ID: LAB-001

SINE NOMINE
ASSOCIATES

Title: Linux Lab Workbook

Version: 1.0.0

Date: 10 August 2015

Author: Neale Ferguson

Contents

1.0 Linux Lab Workbook

For each of the labs take the time to find out what each of the commands do. Use the **`man`** command to display what options the command takes, what its effects are, and what type of things to expect.

One of the key features of any UNIX type system is there is usually a number of different ways to achieve the same result. If your answer doesn't match mine but you get the right result then consider it correct!

1.1 Requirements

Before you start the exercises check the following sections to ensure you have the materials you need to run.

1.1.1 User ID

You will need to have a user id and password to logon to the host being used for this lab.

1.1.2 SSH Client

To perform these lab exercises you will need access to a decent SSH client. The default Windows client is pretty lousy, but it will work.

If you want one that I've found quite useful then go to http://www.chiark.greenend.org.uk/~sgtatham/putty/ and download the file to disk. Running this program will enable you to simply fire off a SSH session or allow you to configure and save various settings.

2.0 Labs

2.1 Lab One

The objective of this exercise is to give you experience in using SSH to connect to the Linux host, login using the user id and password provided, and then to logout.

1. Fire up the SSH client specifying the host name provided
2. When prompted provide your user id and password
3. Logout or exit from the session

2.2 Lab Two

The objectives of this exercise are to

- Familiarize yourself with a couple of commonly used commands, and,
- Let you explore the system a little.

1. Get help on the `ls` command
2. Find out who else is on the system
3. What is your current directory (present working directory)?
4. Pipe the output of the `ls -l /` command to `ls.output` and see what you get

2.3 Lab Three

The objectives of these exercises are to find out how you can see what a system is running and what resources the system is using.

1. Use the `ps -ef | more` command to locate what daemons are running on the system
2. Use the `top` command to display the system activity

2.4 Lab Four

The objectives of these exercises are to:

- See how Linux can handle multiple file systems
- Examine the `/proc` file system which Linux uses to provide information about its internal operation

1. Find out what devices are mounted and what file systems are in use
3. Examine a couple of the `/proc` files using the more command (hint, use the `ls` command to see what files exist within the `/proc` system)

2.5 Lab Five

The objective of this exercise is to familarize you with the hierarchy of files within a file system.

1. Use the `cd` command to go to the "root" of the file system
4. Use the `ls` command to display the files and directories
5. Use the `cd` command to go to your home directory
6. Use the `pwd` command to display the name of the present working directory

2.6 Lab Six

The objective of this exercise is to get your hands dirty playing with files, directories and links.

1. Explore your filesystem:
   - Identify 1st level directories
   - Locate a symbolic link
   - Use the `umask` command to display current default
7. Create 3 files (‘`all`’, ‘`group`’, ‘`owner`’) & assign permissions:
   - `all` - r/w to owner, group, and others
   - `group` - r/w to owner and group, r/o to others
   - `owner` - r/w to owner, r/o to group, none to others
8. Create a directory ‘`test`’ under your home directory
9. Create a file ‘`real.file`’ within this directory
10. Create a symbolic link in your home directory to ‘`real.file`’ called ‘`symbolic.link`’

2.7 Lab Seven

1. What shell are you using
11. Editing the command line:
    - Scrolling through past commands
    - Inserting/deleting characters on command line
    - Using editing key: CTRL-T
12. Try command completion:
    - Note what happens when you issue: `ls /etc/pro<TAB>`
13. Invoke the C shell (and then exit)

2.8 Lab Eight

The objectives of these exercises are to increase your competency and confidence in dealing with files and directories.

1. Use the `ls -a` command to display directories. Where did all those files come from? When you use the SuSE YaST facility to add users, these files get put there as part of the creation process.
14. Use the `-R` option of `ls` to display down file tree
15. Use `cat` to display a file
16. Use `more` to display a file one page at a time
17. Erase the link '`symbolic.link`', erase the '`test`' directory and its contents, then erase the '`all`', '`group`', and '`owner`' files.

3.0 Lab Nine - vi Primer and Exercises

Some things you should know right away:

It is pronounced, “vee-eye”. That's important because you don't want people to think you are completely illiterate, and they will if you say “veye” or “vee”.

There are people who will differ with this, but here is the deal: those people who pronounce differently know that a great number of people say “vee-eye”, but a lot of the people who do pronounce it “vee-eye” do not realize that there are other ways. So be safe. Go with “vee-eye”.

By the way, if you have any modern version of SCO, what you probably want is the graphical editor that is (of course) available only within the graphical environment. Somebody shut that off a long time ago? Try “startx”. That editor does not begin to have the truly awesome power and beauty of `vi`, but there is no learning curve.

A caveat: it is hard to see leading or trailing spaces with the graphical Edit program, and there are places where spaces in the wrong spot can mysteriously break things. Be careful, and remember this.

Of course, you cannot use the graphical editor on a dumb terminal or over a dial-up connection. Nor can you use it if your graphical environment is kaplooey, or never worked at all. So having at least a minimal knowledge of `vi` is helpful, if not absolutely necessary.

3.1 Getting Started

Honestly, there are only a few things you have to know to use `vi`. There are lots of things you should know, lots of things you could know that could make your life easier now and then, but there really are only a handful of things you need to know to get a job done. It may take you ten times longer than it would if you learned just a little bit more, but you will get it done, and that is better than getting nothing done at all.

So let us get started. First thing to do is type:

```
vi /ect/studentnn
```

No, that is not a misprint: I really want you to type “`vi /ect/studentnn`”. Trust me on this one; it is all in a good cause.

Okay, if you have done that, and your system is not the strangest Unix system in the whole world, you should see something that looks a lot like this:

```
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"/ect/studentnn" [New file]
```

Without doing anything at all, I want you to look at your screen. Notice that the "`/ect/studentnn`" is at the bottom of the screen and it says "[New file]". Memorize what this looks like, because every time you mistype the name of a file, this is what you will get.

Notice that, at least right now, vi doesn't care a bit that you don't (I'm pretty sure you don't) even have a directory "`/ect`", or a file called "`studennn`". Right now, vi just doesn't care. All it knows is that there is no such file right now, so it must therefore be a "New" file. Simple minded, yes.

Notice the "~"'s running down your screen? Those are called "tildes" if you prefer accuracy, or "squigglies" by some people. I do not care what you call them, I just want to to remember that in vi, those "~"'s mean "Nothing is there". That's "nothing" as in absolutely nuttin'. Not a bunch of spaces such as separate every word on this page, but nothing at all. Zip, nothing, empty.

That makes sense, here. If `/ect/studentnn` is a new file, there should not be anything in it. Good so far? Okay, let us try something. Press `<ENTER>`.

Nothing happened, right? Try the arrow keys. Do not hit any letters or anything else, just the arrow keys. Anything happen? Can you move down into those squigglies? No? Why not?

Because vi will not let you move over what is not there. Other editors (like the graphical Edit program we talked about above), would just assume you want to add spaces or empty lines, and would let you move down. Not vi, though. vi is picky about those things, and you are stuck right where you are.

Well, there has to be a way to add text, right? Of course. There are two ways (actually a whole lot more, but we are only going to learn two here-keep it simple, remember?). The first way is to type a lower case “`i`”. If you can remember that “`i`” means insert, that will be good. Go ahead, type an “`i`”, but don't type anything else. What happened?

Nothing, right? Actually, if someone else set up your editing environment, you might have seen “INSERT MODE” appear at the bottom of your screen, but probably not. So, that is the second thing you have learned about vi: if you type “`i`”, nothing happens.

But wait: something did actually happen. Try typing something else now, anything at all. “The quick brown fox was not quick enough”. Wow. Look at that. It is working! Press `<ENTER>`. Type some more. Great fun, right?

Okay, now I'm going to break it. Sorry, but this is the only way you are going to learn. Press `<ESC>`. Go ahead, there is no point in typing more. Press `<ESC>`. Press it again. And again. Wait, then `<ESC>`-`<ESC>`-`<ESC>` really fast, pause for a second and then two more. Did your computer beep at you every time you pressed `<ESC>`? It might have (it depends on a few things like: does your speaker work?), which is vi's charming way of saying “Just what is your basic problem, dude? You already pressed it once; I did what I am supposed to, but NOOO, you have to press it again, and again and again…”

OK, now try the arrow keys. If they don't work, use the “`-`” key to move up, the `<ENTER>` to move down, Backspace to move left, and `<SPACE-BAR>` to move right. Try to remember those in case you are ever in a situation where your arrow keys do not work. If it is easier for you, you can also use “`h`”, “`j`”, “`k`” and “`l`” to move around. Try it.

