

*Linux*环境及开发工具应用实践

-系统操作及管理

zhaofang@email.buptsse.cn



软件学院 赵方

目录

1. 特殊变量和控制

2. **Dot**命令和**eval**命令

3. **Shell**程序的调试

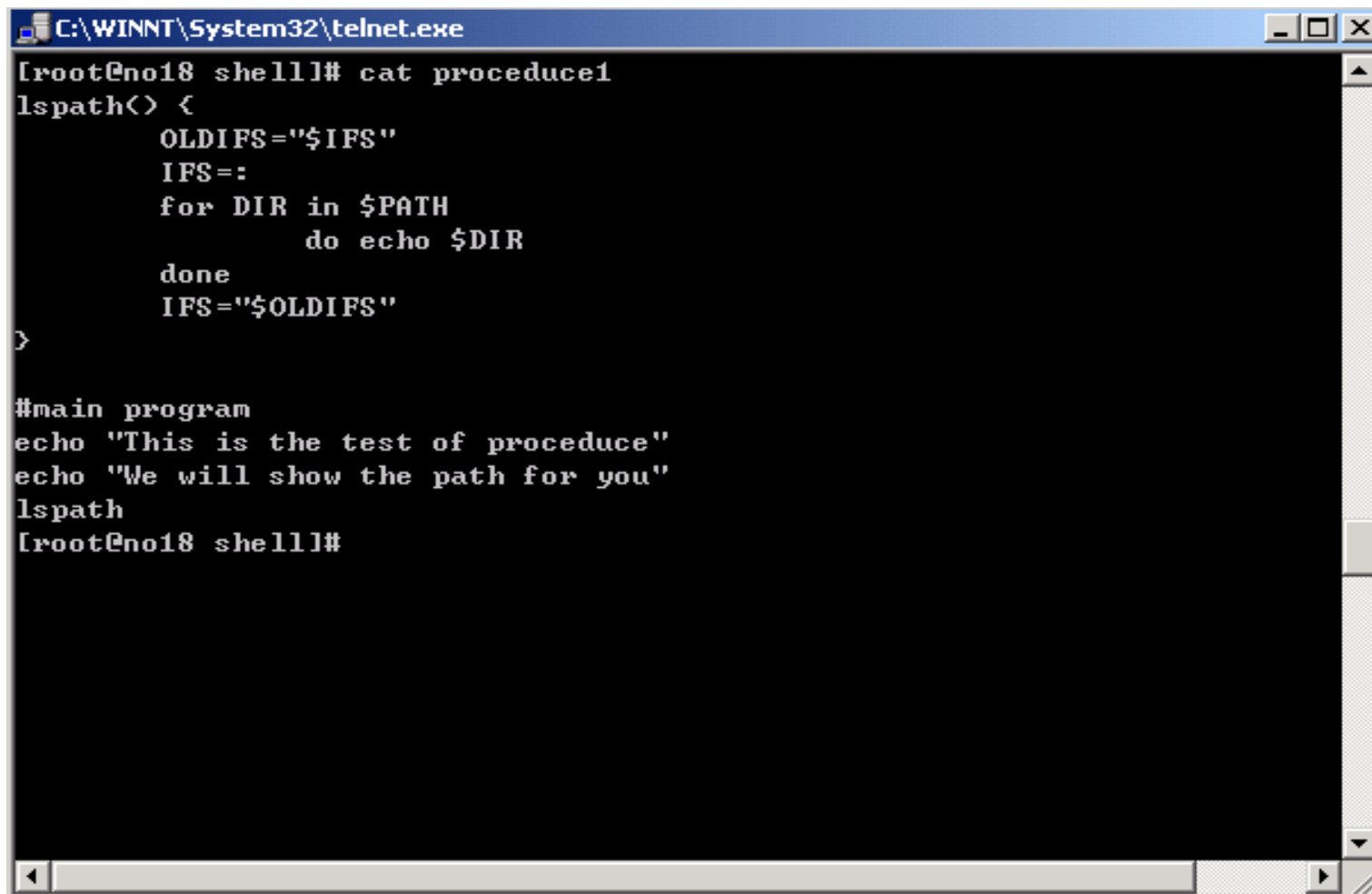
4. 特殊**shell**工具的使用

IFS和循环

- ❖ **IFS**: 是shell的内部域分隔符, 它的值为空格符、跳格符、“:”号或换行符。
- ❖ 分析词表的命令: 如**read**、**set**、**for**和**select**命令可以把它动作一个词来分析。
- ❖ 也可以将一个变量里使用另外一个分隔符。

函数的使用

❖ 实例：

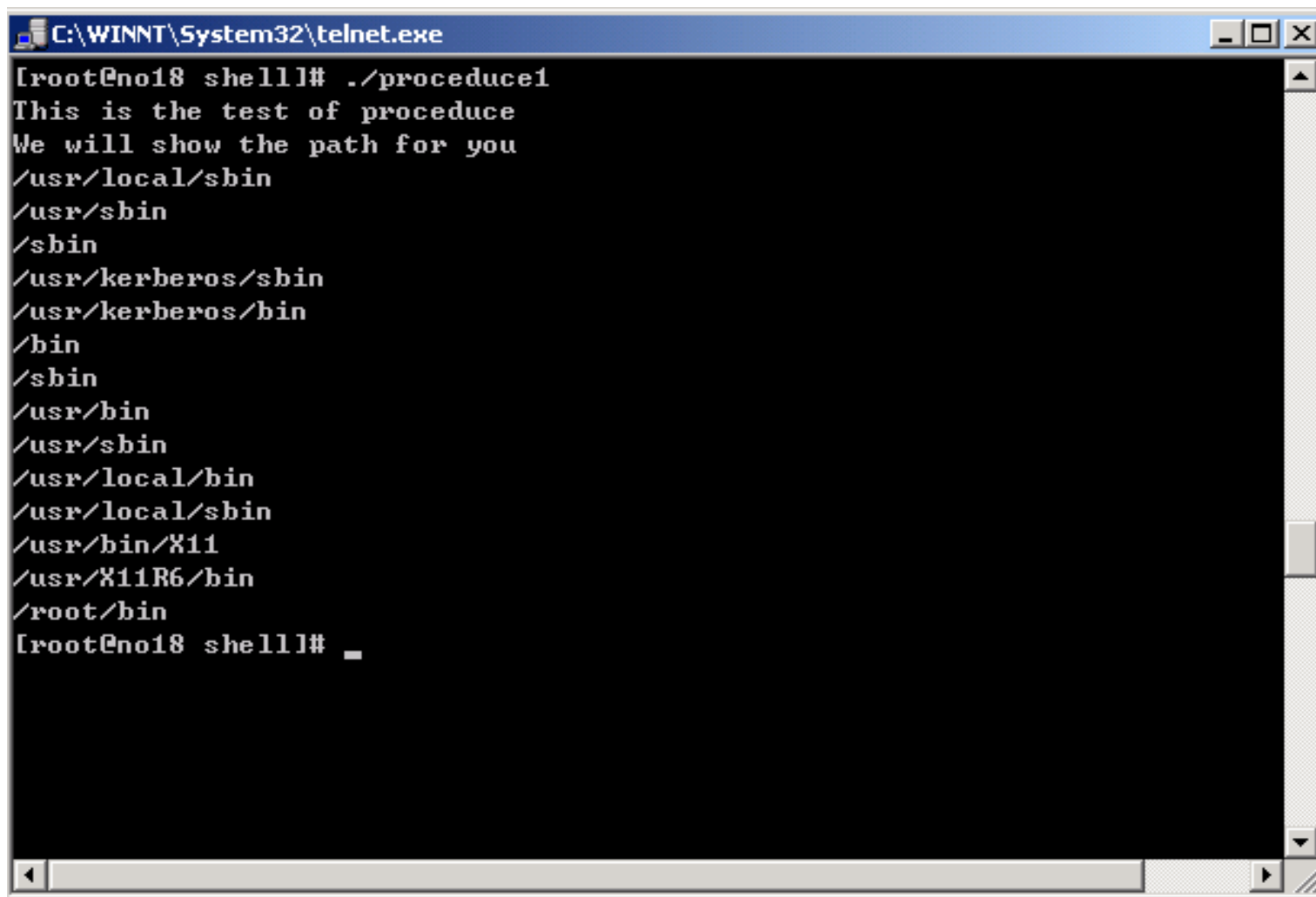


```
C:\WINNT\System32\telnet.exe
[root@no18 shell]# cat procedure1
lspath() {
    OLDIFS="$IFS"
    IFS=:
    for DIR in $PATH
    do echo $DIR
    done
    IFS="$OLDIFS"
}

#main program
echo "This is the test of procedure"
echo "We will show the path for you"
lspath
[root@no18 shell]#
```

函数的使用

❖ 实例：



A screenshot of a Windows command prompt window titled "C:\WINNT\System32\telnet.exe". The window shows a telnet session with a root user on a host named "no18". The user enters the command `./proceduce1`. The script outputs a series of directory paths: `This is the test of proceduce`, `We will show the path for you`, `/usr/local/sbin`, `/usr/sbin`, `/sbin`, `/usr/kerberos/sbin`, `/usr/kerberos/bin`, `/bin`, `/sbin`, `/usr/bin`, `/usr/sbin`, `/usr/local/bin`, `/usr/local/sbin`, `/usr/bin/X11`, `/usr/X11R6/bin`, and `/root/bin`. The prompt returns to `[root@no18 shell]#` with a cursor.

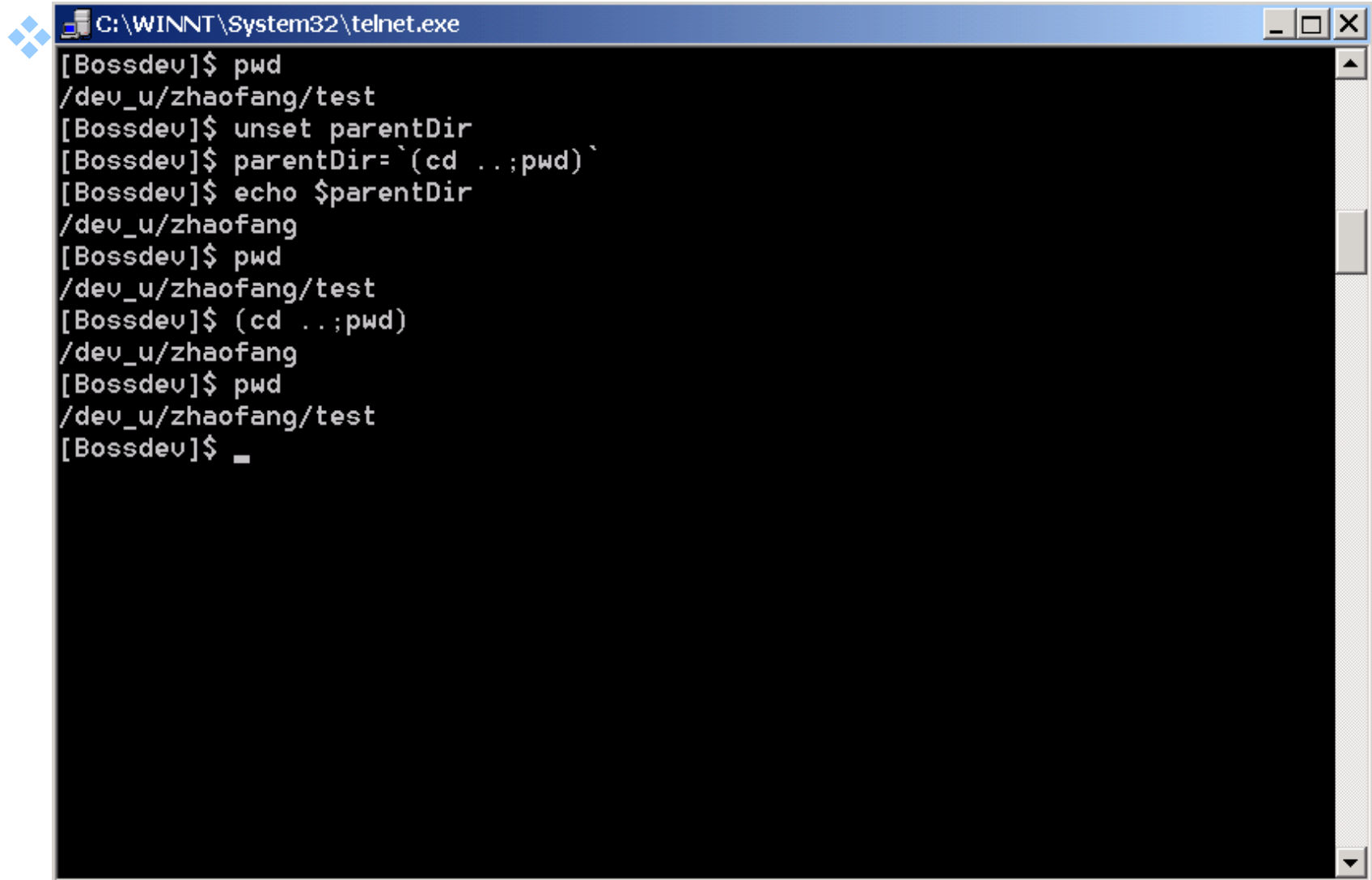
```
C:\WINNT\System32\telnet.exe
[root@no18 shell]# ./proceduce1
This is the test of proceduce
We will show the path for you
/usr/local/sbin
/usr/sbin
/sbin
/usr/kerberos/sbin
/usr/kerberos/bin
/bin
/sbin
/usr/bin
/usr/sbin
/usr/local/bin
/usr/local/sbin
/usr/bin/X11
/usr/X11R6/bin
/root/bin
[root@no18 shell]#
```

控制shell

❖ 1、创建复合命令

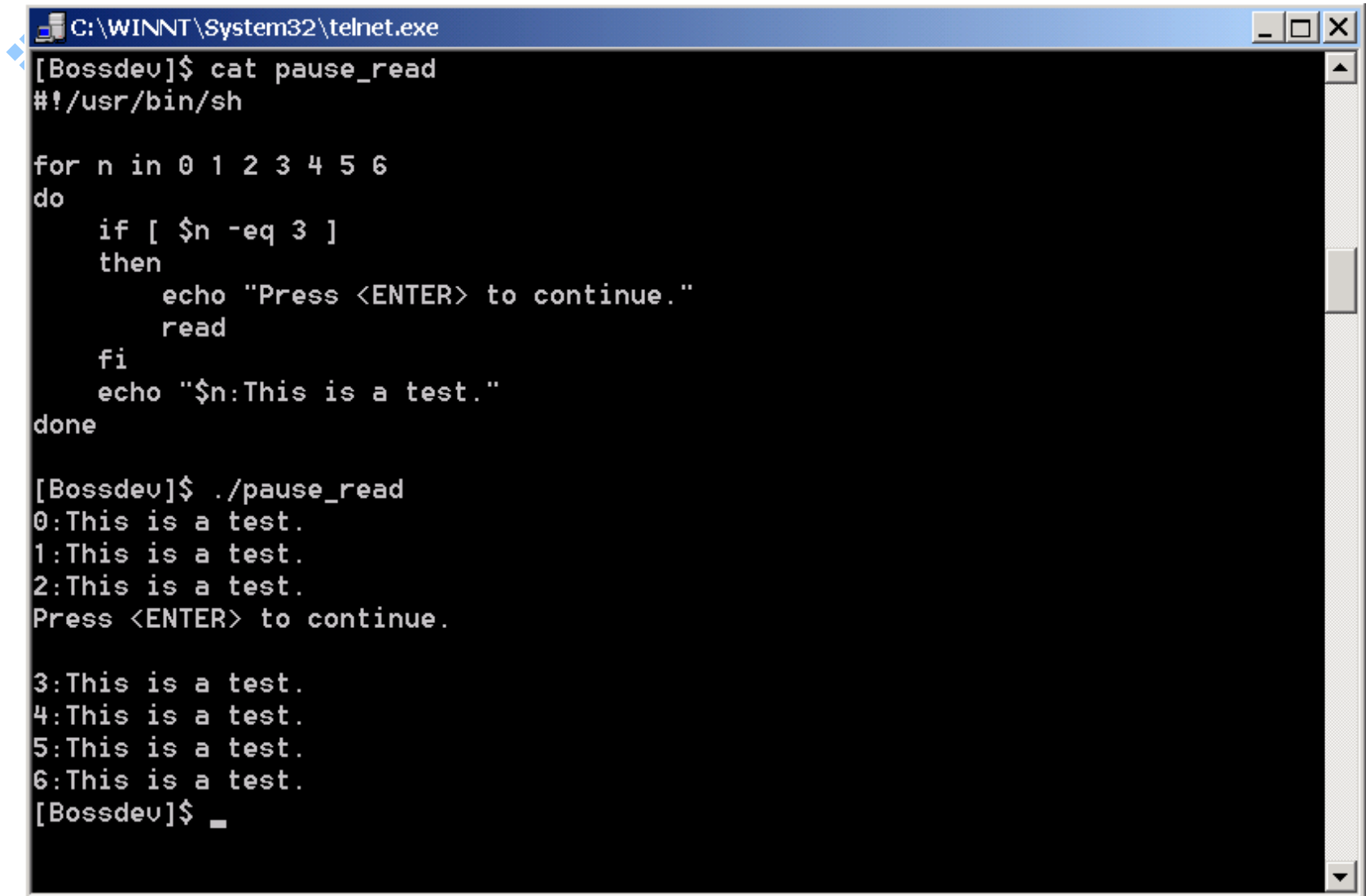
- ❖ 复合命令可以像单个命令那样，执行命令表。表中最后一个命令的退出值作为整个复合命令的退出值。
- ❖ **(list)** 在同一个子shell内部执行该表所有命令。改变环境的命令执行完成后，不留下任何影响。
- ❖ **{ list }** 在当前shell内部执行该表所有命令，在该表执行完成后变量赋值和环境修改的内容仍保留其影响。

控制shell



```
C:\WINNT\System32\telnet.exe
[Bossdev]$ pwd
/dev_u/zhaofang/test
[Bossdev]$ unset parentDir
[Bossdev]$ parentDir=`(cd ../pwd)`
[Bossdev]$ echo $parentDir
/dev_u/zhaofang
[Bossdev]$ pwd
/dev_u/zhaofang/test
[Bossdev]$ (cd ../pwd)
/dev_u/zhaofang
[Bossdev]$ pwd
/dev_u/zhaofang/test
[Bossdev]$ _
```

控制shell



```
C:\WINNT\System32\telnet.exe
[Bossdev]$ cat pause_read
#!/usr/bin/sh

for n in 0 1 2 3 4 5 6
do
    if [ $n -eq 3 ]
    then
        echo "Press <ENTER> to continue."
        read
    fi
    echo "$n:This is a test."
done

[Bossdev]$ ./pause_read
0:This is a test.
1:This is a test.
2:This is a test.
Press <ENTER> to continue.

3:This is a test.
4:This is a test.
5:This is a test.
6:This is a test.
[Bossdev]$
```


局部变量

在某一局部特定环境下使用的变量。

☞ 注册**Shell**在接受到用户输入的命令（非内部命令）后，通常派生出一个子**Shell**，由此子**Shell**负责解释执行该命令。

☞ 子**Shell**有自己的运行环境和变量，这些变量仅在子**Shell**的范围内的特定环境下才能使用。

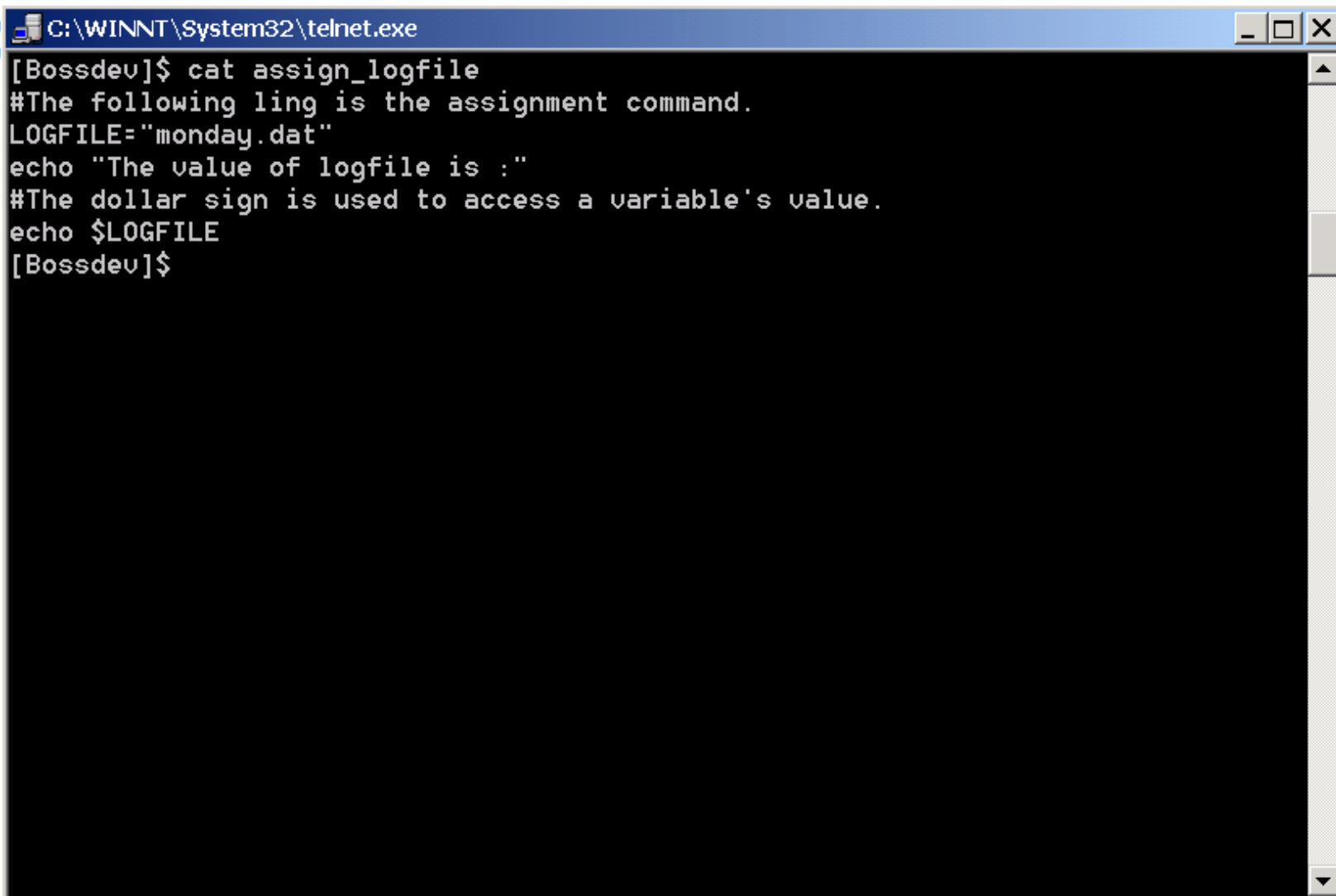
☞ 子**Shell**不能存取由父**Shell**设置的局部变量，也不能改变父**Shell**的变量值。

局部变量（续）

例1: **#** cat vartest1
 echo :\$x:
 # x=100
 # vartest1
 ::

例2: **#** cat vartest2
 x=60
 echo :\$x:
 # x=10
 # vartest2
 :60:
 # echo \$x
 10

一个shell脚本的环境



```
C:\WINNT\System32\telnet.exe
[Bossdev]$ cat assign_logfile
#The following ling is the assignment command.
LOGFILE="monday.dat"
echo "The value of logfile is :\"
#The dollar sign is used to access a variable's value.
echo $LOGFILE
[Bossdev]$
```

一个shell脚本的环境

C:\WINNT\System32\telnet.exe

[Bossdev]\$./assign_logfile

The value of logfile is :

monday.dat

[Bossdev]\$ echo \$LOGFILE

sh: LOGFILE: 参数没有设置。

[Bossdev]\$ _

全局变量

全局变量是一种特殊的变量，可以被任何运行的子Shell来引用。全局变量通过**export**命令来定义，格式如下：

export variables

其中 **variables** 是要变成全局变量的变量表名。

- ☞ 一旦变量被定义为全局变量，则对于以后的所有子Shell来说这些都是全局变量；
- ☞ 子Shell中无法改变全局变量的值；
- ☞ 若在子Shell中改变全局变量的值，实际是对全局变量的副本进行更改，不影响全局变量值；
- ☞ 子Shell中局部变量的使用优先于全局变量。

全局变量（续）

例：

```
# export g_var
# g_var="GLOBAL"
# cat test_var
export g_var l_var
g_var="sub_shell:g_var"
l_var="sub_shell:l_var"
echo $g_var $l_var
# test_var
sub_shell:g_var sub_shell:l_var
# echo $g_var :$l_var:
GLOBAL ::
#
```

局部变量和全局变量作用域

☆ 任何没有用**export**命令定义过的变量是局部变量，子**Shell**不能存取父**Shell**的局部变量；

⌚ 子**Shell**中可以存取和修改父**Shell**的全局变量，但这种修改对于父**Shell**全局变量没有任何影响；

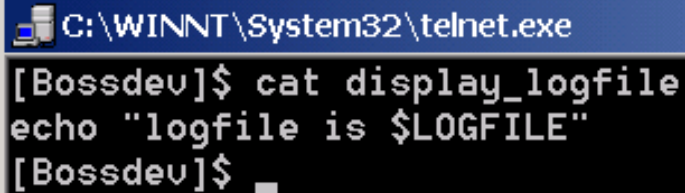
⌚ 在子**Shell**中用**export**命令定义的全局变量和对此变量的修改对父**Shell**变量没有影响；

⌚ 全局变量保持它的全局性，不仅能直接传递给它的子**Shell**，而且子**Shell**还能将它传递给子**Shell**的子**Shell**；

⌚ 在对变量赋值之前和之后的任何时候可以将该变量转换成全局变量。

一个shell脚本的环境

❖ 用export命令可以影响子shell的环境。



A screenshot of a telnet session window. The title bar shows the path 'G:\WINNT\System32\telnet.exe'. The prompt is '[Bossdev]\$'. The user has entered 'cat display_logfile' and 'echo "logfile is \$LOGFILE"'. The prompt is now '[Bossdev]\$ _'.

```
G:\WINNT\System32\telnet.exe
[Bossdev]$ cat display_logfile
echo "logfile is $LOGFILE"
[Bossdev]$ _
```


一个shell脚本的环境



C:\WINNT\System32\telnet.exe

```
[Bossdev]$ cat export_logfile
#The following line is the assignment command.
LOGFILE="monday.dat"
#call the display script before using export
echo "Before Export:"
./display_logfile
export LOGFILE
#call the display script after export
echo "After Export:"
./display_logfile
[Bossdev]$
```

一个shell脚本的环境

C:\WINNT\System32\telnet.exe

```
[Bossdev]$ unset LOGFILE
```

```
[Bossdev]$ ./export_logfile
```

```
Before Export:
```

```
logfile is
```

```
After Export:
```

