LINUX命令

目录

1 ls / 2 cd / 3 pwd /

4 mkdir / 5 rm / 6 mv / 7 cp / 8 cat / 9 more /

10 less / 11 head / 12 tail / 13 which /

14 whereis / 15 locate / 16 find / 17 xargs /

18 wc / 19 grep / 20 cut / 21 paste / 22 tr /

23 sort / 24 uniq / 25 df / 26 du / 27 time /

LINUX命令自备手册

H3 ls

-a	–all 列出目录下的所有文件，包括以 . 开头的隐含文件
-l	除了文件名之外，还将文件的权限、所有者、文件大小等信息详细列出来
-h	–human-readable 以容易理解的格式列出文件大小（例如 1K 234M 2G)
-t	以文件修改时间排序

如`ls -l d*`列出所有以d开头的文件目录详细内容

cd

`cd -`进入上次所在目录

pwd

-P	显示实际物理路径，而非使用连接（link）路径
-L	当目录为连接路径时，显示连接路径

mkdir

-m --

mode= 设定权限<模式>

模式

-m --mode= 模式	设定权限<模式>
-p --parents	可以是一个路径名称。若路径中的某些目录尚不存在，加上此选项后，系统将自动建立好那些尚不存在的目录，即一次可以建立多个目录
-v --verbose	每次创建新目录都显示信息

如

```
mkdir -vp
shiyanlou/{lib/,bin/,doc/{info,product}}
```

rm

-f --force	忽略不存在的文件，从不给出提示
-i --interactive	进行交互式删除
-r --recursive	指示 rm 将参数中列出的全部目录和子目录均递归地删除
-v --verbose	详细显示进行的步骤

mv

-b --back	若需覆盖文件，则覆盖前先行备份

-b --back	若需覆盖文件，则覆盖前先行备份
-f --force	如果目标文件已经存在，不会询问而直接覆盖
-i --interactive	若目标文件已经存在时，就会询问是否覆盖
-u --update	若目标文件已经存在，且源文件比较新，才会更新
-t --target	该选项适用于移动多个源文件到一个目录的情况，此时目标目录在前，源文件在后

H3 cp

-t --target-directory	指定目标目录

-t --target-directory	指定目标目录
-i --interactive	覆盖前询问（使前面的 -n 选项失效）
-n --no-clobber	不要覆盖已存在的文件（使前面的 -i 选项失效）
-s --symbolic-link	对源文件建立符号链接，而非复制文件
-f --force	强行复制文件或目录，不论目的文件或目录是否已经存在
-u --update	使用这项参数之后，只会在源文件的修改时间较目的文件更新时，或是对应的目的文件并不存在，才复制文件

cat

-A --show-all	等价于 -vET
-b --number-nonblank	对非空输出行编号
-e	等价于 -vE
-E --show-ends	在每行结束处显示 $
-n --number	对输出的所有行编号，由 1 开始对所有输出的行数编号
-s --squeeze-blank	有连续两行以上的空白行，就代换为一行的空白行
-t	与 -vT 等价

-A --show-all	等价于 -vET
-T --show-tabs	将跳格字符显示为 ^I
-u	（被忽略）
-v --show-nonprinting	使用 ^ 和 M- 引用，除了 LFD 和 TAB 之外

有趣的是,tac可以将文件的内容反向显示，

```
tac synchro.sh


  expect eof
  send "bye\r"
  # 退出此次ftp会话，并等待服务器的退出提示EOF
  expect "ftp>"
```

more

more 命令会一页一页的显示，按空格键（space）往下一页显示，按 B 键就会往回（back）一页显示

+n	从第 n 行开始显示
-n	定义屏幕大小为 n 行
+/pattern	在每个档案显示前搜寻该字串（pattern），然后从该字串前两行之后开始显示
-c	从顶部清屏，然后显示
-d	提示“Press space to continue，’q’ to quiet”，禁用响铃功能
-p	通过清除窗口而不是滚屏来对文件进行换页，与-c 选项相似