Notice that you still cannot move into those tildes. Nothing has changed; you've added some text that you can move around in, but that is all.

Now let us add a line. Get yourself right on the very first line and then type “`o`”. The “`o`” stands for “open”. Neat, is it not? A whole line opened up underneath where you were and you can type whatever you want until you want to move around again. When you want that, press `<ESC>`, and then you can move again.

If you cheated and used the arrow keys while you were typing, you might have found out that they work, too. But do not count on that: you might not always have arrow keys that work, and some versions of vi don't let you use them when you are typing.

Next lesson: Press `<ESC>` if you haven't already, and put yourself (well, the cursor) anywhere on one of the lines you just typed. Type a lower case “`d`”. Nothing happens (you get a lot of “Nothing happens” with vi). Do it again. Whoops! Did you see that sucker disappear? No? Try it again, and pay closer attention. Type “`d`”, and then type it again. Instant line eradicator!

Of course, sometimes you are going to delete a line you did not mean to delete. Type “`u`” (“undo”). Magic?! Type it again. Wow. Again. And again. It's like stuck, isn't it?

Let's try something else. Get on the very first line and press “`d`” twice. Now move to the very last line and type a “`p`”. Wow, now you can move lines! But wait, there's more: type “`p`” again. And again. And once more. Now you can duplicate lines, too. “`p`” is for “put”.

Sometimes you don't want to remove lines, just characters. vi can do that. Put your cursor on top of a character. Pick a mean looking one, a character that doesn't deserve to be in your file. You are now judge, jury, and executioner. Does this character deserve to die? You bet! Type an “`x`” and the little creep is gone. Changed your mind? Bring it back with “`u`”. Toy with it: “`u`” and it's gone, “`u`” and it's back. Gone, back, gone back. Only you can determine this letter's fate. There's another neat trick that can come in handy if you transpose characters while typing. Say you accidentally type “`lteters`”. Put your cursor on the first “`t`” and hit “`x`”. Then, without moving a muscle, hit “`p`”. You now have “`letters`”. Neat.

One more thing about “`x`” (actually about almost any command, but we'll use “`x`” to demonstrate). Put yourself at the beginning of a line and then type “`i`”, followed by “`hello`”. Hit `<ESC>`, then move back to the “`h`” of “`hello`”. Watch carefully now: type “`5x`”. “`hello`” disappears. Hit “`u`” and then try “`3x`”. Get the point? You could type “`58x`” and the next 58 characters would disappear. The reason I mention this is that sometime you will do it accidentally, and if I didn't give you this hint, you wouldn't have a clue. Forewarned and all that.

You've now learned how to move around, how to insert and delete characters and whole lines, and that's enough. There is no editing task that you cannot accomplish with just this. Yes, there are faster and better ways to do all kinds of things that you might have to do, but there is nothing you cannot do just knowing these few commands.

But you do have to learn how to write your changes and get out, and (important) how NOT to write your changes and get out.

Let's try writing this file. To do that, press `<ESC>` if you are in insert mode, and then type a “`:`”. The cursor moves to the bottom of your screen and your computer puts on a very patient expression. You probably won't want to try this, but you could sit there for 6 or 7 hours and vi would do ABSOLUTELY NOTHING. vi is very, very patient.

But you aren't, right? So type “`w`”, which means “write”. OK, cool, the file is written, and now we can.

What? What do you mean you got an error? What error? Let me see that stupid thing. What did you do now? You probably broke it for good this time, and people are going to be real mad at you because YOU PRESSED “`w`” WHEN YOU WEREN'T SUPPOSED TO!

Yeah, I'm kidding. You got “No such file or directory”, didn't you? It's OK, nothing bad happened. vi just can't write this file because of those crazy directory names we used. I stacked the deck to deliberately create this problem for you.

Great. So you've typed 10,000 words of deathless prose that's due on your boss's desk NOW, and you can't write it. Real amusing, right?

Naw. You can write it, you just can't write it to `/ect/student`*`nn`*. How about we write it to `myfile.safe` instead? To do that, simply hit "`:`" again so you are back at the bottom, and this time type "`w myfile.safe`". You get back something like

```
myfile.safe 3 lines, 64 characters
```

Are you worried what would have happened if your boss had an important file named "`myfile.safe`"? Did you just overwrite that file with a bunch of stupid "brown fox" gibberish? Can you do ANYTHING right?

Stop sweating. It wouldn't have happened. Try it again. Type "`:`", then "`w myfile.safe`". See? It won't overwrite an existing file unless you type "`w!`". You might also want to know that if vi says a file is read-only, but you should be able to write it anyway 'cause you are the superuser, the "`w!`" trick fixes that, too.

3.2 Warning!

That won't save you from complete stupidity. If you had started this session by giving the name of a real file (like "`vi /etc/inittab`"), and then had deleted a bunch of lines and added a bunch of new ones, and then typed "`:w`" (with nothing else, no name, just the bare "`w`"), vi would have happily, efficiently, and mercilessly overwritten `/etc/inittab` with your changes. The theory here is that you saw what you were doing, so you must know what you were doing. So be it.

But let's say you messed up the file and you don't want to write it, you just want to quit. Let's try it: mess up this file a little more. Delete a line, add a line, it doesn't matter, just do something. Now do the "`:`" again, and type "`q`" (for "quit").

Gotcha again. But notice that vi has given you a hint about what to do. It tells you that you need to type "`:quit!`" to get out. Actually, you just need "`:q!`", but you can type it out if it makes you feel better.

That's it. You know the basics. I wish you'd learn more, 'cause it's worth it, but if this is all you can take, it is enough. Quick review and we're out of here:

`i`	insert
`o`	open
`dd`	delete line
`x`	remove characters
`u`	undo
`p`	put

`:w`	write file
`:w!`	write absolutely
`:q`	quit after saving (combine with "`:wq`")
`:q!`	quit without saving

3.3 vi Quick Reference

STARTING AND QUITING VI	
`vi`	starts vi without a named file to save to.
`vi filename.txt`	start vi on an existing file (or supply name to save to if no file yet exits.
`<Esc> key`	puts you in edit mode if you weren't already there (the following commands only work in edit mode).
`:q!`	quit vi WITHOUT SAVING
`:wq`	write (save) to supplied file name and quit
`<Esc>-ZZ`	also saves and quits
`:w newfile.txt`	write to a new file and don't quit (still editing newfile.txt)
`:wq newfile.txt`	write to a new file and quit2) CURSOR MOVEMENTh <j vk ^l >
`^  or 0`	move cursor to start of line
`$`	move cursor to the end of the line
`<ctrl> G`	indicates current line number of file where cursor currently is
`<ctrl> F`	moves cursor ahead one page
`<ctrl> B`	moves cursor back one page
`<shift> H`	moves cursor to top of screen
`<shift> L`	moves cursor to bottom of screen
`1G`	move to line 1
`G`	moves to last line
`w`	advances by a word (W doesn't stop at punctuation)
`b`	backs up by a word (B as well)
`e`	go to the end of a word
BASIC EDITING	

`i`	puts you in insert mode, press <Esc> to exit
`I`	^ then i
`a`	insert mode, but appending after cursor
`A`	$ then a
`o`	insert mode, opening new line below where you are
`O`	insert mode, opening new line above where you are
`u`	undo
`x`	deletes a single character; 5x deletes next 5 characters
`J`	deletes end-eof-line character (does a "Join" of current line with next line)
`.`	repeats last editing command
`r`	replaces current character with another
`cw`	changes remainder of current word to whatever you type; <esc> to end edit
`/word`	finds occurrence of 'word' in the file
`n`	finds next occurrence of 'word'
`dd`	deletes line
`3dd`	deletes 3 lines (from this line down)
`23,50d`	deletes lines 23-50, inclusive
`3Y`	"yank" three lines (place in unnamed buffer)
`"a3Y`	Yank three lines to a buffer called 'a'
`p`	puts deleted (or yanked lines) below this line
`P`	puts deleted (or yanked lines) above this line
`"ap`	places lines from buffer 'a'
COMMAND LINE EDITING	
`:%s/word/WORD`	Replaces first occurrence of word with WORD on every line of the file
`:1,$s/word/WORD/g`	From lines 1 to the end of the file change word to WORD (g means all occurrences on a line).
`:1,23s/^word/WORD/`	From lines 1 to 23 replace "word" at the beginning of any line with "WORD"