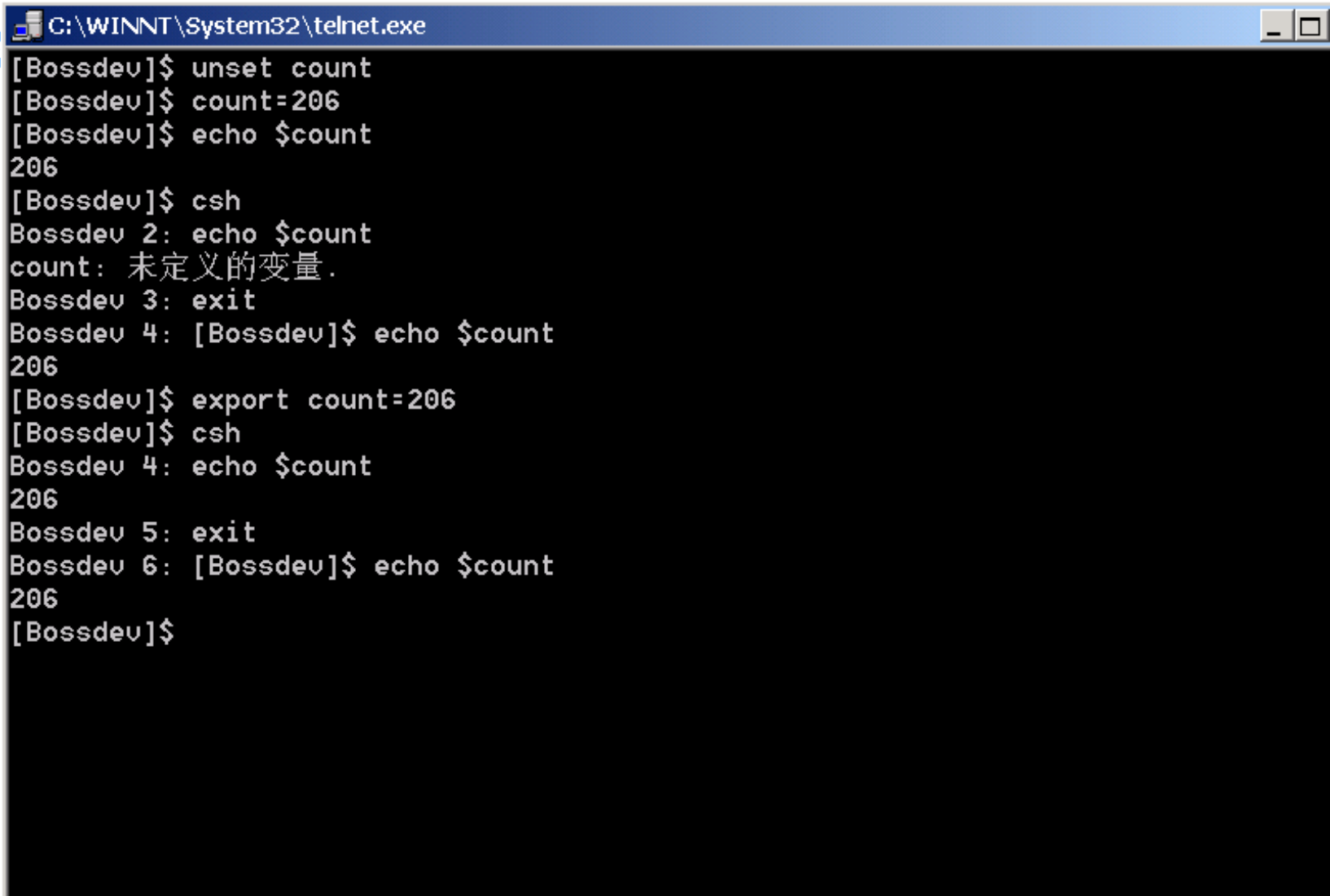
```
logfile is monday.dat
```

```
[Bossdev]$ _
```

一个shell脚本的环境

- ❖ **unset**命令从环境中删除一个变量。
- ❖ 从例子中我们看到**display_logfile**脚本第二次运行时显示出**LOGFILE**变量的值。从而可以看出**export**命令的所产生的效果。
- ❖ 它让子 shell (**display_logfile**的第二次执行) 访问了变量**LOGFILE**。

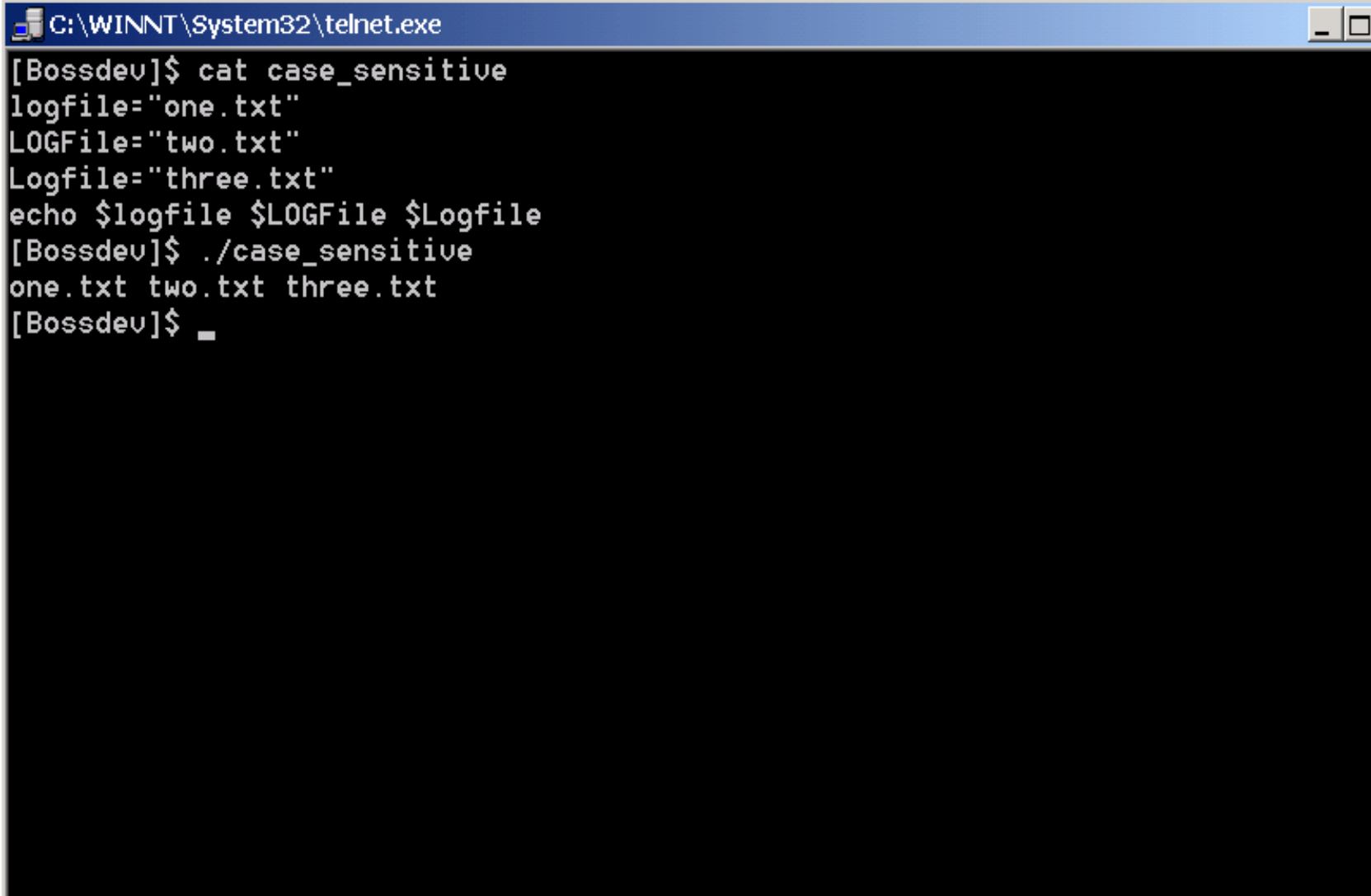
一个shell脚本的环境



A screenshot of a Windows command prompt window titled "C:\WINNT\System32\telnet.exe". The window shows a telnet session with a user named "Bossdev". The session demonstrates variable scope by setting a variable "count" to 206, spawning a subshell with "csh", and then exiting the subshell multiple times. The variable "count" is only accessible within the subshell when it is exported.

```
C:\WINNT\System32\telnet.exe
[Bossdev]$ unset count
[Bossdev]$ count=206
[Bossdev]$ echo $count
206
[Bossdev]$ csh
Bossdev 2: echo $count
count: 未定义的变量.
Bossdev 3: exit
Bossdev 4: [Bossdev]$ echo $count
206
[Bossdev]$ export count=206
[Bossdev]$ csh
Bossdev 4: echo $count
206
Bossdev 5: exit
Bossdev 6: [Bossdev]$ echo $count
206
[Bossdev]$
```

Shell变量的大小写



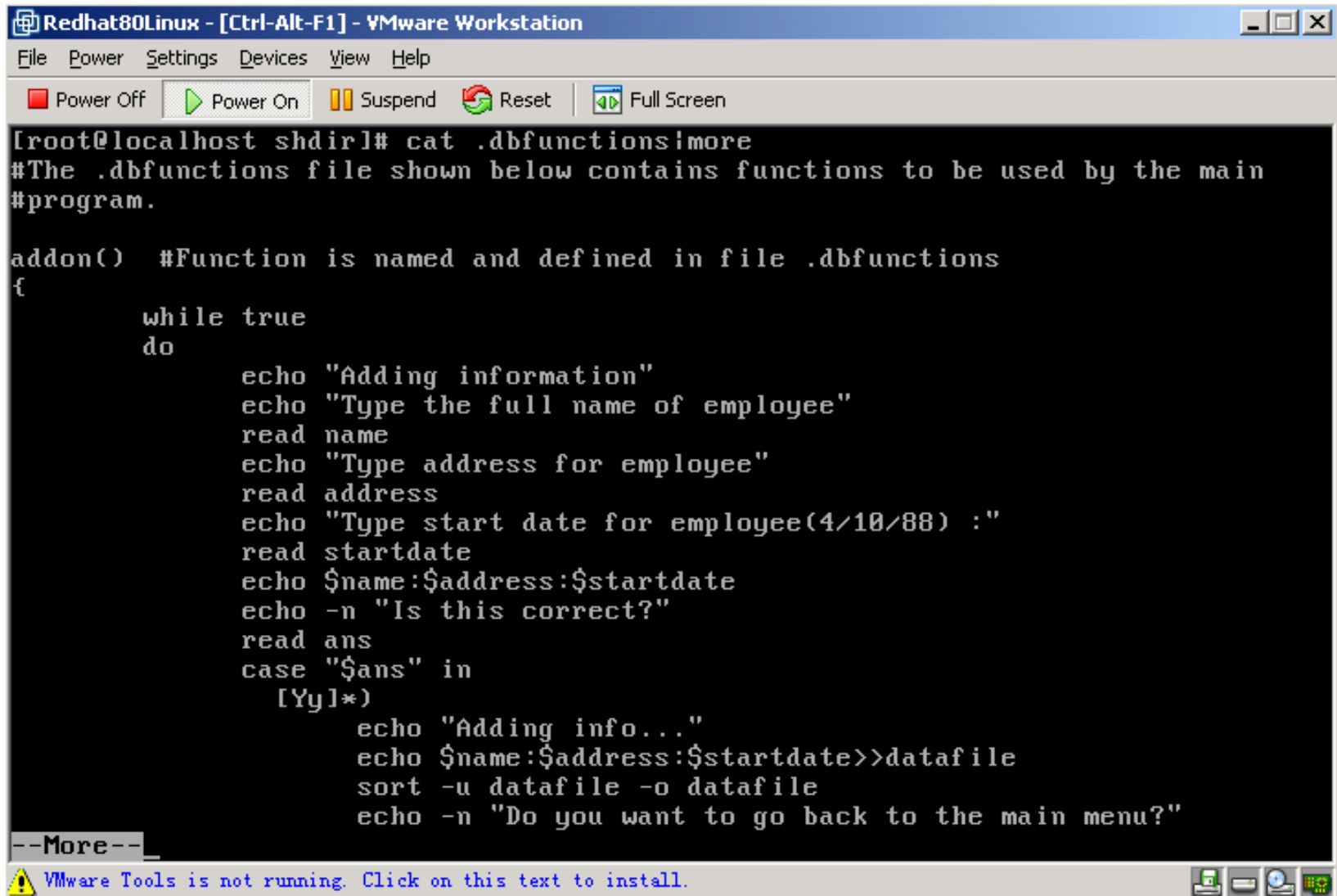
A screenshot of a Windows telnet session window. The title bar shows the path 'C:\WINNT\System32\telnet.exe'. The window has a black background with white text. The user 'Bossdev' is at the prompt. They run 'cat case_sensitive' which shows three lines of variable assignments: 'logfile="one.txt"', 'LOGFile="two.txt"', and 'Logfile="three.txt"'. Then they run 'echo \$logfile \$LOGFile \$Logfile' which outputs 'one.txt two.txt three.txt'. Finally, they run './case_sensitive' which also outputs 'one.txt two.txt three.txt'.

```
C:\WINNT\System32\telnet.exe
[Bossdev]$ cat case_sensitive
logfile="one.txt"
LOGFile="two.txt"
Logfile="three.txt"
echo $logfile $LOGFile $Logfile
[Bossdev]$ ./case_sensitive
one.txt two.txt three.txt
[Bossdev]$
```

函数和dot命令

- ❖ 通常在`.profile`文件里定义了一些函数，以便当用户注册时，它们被定义。如果我们定义的函数不在`.profile`文件中，则无法直接导出函数。这时，我们可以将定义的函数存储在一个文件中，然后当需要调用该函数时，使用`dot`命令，并和该文件名一起连用，来激活该文件内部调用的函数。

函数和dot命令



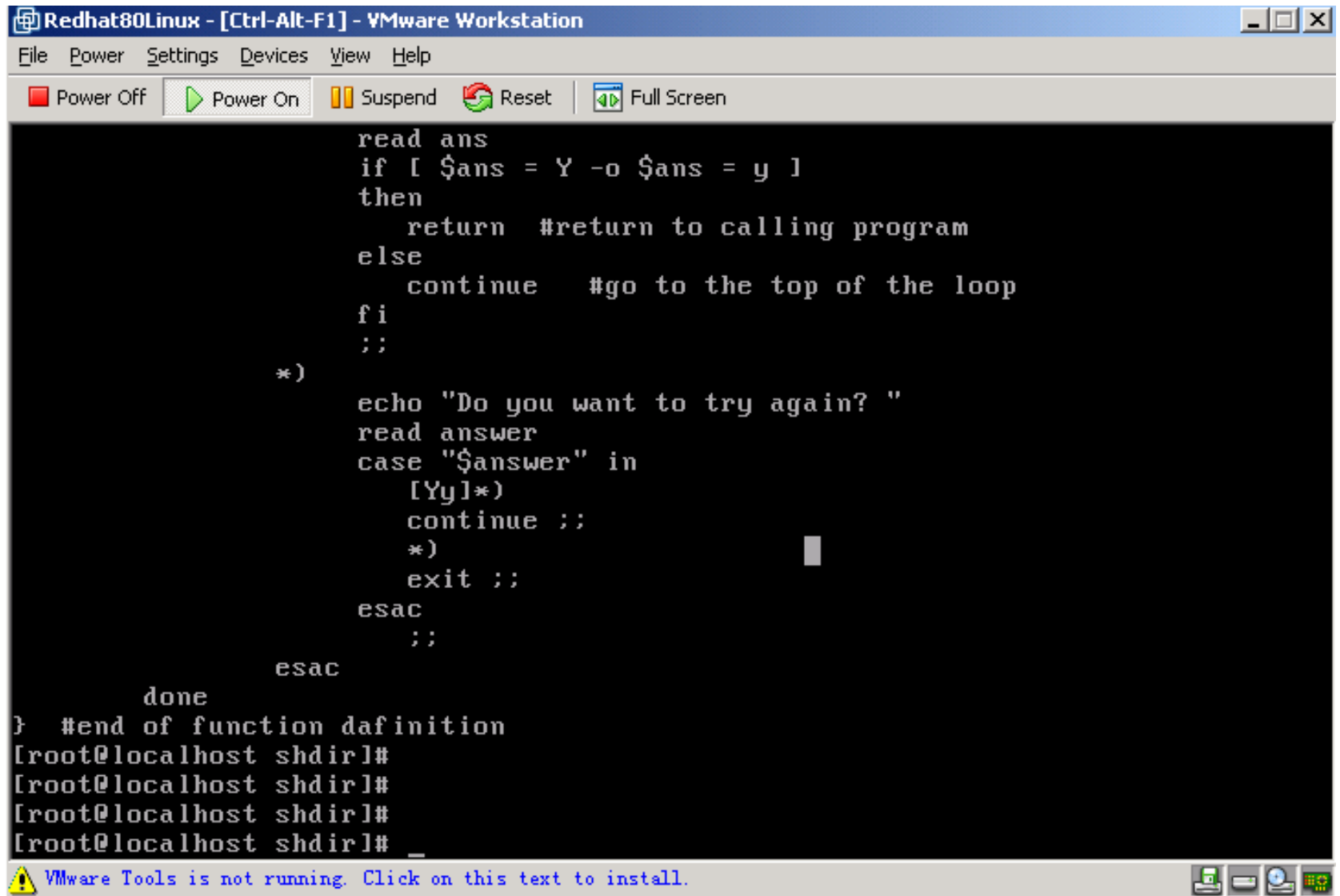
```
Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation
File Power Settings Devices View Help
Power Off Power On Suspend Reset Full Screen

[root@localhost shdir]# cat .dbfunctions;more
#The .dbfunctions file shown below contains functions to be used by the main
#program.

addon() #Function is named and defined in file .dbfunctions
{
    while true
    do
        echo "Adding information"
        echo "Type the full name of employee"
        read name
        echo "Type address for employee"
        read address
        echo "Type start date for employee(4/10/88) :"
        read startdate
        echo $name:$address:$startdate
        echo -n "Is this correct?"
        read ans
        case "$ans" in
            [Yy]*)
                echo "Adding info..."
                echo $name:$address:$startdate>>datafile
                sort -u datafile -o datafile
                echo -n "Do you want to go back to the main menu?"
        esac
    done
}

--More--
! VMware Tools is not running. Click on this text to install.
```

函数和dot命令



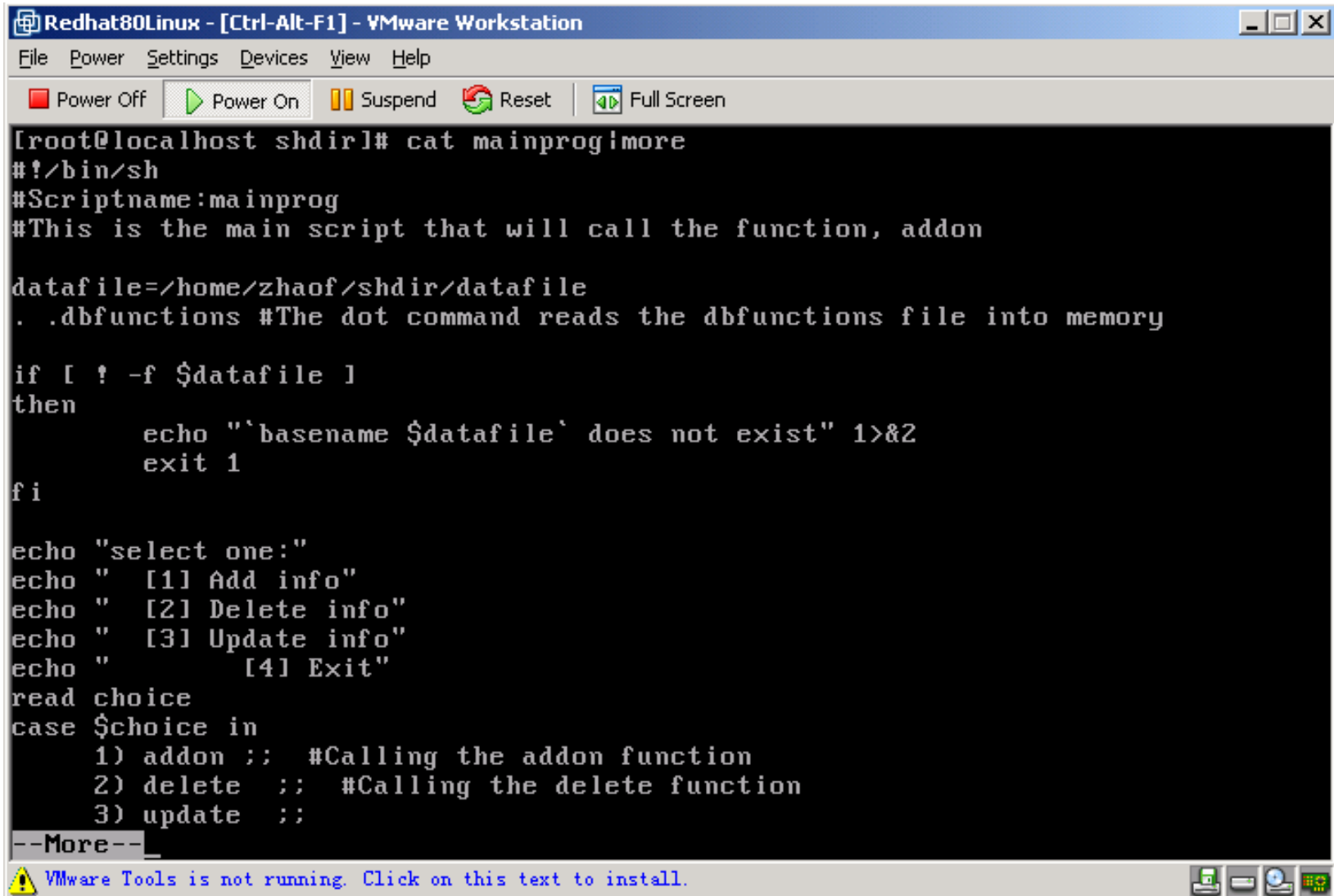
```
Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation
File Power Settings Devices View Help
Power Off Power On Suspend Reset Full Screen

    read ans
    if [ $ans = Y -o $ans = y ]
    then
        return #return to calling program
    else
        continue #go to the top of the loop
    fi
    ;;
*)
    echo "Do you want to try again? "
    read answer
    case "$answer" in
        [Yy]*)
            continue ;;
        *)
            exit ;;
    esac
    ;;
esac

done
} #end of function definition
[root@localhost shdir]#
[root@localhost shdir]#
[root@localhost shdir]#
[root@localhost shdir]#
_

VMware Tools is not running. Click on this text to install.
```


函数和dot命令



```
Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation
File Power Settings Devices View Help
Power Off Power On Suspend Reset Full Screen

[root@localhost shdir]# cat mainprog|more
#!/bin/sh
#Scriptname:mainprog
#This is the main script that will call the function, addon

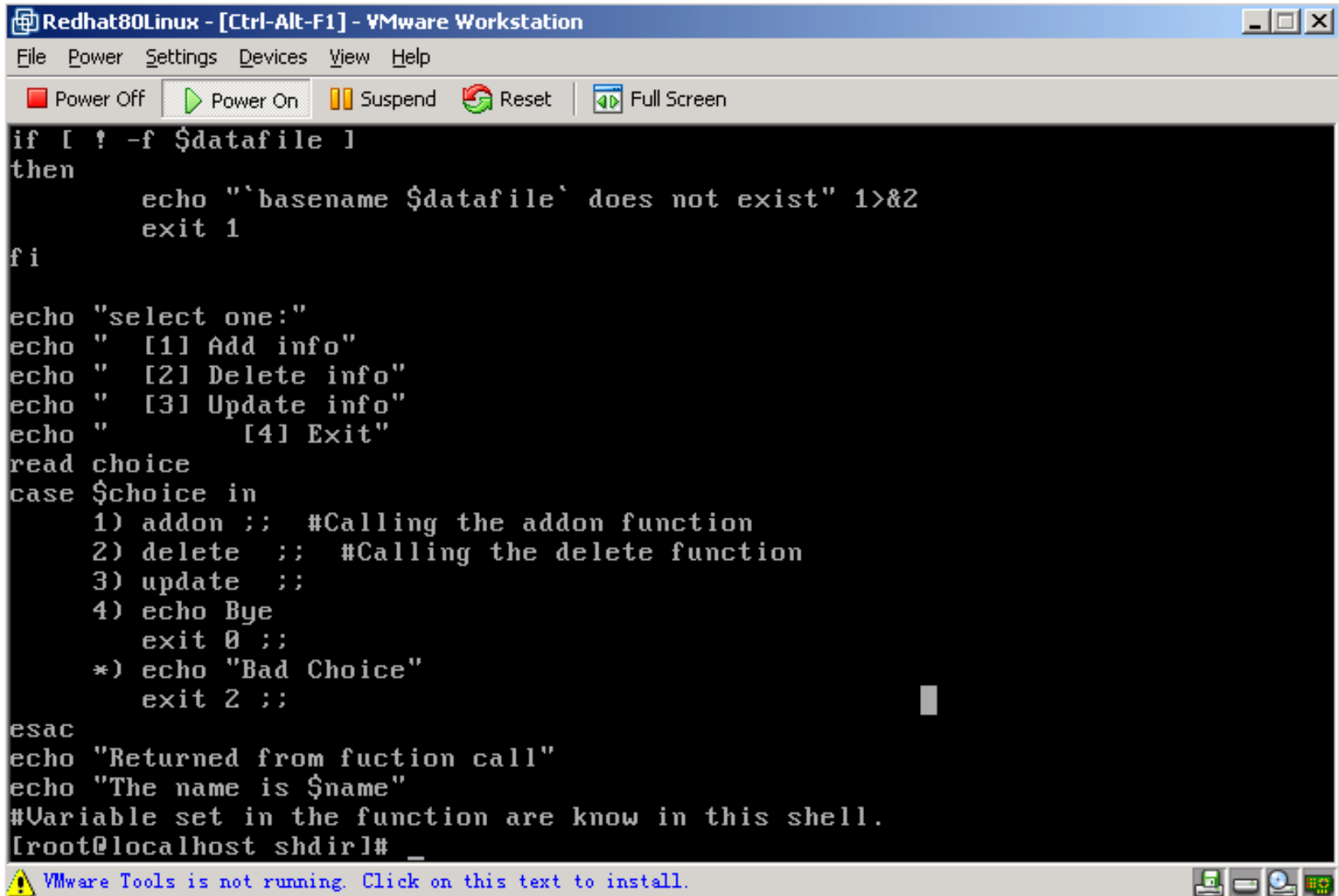
datafile=/home/zhaof/shdir/datafile
. .dbfunctions #The dot command reads the dbfunctions file into memory

if [ ! -f $datafile ]
then
    echo "`basename $datafile` does not exist" 1>&2
    exit 1
fi

echo "select one:"
echo "  [1] Add info"
echo "  [2] Delete info"
echo "  [3] Update info"
echo "      [4] Exit"
read choice
case $choice in
    1) addon ;; #Calling the addon function
    2) delete ;; #Calling the delete function
    3) update ;;
--More--_

! VMware Tools is not running. Click on this text to install.
```

函数和dot命令



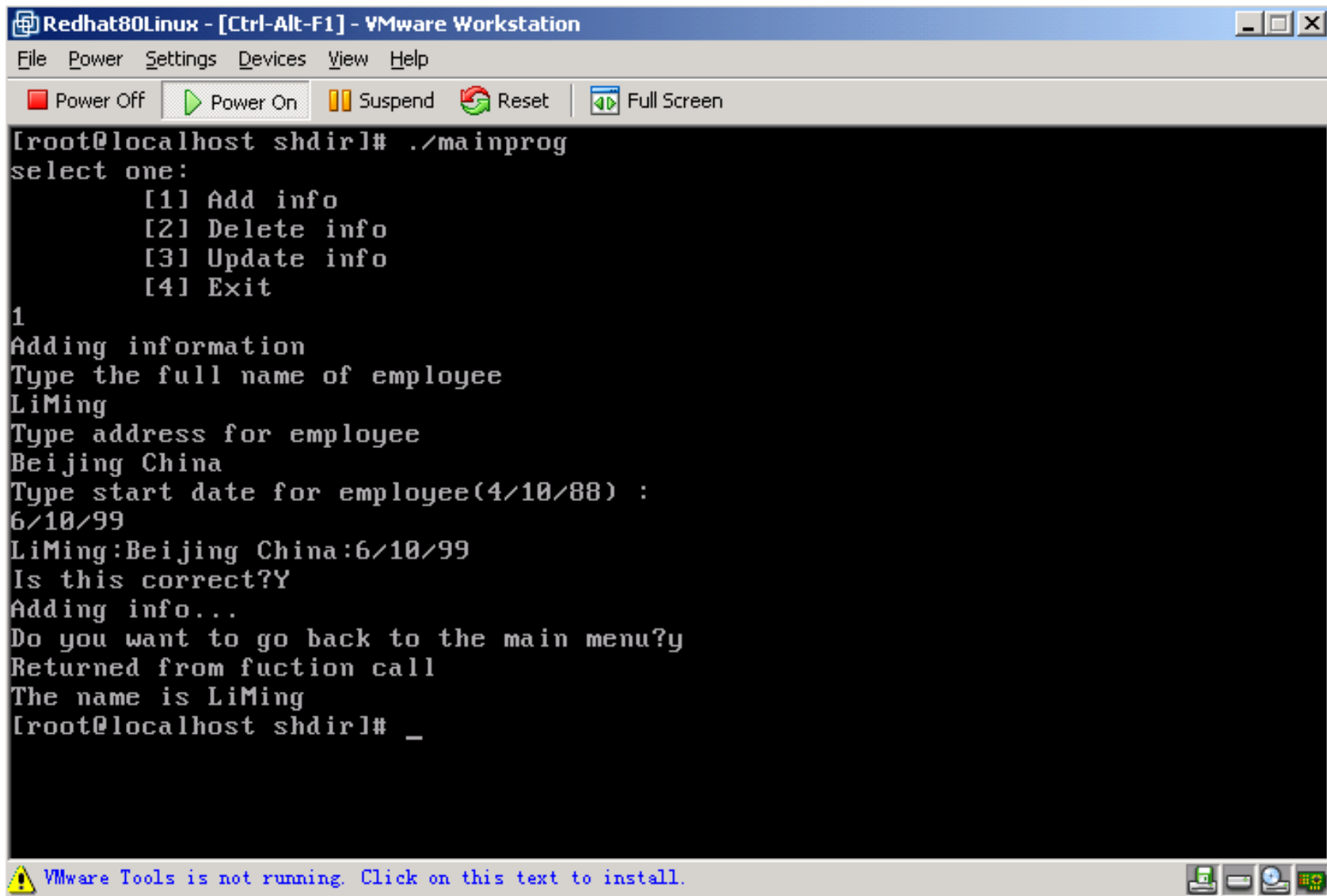
```
Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation
File Power Settings Devices View Help
Power Off Power On Suspend Reset Full Screen

if [ ! -f $datafile ]
then
    echo "`basename $datafile` does not exist" 1>&2
    exit 1
fi

echo "select one:"
echo "  [1] Add info"
echo "  [2] Delete info"
echo "  [3] Update info"
echo "  [4] Exit"
read choice
case $choice in
    1) addon ;; #Calling the addon function
    2) delete ;; #Calling the delete function
    3) update ;;
    4) echo Bye
        exit 0 ;;
    *) echo "Bad Choice"
        exit 2 ;;
esac
echo "Returned from fuction call"
echo "The name is $name"
#Variable set in the function are know in this shell.
[root@localhost shdir]# _
```

! VMware Tools is not running. Click on this text to install.