+n	从第 n 行开始显示
-s	把连续的多个空行显示为一行
-u	把文件内容中的下划线去掉

=	输出当前行的行号
q	退出 more
空格键	向下滚动一屏
b	返回上一屏

如从第五行显示.log文件：`more +5 shiyanlou.log`

如从.log找第一个出现g的字符串并在前两行显示:`more +/g shiyanlou.log`

less

参数	描述
-e	当文件显示结束后，自动离开
-f	强迫打开特殊文件，例如外围设备代号、目录和二进制文件
-i	忽略搜索时的大小写
-m	显示类似 more 命令的百分比
-N	显示每行的行号
-s	显示连续空行为一行

符号	描述

符号	描述
/字符串	向下搜索“字符串”的功能
?字符串	向上搜索“字符串”的功能
n	重复前一个搜索（与 / 或 ? 有关）
N	反向重复前一个搜索（与 / 或 ? 有关）
b	向前翻一页
d	向后翻半页
q	退出 less 命令
空格键	向后翻一页
向上键	向上翻动一行
向下键	向下翻动一行

H3 head

参数	描述
-q	隐藏文件名
-v	显示文件名
-c<字节>	显示字节数
-n<行数>	显示的行数

如

```
head -n 5 xxx.log
```

显示前5行的内容

tail

参数	描述
-f	循环读取
-q	不显示处理信息
-v	显示详细的处理信息
-c<字节>	显示的字节数
-n<行数>	显示行数

tail和head用法类似，不过显示的是末尾的内容。

which

which 命令的作用是，在 PATH 变量指定的路径中搜索可执行文件的所在位置。它一般用来确认系统中是否安装了指定的软件

命令可以是下面四种形式之一：

01. 是一个可执行程序，就像我们所看到的位于目录 `/usr/bin` 中的文件一样。属于这一类的程序，可以编译成二进制文件，诸如用 C 和 C++ 语言写成的程序，也可以是由脚本语言写成的程序，比如说 shell，perl，python，ruby，等等。
02. 是一个内建于 shell 自身的命令。bash 支持若干命令，内部叫做 shell 内部命令 (builtins)。例如，上面我本地环境中的 cd 命令，就是一个 shell 内部命令。
03. 是一个 shell 函数。这些是小规模的 shell 脚本，它们混合到环境变量中。 比如上面讲到的 cd 命令，在某些环境中就是一个 shell 函数。
04. 是一个命令别名。我们可以定义自己的命令，建立在其它命令之上。

H3 whereis

whereis 命令查找速度非常快，这是因为它根本不是在磁盘中漫无目的乱找，而是在一个数据库中（/var/lib/mlocate/mlocate.db）查询。这个数据库是 Linux 系统自动创建的，包含有本地所有文件的信息，并且每天通过自动执行 updatedb 命令更新一次。也正是因为这个数据库要**每天才更新一次**，就会使得 whereis 命令的搜索结果有时候会不准确，比如刚添加的文件可能搜不到。

参数	描述
-b	定位可执行文件
-m	定位帮助文件
-s	定位源代码文件

参数	描述
-u	搜索默认路径下除可执行文件、源代码文件和帮助文件以外的其它文件
-B	指定搜索可执行文件的路径
-M	指定搜索帮助文件的路径
-S	指定搜索源代码文件的路径

如`whereis -m gcc`就是搜索gcc的帮助文档

locate

locate与whereis使用相同数据库，但可以实现更复杂的功能

参数	描述
-q	安静模式，不会显示任何错误讯息
-n	至多显示 n 个输出
-r	使用正则表达式做寻找的条件
-V	显示版本信息

支持正则表达式

H3 find

find 命令主要作用是沿着文件层次结构向下遍历，匹配符合条件的文件

参数	描述

参数	描述
-print	find 命令将匹配的文件输出到标准输出
-exec	find 命令对匹配的文件执行该参数所给出的 shell 命令
-name	按照文件名查找文件
-type	查找某一类型的文件
-prune	使用这一选项可以使 find 命令不在当前指定的目录中查找，如果同时使用 -depth 选项，那么 -prune 将被 find 命令忽略
-user	按照文件属主来查找文件
-group	按照文件所属的组来查找文件

