Linux标准I/O和管道

一，Linux的三种I/O设备

1.Linux为程序提供三种I/O设备

设备	**linux**使用数字表示某设备	默认动作
标准输入	使用0表示	默认接收来自终端窗口的输入
标准输出	使用1表示	标准输出默认将内容输出到终端窗口
标准错误	使用2表示	标准错误默认直接输出到终端窗口

- 使用**I/O**重定向技术可以改变上面三种设备的默认行为。

2.使用I/O重定向

- 把标准输出和标准错误重新定向到文件而不打印到终端

```
用法：
    命令 操作符号 文件名
        支持的操作符号包括：
            >  把STDOUT重定向到文件
            2> 把STDERR重定向到文件
            &> 把所有输出重定向到文件
```

```
#### >  把STDOUT重定向到文件
[root@centos8 /data  $]ls
1  2  3  333  add1.txt  add.txt  ASCII-WELCOME.txt  cal.log  hello  if
jdsfjllfj.log  log1  log2  log3  my_file_1.txt  one  passwd  passwdd
pattern.example  so  so.link
[root@centos8 /data  $]ls > ls.log
[root@centos8 /data  $]cat ls.log
1
2
3
333
add1.txt
add.txt
ASCII-WELCOME.txt
cal.log
hello
if
jdsfjllfj.log
log1
log2
```

```
log3
ls.log
my_file_1.txt
one
passwd
passwdd
pattern.example
so
so.link

####  2> 把STDERR重定向到文件
[root@centos8 /data  $]lls 2> error.log
[root@centos8 /data  $]cat error.log
bash: lls: command not found...
Similar command is: 'ls'

####  &> 把所有输出重定向到文件
[root@centos8 /data  #]ls /data/ /noshuchdir &> allerror.log
[root@centos8 /data  #]cat allerror.log
ls: cannot access '/noshuchdir': No such file or directory  # STDERR
/data/:
1
2
3
333
add1.txt
add.txt
allerror.log
ASCII-WELCOME.txt
cal.log
error.log
hello
if
jdsfjllfj.log
log1
log2
log3
ls.log
my_file_1.txt
one
passwd
passwdd
pattern.example
so
so.link
```

- 使用>符号会覆盖已有文件，若追加则需要用>>。
- 将标准输出和错误输出各自定向至不同文件如下：

```
COMMAND > /path/to/success.out 2> /path/to/error.out
```

- 合并标准输出和错误输出为同一个数据流并重定向到某个文件如下：

```
&>    覆盖重定向
&>>   追加重定向
COMMAND > /path/to/successanderror.out 2>&1 （注意顺序）
COMMAND >> /path/to/successanderror.out 2>&1
```

3.tr命令

- tr命令用于替换、删除或者压缩从标准输入的字符，并把处理的结果写到标准输出
- 用法：tr [OPTION]... SET1 [SET2]
- 选项：

```
-c, -C, --complement
       使用字符集 SET1 的补集
-d, --delete
       删除字符集SET1中的字符集
-s, --squeeze-repeats
      压缩最后指明的SET字符集中的每个连续重复的字符，使用单个该字符替换
-t, --truncate-set1
      截断SET1字符集到字符集SET2的长度
字符集使用字符串指明，大部分字符串代表自身。可被识别的字符集如下：
\NNN   character with octal value NNN (1 to 3 octal digits)
\\     backslash
\a     audible BEL
\b     backspace
\f     form feed
\n     new line
\r     return
\t     horizontal tab
\v     vertical tab
CHAR1-CHAR2
       all characters from CHAR1 to CHAR2 in ascending order
[CHAR*]
       in SET2, copies of CHAR until length of SET1
[CHAR*REPEAT]
       REPEAT copies of CHAR, REPEAT octal if starting with 0
[:alnum:]
       all letters and digits
[:alpha:]
       all letters
[:blank:]
       all horizontal whitespace
[:cntrl:]
       all control characters
[:digit:]
       all digits
[:graph:]
```

```
               all printable characters, not including space
       [:lower:]
               all lower case letters
       [:print:]
               all printable characters, including space
       [:punct:]
               all punctuation characters
       [:space:]
               all horizontal or vertical whitespace
       [:upper:]
               all upper case letters
       [:xdigit:]
               all hexadecimal digits
       [=CHAR=]
               all characters which are equivalent to CHAR
```