函数和dot命令

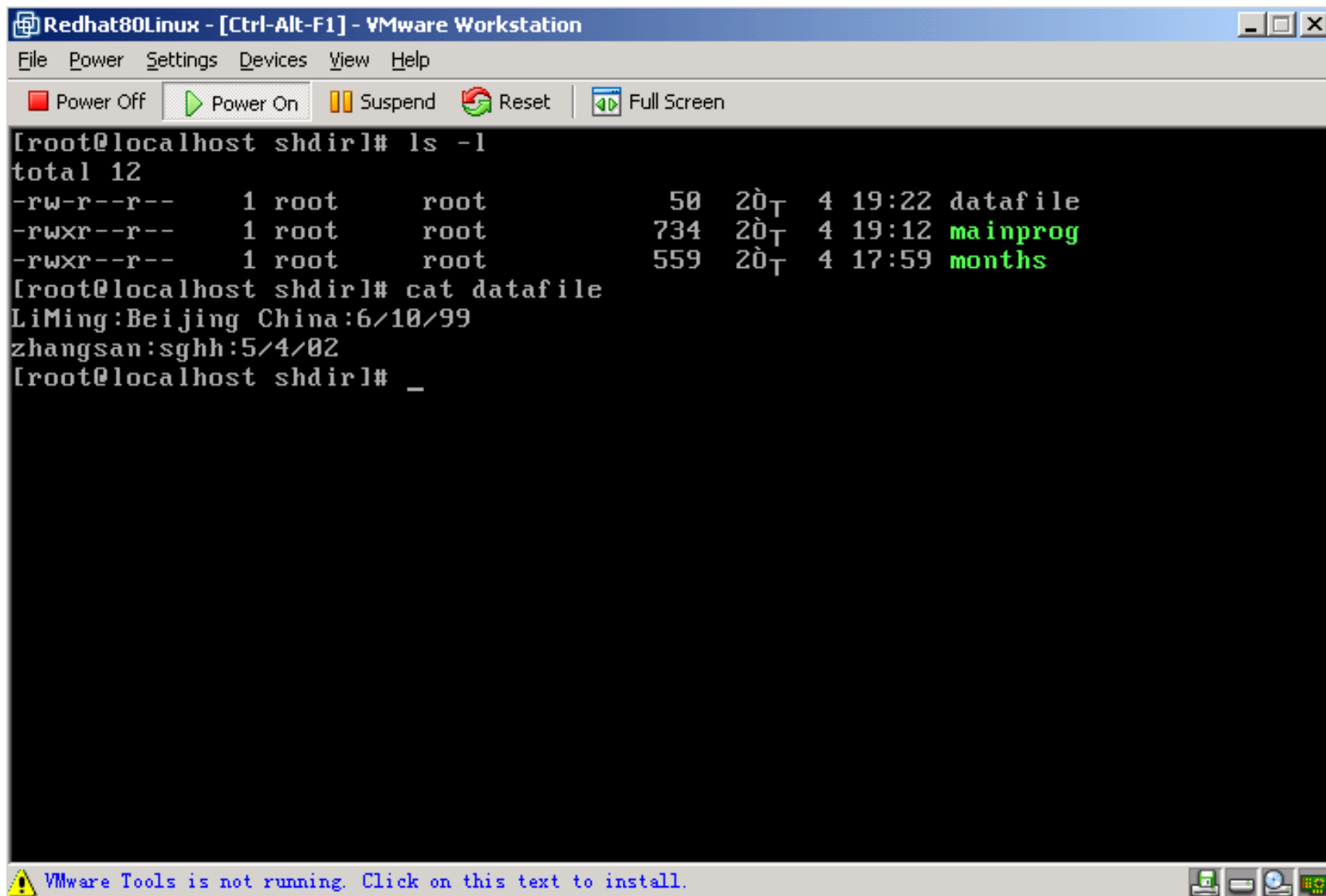


```
Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation
File Power Settings Devices View Help
Power Off Power On Suspend Reset Full Screen

[root@localhost shdir]# ./mainprog
select one:
    [1] Add info
    [2] Delete info
    [3] Update info
    [4] Exit
1
Adding information
Type the full name of employee
LiMing
Type address for employee
Beijing China
Type start date for employee(4/10/88) :
6/10/99
LiMing:Beijing China:6/10/99
Is this correct?Y
Adding info...
Do you want to go back to the main menu?y
Returned from fuction call
The name is LiMing
[root@localhost shdir]# _
```

! VMware Tools is not running. Click on this text to install.

函数和dot命令



```
Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation
File Power Settings Devices View Help
Power Off Power On Suspend Reset Full Screen

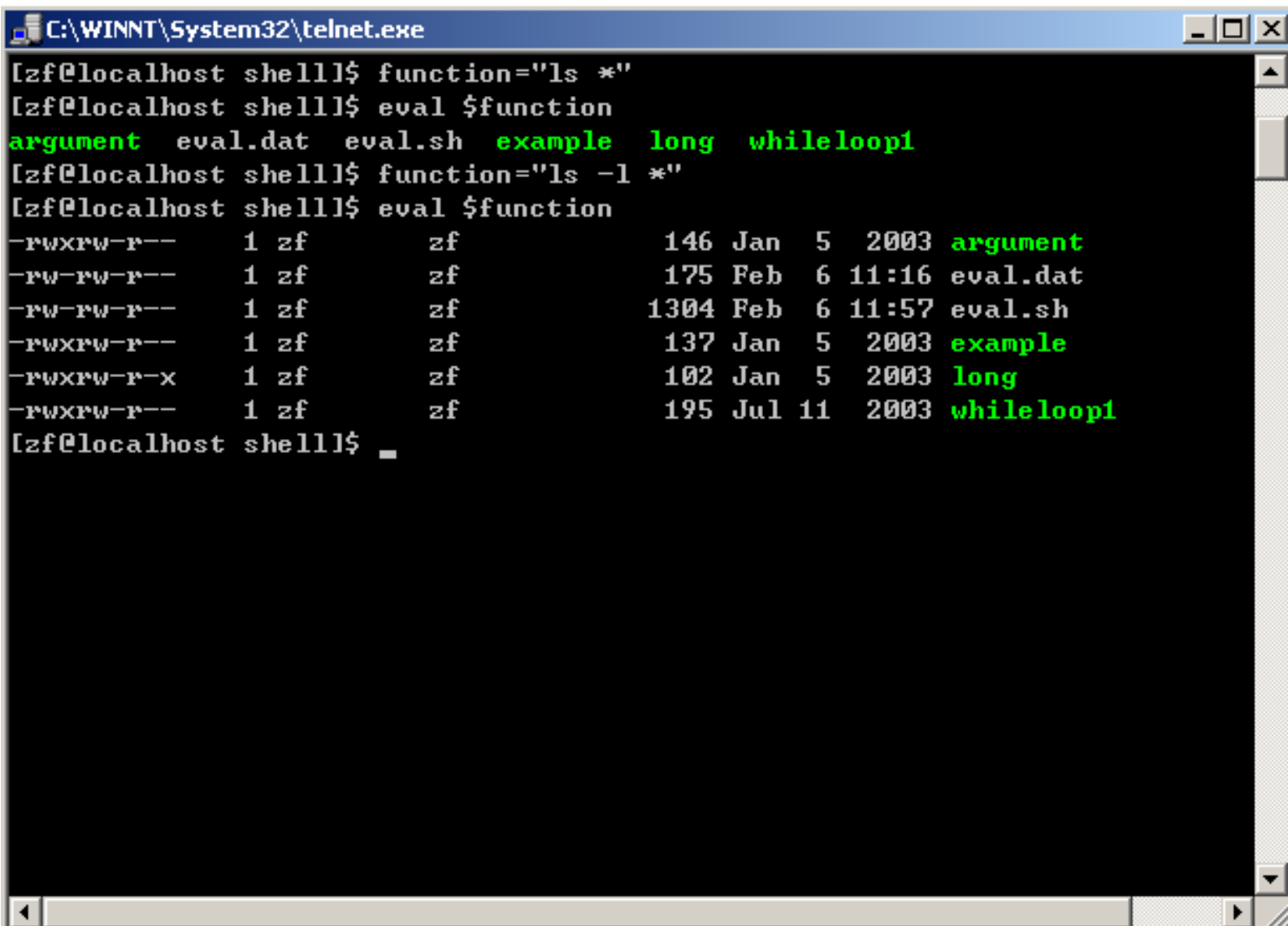
[root@localhost shdir]# ls -l
total 12
-rw-r--r-- 1 root root 50 2017-04-19 19:22 datafile
-rwxr--r-- 1 root root 734 2017-04-19 19:12 mainprog
-rwxr--r-- 1 root root 559 2017-04-17 17:59 months
[root@localhost shdir]# cat datafile
LiMing:Beijing China:6/10/99
zhangsan:sghh:5/4/02
[root@localhost shdir]# _
```

VMware Tools is not running. Click on this text to install.

使用eval命令

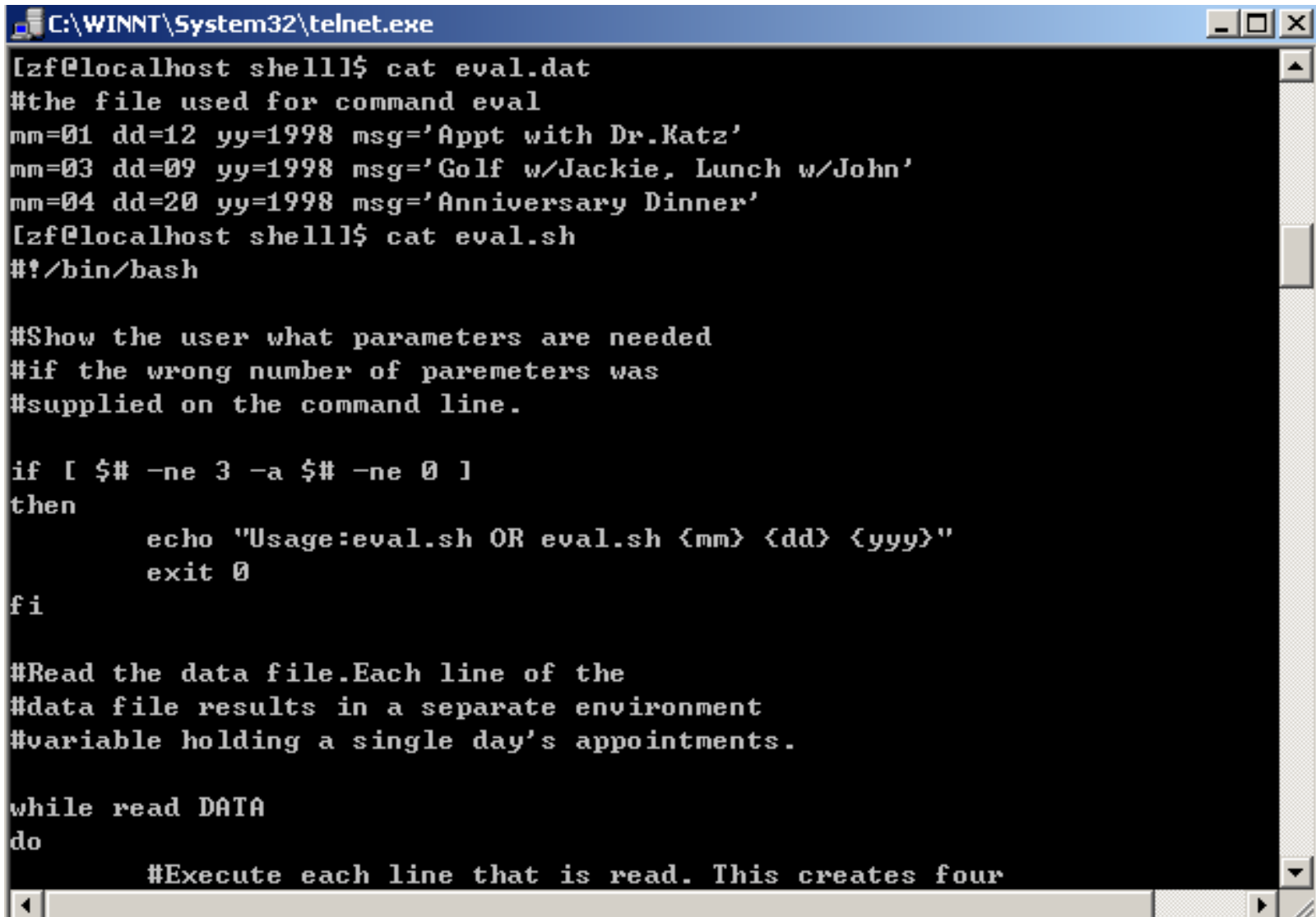
- ❖ eval命令：
- ❖ 可以用于把它的参数连在一起使用，然后执行新创建的命令，此命令实现了有效地引起第二轮变量替换。
- ❖ 利用这种技术，用户可以间接运行命令。

使用eval命令



```
C:\WINNT\System32\telnet.exe
[zf@localhost shell]$ function="ls *"
[zf@localhost shell]$ eval $function
argument eval.dat eval.sh example long whileloop1
[zf@localhost shell]$ function="ls -l *"
[zf@localhost shell]$ eval $function
-rwxrwxr--  1 zf      zf          146 Jan  5  2003 argument
-rw-rw-r--  1 zf      zf          175 Feb  6 11:16 eval.dat
-rw-rw-r--  1 zf      zf        1304 Feb  6 11:57 eval.sh
-rwxrwxr--  1 zf      zf          137 Jan  5  2003 example
-rwxrwxr-x  1 zf      zf          102 Jan  5  2003 long
-rwxrwxr--  1 zf      zf          195 Jul 11  2003 whileloop1
[zf@localhost shell]$
```

使用eval命令



```
C:\WINNT\System32\telnet.exe
[zf@localhost shell]$ cat eval.dat
#the file used for command eval
mm=01 dd=12 yy=1998 msg='Appt with Dr.Katz'
mm=03 dd=09 yy=1998 msg='Golf w/Jackie, Lunch w/John'
mm=04 dd=20 yy=1998 msg='Anniversary Dinner'
[zf@localhost shell]$ cat eval.sh
#!/bin/bash

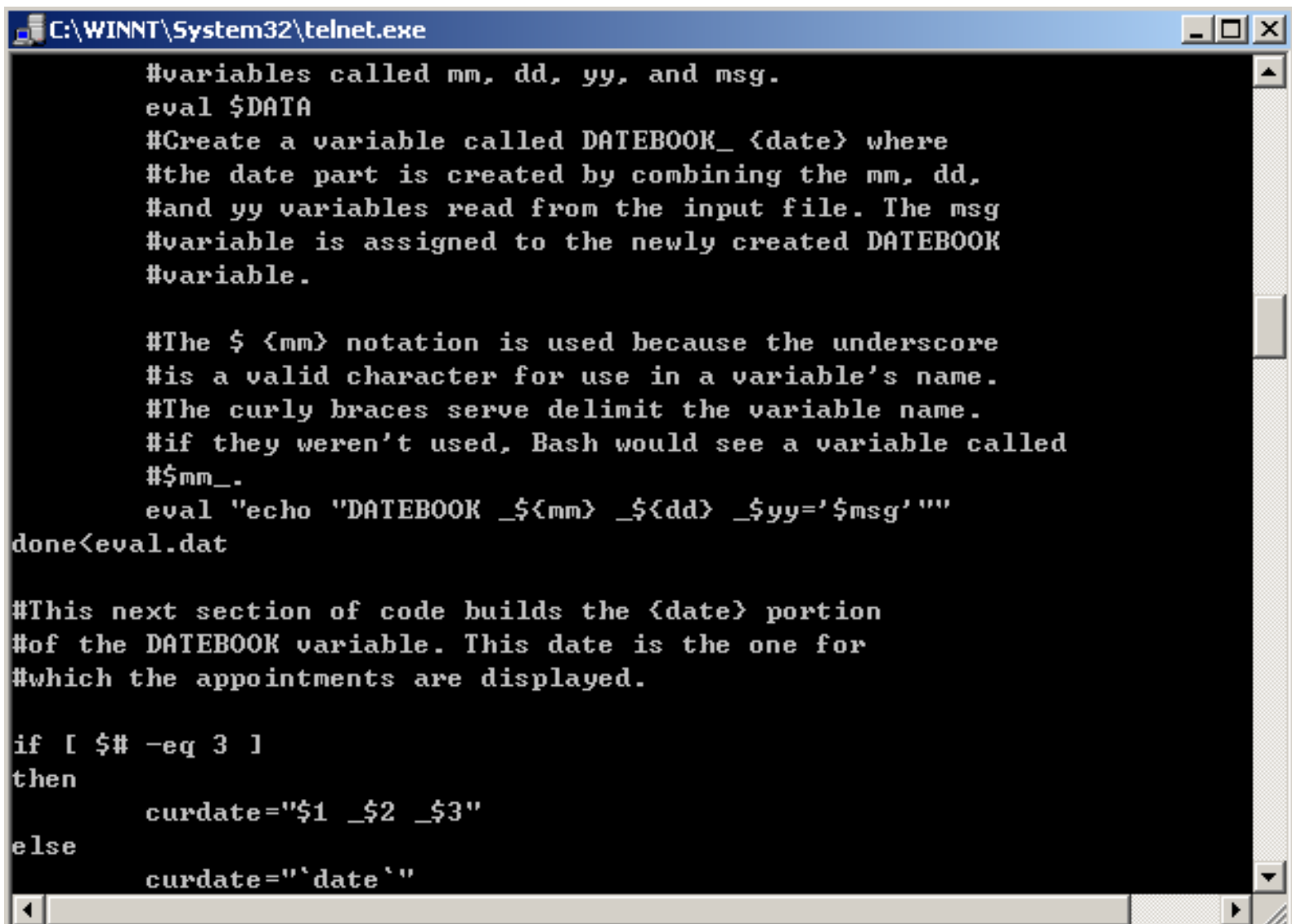
#Show the user what parameters are needed
#if the wrong number of paremeters was
#supplied on the command line.

if [ $# -ne 3 -a $# -ne 0 ]
then
    echo "Usage:eval.sh OR eval.sh {mm} {dd} {yyy}"
    exit 0
fi

#Read the data file.Each line of the
#data file results in a separate environment
#variable holding a single day's appointments.

while read DATA
do
    #Execute each line that is read. This creates four
```

使用eval命令



```
C:\WINNT\System32\telnet.exe

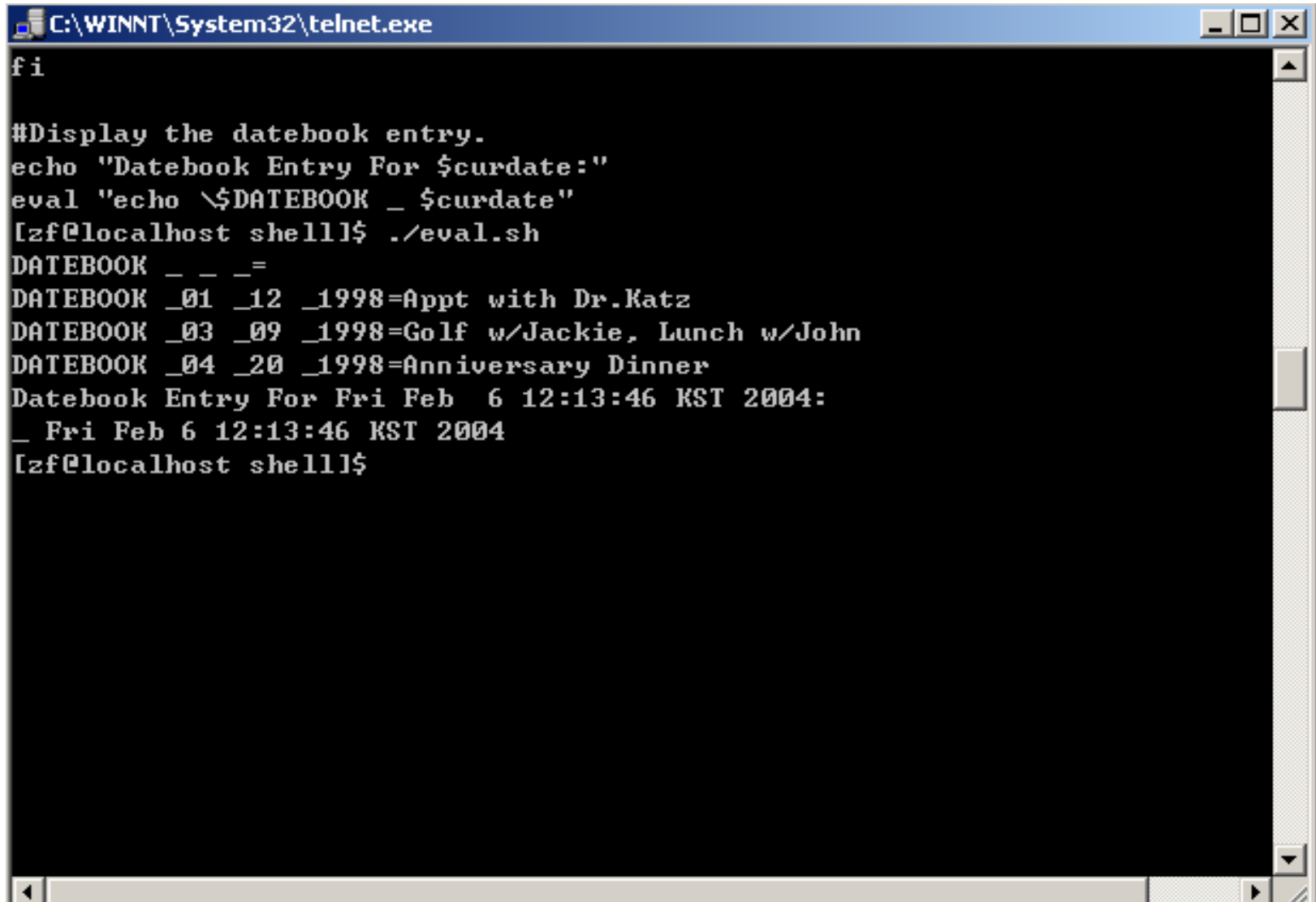
#variables called mm, dd, yy, and msg.
eval $DATA
#Create a variable called DATEBOOK_ {date} where
#the date part is created by combining the mm, dd,
#and yy variables read from the input file. The msg
#variable is assigned to the newly created DATEBOOK
#variable.

#The $ {mm} notation is used because the underscore
#is a valid character for use in a variable's name.
#The curly braces serve delimit the variable name.
#if they weren't used, Bash would see a variable called
#$_mm_.
eval "echo "DATEBOOK _${mm} _${dd} _${yy}='${msg}'""
done<eval.dat

#This next section of code builds the {date} portion
#of the DATEBOOK variable. This date is the one for
#which the appointments are displayed.

if [ $# -eq 3 ]
then
    curdate="$1 _$2 _$3"
else
    curdate="'date'"
```


使用eval命令



```
C:\WINNT\System32\telnet.exe
fi

#Display the datebook entry.
echo "Datebook Entry For $curdate:"
eval "echo \${DATEBOOK _ $curdate}"
[zf@localhost shell]$ ./eval.sh
DATEBOOK _ _ _=
DATEBOOK _01 _12 _1998=Appt with Dr.Katz
DATEBOOK _03 _09 _1998=Golf w/Jackie, Lunch w/John
DATEBOOK _04 _20 _1998=Anniversary Dinner
Datebook Entry For Fri Feb  6 12:13:46 KST 2004:
_ Fri Feb 6 12:13:46 KST 2004
[zf@localhost shell]$
```

程序调试

Shell提供了多种工具以便在调试Shell程序时使用，
这些工具允许观察一个Shell程序的执行。

常用的测试方式有：

Shell程序的详细跟踪；

Shell程序的跟踪执行。

程序调试（续1）

Shell程序的详细跟踪

Shell提供的详细跟踪特性允许用户观察一个Shell程序的读入和执行，如果在读入命令行时发现语法错误，则终止程序的执行。命令行被读入后，Shell按读入时的形式在标准错误输出中显示该命令行，然后执行命令行。详细跟踪Shell程序的执行有两种方式：

整个程序的详细跟踪和局部程序的详细跟踪。

程序调试（续2）

☆ 整个程序的跟踪执行

格式

sh -v shprog

用来实现对整个文件的脚本进行跟踪。

🕒 局部程序的跟踪执行

格式

set -v —— 设置跟踪标志

set +v —— 关闭跟踪标志

用来实现对文件中的部分脚本进行跟踪。

程序调试（续3）

例: **#** cat traced
date
echo \$PATH | wc -c
traced
1998年 11月 05日 星期四 17时 29分 59秒 CST
45
sh -v traced
date
1998年 11月 05日 星期四 17时 30分 08秒 CST
echo \$PATH | wc -c
45

程序调试（续4）

Shell程序的跟踪执行

Shell的跟踪执行功能允许用户观察一个**Shell**程序的执行，它使命令行在执行前完成所有替换之后，在标准错误输出中显示每一个被替换后的命令行，并且在行前加上前缀符号“+”（但变量赋值语句不加“+”符号），然后执行命令。

同详细跟踪一样，对**Shell**程序的跟踪执行也有两种方式：

整个程序的跟踪执行和局部程序的跟踪执行。

程序调试（续5）

☆ 整个程序的跟踪执行

格式

sh -x shprog

用来实现对整个文件脚本的跟踪执行。

🕒 局部程序的跟踪执行

格式

set -x —— 设置跟踪标志

set +x —— 关闭跟踪标志

用来实现对文件中部分脚本的跟踪执行。

程序调试（续6）

例：

```
# cat traced
```

```
date
```

```
echo $PATH | wc -c
```

```
# sh -x traced
```

```
+date
```

```
1998年 11月 05日 星期四 17时 30分 08秒  
CST
```

```
+echo /bin:/usr/bin:/usr/fk/bin
```

```
/bin:/usr/bin:/usr/fk/bin
```

```
+wc -c
```

```
25
```


