3.10** 从 **a.log** 文件中提取包含**“WARNING”**或**“FATAL”**，同时不包含**“IGNOR”**行，然后，提取以**“:”** 分割的第五个字段

awk -F: '/WARNING|FATAL/ && !/IGNOR/{print $5}' a.log