后将结果输出到屏幕上。

也可以对标准输入和输出进行重定位，因为这种情况通常会很有用。对标准输入进行重定位的语法使用一个小于号（<）加上紧接的一个输入文件名。类似的，标准输出可以通过一个大于号（>）进行重定位。允许在一个命令中对两者同时进行重定位。比如，下面的命令：

```
sort <in >out
```

使得sort从文件in中得到输入，并把结果输出到文件out中。由于标准错误没有被重定位，因此所有的错误信息会输出到屏幕中。一个从标准输入中读取数据，对数据进行某种处理，然后输出到标准输出的程序称为过滤器（filter）。

考虑下面一条包括三条独立命令的命令行：

```
sort <in >temp; head -30 <temp; rm temp
```

首先它运行sort，从in得到输入然后将结果输出到temp中。完成后，shell运行head，令其将temp的前30行内容输出到标准输出中，默认为终端。最后，临时文件temp被删除。

常常有把命令行中第一个程序的输出作为下一个程序的输入这种情况。在上面的例子中，我们使用temp文件来保存这个输出。然而，Linux提供了一种更简单的方法来达到相同的结果。在命令行

```
sort <in | head -30
```

中，竖杠，也常被称为管道符（pipe symbol），告诉程序从sort中得到输出并且将其作为输入传给head，由此消除了创建、使用和删除一个临时文件的过程。由管道符连接起来的命令，称为一个管线（pipeline），可以包含任意多的命令。一个由四个部分组成的管线如下所示：

```
grep ter *.t | sort | head -20 | tail -5 >foo
```

这里所有以.t结尾的文件中包含“ter”的行被写到标准输出中，然后被排序。这些内容的前20行被head选择出来并传给tail，它又将最后5行（也即排完序的列表中的第16到20行）传给foo。这个例子显示了Linux是如何提供了一组各负责一项任务的基本单元（一些过滤器）和一个几乎可以用无穷的方式把它们组合起来的机制。

Linux是一种通用多道程序设计系统。一个用户可以同时运行多个程序，每一个作为一个独立的进程存在。在shell中，后台运行一个程序的语法是在原本命令后加一个“&”。因此，

```
wc -l <a >b &
```

运行字数统计程序wc，来统计输入文件a中的行数（-l标志），并将结果输出到b中，不过整个过程都在后台运行。命令一被输入，shell输出提示符就可以接收并处理下一条命令。管线也可以在后台中运行，比如下面的指令：

```
sort <x | head &
```

多个管线也可以同时在后台中运行。

可以把一系列shell命令放到一个文件中，然后将此文件作为shell的输入来运行。第二个shell按照顺序处理这些命令，和处理从键盘输入的命令一样。包含shell命令的文件称为shell脚本。shell脚本可以给shell的变量赋值，然后过一段时间再读取这些变量。shell脚本也可以包含参数，同时使用if、for、while和case等结构。因此，一个shell脚本实际上是一个由shell语言编写的程序。Berkeley C shell是另一种shell，它的设计目标是使得shell脚本（以及一般意义上的命令语言）在很多方面看上去和C程序相似。由于shell也只是一个用户程序，其他人也设计并发行过很多不同的shell。

10.2.4 Linux应用程序

Linux的命令行（shell）用户界面包含大量的标准应用程序。这些程序可以大致分成以下6类：

1) 文件和目录操作命令。

2) 过滤器。

3) 程序设计工具，如编辑器和编译器。

4) 文档处理。