程序调试（续7）

☞ 详细跟踪与跟踪执行的组合

☆ 整个程序的跟踪执行

格式

sh -vx shprog

🕒 局部程序的跟踪执行

格式

set -vx **--** 设置跟踪标志

set +vx **--** 关闭跟踪标志

调试shell程序



```
C:\WINNT\System32\telnet.exe
[Bossdev]$ cat script_oo
#!/usr/bin/sh
#This is to show what a script looks like.
echo "Our first script"
echo "-----"
echo #This inserts an empty line in output.
echo "We are currently in the following directory"
pwd
echo
echo "This directory contains the following files"
ls

[Bossdev]$ ./script_oo
Our first script
-----

We are currently in the following directory
/dev_u/zhaofang/test

This directory contains the following files
assign_logfile      display_logfile      export_logfile      testa
case_sensitive      display_parameters   script_oo
[Bossdev]$ _
```

调试shell程序

- ❖ 调试shell程序，可以把程序的第一行“`#!/usr/bin/sh`”替换为“`#!/usr/bin/sh -x`”。当再执行这个程序，它在执行前把每一行显示在终端屏幕上。在程序中真正执行行的开头显示一个加号“`+`”。之后，显示它的输出。
- ❖ 这个方法用于标识哪个程序行会引起问题。

调试shell程序

```
#!/usr/bin/sh -x
❖ #This is to show what a script looks like.
echo "Our first script"
echo "-----"
echo #This inserts an empty line in output.
echo "We are currently in the following directory"
pwd
echo
echo "This directory contains the following files"
ls

[Bossdev]$ ./script_oo
+ echo Our first script
Our first script
+ echo -----
-----
+ echo

+ echo We are currently in the following directory
We are currently in the following directory
+ pwd
/dev_u/zhaofang/test
+ echo

+ echo This directory contains the following files
This directory contains the following files
+ ls
assign_logfile      display_logfile      export_logfile      testa
case_sensitive      display_parameters   script_oo
```

Shell程序的应用

👉 何时使用**Shell**程序设计语言

☆ 当一个问题的解决方法包含了许多UNIX系统的标准命令操作时，可使用**Shell**程序设计语言；

🕒 如果一个问题能用在UNIX系统中已建立的基本操作所表示，则使用**Shell**程序设计语言能构成更强的功能；

🕒 如果处理问题的基本数据是正文行或文件，则**Shell**可描述一个很好的解决方法；若基本数据是数字或字符，则使用**Shell**可能不是好办法；

🕒 使用**Shell**程序的最后一个准则是程序的开发成本。

Linux下几种重要的工具

- ❖ Linux操作系统不仅仅是一个非常好的管理环境，而且因为它配置了许多强有力的工具，为程序开发、文件处理等应用提供了方便，其中有些工具本身就可以作为一种编程脚本来完成相当重要的功能。
- ❖ 本部分我们介绍以下内容：
- ❖ 1、匹配检索规则
- ❖ 2、机制简单、实用的文件处理工具grep、sed和awk。
- ❖ 3、这几种工具的脚本编辑方法。

文件名替换

❖ * ? [characters]

❖ 功能:

❖ 1、匹配文件前缀

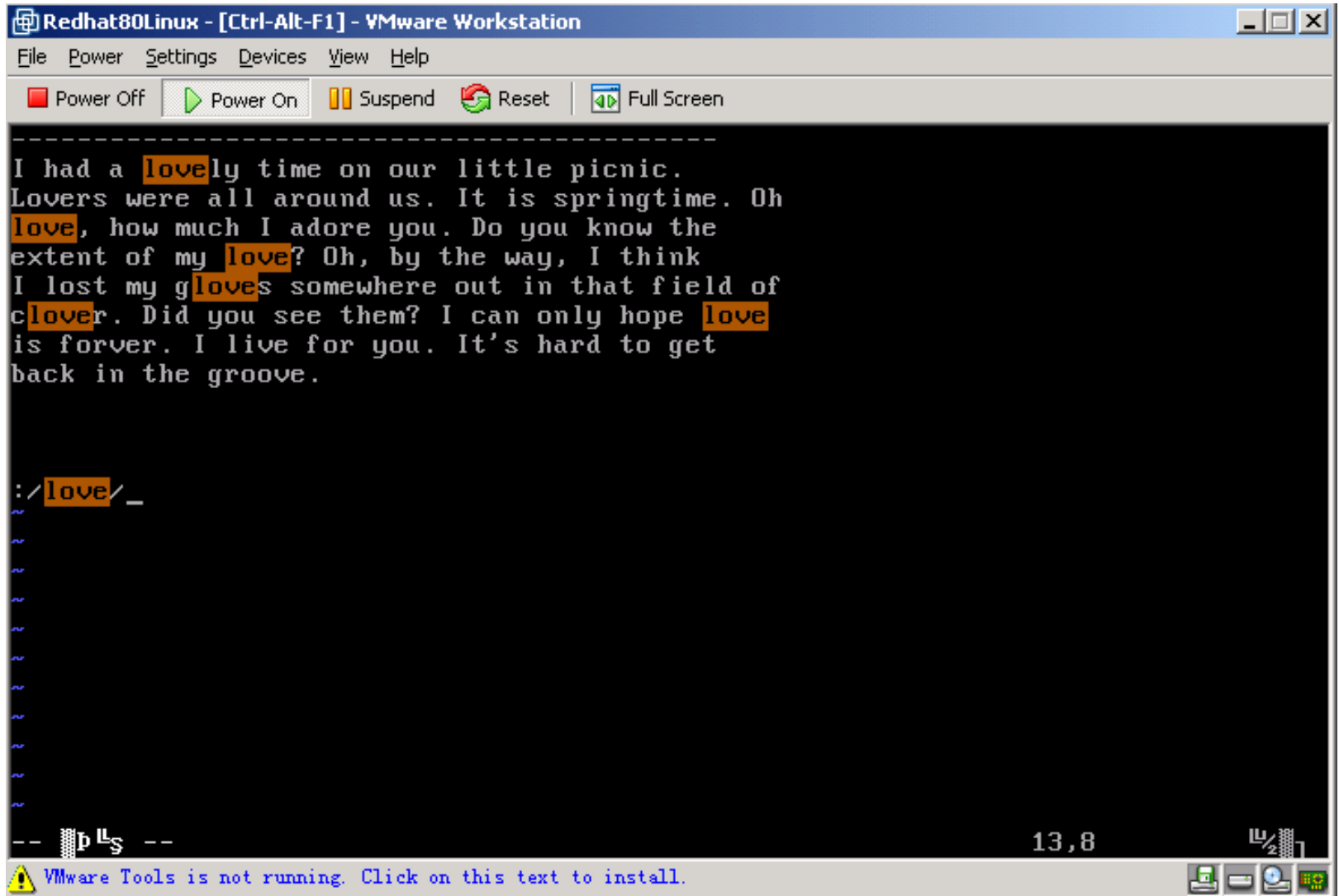
❖ 2、匹配文件后缀

❖ 3、匹配后缀和前缀

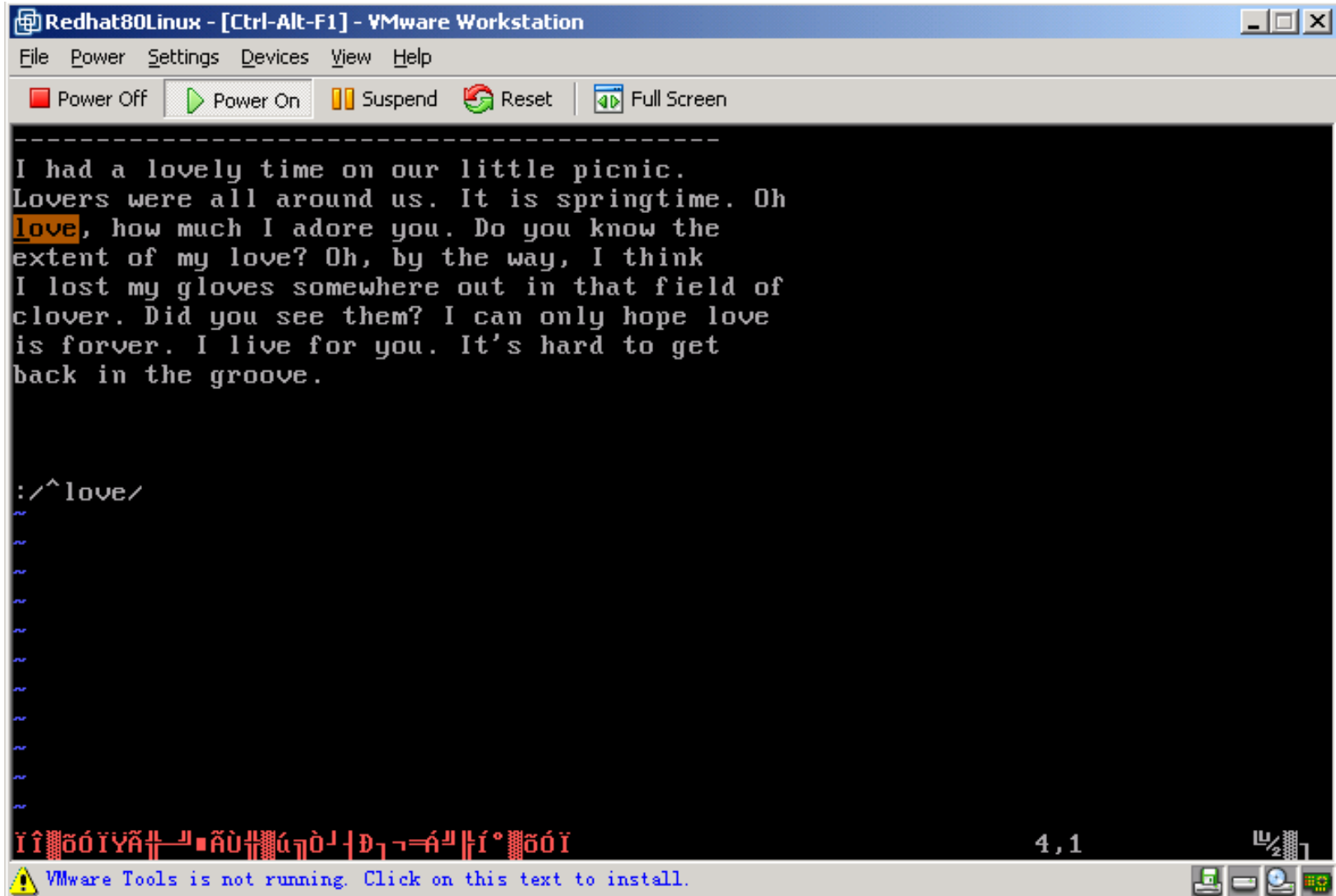
❖ 4、匹配字符集

常用正则表达式

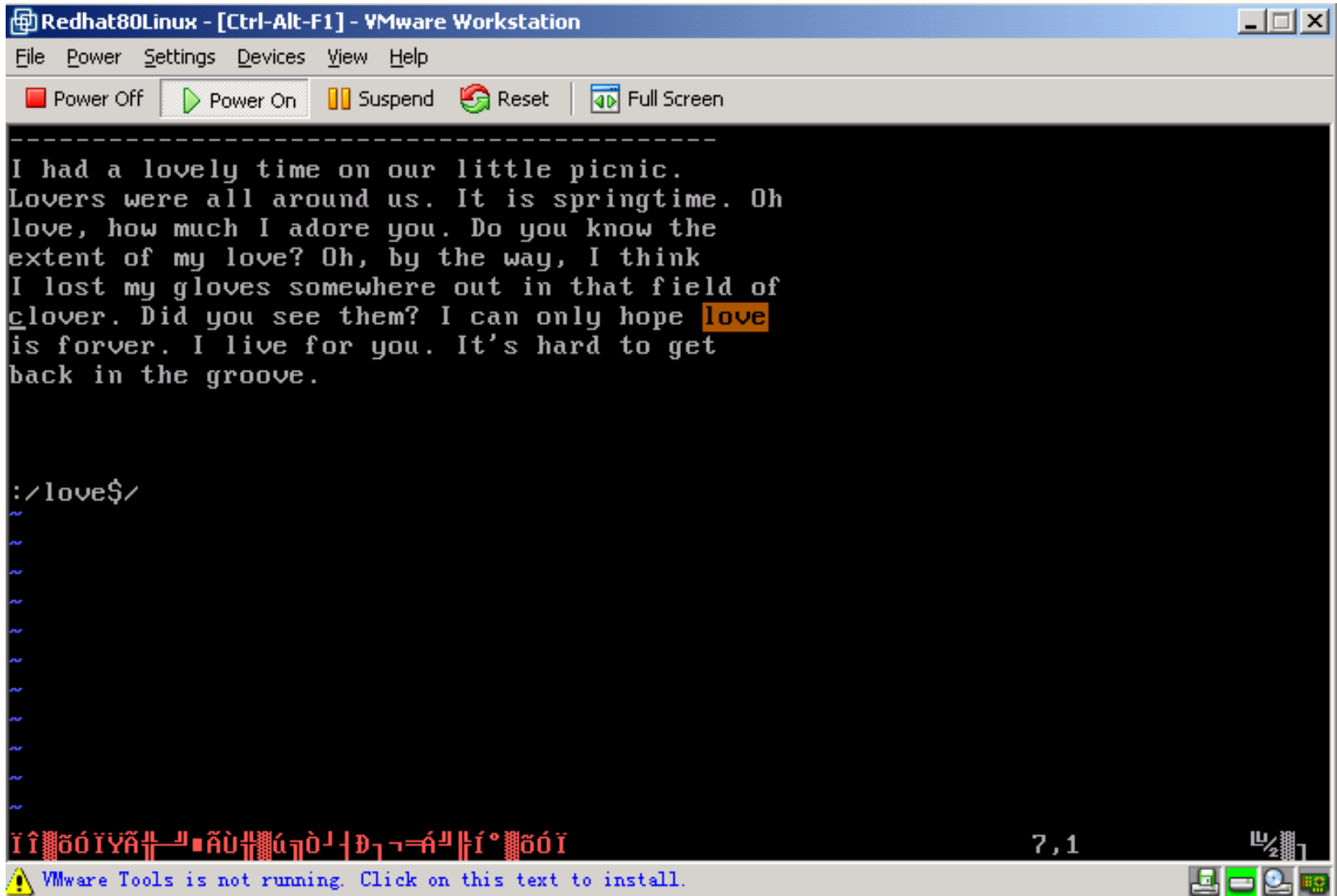
正则表达式



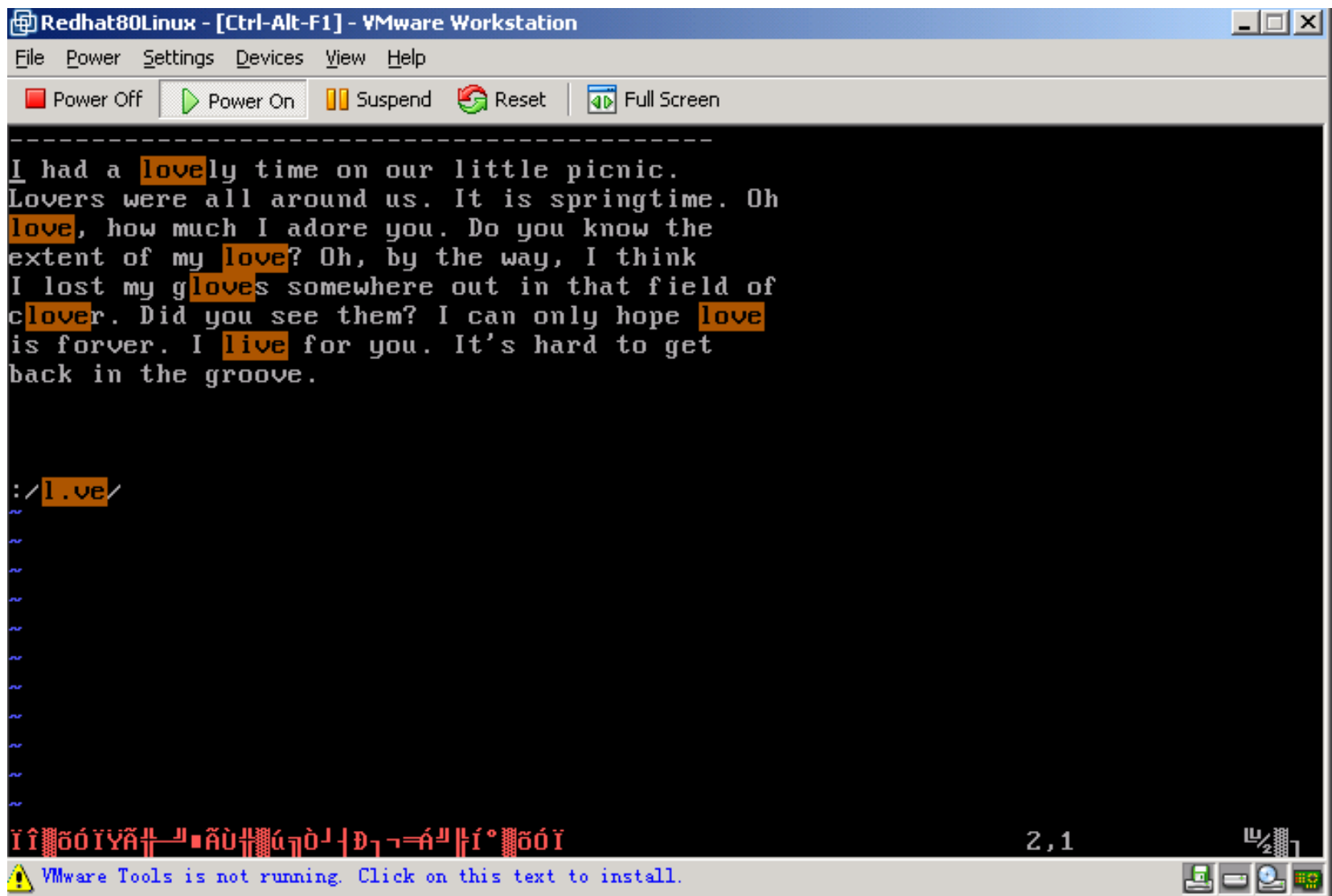
正则表达式



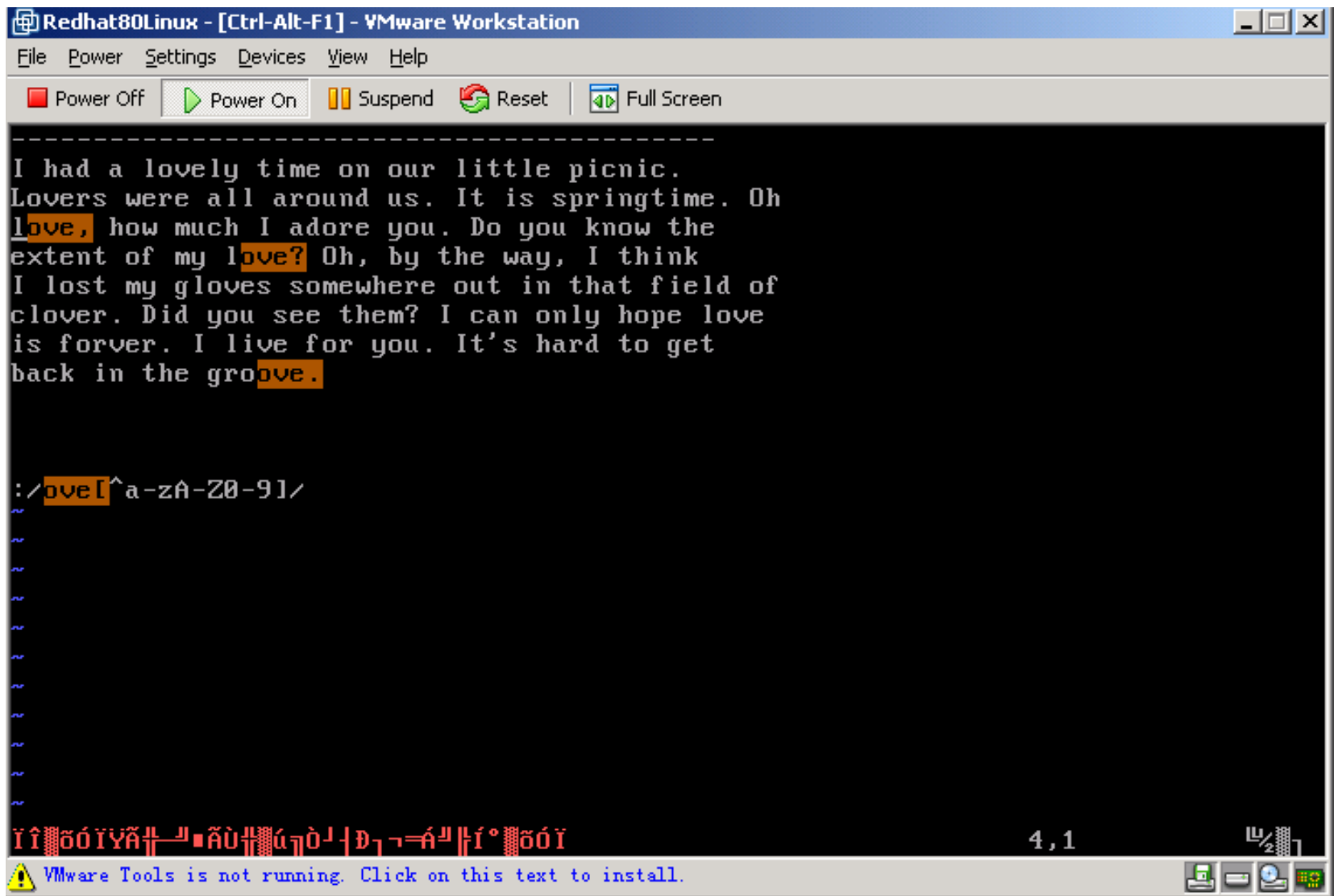
正则表达式



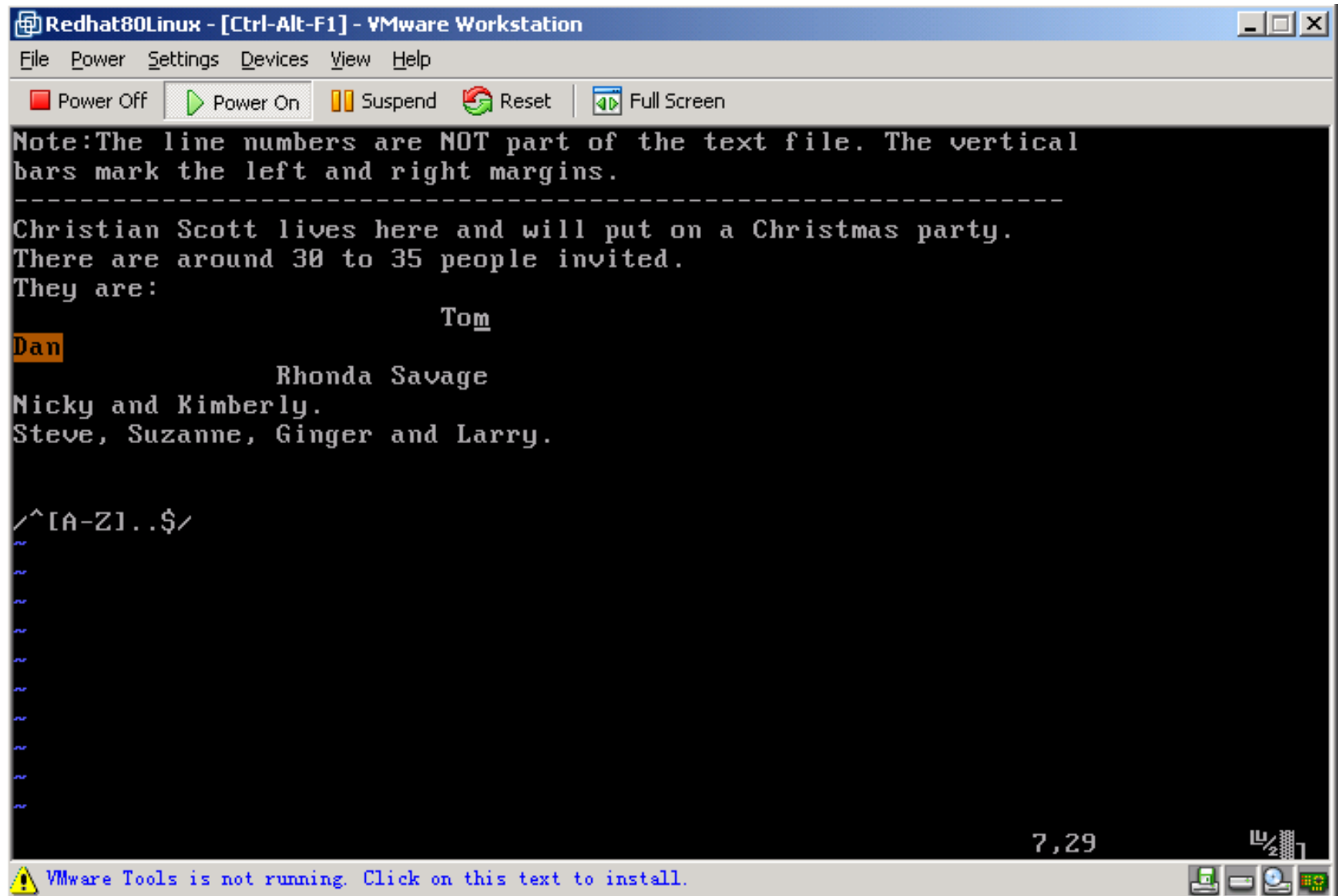
正则表达式



正则表达式



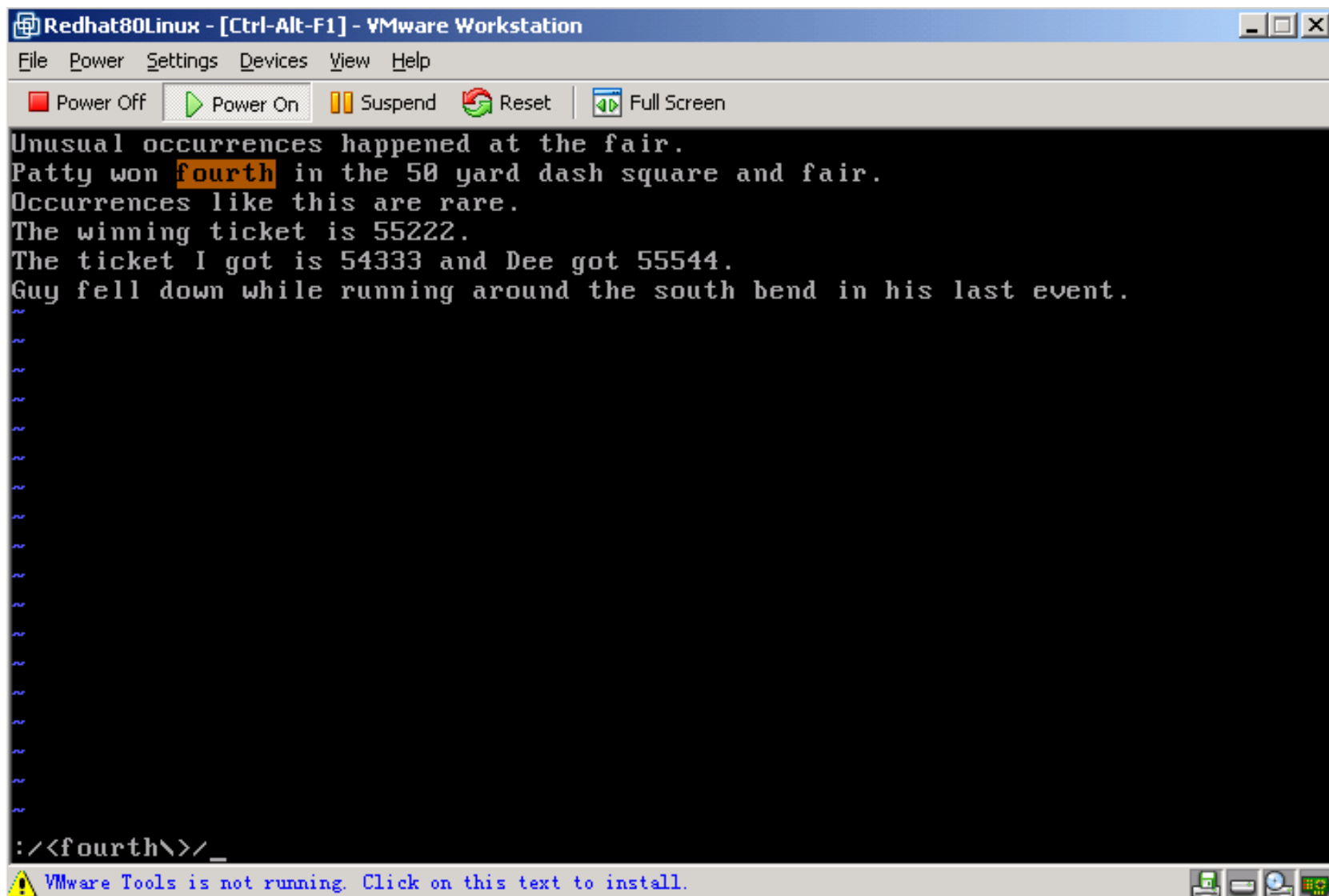
正则表达式



更多的正则表达式元字符

- ❖ 有些元字符不一定可移植到所有使用正则表达式实用程序，但可以用于vi编辑器和sed和grep的一些版本中。
- ❖ 如：
- ❖ \<是词开头定位；
- ❖ \>是词结尾定位。