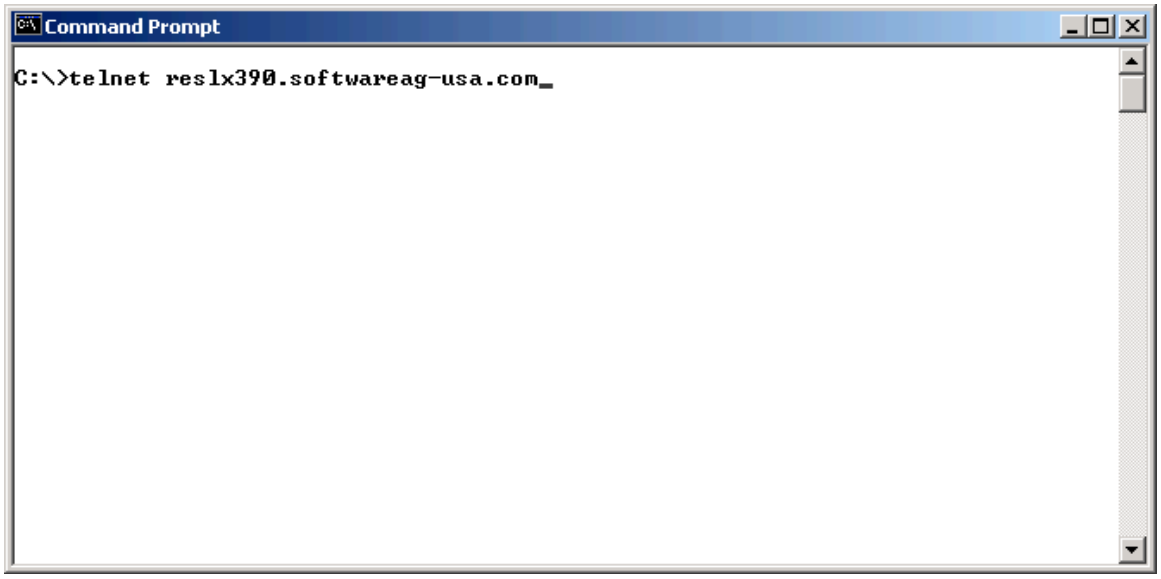
`:1,23s/word$/WORD/`	From lines 1 to 23 replace "word" at the end of any line with "WORD"
`:1,$s/^...//`	From lines 1 to the end of the file remove "..." beginning any of those lines.
`:g/word/d`	Does a "grep" to find lines with 'word', then deletes those lines
`:1,$s/\&/and/g`	Replaces every occurrence of & (escaped) with "and"
`:g/word/p`	Does a "grep" to find lines with 'word', then prints those line to the screen
`:3,15s/^/\#`	Put a # at the beginning of lines 3 through 15
`:%s/$/;`	Append a semicolon to the end of every line (note that "%" = "1,$")
FILE OPERATIONS	
`:r path\filename`	read in the specified file starting on the next line
`:w`	save
:w filename	save as filename
`:wq`	save and quit
`:q`	quit if no modifications since last save
`:q!`	quit no matter what (without saving)

me	edit another file without having to quit and restart vi

4.0 Lab One Answers

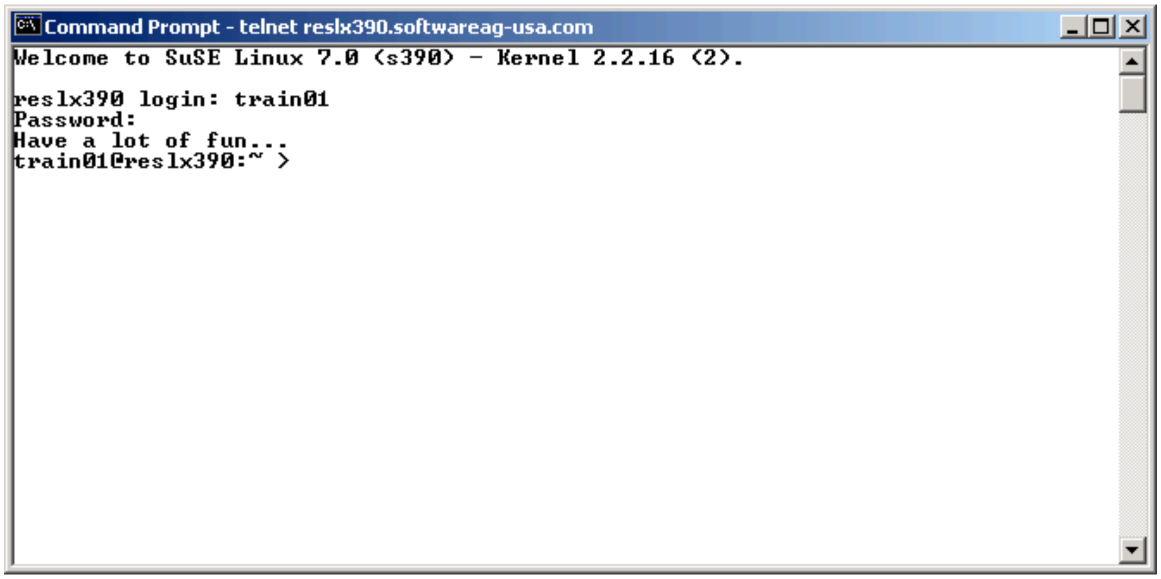
4.1 Using the Windows Client

1. SSH to host



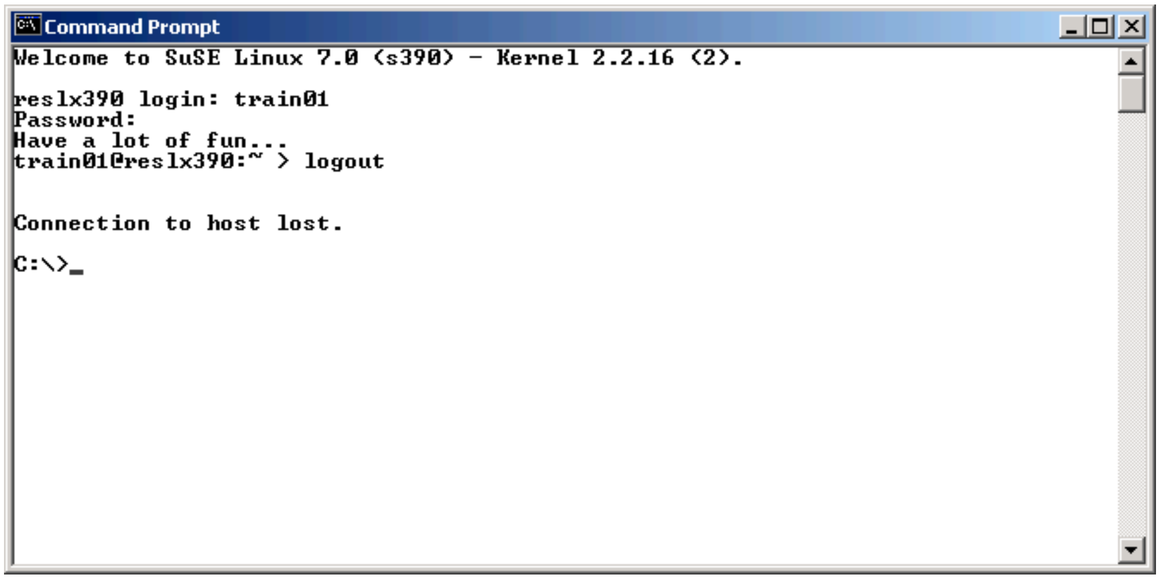
```
C:\>telnet reslx390.softwareag-usa.com_
```

18. Provide user id and password when prompted



```
Welcome to SuSE Linux 7.0 (s390) - Kernel 2.2.16 (2).

reslx390 login: train01
Password:
Have a lot of fun...
train01@reslx390:~ >
```

