训练稿——Linux常用重点命令 grep&sort&find&uniq

目录

[命令：grep 1](#_Toc323572290)

[命令：sort 4](#_Toc323572291)

[命令：find 6](#_Toc323572292)

[命令：uniq 8](#_Toc323572293)

## 命令：grep

|  |
| --- |
| Grep General Regular Expression Parser |
| 【global search regular expression(RE) and print out the line】  全面搜索正则表达式并把行打印出来，是一种强大的文本搜索工具，它使用正则表达式搜索文本，并把匹配的行打印出来。  grep（全局正则表达式版本）允许对文本文件进行模式查找。如果找到匹配模式，grep打印包含模式的所有行。grep支持基本正则表达式，也支持其扩展集。  资料：man grep |

|  |
| --- |
| 功能： |
| 使用正则表达式搜索文本并打印匹配行 |

|  |
| --- |
| 格式: |
| grep [options] PATTERN [Files]  注：输入字符串作为参数，最好双引号括起 “mystr”[ 一以防被误解为shell命令，二可以用来查找多个单词组成的字符串]  在调用变量时，也使用双引号括起 “$MYSTR”  使用正则[匹配模式]是，应使用单引号括起 ‘49[32]’  grep [-acinv] [--color=auto] 'str' filename |

|  |  |
| --- | --- |
| 选项 | |
| -c | 只输出匹配的行数,而不输出匹配的行 |
| -a | 查找binary文件 |
| -i | 不区分大小写 |
| -h | 查询多个文件时不显示文件名【默认是 文件名:符合的记录行】 |
| -l | 查询多个文件时，只输出包含匹配字符的文件名 |
| -n | 显示匹配行及行号 |
| -s | 不显示不存在或无匹配文本的错误信息 |
| -v | 显示不包含匹配文本的所有行 |
| -o | 仅显示匹配的部分 |
| -r | 递归选项。指示GNUgrep和egrep检查做为参数的目录下的文件，然后递归的检查该目录下所有子目录下的文件 |
| -E | 启用扩展表达式，可使用扩展元字符 egrep  　+  　　匹配一个或多个先前的字符。如：'[a-z]+able'，匹配一个或多个小写字母后跟able的串，如loveable,enable,disable等。  　?  匹配零个或1个先前的字符。如：'gr?p'匹配gr后跟一个或没有字符，然后是p的行。  \*  匹配0个或多个  　a|b|c  　　匹配a或b或c。如：grep|sed匹配grep或sed  　()  　　分组符号，如：love(able|rs)ov+匹配loveable或lovers，匹配一个或多个ov。  　x,x{m,},x{m,n}  作用同x\{m\},x\{m,\},x\{m,n\}  {n}必须匹配n次 {n,}至少匹配n次 {n,m}匹配次数在[n,m]之间 |
| -A n1 -B n2 | 匹配行附近前后多少行 |
| -C n | 同上 |

|  |
| --- |
| 常用示例; |
| 1. 查询多个文件   grep “sort” \*.doc 在目录下所有doc文件中查sort  grep “sort” filea fileb 在filea,fileb中查找   1. 计算匹配行数   grep –c “test” data.txt   1. 显示非匹配行   grep –v “test” data.txt   1. 忽略大小写   默认是大小写敏感的，若需要，使用-i进行忽略  grep –i “ignore” data.txt   1. 正则表示   grep ‘48[34]’ data.txt  grep ‘^[^48]’ data.txt 开头不是4,8的  grep ‘[Ss]ept’ data.txt  grep ‘^[0-9][0-5][0-6]’  grep ‘4\{2,\}’ data   1. 使用“与” “或”   必须使用参数 –E [E一定大写]  grep –E ‘aaa|bbb’ data  等价于 grep –c “[L|l]et” dream   1. 空行及特殊字符   grep ‘^$’ data  grep ‘\.’ myfile   1. 类名：   grep 允许使用国际字符串模式匹配或匹配模式的类名   |  |  | | --- | --- | | [[:upper:]] | [A-Z] | | [[:lower:]] | [a-z] | | [[:digit:]] | [0-9] | | [[:alnum:]] | [0-9a-zA-Z] | | [[:space:]] | 空格或tab键 | | [[:alpha:]] | [a-zA-Z] |   grep ‘5[[:upper:]] [[:upper:]]’ data 5开头，两个大写   1. 系统grep命令   ls –l | grep ‘^d’  ls –l | grep ‘^[^d]’ |
| 查找内容前后的多少行 |
| • -A NUM, --after-context=NUM 匹配行之后多少行[ahead]  • -B NUM, --before-context=NUM 匹配行之前多少行[back]  • -C NUM，--context=NUM 匹配行前后多少行[上下文context] |
| 处理二进制文件 |
| Grep查找的是text文件，  如果要查找的是二进制文件，则输出如下结果：Binary file file-name matches  • -a, --text 使用-a进行二进制文件处理，结果类似文本文件 |
| 点亮匹配的内容 |
| 将匹配上的内容加颜色显示出来  --colour[=WHEN], --color[=WHEN] 三种WHEN=never,always,auto |
| 匹配内容的统计与其他 |
| 将匹配的内容进行记录统计  -c, --count -c，统计匹配行数， -cv统计不匹配行数  -o, --only-matching（只显示匹配的内容） |
| 查看目录下面的所有文件（包括子目录） |
| -R, -r, --recursive 递归，查询所有子目录  --include=PATTERN 只查询该模式文件  --exclude=PATTERN 不查询该模式文件  -n, --line-number 输出文件名及匹配行号 |
| 如何匹配以“-”开始的字符 |
| -e PATTERN, --regexp=PATTERN 使用正则 |
| 在哪些文件中出现 |
| -h, --no-filename 结果中不输出文件名  -L, --files-without-match 输出没有匹配上的文件名列表 |
| 反向匹配 |
| -v, --invert-match 反向，未匹配的行  -m NUM, --max-count=NUM 最大匹配次数 |