更多的正则表达式元字符



更多的正则表达式元字符

A screenshot of a Windows NT command prompt window running telnet.exe. The title bar reads "C:\WINNT\System32\telnet.exe". The main area contains the following text:

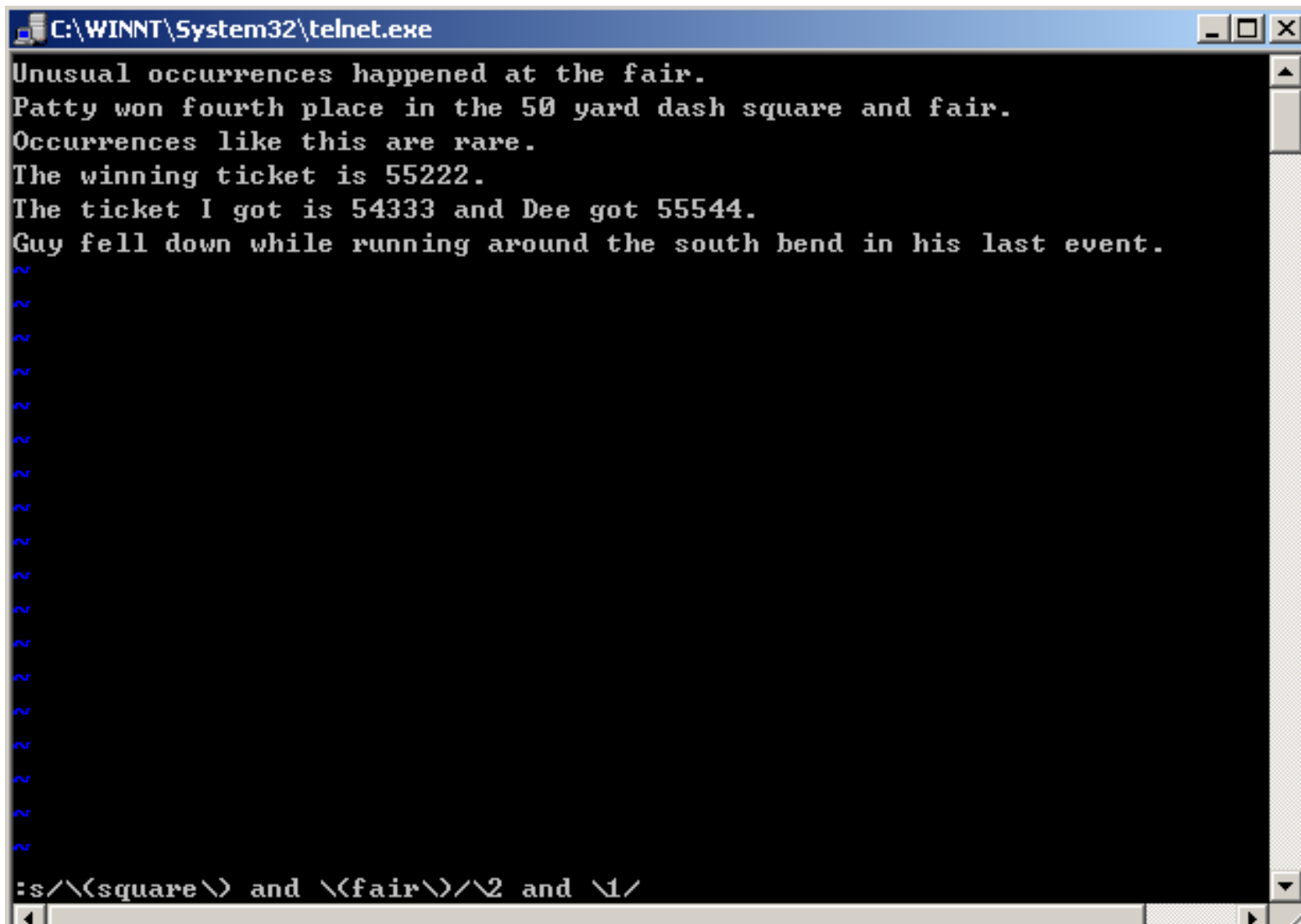
Unusual occurences happened at the fair.
Patty won fourth place in the 50 yard dash square and fair.
Occurences like this are rare.
The winning ticket is 55222.
The ticket I got is 54333 and Dee got 55544.
Guy fell down while running around the south bend in his last event.

This is followed by approximately 18 blue tilde (~) characters arranged vertically. At the bottom left, there is a partial line of text starting with ":1,\$s/\<[Oo]ccur\>ence/\1rence/" which appears to be part of a regular expression search pattern. A vertical scrollbar is visible on the right side of the window.

更多的正则表达式元字符

[illegible]

更多的正则表达式元字符



```
C:\WINNT\System32\telnet.exe
Unusual occurrences happened at the fair.
Patty won fourth place in the 50 yard dash square and fair.
Occurrences like this are rare.
The winning ticket is 55222.
The ticket I got is 54333 and Dee got 55544.
Guy fell down while running around the south bend in his last event.
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
:s/\(square\) and \(fair\)\/\2 and \1/
```

更多的正则表达式元字符

[illegible]

模式搜索程序grep

- ❖ **grep**是**Globally find Regular Expression and Print**的缩写。它的功能是寻找文件中包含有正则表达式的行并将其打印出来。

模式搜索程序grep

```
C:\WINNT\System32\telnet.exe
[zf@localhost shell]$ cat datafile
northwest  NW  Charles Main      3.0    0.98    3      34
western    WE  Sharon Gray       5.3     .97     5      23
southwest  SW  Lewis Dalsass     2.7     .8       2      18
southern   SO  Suan Chin        5.1     .95     4      15
southeast  SE  Patricia Hemenway 4.0     .7       4      17
eastern    EA  TB Savage        4.4     .84     5      20
northeast  NE  AM Main Jr.      5.1     .94     3      13
north      NO  Margot Weber     4.5     .89     5       9
central    CT  Ann Stephens     5.7     .94     5      13
[zf@localhost shell]$
```

模式搜索程序grep

```
C:\WINNT\System32\telnet.exe
[zf@localhost shell]$ grep NW datafile
northwest  NW  Charles Main      3.0    0.98    3      34
[zf@localhost shell]$ grep NW d*
datafile:northwest  NW  Charles Main      3.0    0.98    3      34
[zf@localhost shell]$ grep '^n' datafile
northwest  NW  Charles Main      3.0    0.98    3      34
northeast  NE  AM Main Jr.      5.1    .94     3      13
north      NO  Margot Weber     4.5    .89     5       9
[zf@localhost shell]$ grep '4$' datafile
northwest  NW  Charles Main      3.0    0.98    3      34
[zf@localhost shell]$ grep '5..' datafile
western    WE  Sharon Gray     5.3    .97     5      23
southern   SO  Suan Chin      5.1    .95     4      15
eastern    EA  TB Savage      4.4    .84     5      20
northeast  NE  AM Main Jr.     5.1    .94     3      13
north      NO  Margot Weber     4.5    .89     5       9
central    CT  Ann Stephens    5.7    .94     5      13
[zf@localhost shell]$ grep '\.5' datafile
north      NO  Margot Weber     4.5    .89     5       9
[zf@localhost shell]$ grep '^[we]' datafile
western    WE  Sharon Gray     5.3    .97     5      23
eastern    EA  TB Savage      4.4    .84     5      20
[zf@localhost shell]$
```

模式搜索程序grep

```
C:\WINNT\System32\telnet.exe

[zf@localhost shell]$ grep '[^0-9]' datafile
northwest  NW  Charles Main      3.0   0.98   3    34
western    WE  Sharon Gray       5.3   .97    5    23
southwest  SW  Lewis Dalsass     2.7   .8     2    18
southern   SO  Suan Chin        5.1   .95    4    15
southeast  SE  Patricia Hemenway 4.0   .7     4    17
eastern    EA  TB Savage         4.4   .84    5    20
northeast  NE  AM Main Jr.       5.1   .94    3    13
north      NO  Margot Weber      4.5   .89    5     9
central    CT  Ann Stephens      5.7   .94    5    13
[zf@localhost shell]$ grep '[A-Z][A-Z] [A-Z]' datafile
eastern    EA  TB Savage         4.4   .84    5    20
northeast  NE  AM Main Jr.       5.1   .94    3    13
[zf@localhost shell]$ grep 'ss*' datafile
northwest  NW  Charles Main      3.0   0.98   3    34
southwest  SW  Lewis Dalsass     2.7   .8     2    18
central    CT  Ann Stephens      5.7   .94    5    13
[zf@localhost shell]$
```

模式搜索程序grep

```
C:\WINNT\System32\telnet.exe
[zf@localhost shell]$ grep '[a-z]\{9\}' datafile
northwest  NW  Charles Main      3.0  0.98  3    34
southwest  SW  Lewis Dalsass    2.7   .8    2    18
southeast  SE  Patricia Hemenway 4.0   .7    4    17
northeast  NE  AM Main Jr.      5.1   .94   3    13
[zf@localhost shell]$ grep '\{3\}\.[0-9].*\1 *\1' datafile
northwest  NW  Charles Main      3.0  0.98  3    34
[zf@localhost shell]$ grep '\<north' datafile
northwest  NW  Charles Main      3.0  0.98  3    34
northeast  NE  AM Main Jr.      5.1   .94   3    13
north      NO  Margot Weber      4.5   .89   5     9
[zf@localhost shell]$ grep '\<north>' datafile
[zf@localhost shell]$
```


模式搜索程序grep

```
C:\WINNT\System32\telnet.exe
[zf@localhost test]$ ls -l
total 32
-rw-rw-r-- 1 zf      zf          0 Jun 12  2003 file1
-r---w-r-- 1 zf      zf          0 Jun 12  2003 file2
drwxrwxr-x 2 zf      zf      4096 Feb  9 16:01 mydir
drwxrwxr-x 2 zf      zf      4096 Feb  9 16:01 test
-rwxrwxr-x 1 zf      zf     16913 Jan 21  2003 watch
-rw-r---w- 1 zf      zf          0 Jan 21  2003 watch_test
-rwxrw---- 1 zf      zf      129 Jan 21  2003 watch_test.c
[zf@localhost test]$ ls -l |grep '^d'
drwxrwxr-x 2 zf      zf      4096 Feb  9 16:01 mydir
drwxrwxr-x 2 zf      zf      4096 Feb  9 16:01 test
[zf@localhost test]$
```

模式搜索程序grep

```
C:\WINNT\System32\telnet.exe
[zf@localhost shell]$ grep -n '^south' datafile
3:southwest SW Lewis Dalsass 2.7 .8 2 18
4:southern SO Suan Chin 5.1 .95 4 15
5:southeast SE Patricia Hemenway 4.0 .7 4 17
[zf@localhost shell]$ grep -i 'pat' datafile
southeast SE Patricia Hemenway 4.0 .7 4 17
[zf@localhost shell]$ grep -l 'SE' *
datafile
[zf@localhost shell]$ grep -c 'west' datafile
3
[zf@localhost shell]$ grep -w 'north' datafile
north NO Margot Weber 4.5 .89 5 9
[zf@localhost shell]$
```

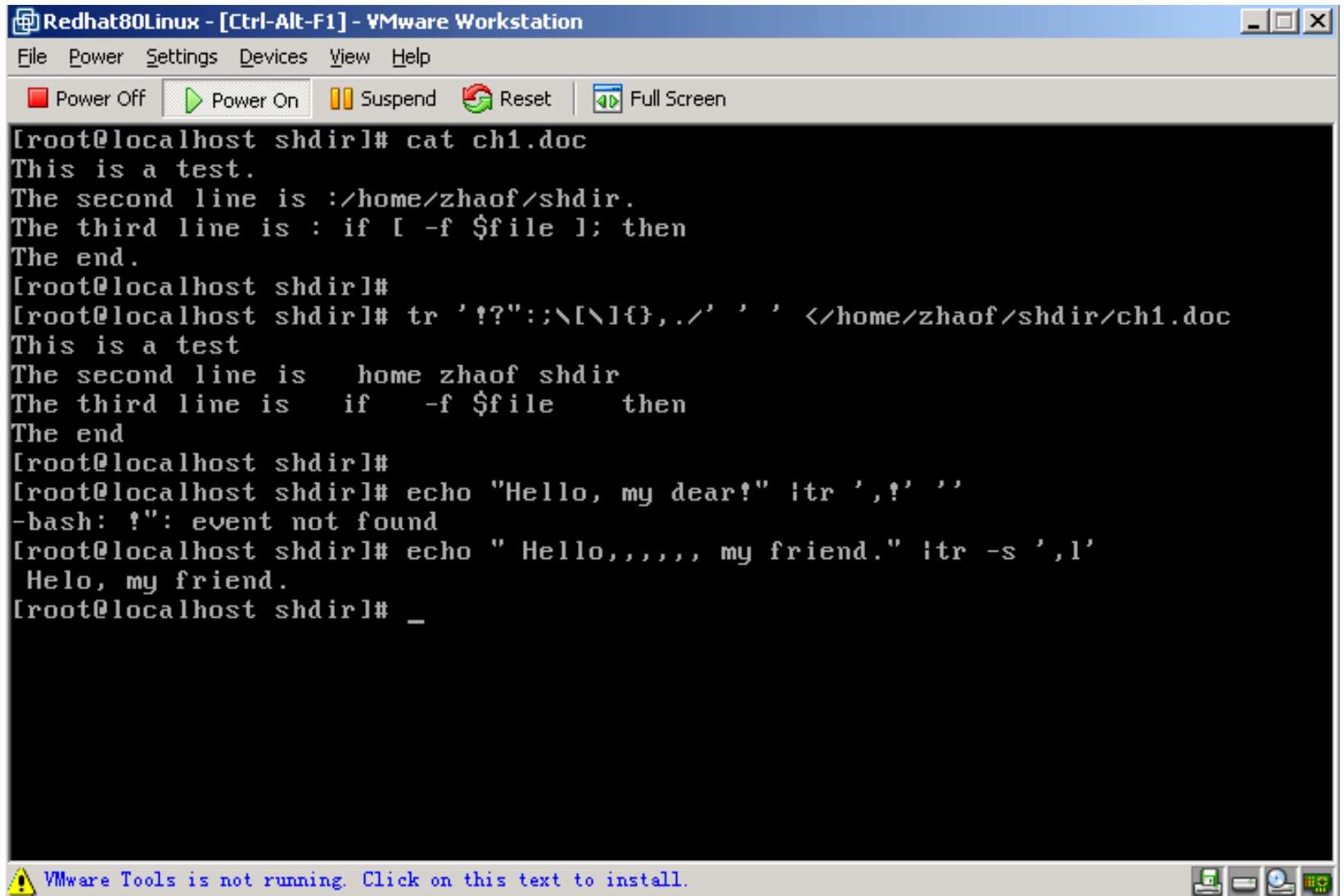
统计单词总数

- ❖ 统计单词最简单的方法是**wc**命令，但它只能显示所有内容的字符、单词或行数目。
- ❖ 当需要统计一个文件中某个特殊单词**word**总数时，可以使用下列命令：
 - ❖ **1、tr**：将一个集合中的所有字符改变成另一个集合中的字符。也可以用于删除字符集。
 - ❖ **2、sort**：为输入文件中的行进行分类，如果没有指定输入文件，则为**STDIN**中给出的行分类。
 - ❖ **3、uniq**：打印出文件中所有的唯一行。如果某行出现多次，只打印该行的一个拷贝。可以列出某个特定行重复的次数。

统计单词总数

- ❖ **tr**命令用于删除输入文件中的所有标点符号和定界符。
格式如下：
- ❖ `tr 'set1' 'set2'`
- ❖ 如：
- ❖ `tr '!?"":;\[\]\{\}(),,.''
'' </home/zhaof/docs/ch1.doc`
- ❖ 注意：在**tr**命令中字符 “[”和 “]”以字符 “\[”和 “\]”
的格式给出，因为它们在**tr**命令中有特殊意义。

统计单词总数



```
Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation
File Power Settings Devices View Help
Power Off Power On Suspend Reset Full Screen

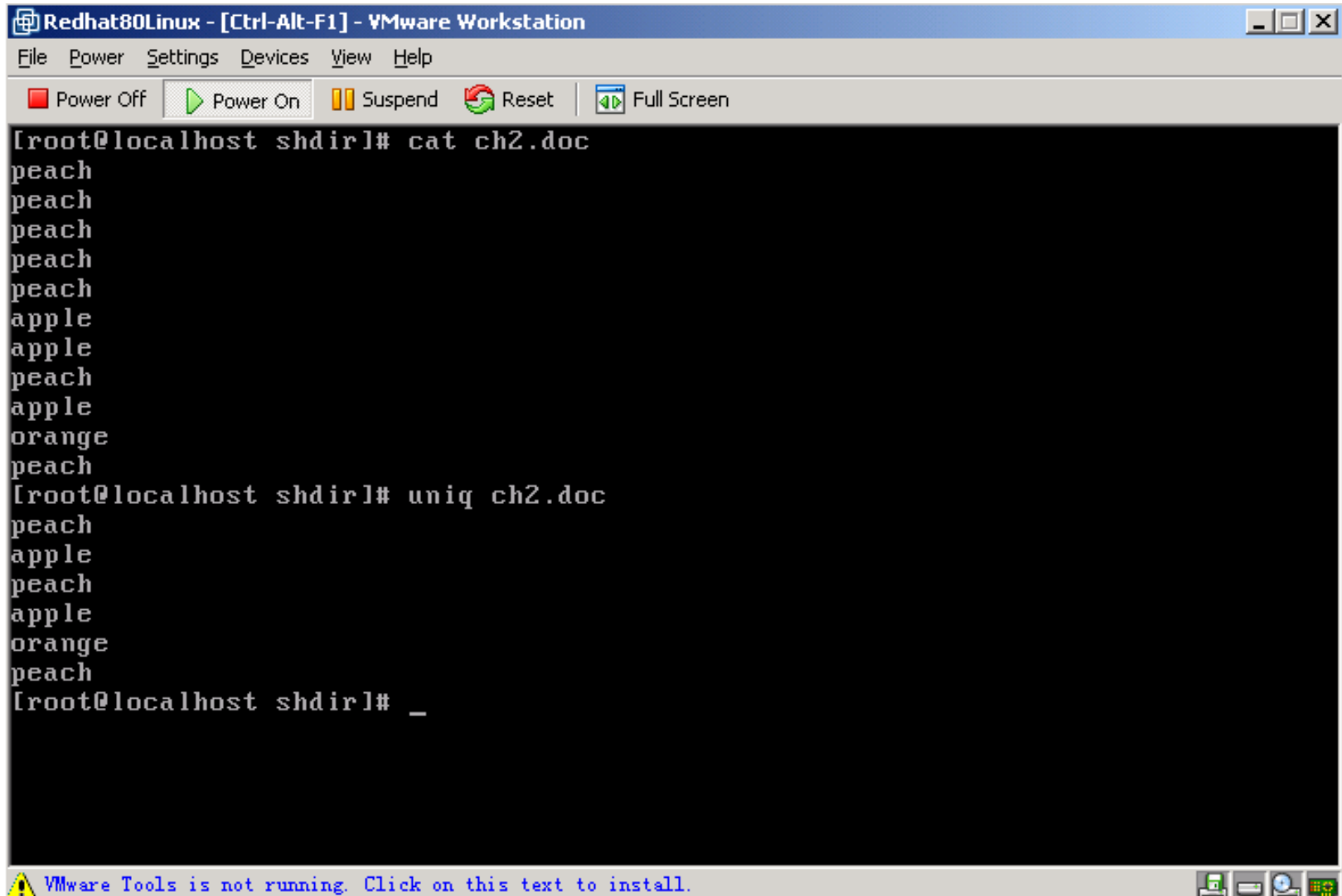
[root@localhost shdir]# cat ch1.doc
This is a test.
The second line is :/home/zhaof/shdir.
The third line is : if [ -f $file ]; then
The end.
[root@localhost shdir]#
[root@localhost shdir]# tr '?!\";:\[\\]{}.,/' ' ' </home/zhaof/shdir/ch1.doc
This is a test
The second line is    home zhaof shdir
The third line is    if    -f $file    then
The end
[root@localhost shdir]#
[root@localhost shdir]# echo "Hello, my dear!" |tr ',!' ''
-bash: !\";: event not found
[root@localhost shdir]# echo " Hello,,,,, my friend." |tr -s ',l' '
Helo, my friend.
[root@localhost shdir]# _

! VMware Tools is not running. Click on this text to install.
```

统计单词总数

- ❖ 我们可以通过**sort -u**命令选项删除所有重复的词，如果需要统计每个单词重复的次数，可以使用**uniq**命令。

统计单词总数



```
Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation
File Power Settings Devices View Help
Power Off Power On Suspend Reset Full Screen

[root@localhost shdir]# cat ch2.doc
peach
peach
peach
peach
peach
apple
apple
peach
apple
orange
peach
[root@localhost shdir]# uniq ch2.doc
peach
apple
peach
apple
orange
peach
[root@localhost shdir]# _
```

VMware Tools is not running. Click on this text to install.

统计单词总数

The screenshot shows a VMware Workstation interface. The title bar reads "Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation". Below it is a menu bar with "File", "Power", "Settings", "Devices", "View", and "Help". A toolbar contains icons for "Power Off", "Power On", "Suspend", "Reset", and "Full Screen". The main area displays a terminal window with the following commands and output:

```
[root@localhost shdirl]# sort ch2.doc  
apple  
apple  
apple  
orange  
peach  
peach  
peach  
peach  
peach  
peach  
peach  
peach  
[root@localhost shdirl]# sort ch2.doc |uniq  
apple  
orange  
peach  
[root@localhost shdirl]# sort ch2.doc |uniq -c  
      3 apple  
      1 orange  
      7 peach  
[root@localhost shdirl]# _
```


At the bottom of the window, there is a status bar with a yellow warning icon and the text "VMware Tools is not running. Click on this text to install." To the right of the status bar are several small icons representing different system components.

流编辑器sed

- ❖ 流编辑程序**sed**是一种线形型、非交互式的编辑器。它允许用户在不和编辑器交互式工作的方式下，执行与在**vi**或**ex**编辑器一样的编辑任务。
- ❖ **sed**编辑器是无破坏性的，它不改变文件，除非用户用**shell**重定向保存输出的结果。
- ❖ 在默认情况下，**sed**把所有的行打印在屏幕上。

流编辑器sed

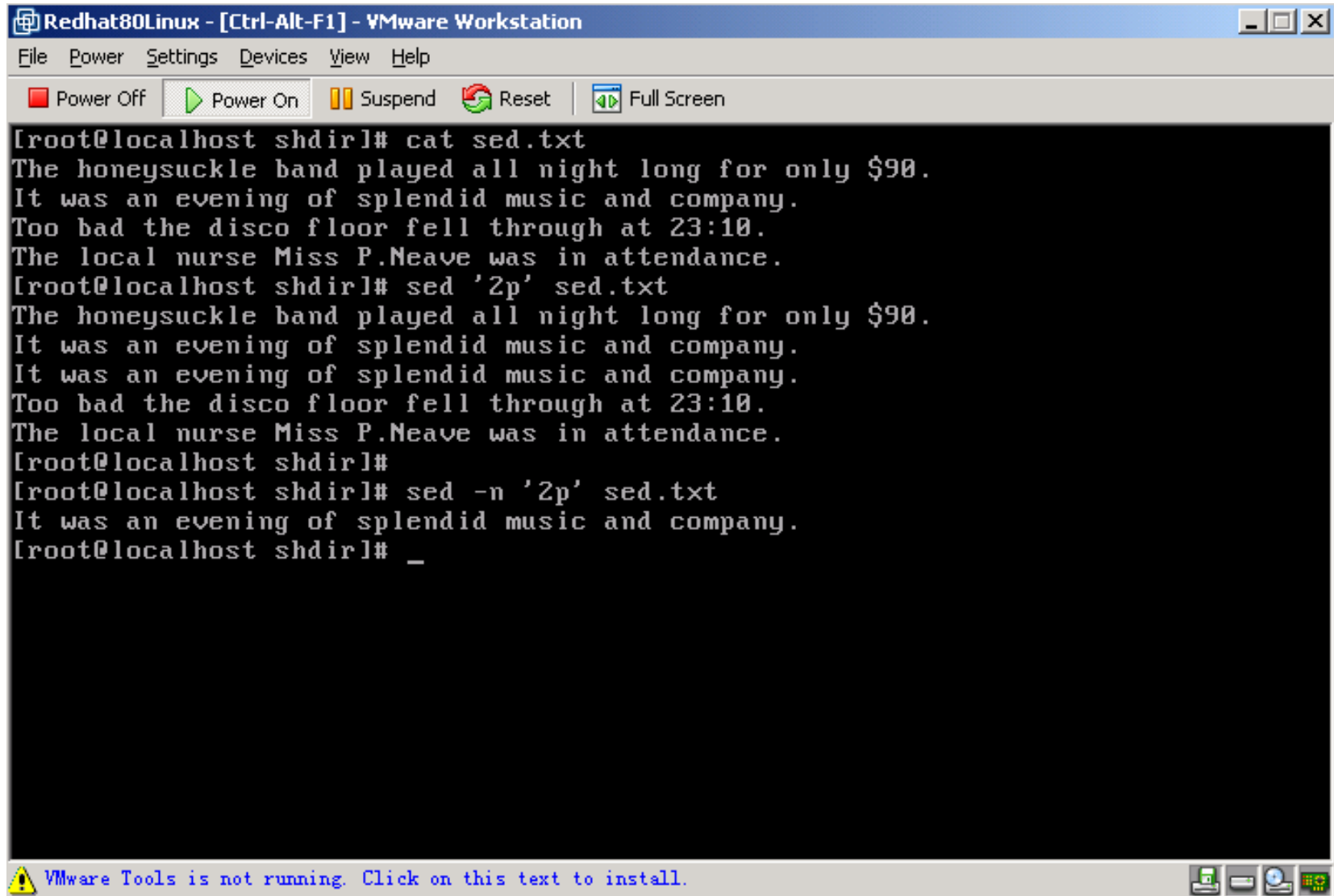
- ❖ 调用sed命令的三种方式：
- ❖ 1、直接键入命令：
❖ `sed [-option] command_line filename`
- ❖ 2、将sed命令插入脚本文件，然后调用sed命令：
❖ `sed [-option] -f program_file filename`
- ❖ 3、将sed命令插入脚本文件，并使脚本可执行。
❖ `sed program_file [-option] filename`
- ❖ 其中，[-option]包括如下内容：
 - ❖ **n** 不打印；默认情况下打印所有行。
 - ❖ **c** 允许多次编辑；即下一个命令是编辑命令。
 - ❖ **f** 通知sed脚本支持所有的sed命令。

流编辑器sed

- ❖ **sed**处理输入文件时，默认从第一行开始，但用户也可以用其他方式来定位文本：
- ❖ 1、使用行号来定位文本；
- ❖ 2、使用正则表达式来定位文本。

Sed命令正则表达式规则

流编辑器sed

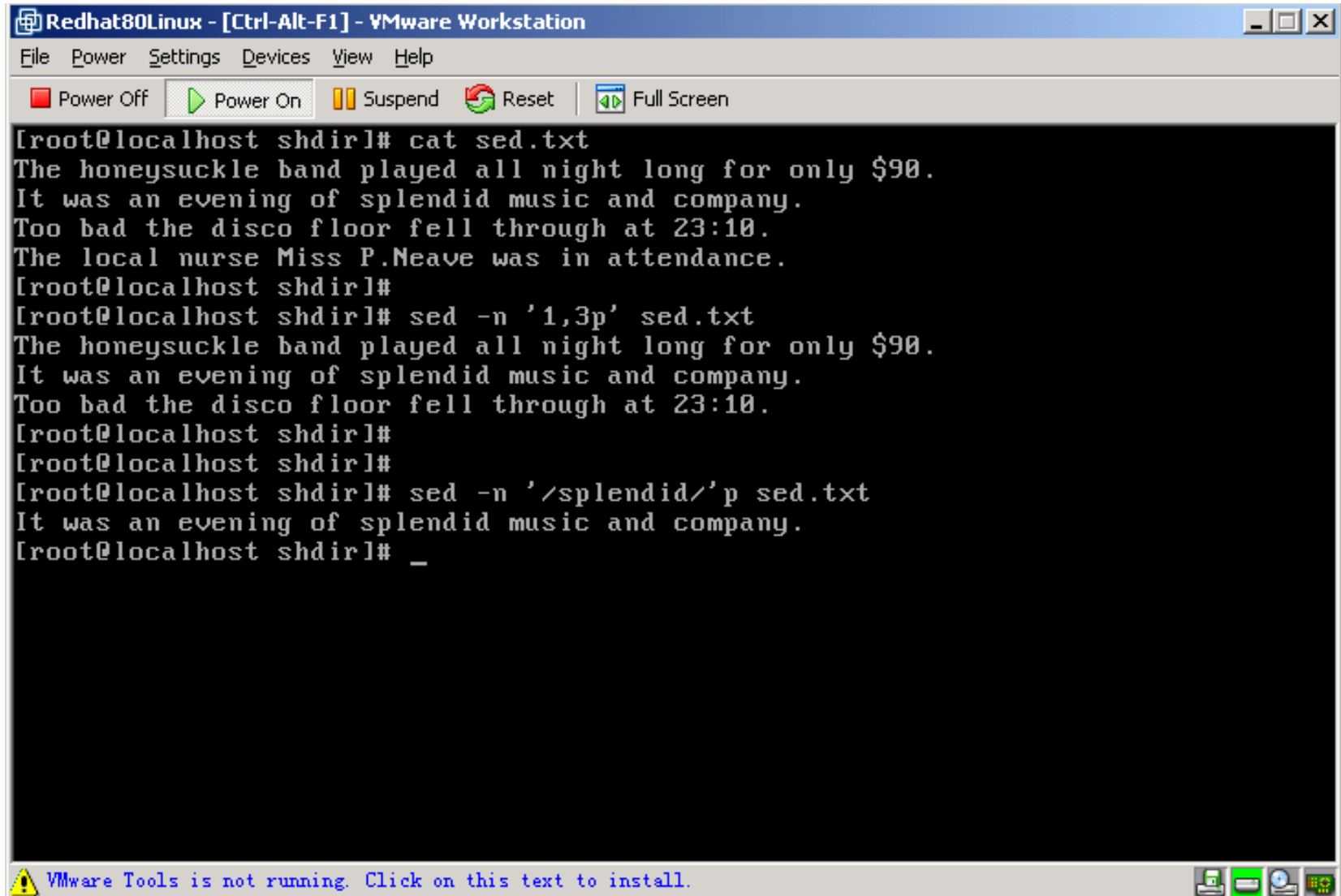


The screenshot shows a terminal window titled "Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation". The terminal displays the following commands and output:

```
[root@localhost shdir]# cat sed.txt
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
[root@localhost shdir]# sed '2p' sed.txt
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
[root@localhost shdir]#
[root@localhost shdir]# sed -n '2p' sed.txt
It was an evening of splendid music and company.
[root@localhost shdir]# _
```

At the bottom of the window, there is a status bar with a yellow warning icon and the text: "VMware Tools is not running. Click on this text to install." On the right side of the status bar, there are icons for VMware Tools, a printer, a network icon, and a power icon.