参数	描述
-mtime -n +n	按照文件的更改时间来查找文件，-n 表示文件更改时间距现在小于 n 天，+n 表示文件更改时间距现在大于 n 天，find 命令还有 -atime 和 -ctime 选项

如打印当前目录下的文件目录列表，可以使用如下命令：

```
find . -print
```

打印当前目录下所有以.txt 结尾的文件名，可以使用如下命令（截图只显示部分）：

```
find . -name "*.txt" -print
```

打印当前目录下所有以.txt 或.pdf 结尾的文件名，可以使用如下命令（截图只显示部分）：

```
find . \( -name "*.pdf" -or -name "*.txt" \)
```

打印当前目录下所有**不以**.txt 结尾的文件名，可以使用如下命令：

```
find . ! -name "*.txt"
```

根据文件类型来查找文件，使用 -type 选项，常见 find 文件类型见下表：

文件类型	描述
b	块设备文件
c	字符设备文件
d	目录
f	普通文件
l	符号链接

根据文件权限查找文件，使用 -perm 选项。所有者使用 -user 选项。

另外，find 命令可以通过逻辑操作符来创建更复杂的逻辑关系，例如 find 命令（一）中的例三就使用了操作符 -or 。find 命令的逻辑操作符见下表：

操作符	描述
-and	匹配如果操作符两边的测试条件都是真。可以简写为 -a。注意若没有使用操作符，则默认使用 -and
-or	匹配若操作符两边的任一个测试条件为真。可以简写为 -o
-not	匹配若操作符后面的测试条件是假。可以简写为一个感叹号（!）
()	把测试条件和操作符组合起来形成更大的表达式。这用来控制逻辑计算的优先级。默认情况下，find 命令按照从左到右的顺序计算。经常有必要重写默认的求值顺序，以得到期望的结果。即使没有必要，有时候包括组合起来的字符，对提高命令的可读性是很有帮助的。注意因为圆括号字符对于 shell 来说有特殊含义，所以在命令行中使用它们的时候，它们必须用引号引起来，才能作为实参传递给 find 命令。通常反斜杠字符被用来转义圆括号字符

打印当前目录下所有以 .txt 结尾的符号链接，可以使用如下命令：

```
find . -type l -name "*.txt" -print
```

打印当前目录下所有权限为 777 的 php 文件（web 服务器上的 php 文件一般需要执行权限），可以使用如下命令：

```
find . -type f -name "*.php" -perm 777
```

打印当前目录下权限不是 777 和 664 的所有文件，可以使用如下命令：

```
find . -type f \( ! -perm 777 -and ! -perm 644 \)
```

找到当前目录下所有 php 文件，并显示其详细信息，可以使用如下命令：

```
find . -name "*.php" -exec ls -l {} \;
```