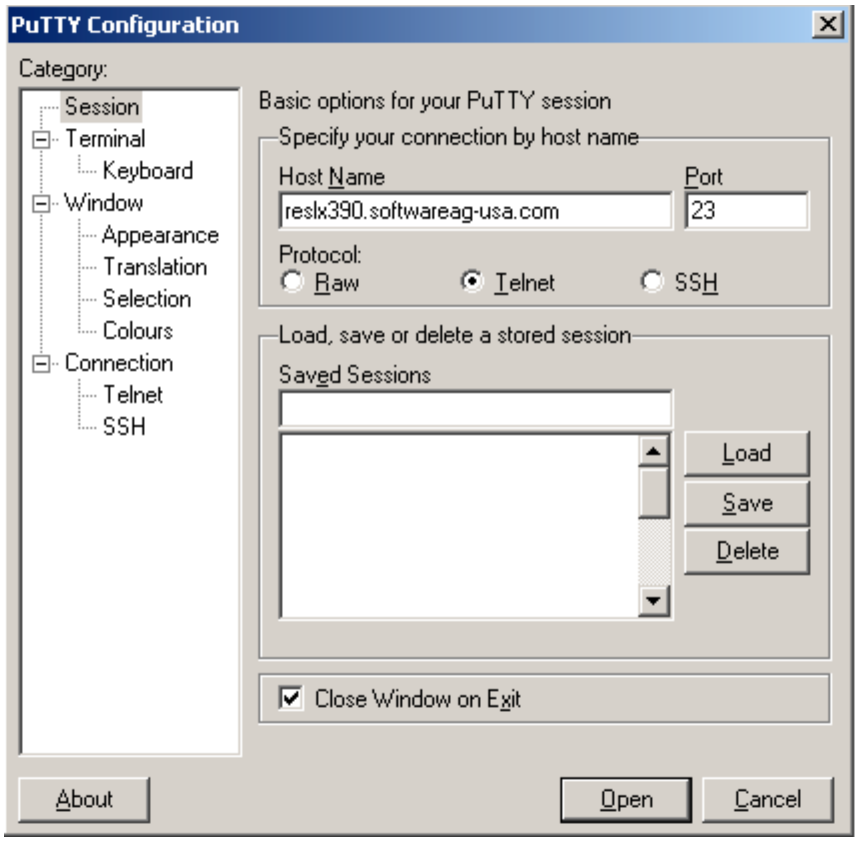
19. Logout or exit to terminate the session



```
Welcome to SuSE Linux 7.0 (s390) - Kernel 2.2.16 (2).

reslx390 login: train01
Password:
Have a lot of fun...
train01@reslx390:~ > logout


Connection to host lost.

C:\>_
```

4.2 Using Putty

1. SSH to host



Basic options for your PuTTY session

Specify your connection by host name

Host Name: reslx390.softwareag-usa.com

Port: 23

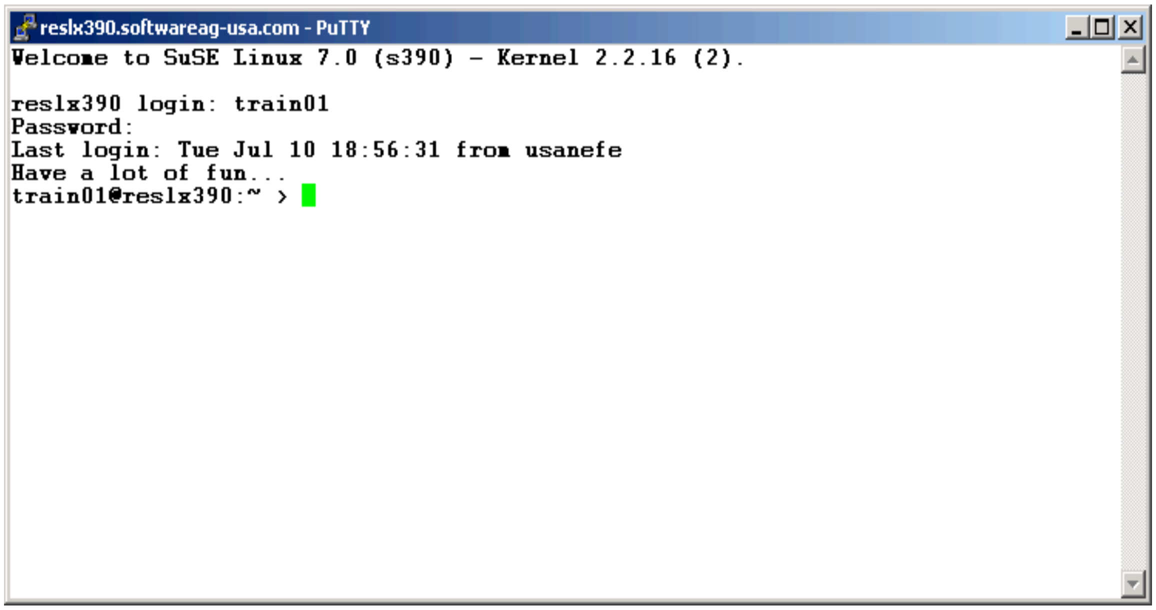
Protocol: Raw [✓ Telnet] SSH

Load, save or delete a stored session

Saved Sessions

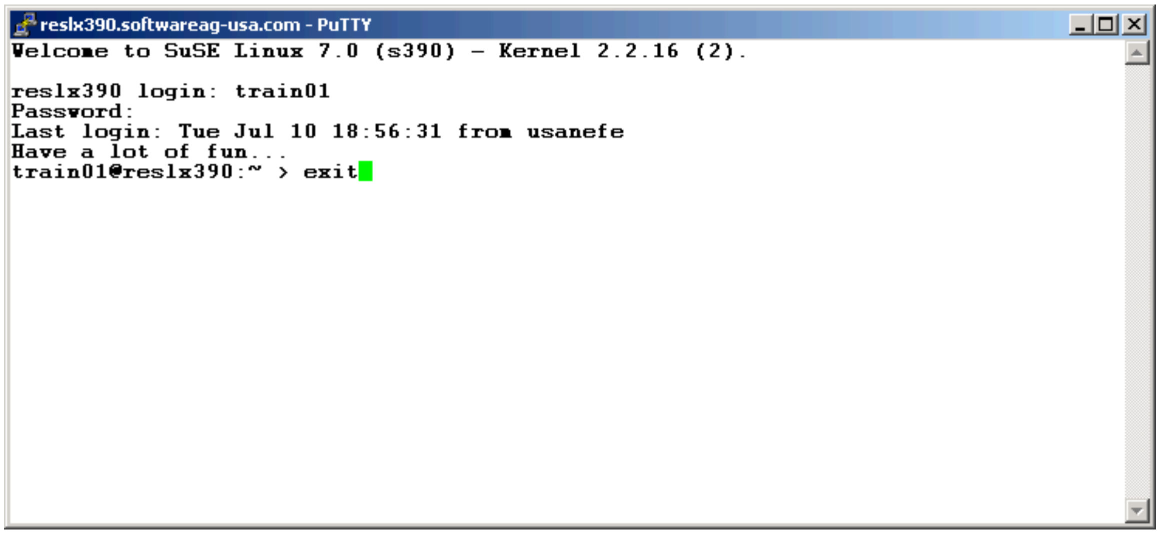
[✓] Close Window on Exit

20. Provide user id and password when prompted



```
Welcome to SuSE Linux 7.0 (s390) - Kernel 2.2.16 (2).

reslx390 login: train01
Password:
Last login: Tue Jul 10 18:56:31 from usanefe
Have a lot of fun...
train01@reslx390:~ >
```