流编辑器sed

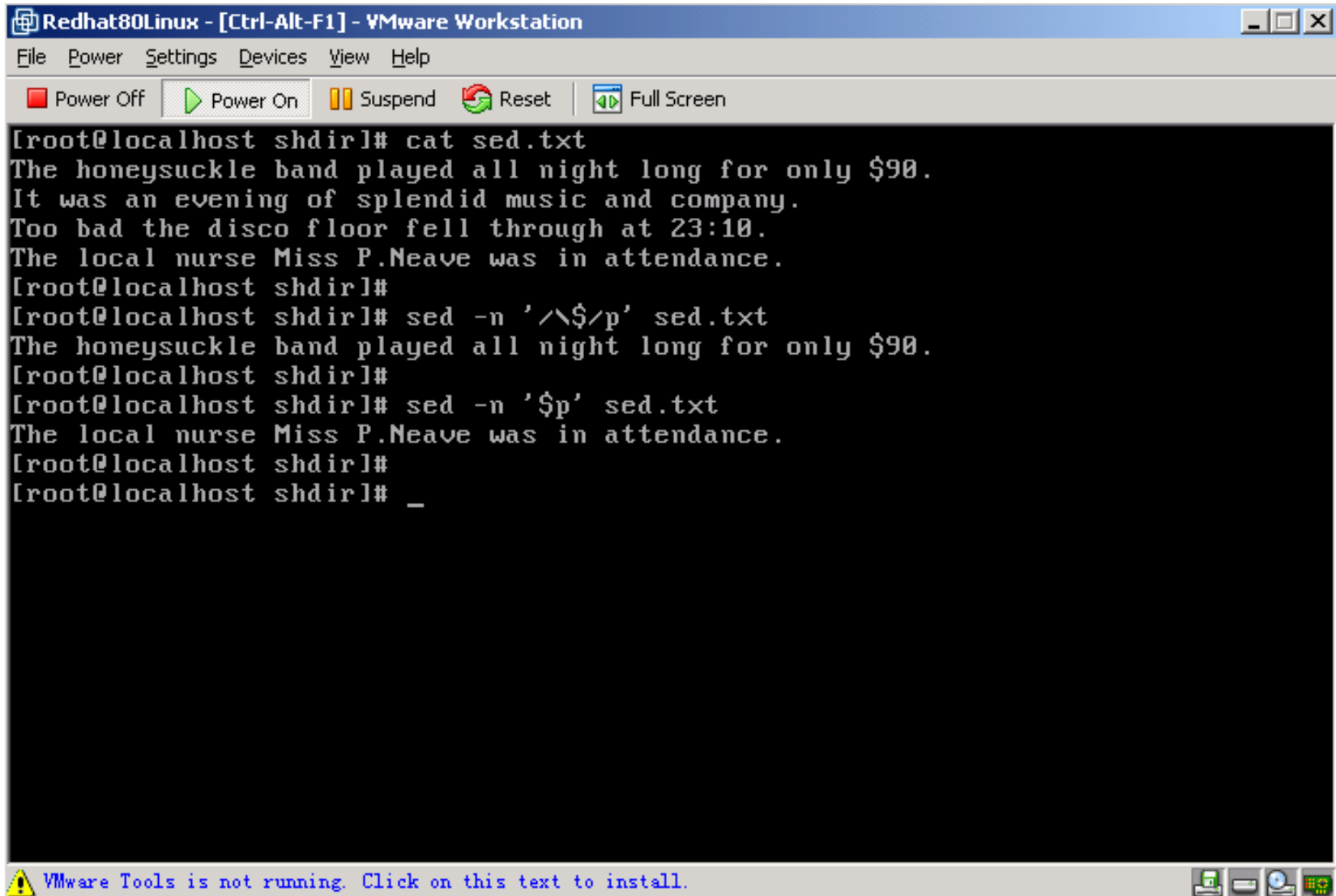


The screenshot shows a terminal window titled "Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation". The window has a menu bar with "File", "Power", "Settings", "Devices", "View", and "Help". Below the menu bar is a toolbar with buttons for "Power Off", "Power On", "Suspend", "Reset", and "Full Screen". The terminal content shows a user at the root@localhost shdir prompt. They first use the 'cat' command to view the contents of 'sed.txt', which contains four lines of text. Then, they use 'sed -n '1,3p' sed.txt' to print lines 1 through 3. Finally, they use 'sed -n '/splendid/'p sed.txt' to print only the line containing the word 'splendid'. The prompt ends with an underscore character.

```
[root@localhost shdir]# cat sed.txt
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Meave was in attendance.
[root@localhost shdir]#
[root@localhost shdir]# sed -n '1,3p' sed.txt
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
[root@localhost shdir]#
[root@localhost shdir]#
[root@localhost shdir]# sed -n '/splendid/'p sed.txt
It was an evening of splendid music and company.
[root@localhost shdir]# _
```

VMware Tools is not running. Click on this text to install.

流编辑器sed



The screenshot shows a terminal window titled "Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation". The window has a menu bar with "File", "Power", "Settings", "Devices", "View", and "Help". Below the menu bar is a toolbar with buttons for "Power Off", "Power On", "Suspend", "Reset", and "Full Screen". The terminal content shows a user at the root prompt in the /shdir directory. They first use the 'cat' command to display the contents of 'sed.txt', which contains a poem. Then, they use 'sed -n '/^\$/p' sed.txt' to print only the empty lines (line 4 and line 6). Finally, they use 'sed -n '\$p' sed.txt' to print only the last line of the file (line 6). The prompt ends with an underscore, indicating the user is waiting for input.

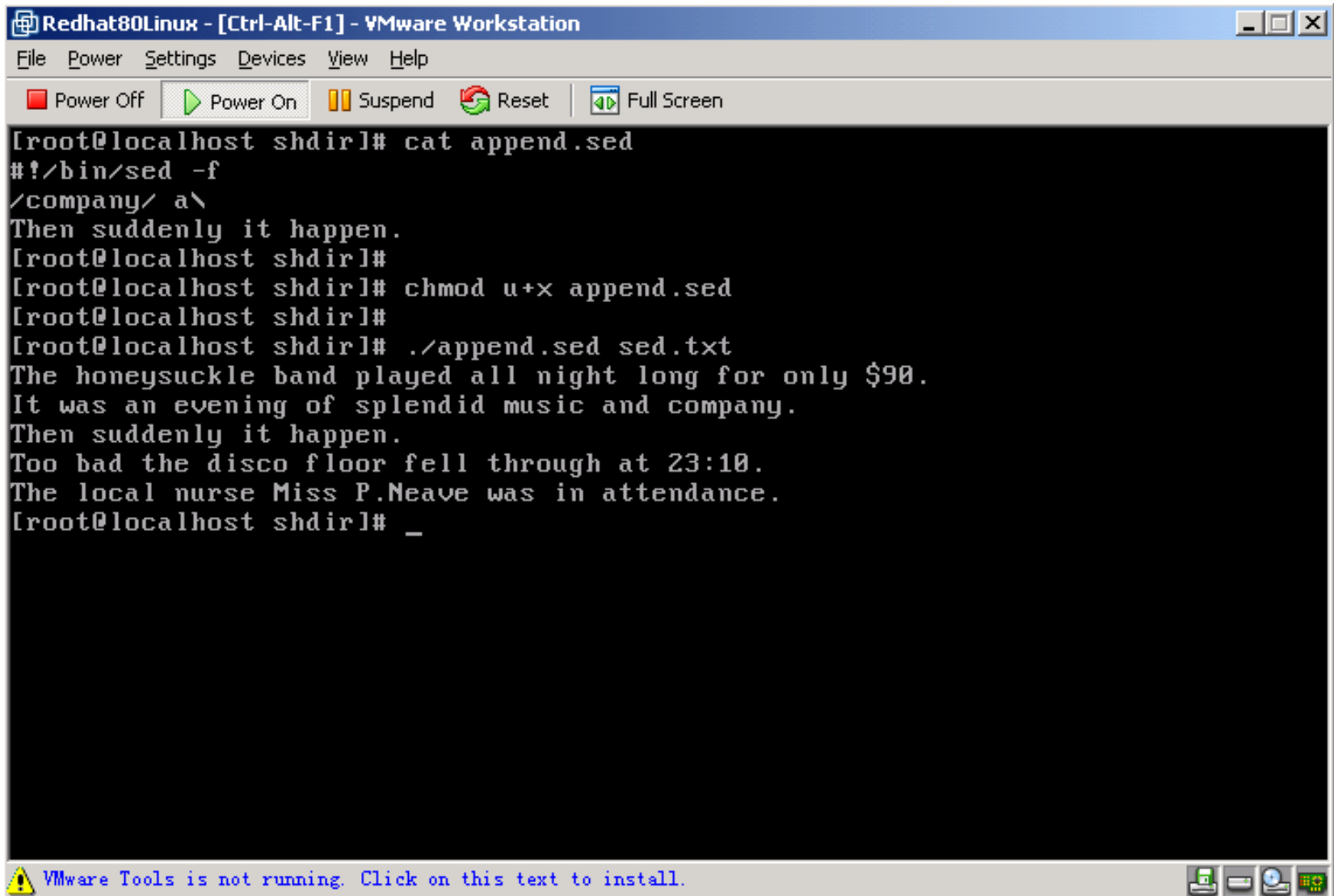
```
[root@localhost shdir]# cat sed.txt
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
[root@localhost shdir]#
[root@localhost shdir]# sed -n '/^$/p' sed.txt
The honeysuckle band played all night long for only $90.
[root@localhost shdir]#
[root@localhost shdir]# sed -n '$p' sed.txt
The local nurse Miss P.Neave was in attendance.
[root@localhost shdir]#
[root@localhost shdir]# _
```

At the bottom of the window, there is a status bar with a yellow warning icon and the text "VMware Tools is not running. Click on this text to install." followed by several small icons.

流编辑器sed

- ❖ 以上**sed**命令都是寻找匹配行，使用**sed**更好的用途是附加或修改文本的操作。
- ❖ 由于附加或修改文本的实例一般比较复杂，所以在多数情况下，都采用**sed**脚本命令格式来完成，即需要预先编辑好脚本文件。

流编辑器sed

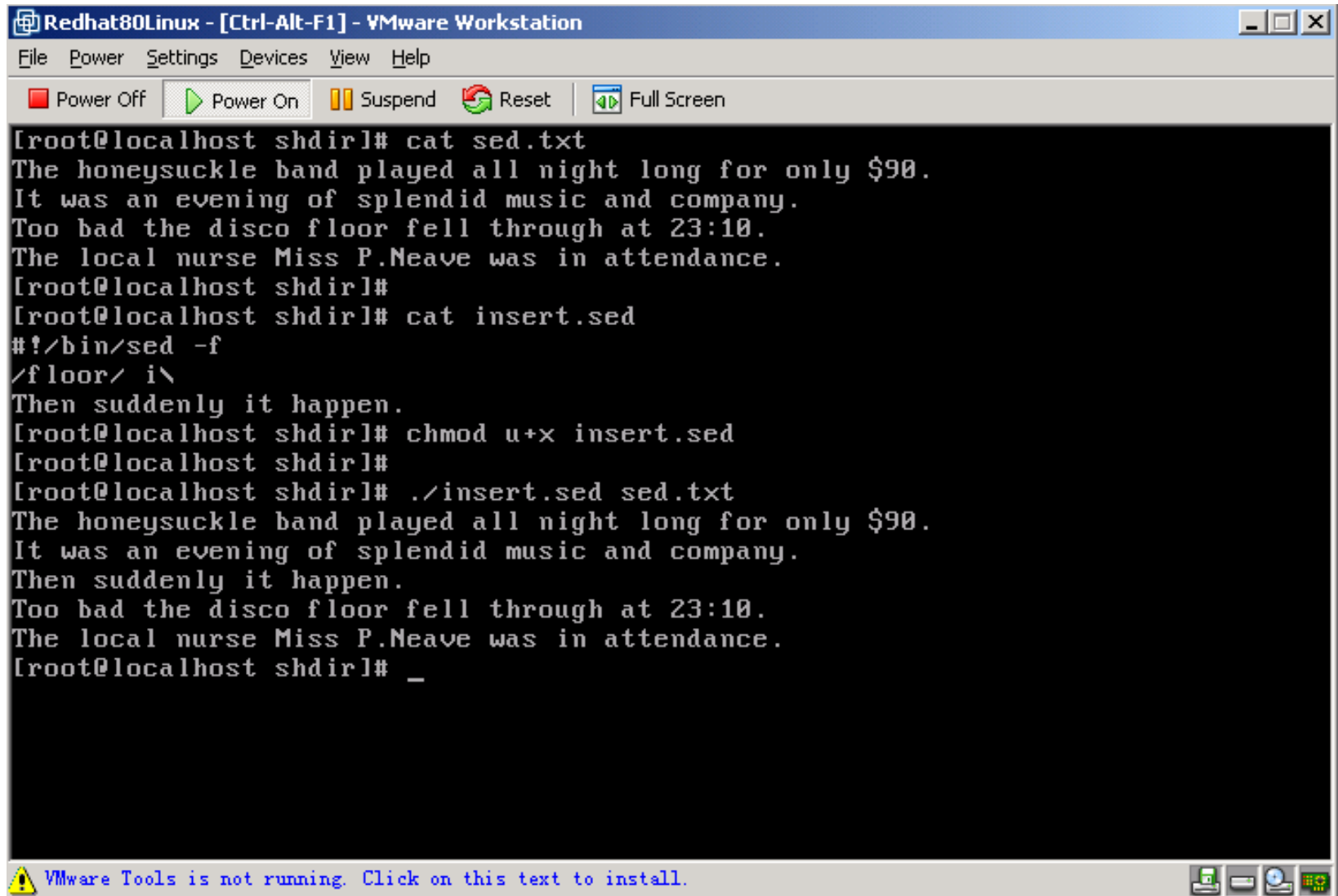


```
Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation
File Power Settings Devices View Help
Power Off Power On Suspend Reset Full Screen

[root@localhost shdir]# cat append.sed
#!/bin/sed -f
/company/ a\
Then suddenly it happen.
[root@localhost shdir]#
[root@localhost shdir]# chmod u+x append.sed
[root@localhost shdir]#
[root@localhost shdir]# ./append.sed sed.txt
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Then suddenly it happen.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
[root@localhost shdir]# _

! VMware Tools is not running. Click on this text to install.
```


流编辑器sed

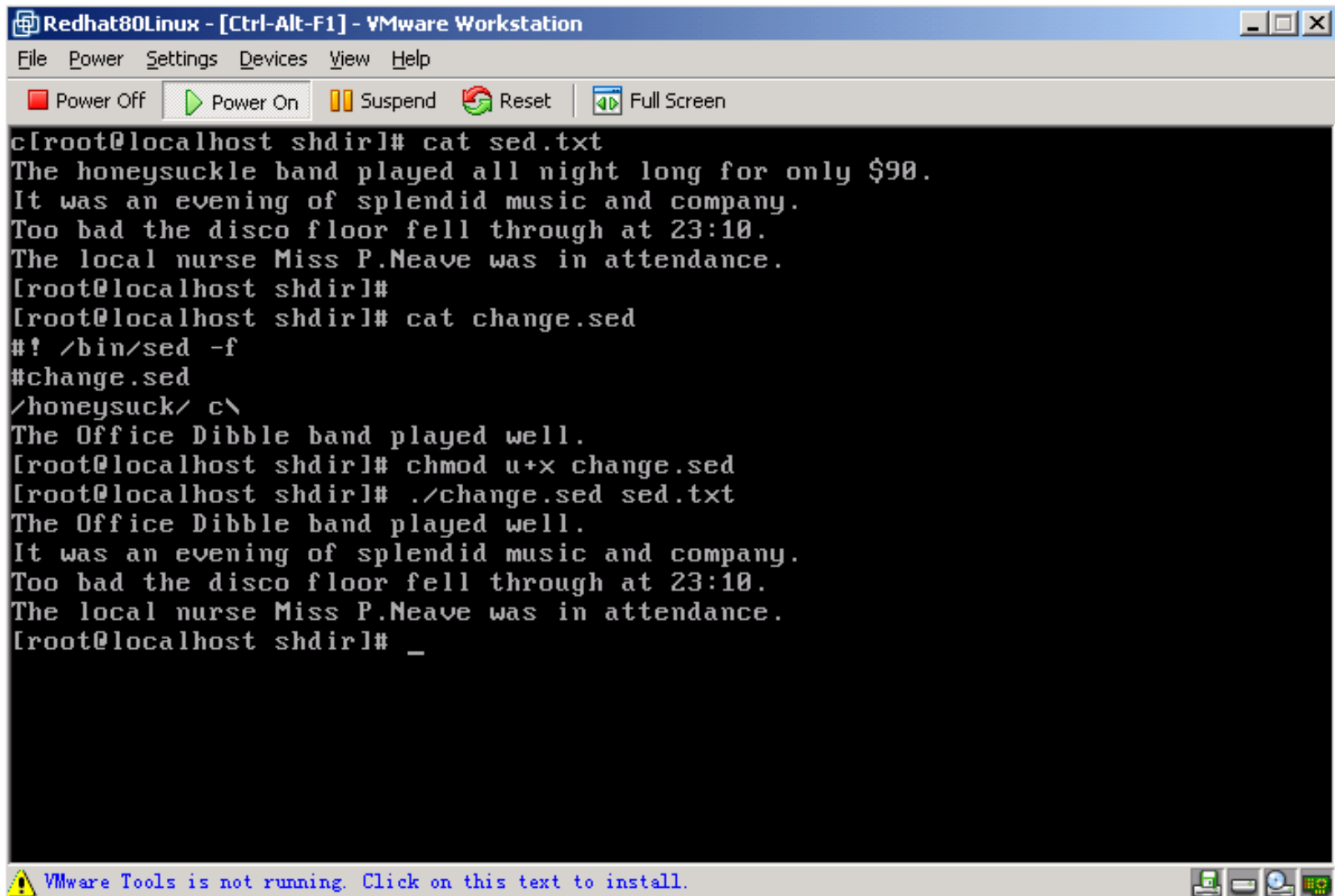


The screenshot shows a VMware Workstation window titled "Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation". The window has a menu bar (File, Power, Settings, Devices, View, Help) and a toolbar with buttons for Power Off, Power On, Suspend, Reset, and Full Screen. The main area is a terminal window with the following text:

```
[root@localhost shdir]# cat sed.txt
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
[root@localhost shdir]#
[root@localhost shdir]# cat insert.sed
#!/bin/sed -f
/floor/ i\
Then suddenly it happen.
[root@localhost shdir]# chmod u+x insert.sed
[root@localhost shdir]#
[root@localhost shdir]# ./insert.sed sed.txt
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Then suddenly it happen.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
[root@localhost shdir]# _
```

At the bottom of the window, there is a yellow warning icon and the text: "VMware Tools is not running. Click on this text to install." The bottom right corner of the window shows several system icons.

流编辑器sed

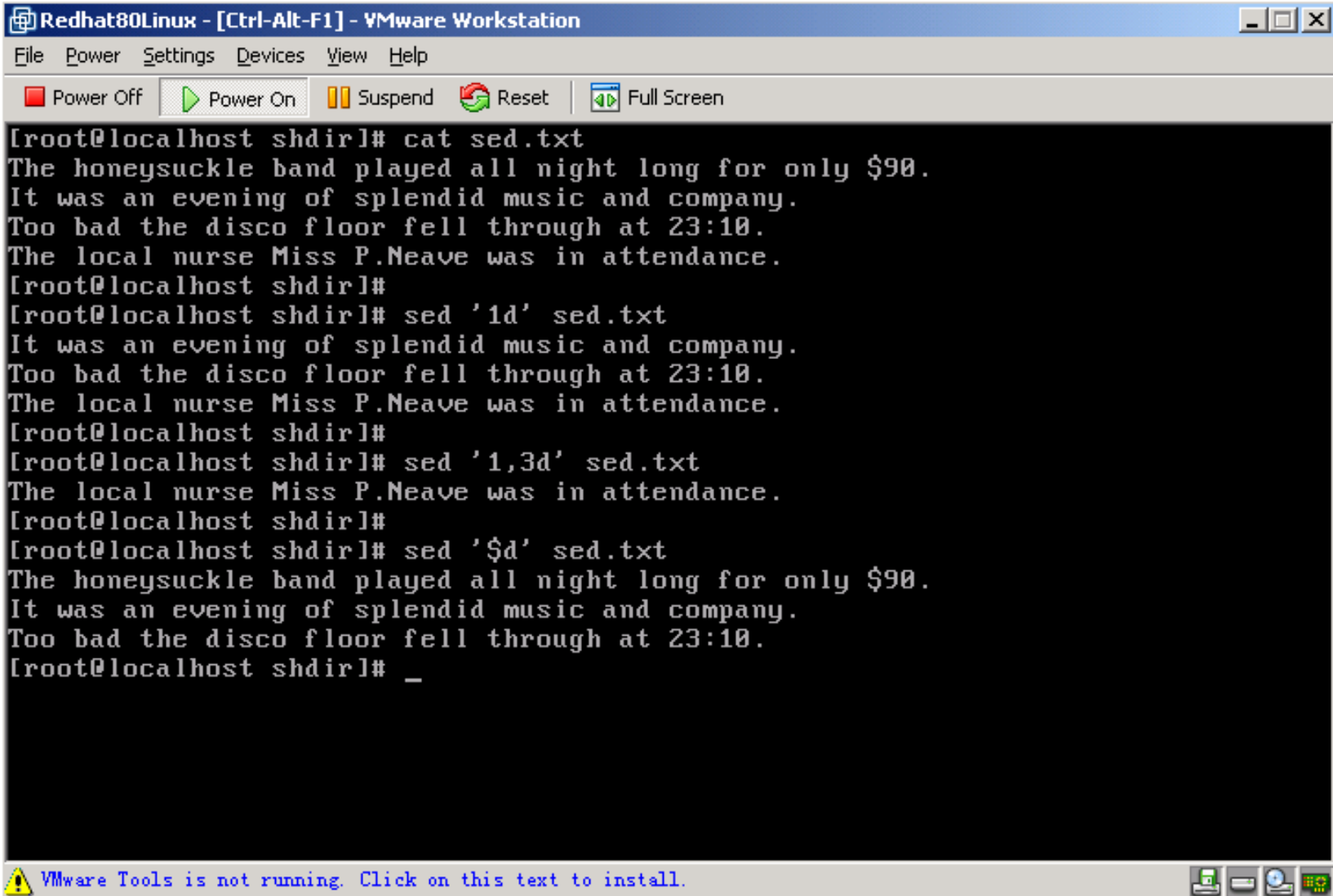


The screenshot shows a VMware Workstation window titled "Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation". The window has a menu bar (File, Power, Settings, Devices, View, Help) and a toolbar with buttons for Power Off, Power On, Suspend, Reset, and Full Screen. The main area is a terminal window with the following text:

```
c[root@localhost shdir]# cat sed.txt
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
[root@localhost shdir]#
[root@localhost shdir]# cat change.sed
#!/bin/sed -f
#change.sed
/honeysuck/ c\
The Office Dibble band played well.
[root@localhost shdir]# chmod u+x change.sed
[root@localhost shdir]# ./change.sed sed.txt
The Office Dibble band played well.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
[root@localhost shdir]# _
```

At the bottom of the window, there is a yellow warning icon and the text: "VMware Tools is not running. Click on this text to install." The bottom right corner shows standard Linux desktop icons.