xargs

xargs 命令可以从标准输入接收输入，并把输入转换为一个特定的参数列表。

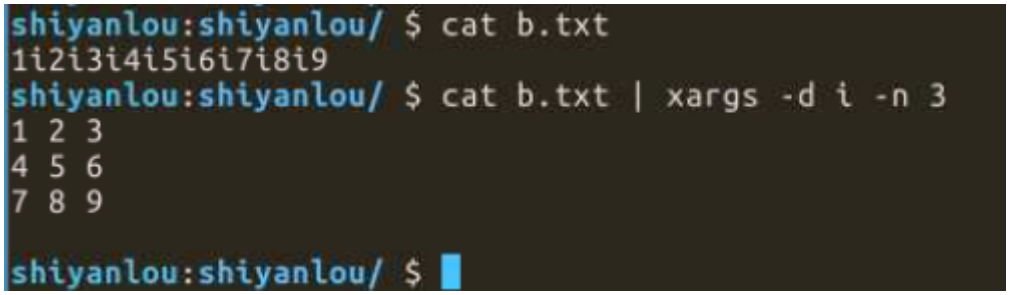
参数	描述
-n	指定每行最大的参数数量
-d	指定分隔符

将单行输入转换为多行输出，可以使用如下命令：

```
echo "1 2 3 4 5 6 7" | xargs -n 3
```

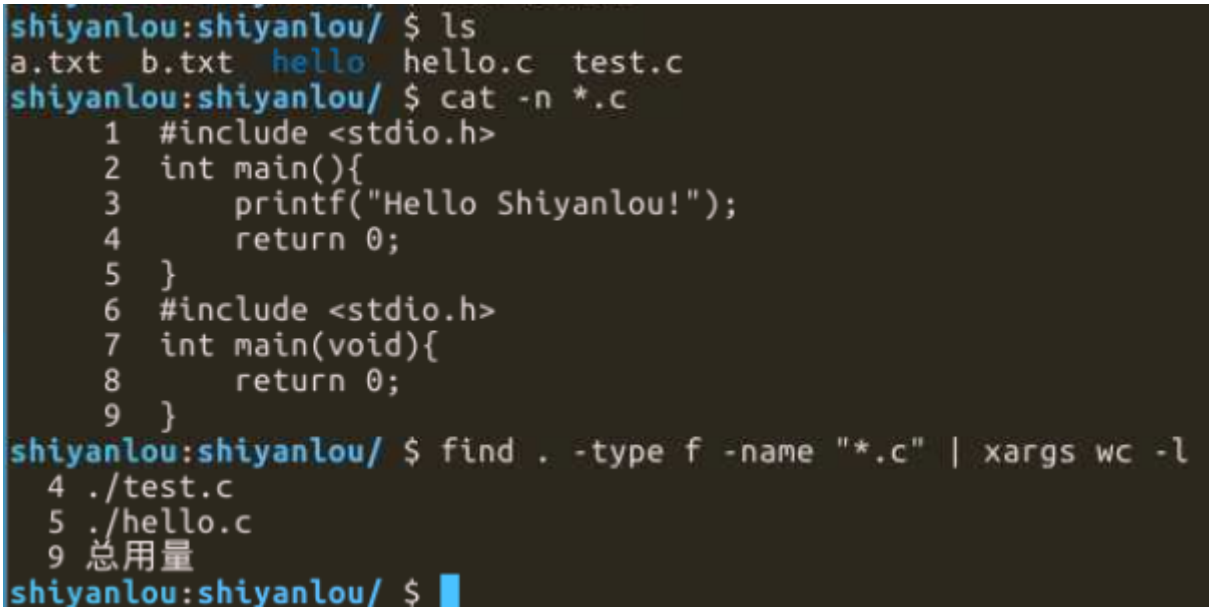
将单行输入转换为多行输出，指定分隔符为 i，可以使用如下命令：

```
cat b.txt | xargs -d i -n 3
```



查找当前目录下所有 c 代码文件，统计总行数，可以使用如下命令：

```
find . -type f -name "*.c" | xargs wc -l
```



wc

是一个统计的工具，主要用来显示文件所包含的行、字和字节数。

参数	描述
-c	统计字节数
-l	统计行数
-m	统计字符数，这个标志不能与 -c 标志一起使用
-w	统计字数，一个字被定义为由空白、跳格或换行字符分隔的字符串
-L	打印最长行的长度

如统计文件的字节数：

```
wc -c c.txt
```

注意，每行结尾的换行符也算一个字符，空格也算一个字符。另外，由于系统采用 UTF-8 编码，所以一个汉字为 3 字节，9 个汉字加上一个换行，一共 28 个字节。