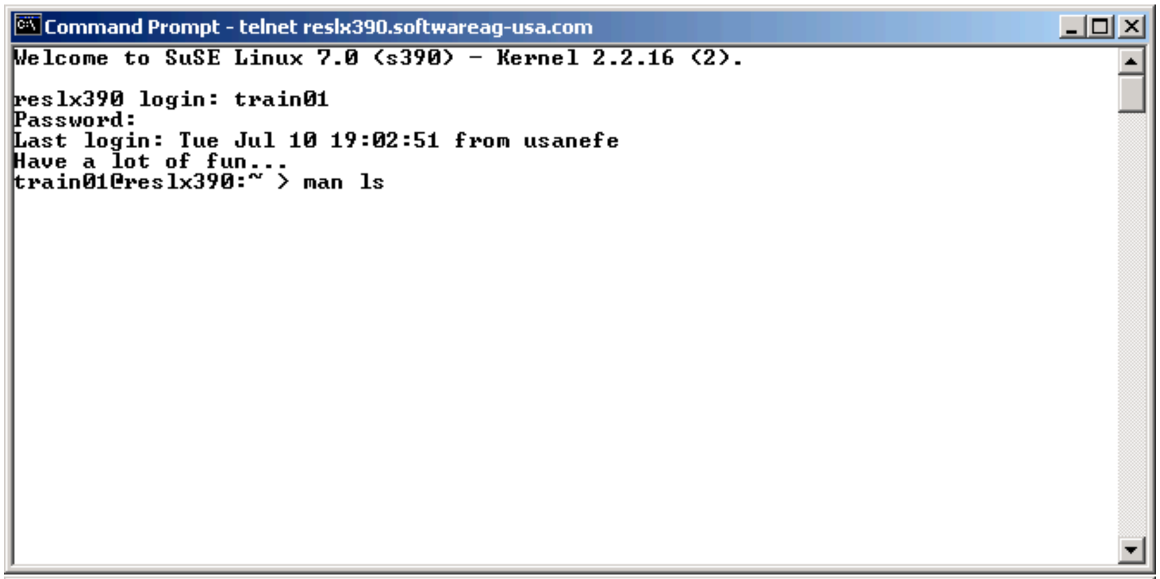
21. Logout or exit to terminate session



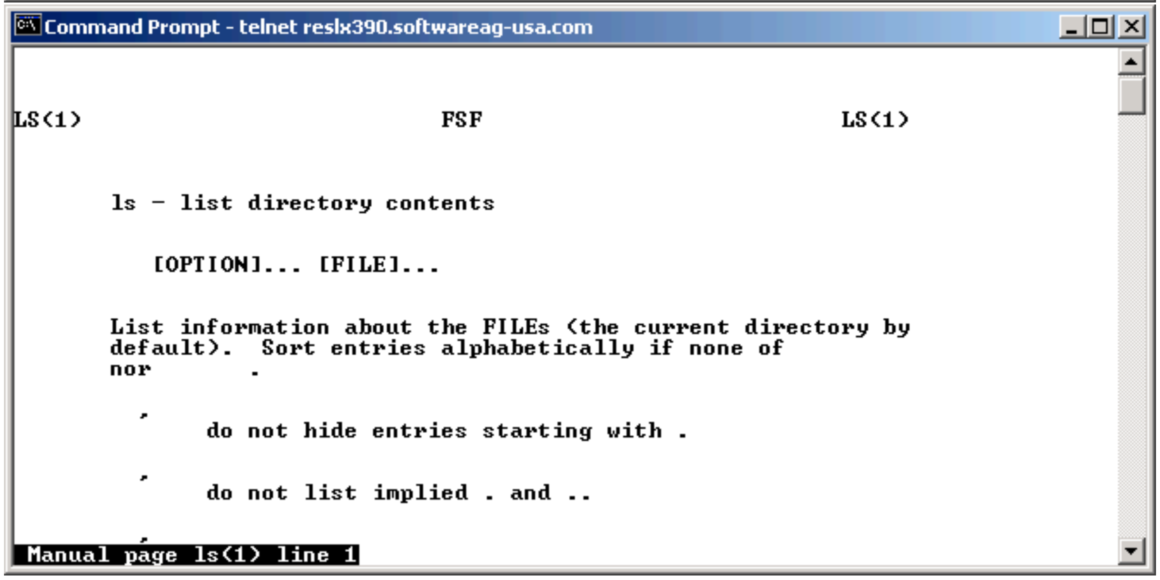
```
Welcome to SuSE Linux 7.0 (s390) - Kernel 2.2.16 (2).

reslx390 login: train01
Password:
Last login: Tue Jul 10 18:56:31 from usanefe
Have a lot of fun...
train01@reslx390:~ > exit
```

5.0 Lab Two Answers

1. Get help on the `ls` command



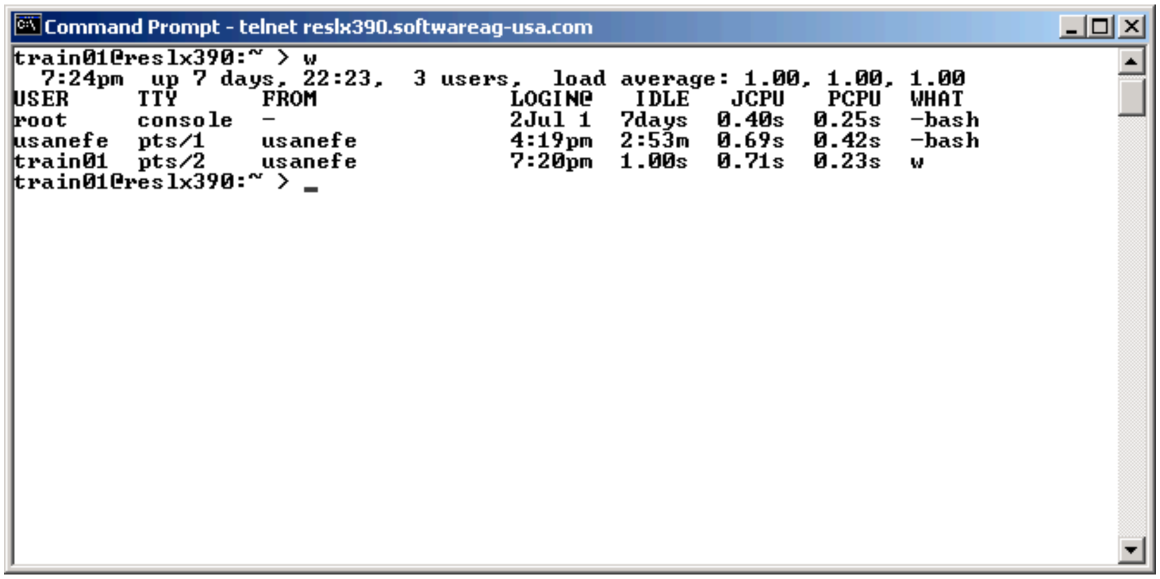
```
Welcome to SuSE Linux 7.0 (s390) - Kernel 2.2.16 (2).

reslx390 login: train01
Password:
Last login: Tue Jul 10 19:02:51 from usanefe
Have a lot of fun...
train01@reslx390:~ > man ls
```



```
LS(1)                          FSF                          LS(1)


       ls - list directory contents

          [OPTION]... [FILE]...

       List information about the FILEs (the current directory by
       default).  Sort entries alphabetically if none of
       nor       .

         ,
              do not hide entries starting with .

         ,
              do not list implied . and ..

         ,
 Manual page ls(1) line 1
```

(Press q to exit from "`man`")

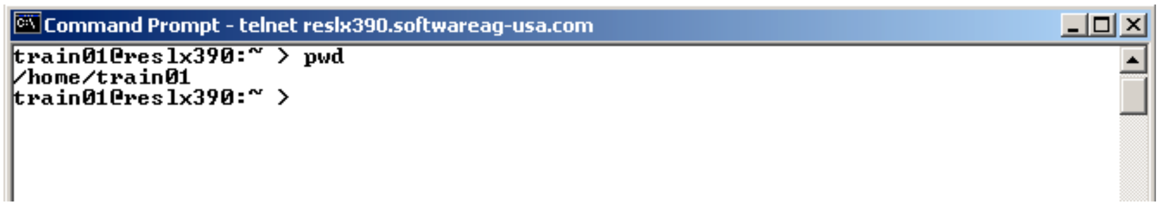
22. Find out who else is on the system



Command Prompt - telnet reslx390.softwareag-usa.com

```
train01@reslx390:~ > w
  7:24pm  up 7 days, 22:23,  3 users,  load average: 1.00, 1.00, 1.00
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU  WHAT
root     console  -                 2Jul 1  7days  0.40s  0.25s  -bash
usanefe  pts/1    usanefe           4:19pm  2:53m  0.69s  0.42s  -bash
train01  pts/2    usanefe           7:20pm  1.00s  0.71s  0.23s  w
train01@reslx390:~ > _
```