流编辑器sed

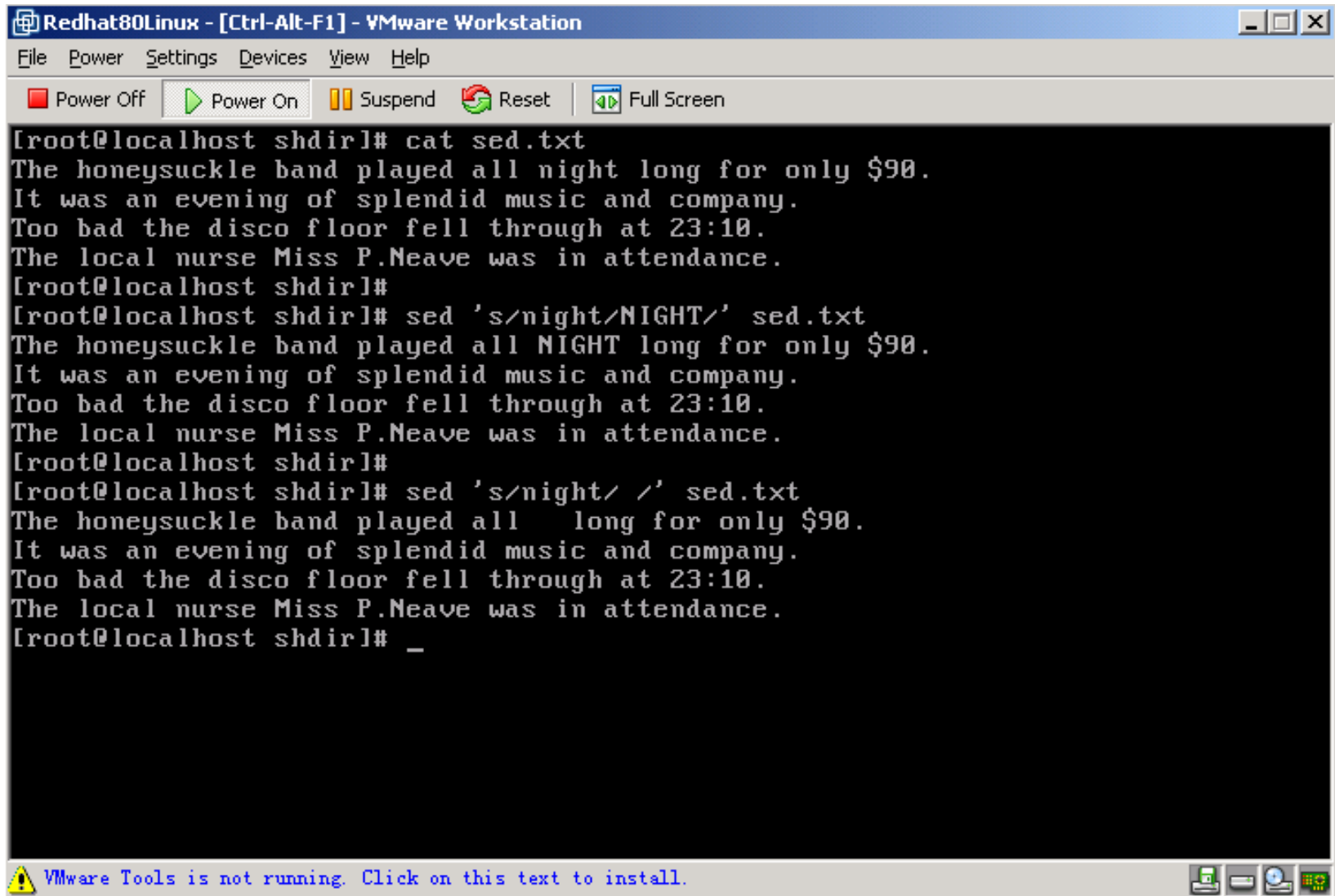


```
Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation
File Power Settings Devices View Help
Power Off Power On Suspend Reset Full Screen

[root@localhost shdir]# cat sed.txt
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
[root@localhost shdir]#
[root@localhost shdir]# sed '1d' sed.txt
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
[root@localhost shdir]#
[root@localhost shdir]# sed '1,3d' sed.txt
The local nurse Miss P.Neave was in attendance.
[root@localhost shdir]#
[root@localhost shdir]# sed '$d' sed.txt
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
[root@localhost shdir]# _

VMware Tools is not running. Click on this text to install.
```

流编辑器sed

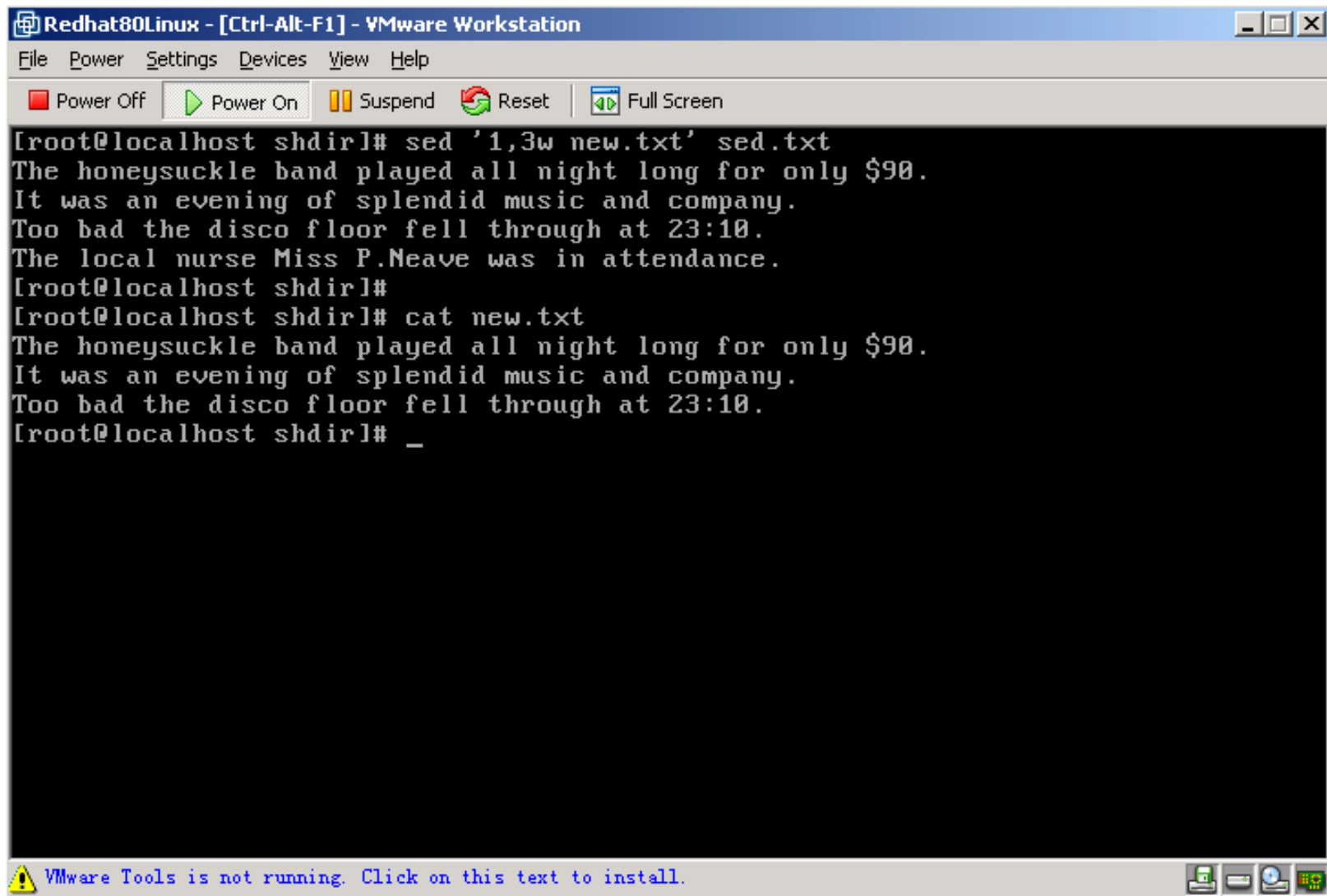


```
Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation
File Power Settings Devices View Help
Power Off Power On Suspend Reset Full Screen

[root@localhost shdir]# cat sed.txt
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
[root@localhost shdir]#
[root@localhost shdir]# sed 's/night/NIGHT/' sed.txt
The honeysuckle band played all NIGHT long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
[root@localhost shdir]#
[root@localhost shdir]# sed 's/night/ /' sed.txt
The honeysuckle band played all    long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
[root@localhost shdir]# _

! VMware Tools is not running. Click on this text to install.
```

流编辑器sed



Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation

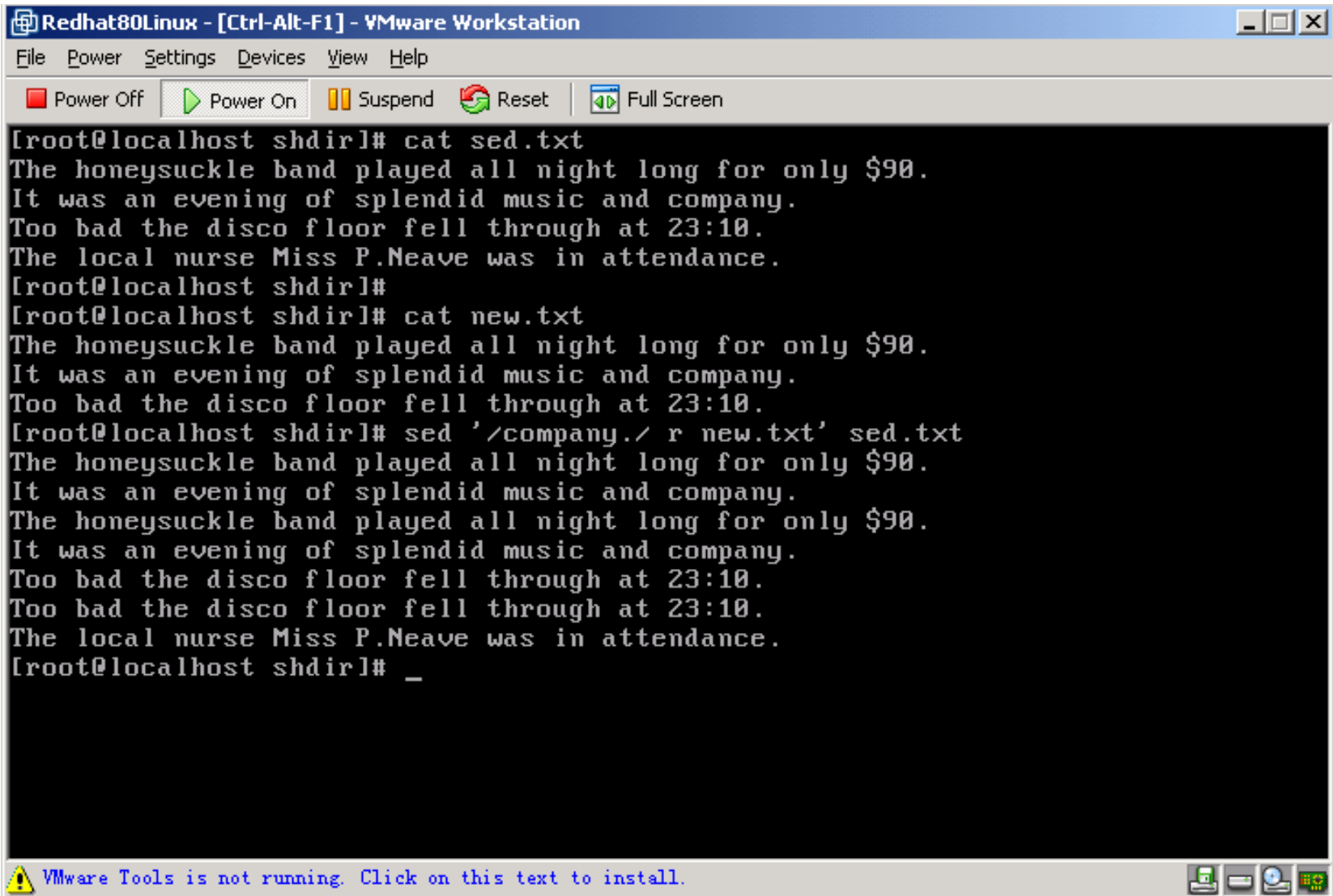
File Power Settings Devices View Help

Power Off Power On Suspend Reset Full Screen

```
[root@localhost shdir]# sed '1,3w new.txt' sed.txt
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
[root@localhost shdir]#
[root@localhost shdir]# cat new.txt
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
[root@localhost shdir]# _
```

VMware Tools is not running. Click on this text to install.

流编辑器sed



The screenshot shows a terminal window titled "Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation". The terminal displays the following commands and output:

```
[root@localhost shdir]# cat sed.txt
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
[root@localhost shdir]#
[root@localhost shdir]# cat new.txt
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
[root@localhost shdir]# sed '/company./ r new.txt' sed.txt
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
[root@localhost shdir]# _
```

At the bottom of the window, a message states: "VMware Tools is not running. Click on this text to install." The VMware Workstation interface includes a menu bar (File, Power, Settings, Devices, View, Help) and a toolbar with buttons for Power Off, Power On, Suspend, Reset, and Full Screen.

数据加工和检索工具awk

- ❖ awk最早是由三个人共同开发的，他们分别是Aho, Wenberger和Kernighan。取其三者的姓名首字母就是awk。
- ❖ awk是用于完成与数据加工和信息检索有关任务的一种编程语言。
- ❖ 特点：
 - ❖ 1、容易入门
 - ❖ 2、功能强大
 - ❖ 3、解释型语言

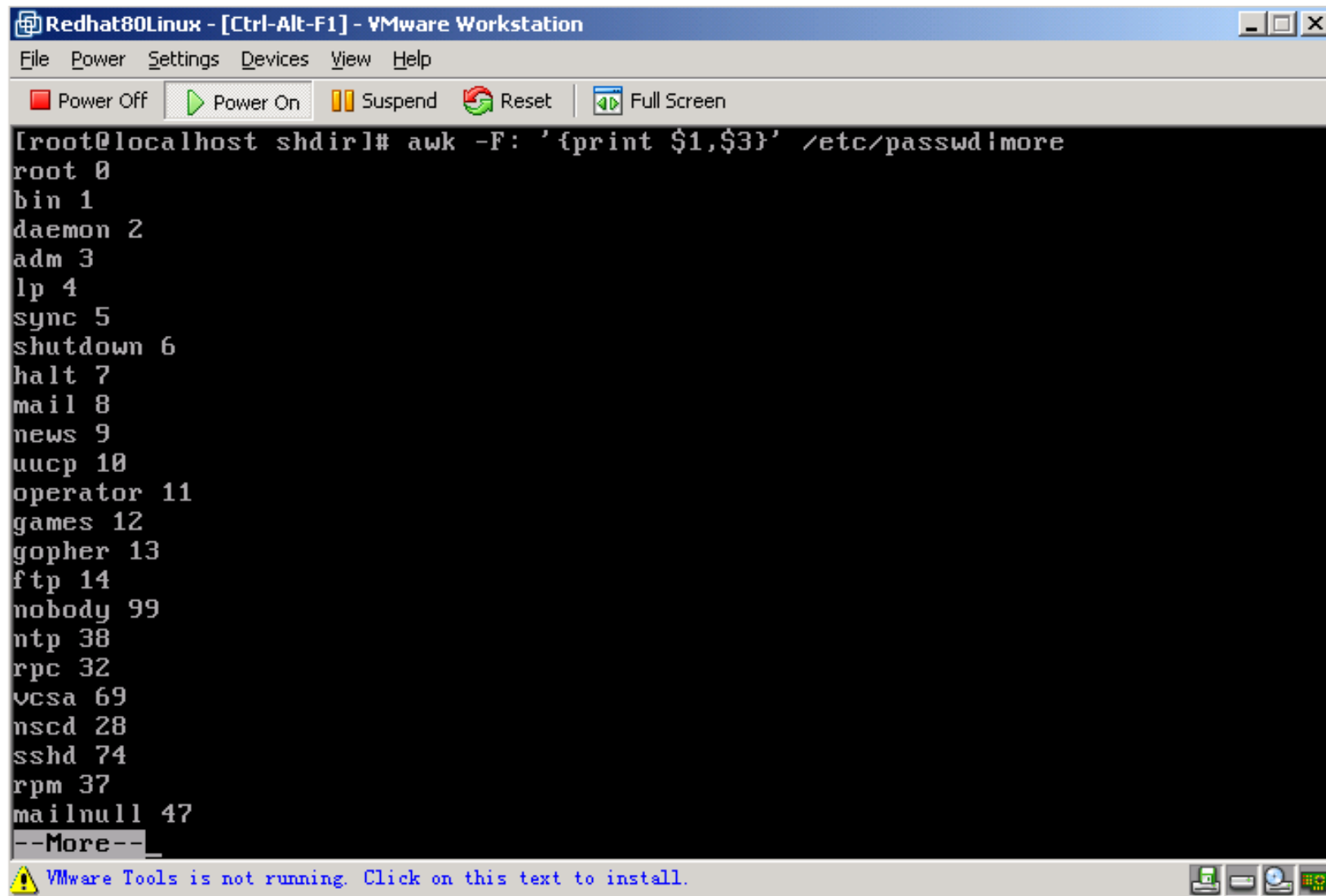
数据加工和检索工具awk

- ❖ awk的基本功能是：
- ❖ 1、逐行扫描输入行；
- ❖ 2、寻找与特定模式相匹配的行；
- ❖ 3、对该行进行相应的动作。
- ❖ 其格式如下：
- ❖ `awk [-Fchar] 'command_line' filename`
- ❖ 或者：
- ❖ `awk -f program_file filename`
- ❖ 前一种形式“-Fchar”表示确定间隔符，
“command_line”表示对一行匹配记录的操作动作，
“filename”表示作用对象文件。
- ❖ 后一种形式的“program_file”是指用户按照一定格式编制好的对象文件匹配与操作。比起前一种形式，用户可以自定义匹配模式和操作方式，所以更加灵活。

Awk的基本概念

- ❖ 字段（field）
- ❖ 每个读入记录行都可能会有好几个字段，中间用一定的分隔符间隔开，**awk**将用**\$1**、**\$2**、**\$3**.....这样的变量来访问这些字段。默认的分隔符一般是空格和制表符。如果在对象文件中表示以默认分隔符来间隔字段的那么就需要使用**-F**来说明。如：
- ❖ `awk -F: '{print $1,$3}' /etc/passwd`
- ❖ `awk -F: '{print $1,$7}' /etc/passwd |grep '/home/zhaof'`

数据加工和检索工具awk

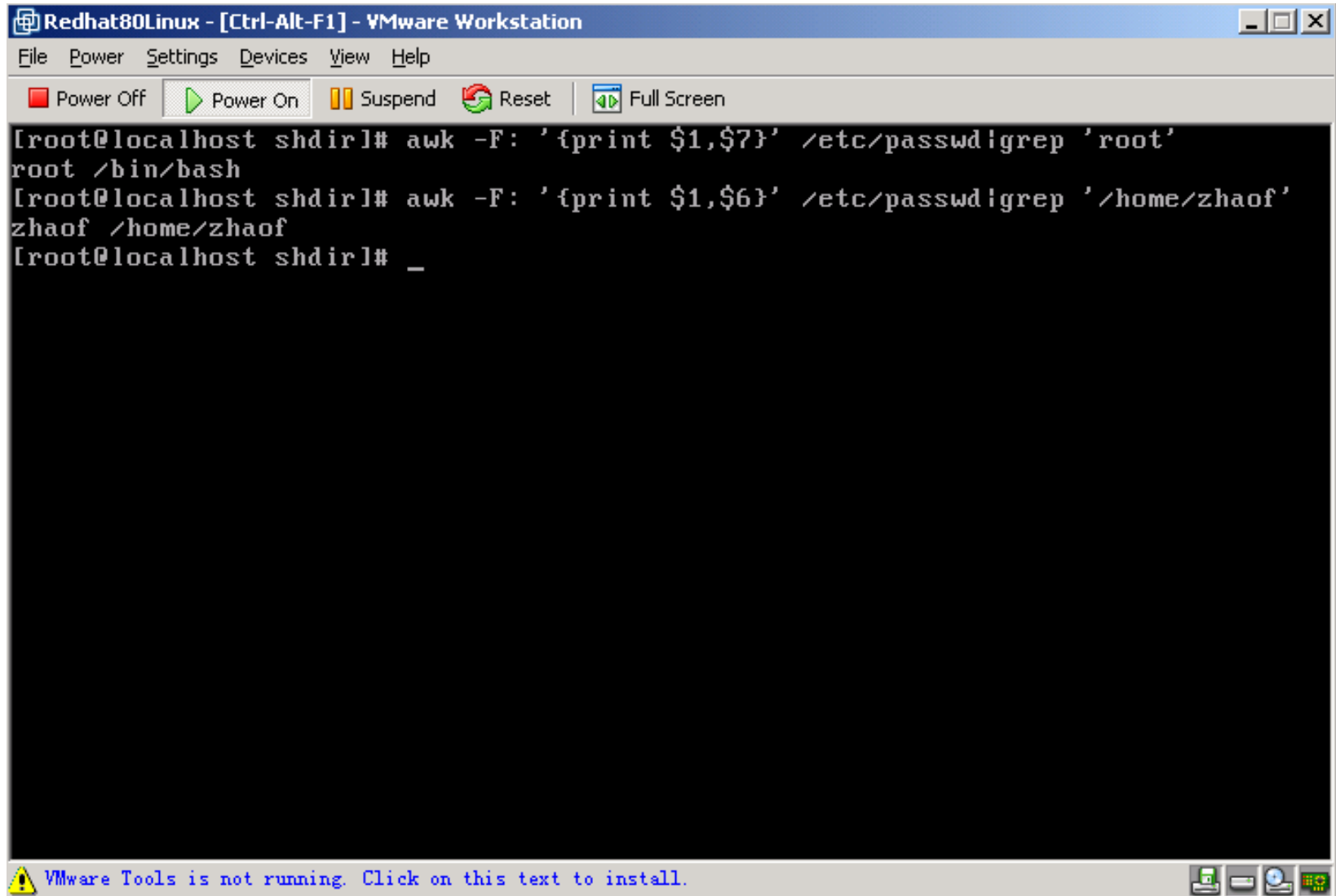


```
Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation
File Power Settings Devices View Help
Power Off Power On Suspend Reset Full Screen

[root@localhost shdir]# awk -F: '{print $1,$3}' /etc/passwd
root 0
bin 1
daemon 2
adm 3
lp 4
sync 5
shutdown 6
halt 7
mail 8
news 9
uucp 10
operator 11
games 12
gopher 13
ftp 14
nobody 99
ntp 38
rpc 32
vcsa 69
nscd 28
sshd 74
rpm 37
mailnull 47
--More--
```

! VMware Tools is not running. Click on this text to install.

数据加工和检索工具awk



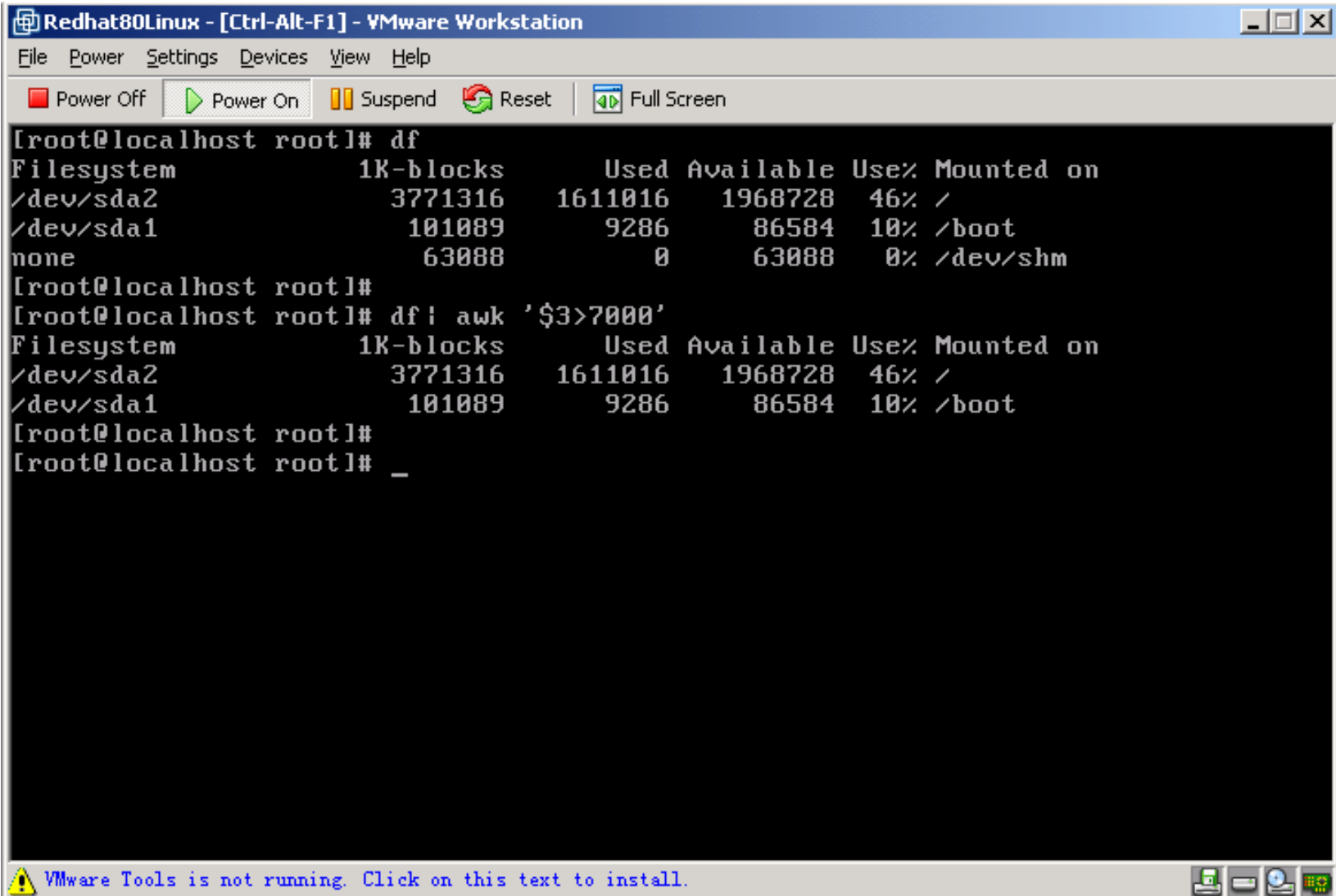
```
Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation
File Power Settings Devices View Help
Power Off Power On Suspend Reset Full Screen
[root@localhost shdir]# awk -F: '{print $1,$7}' /etc/passwd | grep 'root'
root /bin/bash
[root@localhost shdir]# awk -F: '{print $1,$6}' /etc/passwd | grep '/home/zhaof'
zhaof /home/zhaof
[root@localhost shdir]# _
```

VMware Tools is not running. Click on this text to install.