统计/bin目录下的命令个数：

```
ls /bin | wc -l
```

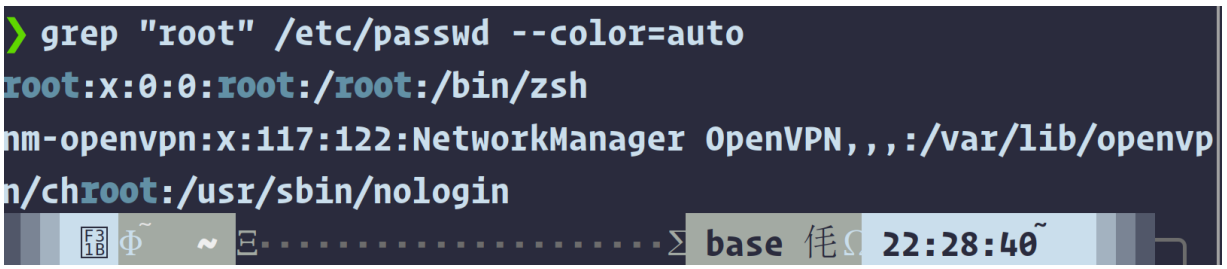
grep

grep 可用于 shell 脚本，因为 grep 通过返回一个状态值来说明搜索的状态，如果模板搜索成功，则返回 0，如果搜索不成功，则返回 1，如果搜索的文件不存在，则返回 2。

参数	描述
-c	计算找到‘搜寻字符串’（即 pattern）的次数
-i	忽略大小写的不同，所以大小写视为相同
-n	输出行号
-v	反向选择，打印不匹配的行
-r	递归搜索
--color=auto	将找到的关键词部分加上颜色显示

如取出/etc/passwd中出现root的行并标识:

```
grep "root" /etc/passwd --color=auto
```



如取出没有root,nologin的行：

```
grep -v "root" /etc/passwd | grep -v "nologin"
```

如在当前目录下递归搜索文件中包含 main() 的文件

```
grep -r "main()"
```

grep **支持正则**

正则表达式	描述	示例
^	行起始标记	^shiyan 匹配以 shiyan 起始的行
$	行尾标记	$shiyan 匹配以 shiyan 结尾的行
.	匹配任意一个字符	a.c 匹配 abc、aac，但不匹配 abbc
[]	匹配包含在[字符]之中的任意一个字符	ab[cd] 匹配 abc 或 abd
[^]	匹配除[^字符]之外的任意一个字符	1[^01] 匹配 12、13，但不匹配 10、11
[-]	匹配[ ]指定范围内的任意一个字符	[1-5] 匹配 1-5 的任意一个数字
{n}	匹配之前的项 n 次	[0-9]{2} 匹配任意一个两位数，相当于[0-9][0-9]
{n,}	之前的项至少需要匹配 n 次	[0-9]{2,} 匹配任意一个两位或更多位的数字
{n,m}	指定之前的项需要匹配的最小和最大次数	[0-9]{2,5} 匹配从两位数到五位数之间的任意一个数字
?	匹配之前的项 1 次或者 0 次	shiy?an 匹配 shiyan 或 shian
*	匹配之前的项 0 次或者多次	shiy*an 匹配 shian、shiyan、shiyyan
+	匹配之前的项 1 次或者多次	shiy+an 匹配 shiyan、shiyyan
()	创建一个用于匹配的子串	ma(in)? 匹配 ma 或 main
|	匹配 | 两边的任意一项	Dec (1st | 2nd) 匹配 Dec 1st 或者 Dec 2nd
\	将上面的特殊字符进行转义	a\+b 匹配 a+b

利用 Linux 系统自带的字典查找一个五个字母的单词，第三个字母为 j,最后一个字母为 r，`/usr/share/dict`目录下存放字典文件（若没有可手动建立），可以使用如下命令：

```
grep '^..j.r$' words
```

验证固定电话，打印符合条件的电话，固定电话格式基本都是带有 0 的区号+连接符“-”＋电话号码，另外还有可能有分机号，区号有 3 位、4 位，电话号码有 7 位和 8 位的，可以使用如下命令：

```
grep -E "^0[0-9]{2,3}-[0-9]{7,8}(-[0-9]{3,4})?$" telphone.txt
```