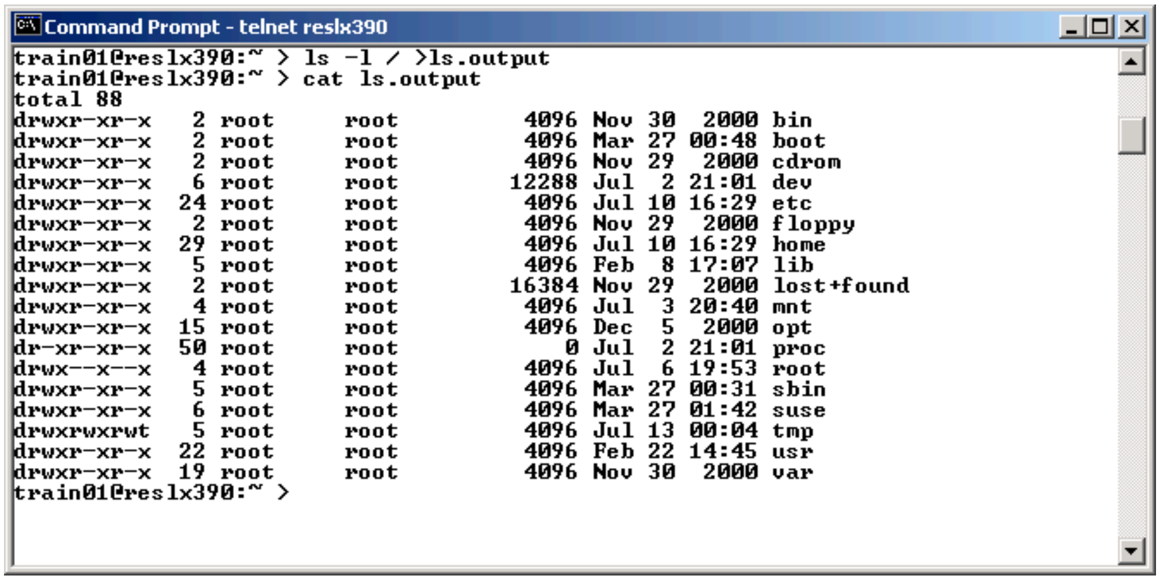
23. What is your current directory



Command Prompt - telnet reslx390.softwareag-usa.com

```
train01@reslx390:~ > pwd
/home/train01
train01@reslx390:~ >
```

24. Pipe the output of the `ls -l /` command to `ls.output` and see what you get

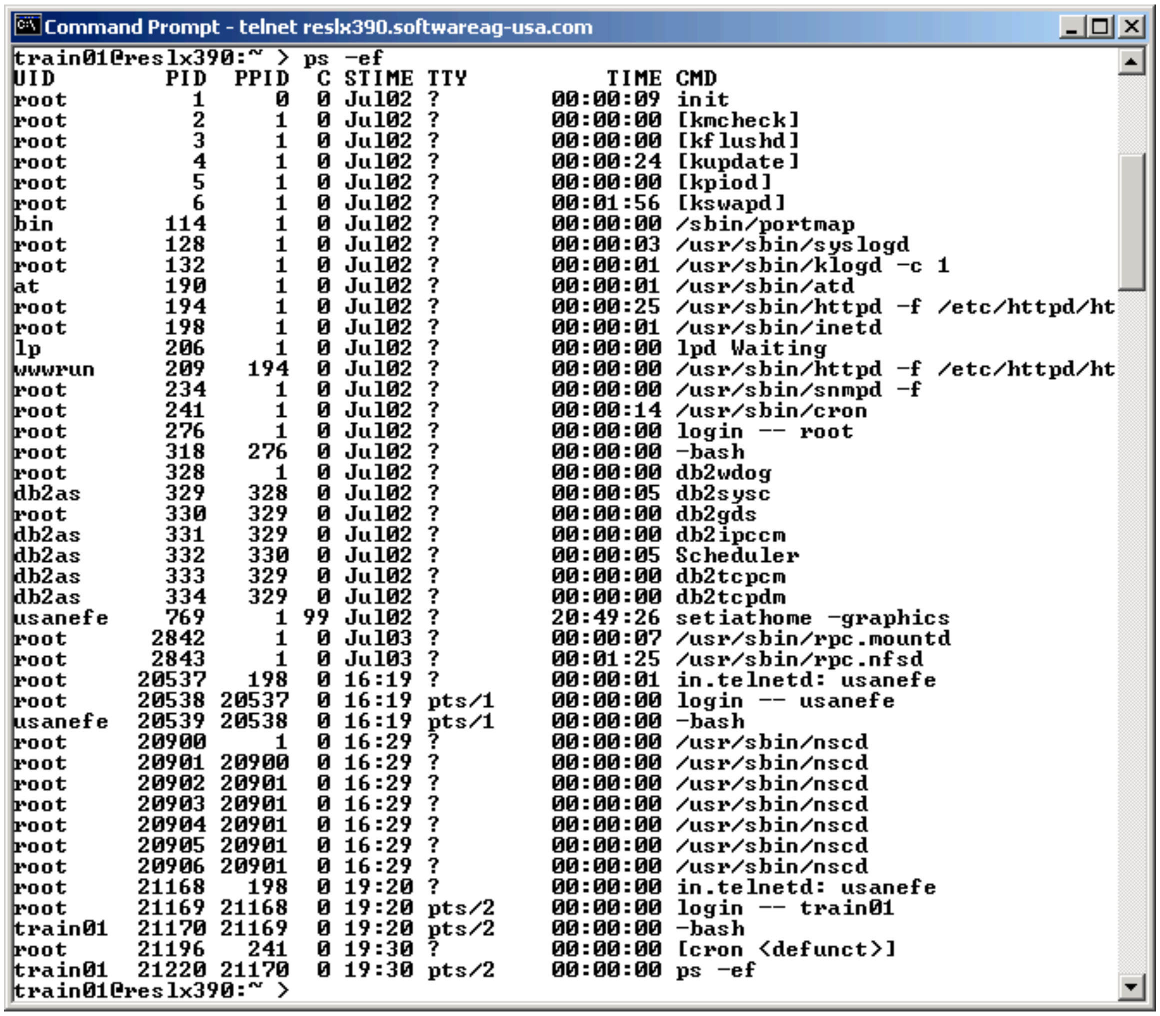


Command Prompt - telnet reslx390

```
train01@reslx390:~ > ls -l / >ls.output
train01@reslx390:~ > cat ls.output
total 88
drwxr-xr-x   2 root     root         4096 Nov 30  2000 bin
drwxr-xr-x   2 root     root         4096 Mar 27 00:48 boot
drwxr-xr-x   2 root     root         4096 Nov 29  2000 cdrom
drwxr-xr-x   6 root     root        12288 Jul  2 21:01 dev
drwxr-xr-x  24 root     root         4096 Jul 10 16:29 etc
drwxr-xr-x   2 root     root         4096 Nov 29  2000 floppy
drwxr-xr-x  29 root     root         4096 Jul 10 16:29 home
drwxr-xr-x   5 root     root         4096 Feb  8 17:07 lib
drwxr-xr-x   2 root     root        16384 Nov 29  2000 lost+found
drwxr-xr-x   4 root     root         4096 Jul  3 20:40 mnt
drwxr-xr-x  15 root     root         4096 Dec  5  2000 opt
dr-xr-xr-x  50 root     root            0 Jul  2 21:01 proc
drwx--x--x   4 root     root         4096 Jul  6 19:53 root
drwxr-xr-x   5 root     root         4096 Mar 27 00:31 sbin
drwxr-xr-x   6 root     root         4096 Mar 27 01:42 suse
drwxrwxrwt   5 root     root         4096 Jul 13 00:04 tmp
drwxr-xr-x  22 root     root         4096 Feb 22 14:45 usr
drwxr-xr-x  19 root     root         4096 Nov 30  2000 var
train01@reslx390:~ >
```