数据加工和检索工具awk

- ❖ 匹配模式一般分为三类：
- ❖ 1、关系表达式：**awk**通过一些关系运算符来说明字段是否与要求符合。
- ❖ **awk**的关系操作符有如下定义：
 - ❖ < 小于
 - ❖ <= 小于或等于
 - ❖ == 等于
 - ❖ != 不等于
 - ❖ >= 大于或等于
 - ❖ > 大于
 - ❖ ~ 与正则表达式匹配
 - ❖ !~ 不与正则表达式匹配
- ❖ 2、正则表达式：**awk**中的正则表达式用//括起来。它的规则与**sed**的正则表达式规则相同。

数据加工和检索工具awk

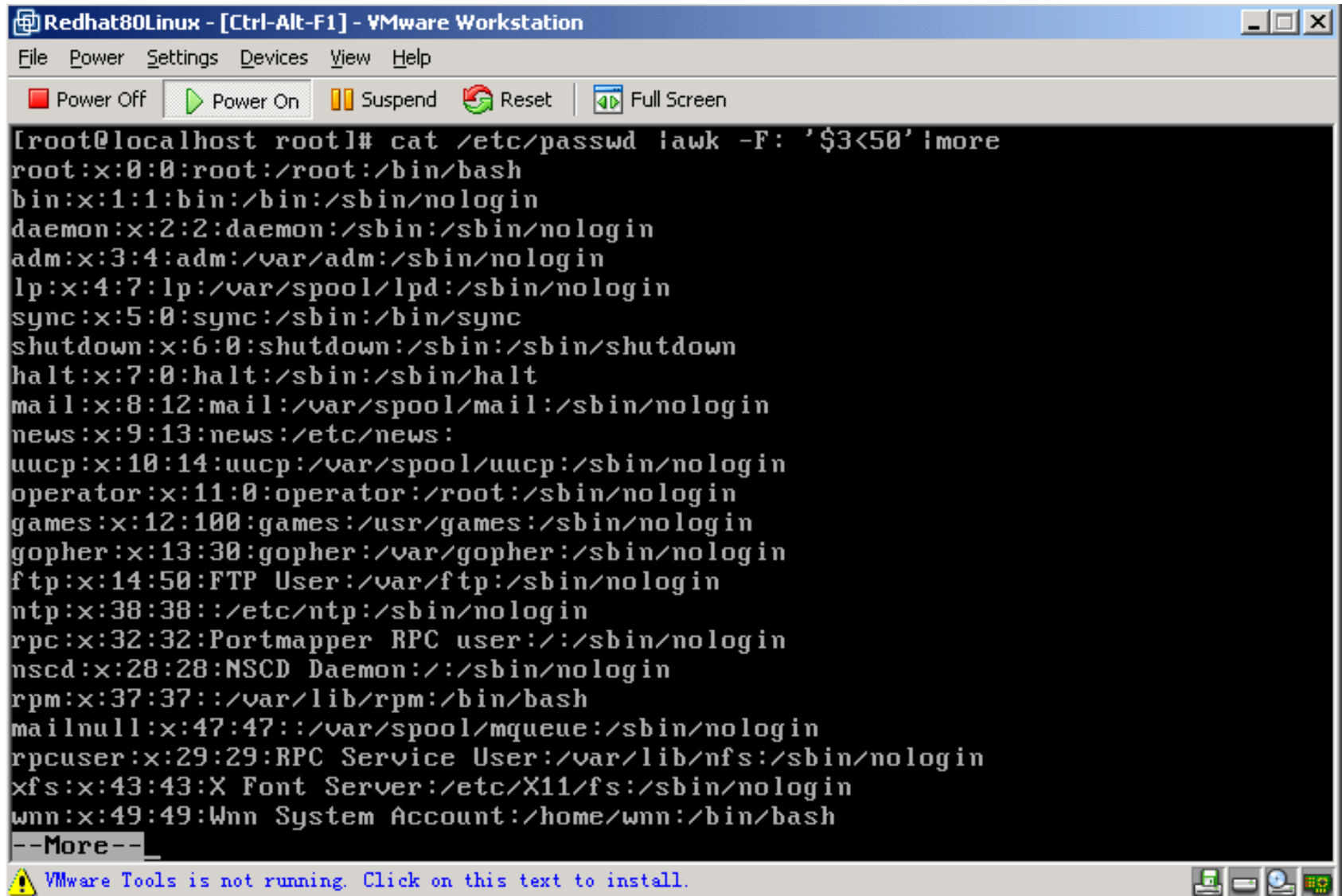


```
Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation
File Power Settings Devices View Help
Power Off Power On Suspend Reset Full Screen

[root@localhost root]# df
Filesystem      1K-blocks    Used Available Use% Mounted on
/dev/sda2        3771316   1611016   1968728   46% /
/dev/sda1        101089     9286     86584    10% /boot
none             63088        0     63088     0% /dev/shm
[root@localhost root]#
[root@localhost root]# df | awk '$3>7000'
Filesystem      1K-blocks    Used Available Use% Mounted on
/dev/sda2        3771316   1611016   1968728   46% /
/dev/sda1        101089     9286     86584    10% /boot
[root@localhost root]#
[root@localhost root]# _
```

VMware Tools is not running. Click on this text to install.

数据加工和检索工具awk



```
Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation
File Power Settings Devices View Help
Power Off Power On Suspend Reset Full Screen

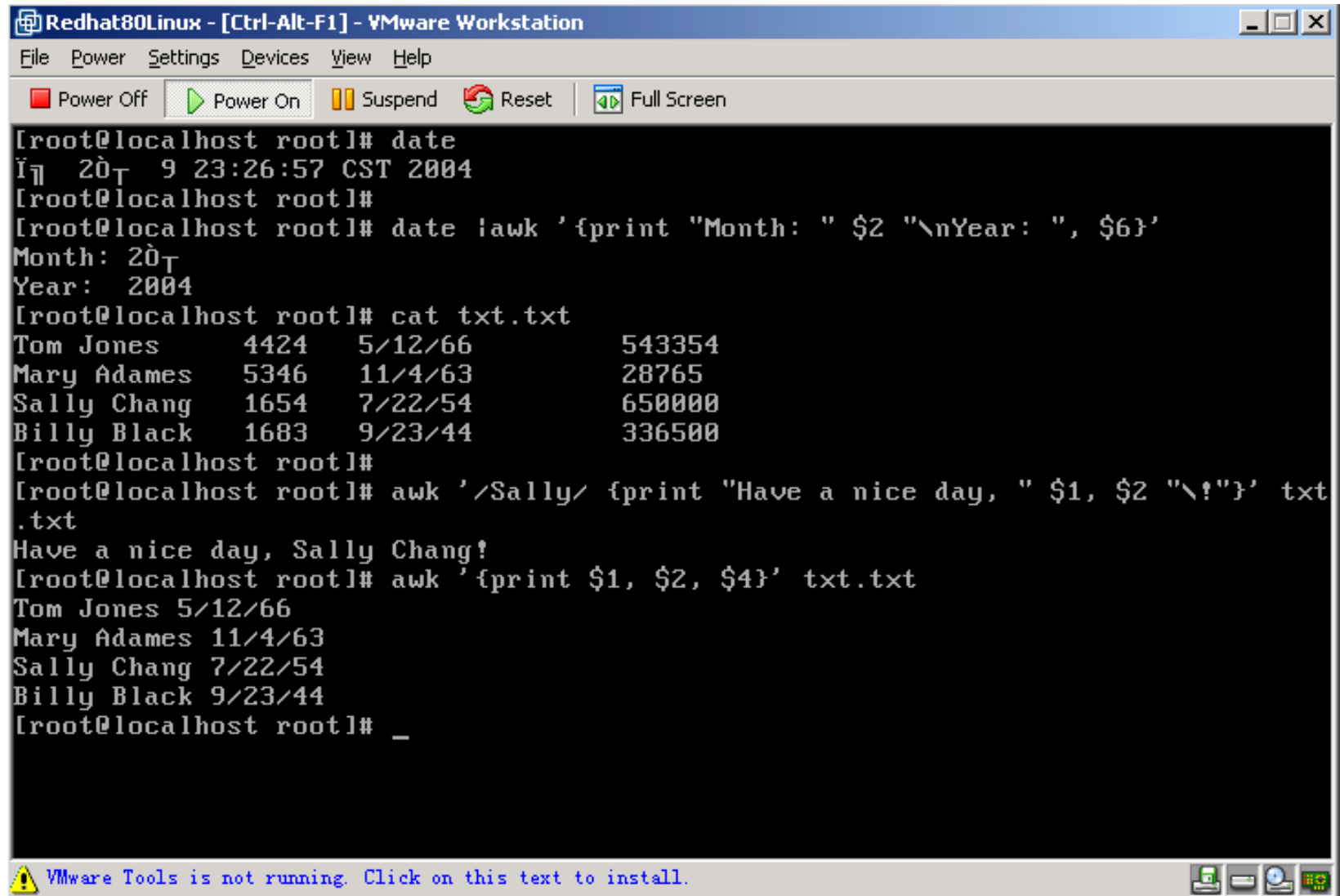
[root@localhost root]# cat /etc/passwd |awk -F: '$3<50' {more
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
news:x:9:13:news:/etc/news:
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
gopher:x:13:30:gopher:/var/gopher:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
ntp:x:38:38::/etc/ntp:/sbin/nologin
rpc:x:32:32:Portmapper RPC user::/sbin/nologin
nscd:x:28:28:NSCD Daemon::/sbin/nologin
rpm:x:37:37::/var/lib/rpm:/bin/bash
mailnull:x:47:47::/var/spool/mqueue:/sbin/nologin
rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin
xfs:x:43:43:X Font Server:/etc/X11/fs:/sbin/nologin
wnn:x:49:49:Wnn System Account:/home/wnn:/bin/bash
--More--_

! VMware Tools is not running. Click on this text to install.
```

数据加工和检索工具awk

- ❖ 动作（**action**）
- ❖ 有关需要**awk**执行的操作部分封装在花括号中。
- ❖ 执行的操作可以是打印输出“**print**”、字符串运算、数学计算和统计工作。而且**awk**还可以定义变量、数组和函数等。
- ❖ **print**函数可以用来显示**awk**操作的变元。**Print**函数接受如变量、计算所得的数值或串常数等变元。其中：串必须封装在双引号中，逗号用来分隔变元，如果没有分隔符，则输出的变元就串联在一起。
- ❖ 在**print**函数中还可以使用下列转义符：
 - ❖ **\b** 退格 **\f** 换页 **\n** 换行
 - ❖ **\r** 回车 **\t** 跳格
 - ❖ **\047** 八进制数**47**，即一个单引号
 - ❖ **\c** **c**代表其他任意字符，例如\

数据加工和检索工具awk

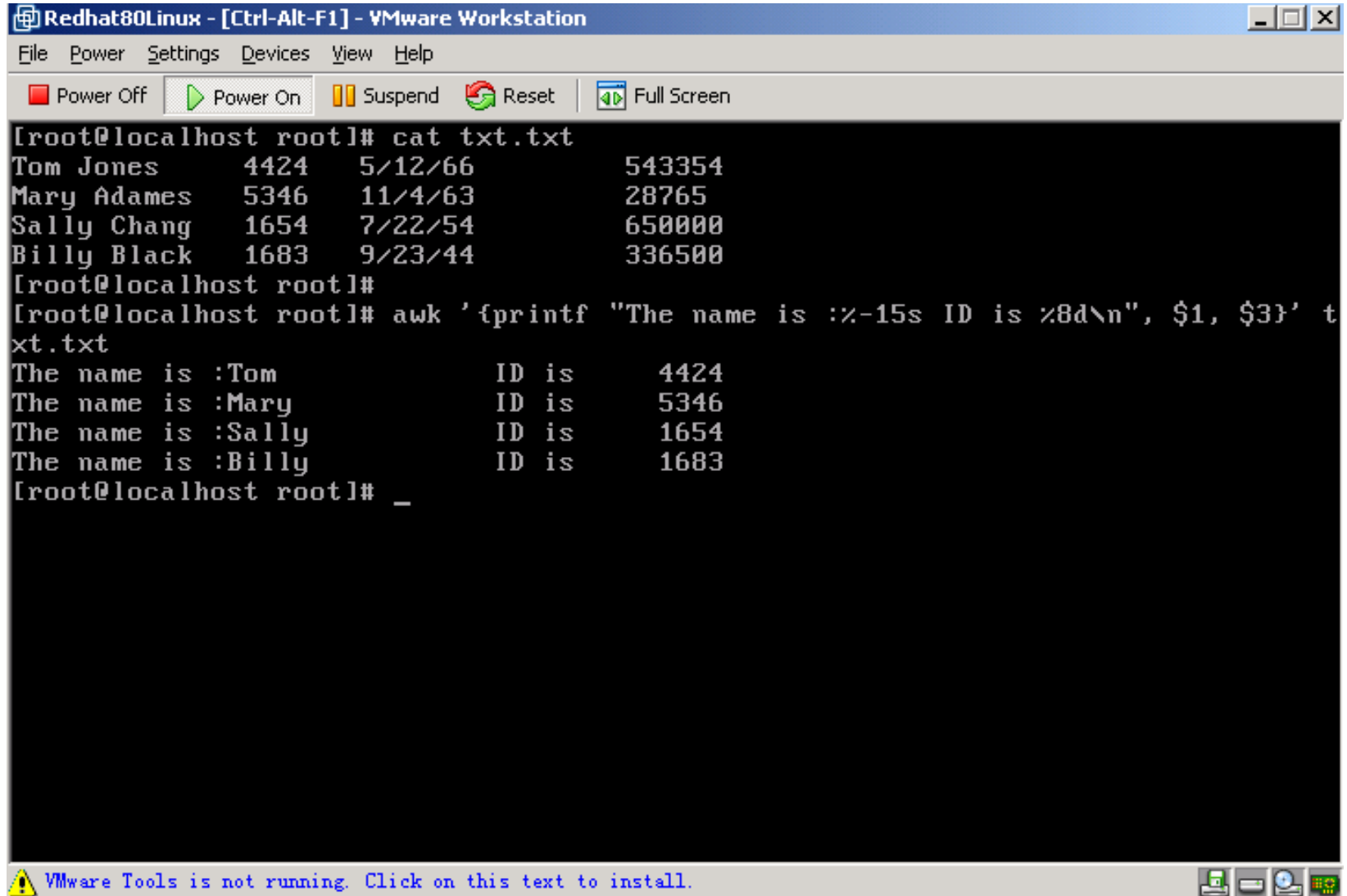


```
Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation
File Power Settings Devices View Help
Power Off Power On Suspend Reset Full Screen

[root@localhost root]# date
i  2004  9 23:26:57 CST 2004
[root@localhost root]#
[root@localhost root]# date |awk '{print "Month: " $2 "\nYear: ", $6}'
Month: 2004
Year: 2004
[root@localhost root]# cat txt.txt
Tom Jones      4424    5/12/66      543354
Mary Adames    5346    11/4/63      28765
Sally Chang    1654    7/22/54      650000
Billy Black    1683    9/23/44      336500
[root@localhost root]#
[root@localhost root]# awk '/Sally/ {print "Have a nice day, " $1, $2 "\n!"}' txt.txt
Have a nice day, Sally Chang!
[root@localhost root]# awk '{print $1, $2, $4}' txt.txt
Tom Jones 5/12/66
Mary Adames 11/4/63
Sally Chang 7/22/54
Billy Black 9/23/44
[root@localhost root]# _
```

VMware Tools is not running. Click on this text to install.

数据加工和检索工具awk



The screenshot shows a VMware Workstation window titled "Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation". The window has a menu bar (File, Power, Settings, Devices, View, Help) and a toolbar with buttons for Power Off, Power On, Suspend, Reset, and Full Screen. The terminal window displays the following commands and output:

```
[root@localhost root]# cat txt.txt
Tom Jones      4424    5/12/66      543354
Mary Adames    5346    11/4/63     28765
Sally Chang    1654    7/22/54     650000
Billy Black    1683    9/23/44     336500
[root@localhost root]#
[root@localhost root]# awk '{printf "The name is :%-15s ID is %8d\n", $1, $3}' t
xt.txt
The name is :Tom           ID is      4424
The name is :Mary         ID is      5346
The name is :Sally        ID is      1654
The name is :Billy        ID is      1683
[root@localhost root]# _
```

At the bottom of the window, there is a yellow warning icon and the text: "VMware Tools is not running. Click on this text to install." To the right of this text are icons for a CD, a floppy disk, and a network connection.

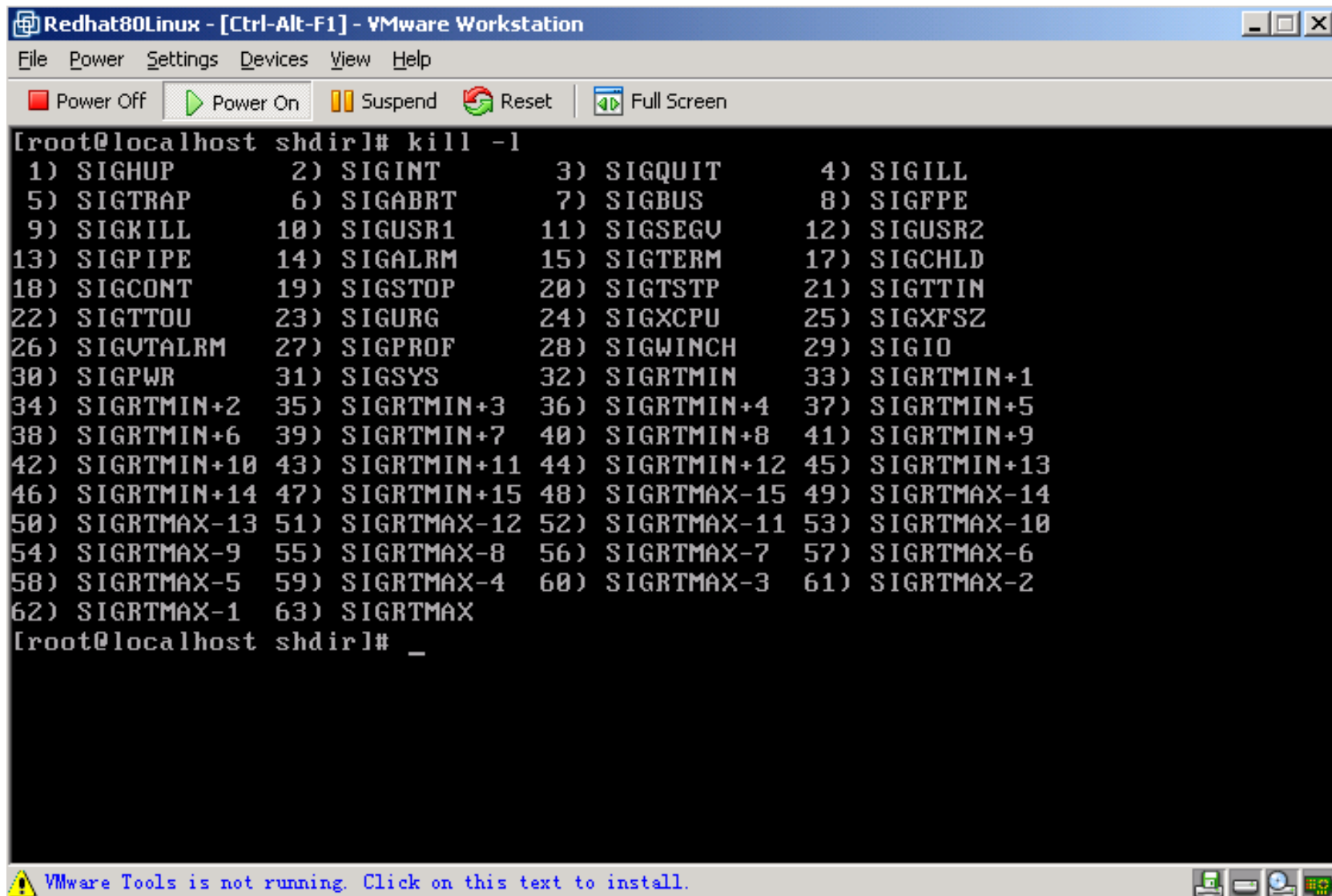
数据加工和检索工具awk

- ❖ 除了以上基本功能和基础操作外，**awk**还为用户提供了丰富的内部变量、用户定义变量、算术运算、赋值运算等丰富的运算功能。
- ❖ **awk**还可以使用C语言的许多高级运算符，如“++”、“+=”等都可以使用。
- ❖ 同时**awk**还提供了如“atan2(y,x)”、“cos(x)”、“exp()”、“log()”等很多内部算术函数。
- ❖ **awk**还和C语言一样，提供了针对字符串操作的运算符。同时，可以通过连接常量、变量、数组元素、函数和其他表达式来创建串表达式，提供了丰富的串操作功能。而且，**awk**利用内部自身所具备的强大的字符串操作函数完成字符串操作。

UNIX的基本概念

- ❖ 信号（signal）：又称为软中断信号，用于在进程或线程中实现同步或异步通信。通知进程发生了某个事件。
- ❖ 信号的特性：异步性，即进程在执行期中任何时刻都可能收到信号。
- ❖ 进程对信号的处理：
 - ❖ 执行预先安排的处理程序。
 - ❖ 忽略某个信号。
 - ❖ 对信号进行处理，并保留系统的默认值。

UNIX的基本概念



The screenshot shows a terminal window titled "Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation". The window has a menu bar with "File", "Power", "Settings", "Devices", "View", and "Help". Below the menu bar is a toolbar with buttons for "Power Off", "Power On", "Suspend", "Reset", and "Full Screen". The terminal content shows the command `[root@localhost shdir]# kill -l` and its output, which is a list of 64 signals arranged in four columns. The signals are: 1) SIGHUP, 2) SIGINT, 3) SIGQUIT, 4) SIGILL, 5) SIGTRAP, 6) SIGABRT, 7) SIGBUS, 8) SIGFPE, 9) SIGKILL, 10) SIGUSR1, 11) SIGSEGV, 12) SIGUSR2, 13) SIGPIPE, 14) SIGALRM, 15) SIGTERM, 17) SIGCHLD, 18) SIGCONT, 19) SIGSTOP, 20) SIGTSTP, 21) SIGTTIN, 22) SIGTTOU, 23) SIGURG, 24) SIGXCPU, 25) SIGXFSZ, 26) SIGUTALRM, 27) SIGPROF, 28) SIGWINCH, 29) SIGIO, 30) SIGPWR, 31) SIGSYS, 32) SIGRTMIN, 33) SIGRTMIN+1, 34) SIGRTMIN+2, 35) SIGRTMIN+3, 36) SIGRTMIN+4, 37) SIGRTMIN+5, 38) SIGRTMIN+6, 39) SIGRTMIN+7, 40) SIGRTMIN+8, 41) SIGRTMIN+9, 42) SIGRTMIN+10, 43) SIGRTMIN+11, 44) SIGRTMIN+12, 45) SIGRTMIN+13, 46) SIGRTMIN+14, 47) SIGRTMIN+15, 48) SIGRTMAX-15, 49) SIGRTMAX-14, 50) SIGRTMAX-13, 51) SIGRTMAX-12, 52) SIGRTMAX-11, 53) SIGRTMAX-10, 54) SIGRTMAX-9, 55) SIGRTMAX-8, 56) SIGRTMAX-7, 57) SIGRTMAX-6, 58) SIGRTMAX-5, 59) SIGRTMAX-4, 60) SIGRTMAX-3, 61) SIGRTMAX-2, 62) SIGRTMAX-1, 63) SIGRTMAX. The prompt `[root@localhost shdir]#` is shown at the bottom of the terminal.

```
[root@localhost shdir]# kill -l
 1) SIGHUP       2) SIGINT       3) SIGQUIT      4) SIGILL
 5) SIGTRAP      6) SIGABRT      7) SIGBUS       8) SIGFPE
 9) SIGKILL     10) SIGUSR1     11) SIGSEGV     12) SIGUSR2
13) SIGPIPE     14) SIGALRM     15) SIGTERM     17) SIGCHLD
18) SIGCONT     19) SIGSTOP     20) SIGTSTP     21) SIGTTIN
22) SIGTTOU     23) SIGURG      24) SIGXCPU     25) SIGXFSZ
26) SIGUTALRM   27) SIGPROF     28) SIGWINCH    29) SIGIO
30) SIGPWR      31) SIGSYS      32) SIGRTMIN    33) SIGRTMIN+1
34) SIGRTMIN+2  35) SIGRTMIN+3  36) SIGRTMIN+4  37) SIGRTMIN+5
38) SIGRTMIN+6  39) SIGRTMIN+7  40) SIGRTMIN+8  41) SIGRTMIN+9
42) SIGRTMIN+10 43) SIGRTMIN+11 44) SIGRTMIN+12 45) SIGRTMIN+13
46) SIGRTMIN+14 47) SIGRTMIN+15 48) SIGRTMAX-15 49) SIGRTMAX-14
50) SIGRTMAX-13 51) SIGRTMAX-12 52) SIGRTMAX-11 53) SIGRTMAX-10
54) SIGRTMAX-9  55) SIGRTMAX-8  56) SIGRTMAX-7  57) SIGRTMAX-6
58) SIGRTMAX-5  59) SIGRTMAX-4  60) SIGRTMAX-3  61) SIGRTMAX-2
62) SIGRTMAX-1  63) SIGRTMAX

[root@localhost shdir]#
```

! VMware Tools is not running. Click on this text to install.

UNIX的基本概念

- ❖ 用户系统中所有能被理解的信号都列在C语言的头文件`signal.h`中。该文件的位置随着版本的不同而不同。
- ❖ Linux: `/usr/include/asm/signal.h`
- ❖ 一些供应商提供该文件的帮助文件:
- ❖ 在Linux中: `man 7 signal`

缺省动作

- ❖ 一个信号的缺省动作是一个脚本或程序在接收到一个信号时执行的动作。
- ❖ 一些可能的缺省动作是：
 - ❖ 1、终止进程。
 - ❖ 2、忽略信号。
 - ❖ 3、内核转储：创建一个名为**core**的文件，保存接到信号时进程的内存镜像。
 - ❖ 4、停止进程。
 - ❖ 5、继续一个停止的进程。

传递信号

- ❖ 最常用的是在脚本执行时按下`control_C`或`INTERRUPT`键。向脚本传送一个`SIGINT`信号并终止脚本。
- ❖ 其他传递信号的常用方法是使用`kill`命令。
- ❖ `kill -signal pid`
- ❖ 如：
 - ❖ 1、 `kill pid`
 - ❖ 等价于： `kill -s SIGTERM pid`
 - ❖ 2、 `kill -s SIGHUP pid`
 - ❖ 3、 `kill -s SIGQUIT pid` 或 `kill -s SIGINT pid`
 - ❖ 4、 `kill -9 pid`

处理信号

- ❖ 一个程序或脚本可以用三种不同方式来处理信号：
- ❖ 不做任何处理而让缺省动作发生。
- ❖ 忽略信号并继续执行。
- ❖ 捕获信号并执行一些信号特定命令。使用这种方式的脚本在收到一个信号时有一个特殊的过程来执行。这是处理信号最复杂也是最强大的方式。

俘获信号

- ❖ 格式:
- ❖ `trap `command; command` signal-number`
- ❖ 实例:
- ❖ `trap `rm tmp*; exit 1 ` 1 2 15`
- ❖ 重置信号: 把信号重置为它的默认行为。
- ❖ `trap 2`
- ❖ 忽略信号: 如果`trap`命令后跟一对空引号, 则列出的信号将被进程忽略。
- ❖ `trap "" 1 2`
- ❖ 列出陷阱: 列出所有陷阱和赋给它的命令。
- ❖ `trap`

俘获信号

- ❖ 在shell脚本中有三种trap的常用方法：
- ❖ 1、清除临时文件。
- ❖ 2、一直忽略信号。
- ❖ 3、只在关键操作期间忽略信号。
- ❖ 另外还有设置一个计时器。

Trap命令

- ❖ **trap**命令最简单的用途就是用它忽略信号。
- ❖ 比如：用它忽略信号**CTRL_C**组合键。
- ❖ **trap "" SIGINT**
- ❖ 或
- ❖ **trap "" 2**
- ❖ 用户按**CTRL_C**将不会导致程序中断。

清除临时文件

❖ cleanUp()

```
❖ {  
    if [ -f "$OUTFILE" ]  
    then  
        printf "Cleaning up ..."  
        rm -f "$OUTFILE" 2> /dev/null  
        echo "Done..."  
    fi  
}
```

```
#main program  
echo "Please input the file you want to delete"  
read OUTFILE  
trap cleanUp 1 2 3 15
```


Trap命令



```
C:\WINNT\System32\telnet.exe
[Bossdev]$ cat test_trap
#!/usr/bin/sh
#trap "" SIGINT
input="a"
while [ $input = "a" ]
do
    echo "Type 'a' to continue,anything else to quit:"
    read input
done
echo "The script is done."

[Bossdev]$ ./test_trap
Type 'a' to continue,anything else to quit:
a
Type 'a' to continue,anything else to quit:
a
Type 'a' to continue,anything else to quit:
[Bossdev]$ _
```

Trap命令



```
C:\WINNT\System32\telnet.exe
[Bossdev]$ cat test_trap
#!/usr/bin/sh
trap "" SIGINT
input="a"
while [ $input = "a" ]
do
    echo "Type 'a' to continue,anything else to quit:"
    read input
done
echo "The script is done."

[Bossdev]$ ./test_trap
Type 'a' to continue,anything else to quit:
a
Type 'a' to continue,anything else to quit:
x
The script is done.
[Bossdev]$ _
```

函数里的陷阱

- ❖ 如果用户在一个函数里使用陷阱来处理信号，一旦函数被调用，它将会影响整个命令表。该陷阱对于系统命令表是全局的。
- ❖ 在下面的例子中，设置陷阱来忽略中断键 $\wedge C$ 。注意，必须用`kill`命令来取消该命令，才能停止程序的循环，当函数使用陷阱时，有潜在的副作用。

函数里的陷阱

Redhat80Linux - [Ctrl-Alt-F1] - VMware Workstation

File Power Settings Devices View Help

Power Off Power On Suspend Reset Full Screen

```
#!/bin/sh
trapper()
{
    echo "In trapper "
    _trap `echo "Caught in a trap!"` 2
    #Once set, this trap affects the entire script, Anytime
    #^c is entered, the script will ignore it.
}

#mainprog
while true
do
    echo "In the main script"
    trapper
    echo "Strill in main"
    sleep 5
done

~
~
~
~
~
~
~

"trapper" [17L, 278C] 5,1-8
```

VMware Tools is not running. Click on this text to install.

Thank You !

zhaofang@email.buptsse.cn