区号：前面一个 0，后面跟 2-3 位数字 0[0-9]{2,3}

电话号码：7-8 位数字 [0-9]{7,8}

分机号：一般都是 3-4 位数字 [0-9]{3,4}

cut

cut 命令是一个将文本按列进行切分的小工具，它可以指定分隔每列的定界符。

参数	描述

参数	描述
-b	以字节为单位进行分割
-c	以字符为单位进行分割
-d	自定义分隔符，默认为制表符
-f	自定义字段
--complement	抽取整个文本行，除了那些由 -c 或 -f 选项指定的文本

取出`student.txt`文件中的第一列和第三列，可以使用如下命令：

```
cut -f 1,3 -d ' ' student.txt
```

取出`student.txt`文件中的前三列，可以使用如下命令：

```
cut -f 1-3 -d ' ' student.txt
```

取出`student.txt`文件中除第一列的其他列，可以使用如下命令：

```
cut -f 1 -d ' ' student.txt --complement
```

H3 paste

paste 命令的功能正好与 cut 相反。它会添加一个或多个文本列到文件中

参数	描述
-s	将每个文件合并成行而不是按行粘贴
-d	自定义分隔符，默认为制表符

将`student.txt`和`telphone.txt`文件中的内容按列拼接，可以使用如下命令：

```
paste student.txt telphone.txt
```

将`student.txt`和`telphone.txt`文件中的内容按列拼接，指定分隔符为`:`，可以使用如下命令：

```
paste student.txt telphone.txt -d ':'
```

tr

tr 命令常被用来更改字符，我们可以把它看作是一种基于字符的查找和替换操作。

参数	描述
-d	删除匹配 SET1 的内容，并不作替换

将输入的字符大写转换为小写，可以使用如下命令：

```
echo 'THIS IS SHIYANLOU!' | tr 'A-Z' 'a-z'
```

将输入的字符中的数字删除，可以使用如下命令：

```
echo 'THIS 123 IS S1HIY5ANLOU!' | tr -d '0-9'
```

tr 命令的一个有趣的用法是执行 ROT13 文本编码。ROT13 是一款微不足道的基于一种简易的替换暗码的加密类型。把 ROT13 称为“加密”是不严格的，“文本模糊处理”更准确些。有时候它被用来隐藏文本中潜在的攻击内容。这个方法就是简单地把每个字符在字母表中向前移动 13 位。因为移动的位数是所有 26 个字母的一半，所以对文本再次执行这个算法，就恢复到了它最初的形式。可以使用如下命令：

```
#加密
echo 'shiyanlou' | tr 'a-zA-Z' 'n-za-mN-ZA-M'
```

得到结果 fuvlnaybh

```
#解密
echo 'fuvlnaybh' | tr  'a-zA-Z'  'n-za-mN-ZA-M'
```

得到结果 shiyanlou

sort

参数	描述
-n	基于字符串的长度来排序，使用此选项允许根据数字值排序，而不是字母值

参数	描述
-k	指定排序关键字
-b	默认情况下，对整行进行排序，从每行的第一个字符开始。这个选项导致 sort 程序忽略每行开头的空格，从第一个非空白字符开始排序
-m	只合并多个输入文件
-r	按相反顺序排序，结果按照降序排列，而不是升序
-t	自定义分隔符，默认为制表符

如列出/usr/share下使用空间最多的前10个目录，

```
du -s /usr/share/* | sort -nr  | head -10
```

指定按照自定义排序字段来排序：

```
sort  -k 1,1  -k 2n data.txt
```

第一个-k指明只对第一个字段排序，1,1意味着从第一个字段开始，在第一个字段结束。第二个k的2n表示对第二个字段按照数值排序。在比如-k 3.4n意味着始于第三个字段的第四个字符，按数值排序。