## 命令：sort

sort命令可以用来对文件的内容进行排序。

sort认为文件的每一行由一个个由空格（可以用-t选项指定其他分隔符）分隔的field组成的。

Sort可将许多不同的域按不同的列顺序分类

sort将文件的每一行作为一个单位，相互比较，比较原则是从首字符向后，依次按ASCII码值进行比较，最后将他们按升序输出

|  |
| --- |
| 格式： |
| sort –cmu –o out-putfile [other options] \*pos1 \*pos2 input-files |

|  |  |
| --- | --- |
| 选项： | |
| -c | 测试文件是否已经分类，如果乱序，则输出第一个乱序的行的相关信息，最后返回1.若非乱序，无输出，返回0 |
| -m | 合并两个分类文件 |
| -u | 删除所有重复行 |
| -o | 存储sort结果的输出文件名  【sort默认是把结果输出到标准输出，所以需要用重定向才能将结果写入文件，但是如果你想把排序结果输出到原文件中，用重定向不可使用.-o就可以解决这个问题】 |
| -t | 域分隔符；默认 空格，tab，可定义  用-t选项指定关键字。  它的格式是POS1[,POS2]  POS1指定关键字开始的列，POS2指定关键字结束的列，如果没有POS2，则从POS1到行末都是关键字。  其中POS的格式是F[.C][OPTS],F[.C][OPTS]。F指定第几个field（从1开始数）,C指定这个field里的第几个字符（也是从1开始数）。 |
| -b | 使用域进行分类时，忽略首个空格【会忽略每一行前面的所有空白部分，从第一个可见字符开始比较】 |
| -n | 指定分类是域上数字分类【默认使用字符排序的，若是要数字，使用-n】 |
| -r | 对分类次序或比较求逆【默认是升序】 |
| –kn | 按照第n个域进行排序  【sort [-t delimiter][+filed[column]] [option]】  【常用组合 –t ‘分隔符’ -kn】  -k选项的语法格式，如下：  [ FStart [ .CStart ] ] [ Modifier ] [ , [ FEnd [ .CEnd ] ][ Modifier ] ]  逗号（“，”）分为Start部分和End部分  如果不设定End部分，那么就认为End被设定为行尾  Modifier部分类似n和r的选项部分。  FStart.CStart，其中FStart就是表示使用的域，而CStart则表示在FStart域中从第几个字符开始算“排序首字符”, CStart可以省略的，省略的话就表示从本域的开头部分开始。  同理，在End部分中，你可以设定FEnd.CEnd，如果你省略.CEnd，则表示结尾到“域尾”，即本域的最后一个字符。或者，如果你将CEnd设定为0(零)，也是表示结尾到“域尾”。 |

|  |
| --- |
| 常用示例: |
| 1. 保存输出   $sort –o result sortfile  $sort sortfile > result  Sort的启动方式，sort认为一空格/多空格为分隔符，要加入其他的，必须使用-t，执行时，先查看-t,若是有，使用其进行分割，若是无，使用空格   1. 查看文件是否已排序   $sort –c sortfile  若未排序，输出信息 $?=1  若已排序，无信息输出$?=0   1. 使用其他分隔符   $sort –t : sortfile   1. 排完序后求逆   $sort –t : -r sortfile   1. 唯一分类，原文件中重复行去除   $sort –u sortfile   1. 指定分类域，1开始   $sort –t : -k 4 sortfile 【第四个域开始】  $sort –t : -k 4 –k 1 sortfile  【sort分隔符+ -k指定列 完成根据某列进行排序】   1. 指定sort序列   $sort +0 -2 +3 sortfile   1. 将两个分类文件合并   $sort –m sorted-file1 sorted\_file2  【将两个已排序文件整合，类似合并排序的后半部分】  具体示例：   1. 对以下文件进行排序   153 6 314 5 222 4 114 3  sort -k 1.1,1.2 sort.test  指定了按开始的两列进行排序，即比较"15","31","22","11"来排序  结果:  114 3 153 6 222 4 314 5  sort -k 1.3,2.1 sort.test  指定了通过比较"3 6","4 5", "2 4", "4 3"（注意其中的空格）来排序  结果：  222 4 153 6 114 3 314 5  sort -k 1.3,1.3 sort.test  指定只按第3列来排序，则因为第2和第4行都 是"4"，它们相同，则会再比较一下整行  [如果关键字相同，则sort会把整行再比较一次]  222 4 153 6 114 3 314 5 |