6.0 Lab Three Answers

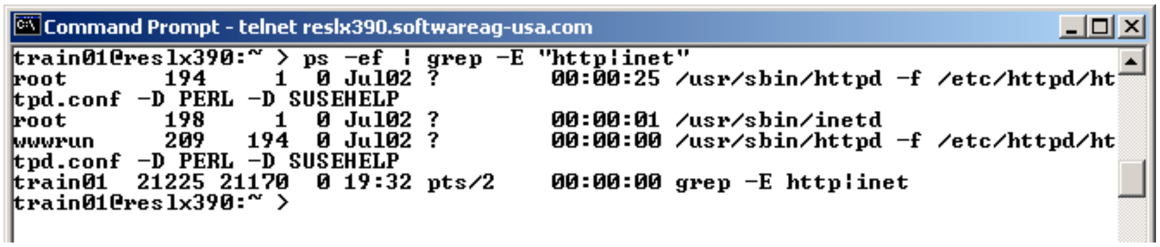
1. Use `ps -ef` to locate daemons



Command Prompt - telnet reslx390.softwareag-usa.com

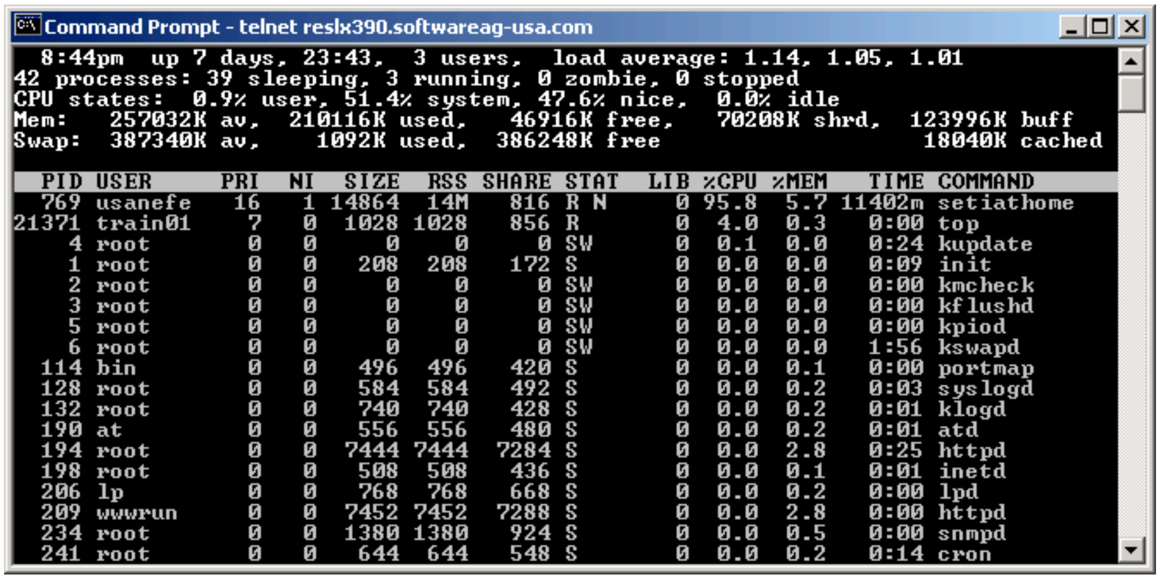
```
train01@reslx390:~ > ps -ef
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 Jul02 ?        00:00:09 init
root         2     1  0 Jul02 ?        00:00:00 [kmcheck]
root         3     1  0 Jul02 ?        00:00:00 [kflushd]
root         4     1  0 Jul02 ?        00:00:24 [kupdate]
root         5     1  0 Jul02 ?        00:00:00 [kpiod]
root         6     1  0 Jul02 ?        00:01:56 [kswapd]
bin        114     1  0 Jul02 ?        00:00:00 /sbin/portmap
root       128     1  0 Jul02 ?        00:00:03 /usr/sbin/syslogd
root       132     1  0 Jul02 ?        00:00:01 /usr/sbin/klogd -c 1
at         190     1  0 Jul02 ?        00:00:01 /usr/sbin/atd
root       194     1  0 Jul02 ?        00:00:25 /usr/sbin/httpd -f /etc/httpd/ht
root       198     1  0 Jul02 ?        00:00:01 /usr/sbin/inetd
lp         206     1  0 Jul02 ?        00:00:00 lpd Waiting
wwwrun     209   194  0 Jul02 ?        00:00:00 /usr/sbin/httpd -f /etc/httpd/ht
root       234     1  0 Jul02 ?        00:00:00 /usr/sbin/snmpd -f
root       241     1  0 Jul02 ?        00:00:14 /usr/sbin/cron
root       276     1  0 Jul02 ?        00:00:00 login -- root
root       318   276  0 Jul02 ?        00:00:00 -bash
root       328     1  0 Jul02 ?        00:00:00 db2wdog
db2as      329   328  0 Jul02 ?        00:00:05 db2sysc
root       330   329  0 Jul02 ?        00:00:00 db2gds
db2as      331   329  0 Jul02 ?        00:00:00 db2ipccm
db2as      332   330  0 Jul02 ?        00:00:05 Scheduler
db2as      333   329  0 Jul02 ?        00:00:00 db2tcpcm
db2as      334   329  0 Jul02 ?        00:00:00 db2tcpdm
usanefe    769     1 99 Jul02 ?        20:49:26 setiathome -graphics
root      2842     1  0 Jul03 ?        00:00:07 /usr/sbin/rpc.mountd
root      2843     1  0 Jul03 ?        00:01:25 /usr/sbin/rpc.nfsd
root     20537   198  0 16:19 ?        00:00:01 in.telnetd: usanefe
root     20538 20537  0 16:19 pts/1    00:00:00 login -- usanefe
usanefe  20539 20538  0 16:19 pts/1    00:00:00 -bash
root     20900     1  0 16:29 ?        00:00:00 /usr/sbin/nscd
root     20901 20900  0 16:29 ?        00:00:00 /usr/sbin/nscd
root     20902 20901  0 16:29 ?        00:00:00 /usr/sbin/nscd
root     20903 20901  0 16:29 ?        00:00:00 /usr/sbin/nscd
root     20904 20901  0 16:29 ?        00:00:00 /usr/sbin/nscd
root     20905 20901  0 16:29 ?        00:00:00 /usr/sbin/nscd
root     20906 20901  0 16:29 ?        00:00:00 /usr/sbin/nscd
root     21168   198  0 19:20 ?        00:00:00 in.telnetd: usanefe
root     21169 21168  0 19:20 pts/2    00:00:00 login -- train01
train01  21170 21169  0 19:20 pts/2    00:00:00 -bash
root     21196   241  0 19:30 ?        00:00:00 [cron <defunct>]
train01  21220 21170  0 19:30 pts/2    00:00:00 ps -ef
train01@reslx390:~ >
```

1. Or to be tricky…



Command Prompt - telnet reslx390.softwareag-usa.com

```
train01@reslx390:~ > ps -ef | grep -E "http|inet"
root       194     1  0 Jul02 ?        00:00:25 /usr/sbin/httpd -f /etc/httpd/ht
tpd.conf -D PERL -D SUSEHELP
root       198     1  0 Jul02 ?        00:00:01 /usr/sbin/inetd
wwwrun     209   194  0 Jul02 ?        00:00:00 /usr/sbin/httpd -f /etc/httpd/ht
tpd.conf -D PERL -D SUSEHELP
train01  21225 21170  0 19:32 pts/2    00:00:00 grep -E http|inet
train01@reslx390:~ >
```

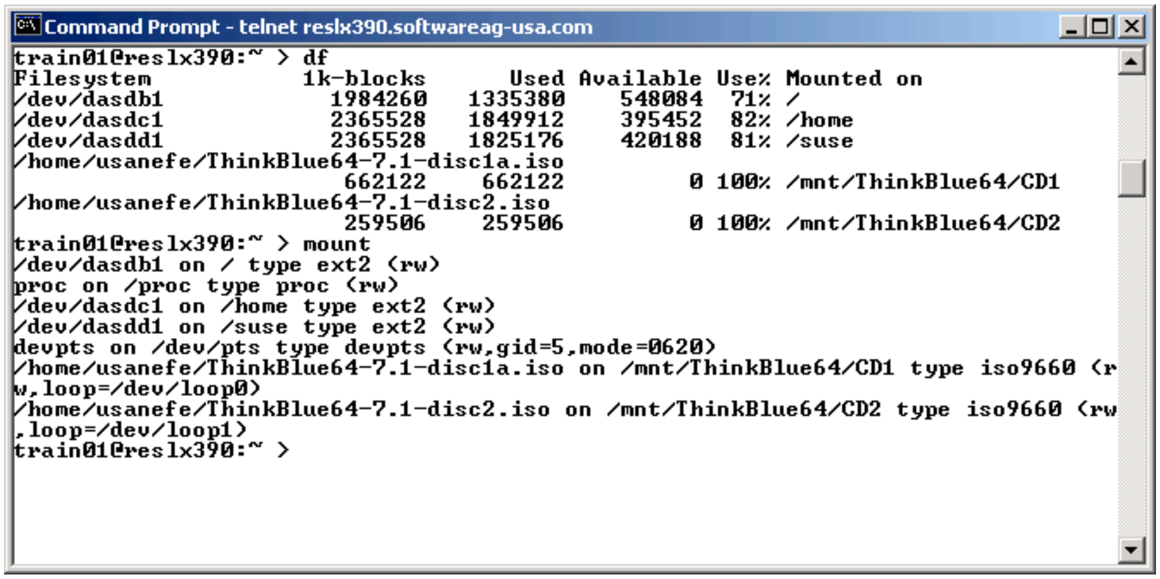


```
  8:44pm  up 7 days, 23:43,  3 users,  load average: 1.14, 1.05, 1.01
42 processes: 39 sleeping, 3 running, 0 zombie, 0 stopped
CPU states:  0.9% user, 51.4% system, 47.6% nice,  0.0% idle
Mem:   257032K av,  210116K used,   46916K free,   70208K shrd,  123996K buff
Swap:  387340K av,    1092K used,  386248K free                   18040K cached

  PID USER      PRI  NI  SIZE  RSS SHARE STAT  LIB %CPU %MEM   TIME COMMAND
  769 usanefe    16   1 14864  14M   816 R N     0 95.8  5.7 11402m setiathome
21371 train01     7   0  1028 1028   856 R       0  4.0  0.3   0:00 top
    4 root        0   0     0    0     0 SW      0  0.1  0.0   0:24 kupdate
    1 root        0   0   208  208   172 S       0  0.0  0.0   0:09 init
    2 root        0   0     0    0     0 SW      0  0.0  0.0   0:00 kmcheck
    3 root        0   0     0    0     0 SW      0  0.0  0.0   0:00 kflushd
    5 root        0   0     0    0     0 SW      0  0.0  0.0   0:00 kpiod
    6 root        0   0     0    0     0 SW      0  0.0  0.0   1:56 kswapd
  114 bin         0   0   496  496   420 S       0  0.0  0.1   0:00 portmap
  128 root        0   0   584  584   492 S       0  0.0  0.2   0:03 syslogd
  132 root        0   0   740  740   428 S       0  0.0  0.2   0:01 klogd
  190 at          0   0   556  556   480 S       0  0.0  0.2   0:01 atd
  194 root        0   0  7444 7444  7284 S       0  0.0  2.8   0:25 httpd
  198 root        0   0   508  508   436 S       0  0.0  0.1   0:01 inetd
  206 lp          0   0   768  768   668 S       0  0.0  0.2   0:00 lpd
  209 wwwrun      0   0  7452 7452  7288 S       0  0.0  2.8   0:00 httpd
  234 root        0   0  1380 1380   924 S       0  0.0  0.5   0:00 snmpd
  241 root        0   0   644  644   548 S       0  0.0  0.2   0:14 cron
```