H3 uniq

uniq 从标准输入或单个文件名参数接受数据有序列表，默认情况下，从数据列表中删除任何重复行。

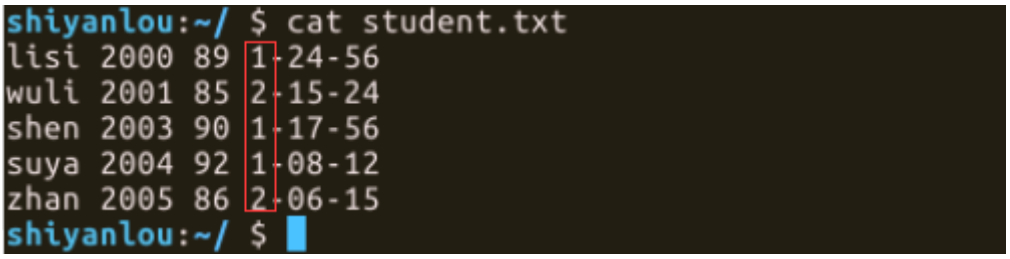
参数	描述

参数	描述
-c	在每行前加上表示相应行目出现次数的前缀编号
-d	只输出重复的行
-u	只显示唯一的行
-D	显示所有重复的行
-f	比较时跳过前 n 列
-i	在比较的时候不区分大小写
-s	比较时跳过前 n 个字符
-w	对每行第 n 个字符以后的内容不作对照

如找出/bin与/usr/bin下所有相同的命令，

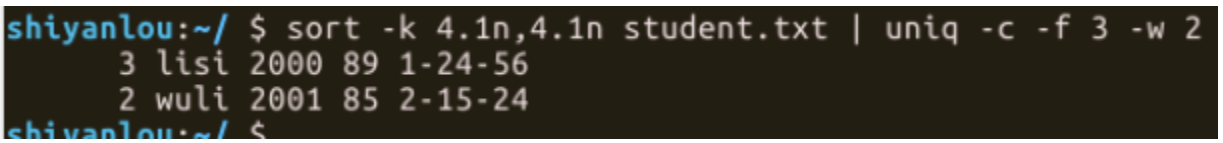
```
ls /bin /usr/bin | sort | uniq -d
```

现有文件内容如下，红色方框里的内容表示区号，现在要统计出各个区号的总人数。



实现思路：首先按区号对每行信息排序，然后使用 uniq 命令对区号进行重复行统计。使用命令如下：

```
sort -k 4.1n,4.1n student.txt | uniq -c -f 3 -w 2
```



`sort -k 4.1n,4.1n` 表示对第四个字段的第一个字符按数值排序。

`uniq -c -f 3 -w 2` 中 -f 3 表示跳过前三列的比较，那么现在只剩下最后一列，-w 2 表示第 2 个字符后的内容不做比较，为什么是 2 呢，因为跳过前三列时并没有跳过最后一列前面的空格分隔符，区号前都还有一个空格。

H3 df

df 命令的功能是用来检查 linux 服务器的文件系统的磁盘空间占用情况。可以利用该命令来获取硬盘被占用了多少空间，目前还剩下多少空间等信息。

参数	描述
-a	全部文件系统列表

参数	描述
-h	方便阅读方式显示
-i	显示 inode 信息
-T	文件系统类型
-t<文件系统类型>	只显示选定文件系统的磁盘信息
-x<文件系统类型>	不显示选定文件系统的磁盘信息

H3 du

du 命令也是查看使用空间的，但是与 df 命令不同的是 Linux du 命令是对文件和目录磁盘使用的空间的查看。

参数	描述

参数	描述
-a	显示目录中所有文件的大小。
-b	显示目录或文件大小时，以 byte 为单位。
-c	除了显示个别目录或文件的大小外，同时也显示所有目录或文件的总和。
-k	以 KB(1024bytes)为单位输出。
-m	以 MB 为单位输出。
-s	仅显示总计，只列出最后加总的值。
-h	以 K，M，G 为单位，提高信息的可读性。

H3 time

将 time 命令的执行结果保存到文件中，可以使用如下命令：

```
{ time date; } 2>1.txt
(time date) 2>2.txt
```