## 命令：find

|  |
| --- |
| 功能 |
| 查找特定字符串的文件或目录 |

|  |
| --- |
| 格式 |
| find [pathName] [options ] [tests] [actions] |
| pathname:路径，当前为. 用户根目录为~ 系统根目录/  options：命令选项 见列表  test 一些针对属性的测试  actions [-print -exec -ok]  -print:将匹配文件输出到标准输出  -exec command:对匹配文件执行参数所给的shell命令，这个动作必须使用\;字符对结尾  -ok command:同上，但存在询问选择  -ls：对当前文件使用ls-dils |

|  |  |
| --- | --- |
| 主要选项： | |
| -name pattern | 按文件名查找，不包含路径名，提供匹配模式pattern,pattern必须总是用引号括起 |
| -perm 权限数值 | 按文件权限查找 |
| -user username | 按文件属主查找 find –user ken |
| -nouser | 搜索不属于本机用户的文件 |
| -group 组名 | 按文件所属的组查找 |
| -nogroup | 搜索不属于本机组的文件 |
| -type | 按文件类型查找 b块设备，p管道，d目录，l符号链接  C字符设备，f普通文件 |
| --size n[c] | 查文件长度为n的文件，c表以字符计，默认以块计算 |
| -depth | 查找时，首先在当前目录，之后再其子目录 |
| -maxdepth N | 最多搜索N层目录 |
| -prune | 使find不在当前指定的目录中查找，若使用-depth，本选项忽略 |
| -atime n | 文件在n天之前被最后访问过 |
| -amin -n |  |
| -mtime –n +n | 按文件更改时间查找 –n n天内， +n n天前 |
| -mmin -n | 查找在系统中最后n分钟里修改过的文件 |
| -newer file1 ! file2 | 查找更改时间比file1新比file2旧的文件 |
| -mount 或 -xdev | 查找文件时不跨越文件系统，不搜索其他文件系统中的目录 |
| -fstype | 查找位于某一类型文件系统中的文件，文件系统类型可在/etc/fstab中找到 |
| -follow | 跟随符号链接 |
| -cpio | 对匹配文件使用cpio命令备份到磁带设备文件中 |
| -empty | 查找在系统中为空的文件或者文件夹 |

|  |
| --- |
| 常用示例： |
| 1. 按文件名   find . –name “myfile” –print  find ~ -name “[A-Z]\*” –print 【使用了正则】  find . –name “\*” -print   1. 按权限数值查找   find . –perm 755 -print   1. 按文件属主或属组   find . –user ken –print  find . –group mygroup -print   1. 按时间范围查找   find / -mtime -5 –print 【5天内的文件，内部计算其实精确到秒，从现在往前五天】   1. 按文件类型查找   find ~ -type f –print 【普通文件】   1. 查找比某个文件新或旧的文件   技巧：可通过touch –t 05042121 locateFile 创建时间坐标文件  find . –newer file1 ! –newer file2 –print [比file1新，比file2旧]  【使用-exec执行 查看找到的文件列表详情】  find . –newer file1 ! –newer file2 –exec ls –l {} \; 【注意分号和反斜杠间无空格】   1. 根据文件大小   默认单位以block，加c表字节  find . –size +1000c –print 【大于1000字节，若改为 -300c，表小于300字节】   1. 在当前文件系统查找，不进入其他文件系统   find . –name “\*.xc” –mount -print   1. 使用-exec 或 –f 来执行shell命令   find . –type f –exec ls –l {} \;  find . –type f –mtime +5 –exec rm {} \;  find . –type f –mtime +5 –ok rm {} \;  二者区别是exec直接执行，ok还进行询问 【注意反斜扛和大括号之间的空格】 |

## 命令：uniq

|  |
| --- |
| Uniq |
| 从文本文件中去除或禁止重复行，一般uniq假定文件已分类排序，并且结果正确[sort –u唯一性选项去除所有的重复行]  Uniq中重复行指持续不断重复出现的行 |

|  |
| --- |
| 格式: |
| uniq –udc –f input-file output-file |

|  |
| --- |
| 选项： |
| -u 只显示不重复行 unique,唯一  -d 只显示有重复的行，每种重复显示一行  -I 忽略大小写 ???  -c打印每一重复行出现的次数【计数】  -f n为数字，前n个域被忽略 |

|  |
| --- |
| 常用示例 |
| 1.只显示非重复行  $uniq –u sortfile  2.提取非重复行到某个文件  $uniq –u sortfile result  3.只显示重复行  $uniq –d sortfile  4.打印重复行及其出现的次数  $uniq –c sortfile  5.忽略比较指定列  $uniq –f2 parts.txt  6.对文本文件做交集，并集&差集  cat filea fileb | sort | uniq > filec # 并集  cat filea fileb | sort | uniq –d > filec # 交集  cat filea fileb | sort | uniq –u > filec # 仅存在于filea 或仅存在于 fileb的 |