如果 -d 选项没有指定并且SET1和SET2都被指定，则输入的文本中属于SET1的字符将被转换为SET2所代表的字符。-t选项只用于转换字符的情况。

4.使用文件中的内容来代替终端输入给STDIN

- 使用使用 < 来重定向标准输入

```
某些命令能够接受从文件中重定向的标准输入STDIN
    tr ‘a-z’   ‘A-Z’< /etc/issue
    该命令会把/etc/issue中的小写字符都转换成大写字符
    tr –d abc < /etc/fstab
    删除fstab文件中的所有abc中任意字符
```

```
[root@centos8 /data  $]cat > log22
hello
this is a log file
^C
[root@centos8 /data  $]cat log2
log2   log22
[root@centos8 /data  $]cat log22
hello
this is a log file
```

- 可以使用文件来代替键盘的输入

```
    cat  < file1 >  file2  # log22的内容被cat命令视为标准输入，其写给标准输出，而标准输出被重定向给file2

    cat  < file1 >> file1 # 追加
[root@centos8 /data  $]cat  < log22 >  file2
[root@centos8 /data  $]cat file2
```

```
hello
this is a log file
```

- 把多行发送给**STDIN**

```
使用“<<终止词”命令从键盘把多行重导向给STDIN
    直到 终止词 位置的所有文本都发送给STDIN
    有时被称为“就地文本（here documents）”
mail -s "Please Call"  admin@magedu.com <<END
> Hi Wang
>
> Please give me a call when you get in. We may need
> to do some maintenance on server1.
>
> Details when you're on-site
> Zhang
> END
```

二，管道

1.管道

> 管道是**linux**或者其他类**unix**系统中用来将一个命令（程序、进程）的结果发送给另一个命令（程序、进程）以便进一步处理的一种重定向技术。类**Unix/Linux**系统允许某个命令的标准输出被定向到另一个命令的输入。使用管道符‘**|**’来实现。

2.使用管道

```
使用管道的格式
    命令1 | 命令2 | 命令3 | …
    将命令1的STDOUT发送给命令2的STDIN，命令2的STDOUT发送到命令3的
    STDIN和STDERR默认不能通过管道转发，可利用2>&1 或 |& 实现
使用管道可以组合多种工具的功能
    ls  | tr ‘a-z’ ‘A-Z
```

- 注意注意注意：最后一个命令会在当前**shell**进程的子**shell**进程中执行

3.结合管道实现一些小功能

> less ： 一页一页地查看输入
>
> > ls -l /etc | less

> mail： 通过电子邮件发送输入
>
> > echo "hello email" | mail -s "hello" steve@example.com

> bc：算术运算
>
> > echo "2^3" |bc

4.管道中 - 符号

```
示例：将 /home 里面的文件打包，但打包的数据不是记录到文件，而是传送到 stdout，
经过管道后，将 tar -cvf - /home 传送给后面的 tar -xvf - ，后面的这个 - 则是取
前一个命令的 stdout， 因此，就不需要使用临时file了

    tar -cvf - /home | tar -xvf  -
```

5.结合tee命令重定向到多个目标

```
用法：
    命令1 | tee [-a ] 文件名 | 命令2
        把命令1的STDOUT保存在文件中，做为命令2的输入
        -a 追加
    作用用：
        保存不同阶段的输出
        复杂管道的故障排除
        同时查看和记录输出
```

三，练习

```
1、将/etc/issue文件中的内容转换为大写后保存至/tmp/issue.out文件中
    cat /etc/issue | tr 'a-z' 'A-Z' > /tmp/issue

2、将当前系统登录用户的信息转换为大写后保存至/tmp/who.out文件中
    whoami | tr 'a-z' 'A-Z' > /tmp/who.out

3、一个linux用户给root发邮件，要求邮件标题为”help”，邮件正文如下：
     Hello, I am 用户名,The system version is here,please help me to  check
it ,thanks!
操作系统版本信息
    mail -s help root<<EOF
Hello,I am `whoami`,The system version is here ,please help me check it, Thanks!
`cat /etc/redhat-release`
EOF

4、将/root/下文件列表，显示成一行，并文件名之间用空格隔开
    ls -1 | tr '\n' ' '

5、计算1+2+3+...+99+100的总和
    echo {1..100} | tr ' ' '+' | bc
    seq -s+ 100 | bc
```

6、删除Windows文本文件中的回车字符 ，即“\r”

```
echo file.txt | tr '\r' ''
```

7、处理字符串“xt.,l 1 jr#!$mn 2 c*/fe 3 uz 4”，只保留其中的数字和空格

```
echo "xt.,l 1 jr#!$mn 2 c*/fe 3 uz 4" | tr -dc "[0-9]|[[:space:]]"
```

8、将PATH变量每个目录显示在独立的一行

```
echo $PATH | tr ':' '\n'
```

9、将指定文件中0-9分别替代成a-j

```
cat file1 | tr '[0-9]' '[a-j]'
```

10、将文件/etc/centos-release中每个单词（由字母组成）显示在独立一行，并无空行

```
cat /etc/redhat-release  | tr -s ' ' '\n'
```