25. Use the `top` command to display system activity

7.0 Lab Four Answers

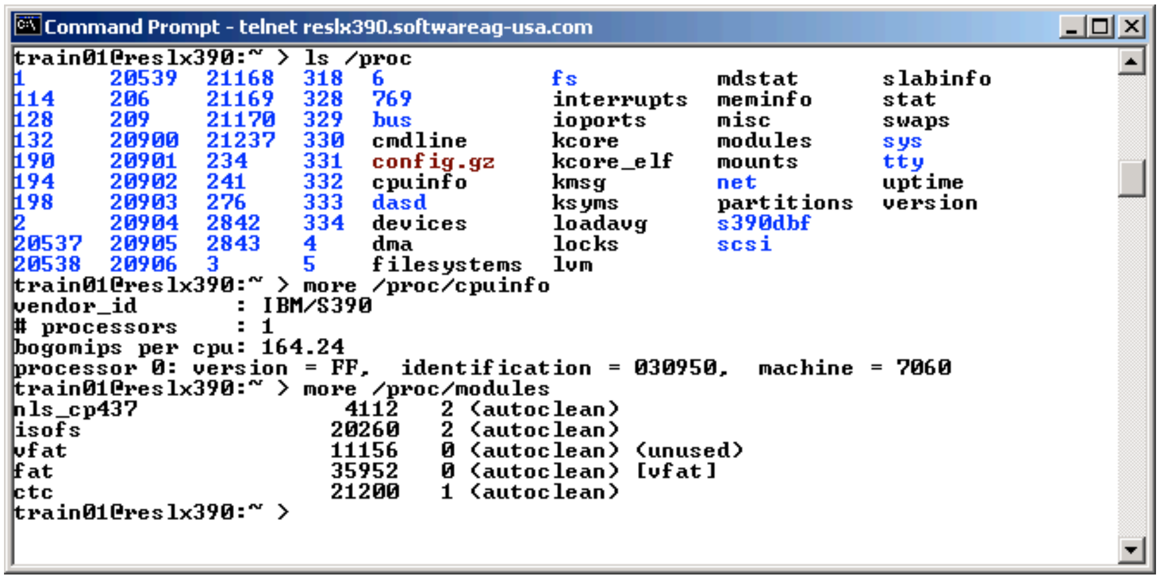
1. Find out what devices are mounted and what file systems are in use



Command Prompt - telnet reslx390.softwareag-usa.com

```
train01@reslx390:~ > df
Filesystem           1k-blocks      Used Available Use% Mounted on
/dev/dasdb1            1984260   1335380    548084  71% /
/dev/dasdc1            2365528   1849912    395452  82% /home
/dev/dasdd1            2365528   1825176    420188  81% /suse
/home/usanefe/ThinkBlue64-7.1-disc1a.iso
                        662122    662122         0 100% /mnt/ThinkBlue64/CD1
/home/usanefe/ThinkBlue64-7.1-disc2.iso
                        259506    259506         0 100% /mnt/ThinkBlue64/CD2
train01@reslx390:~ > mount
/dev/dasdb1 on / type ext2 (rw)
proc on /proc type proc (rw)
/dev/dasdc1 on /home type ext2 (rw)
/dev/dasdd1 on /suse type ext2 (rw)
devpts on /dev/pts type devpts (rw,gid=5,mode=0620)
/home/usanefe/ThinkBlue64-7.1-disc1a.iso on /mnt/ThinkBlue64/CD1 type iso9660 (r
w,loop=/dev/loop0)
/home/usanefe/ThinkBlue64-7.1-disc2.iso on /mnt/ThinkBlue64/CD2 type iso9660 (rw
,loop=/dev/loop1)
train01@reslx390:~ >
```

26. Examine a couple of `/proc` files using the `more` command

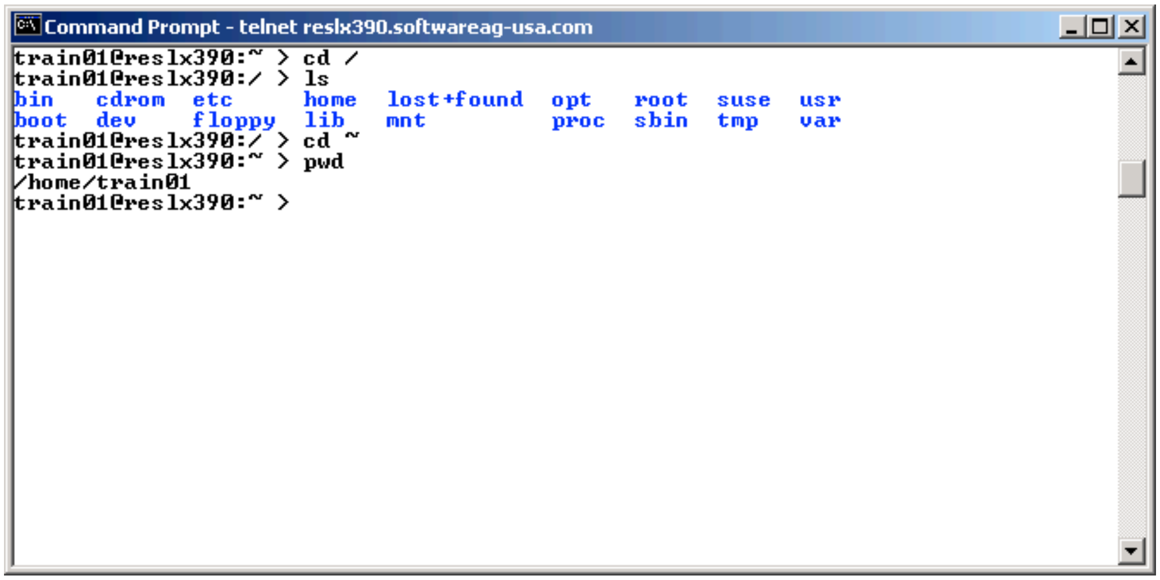


Command Prompt - telnet reslx390.softwareag-usa.com

```
train01@reslx390:~ > ls /proc
1      20539  21168  318  6            fs           mdstat      slabinfo
114    206    21169  328  769          interrupts   meminfo     stat
128    209    21170  329  bus          ioports      misc        swaps
132    20900  21237  330  cmdline      kcore        modules     sys
190    20901  234    331  config.gz    kcore_elf    mounts      tty
194    20902  241    332  cpuinfo      kmsg         net         uptime
198    20903  276    333  dasd         ksyms        partitions  version
2      20904  2842   334  devices      loadavg      s390dbf
20537  20905  2843   4    dma          locks        scsi
20538  20906  3      5    filesystems  lvm
train01@reslx390:~ > more /proc/cpuinfo
vendor_id       : IBM/S390
# processors    : 1
bogomips per cpu: 164.24
processor 0: version = FF,  identification = 030950,  machine = 7060
train01@reslx390:~ > more /proc/modules
nls_cp437               4112   2 (autoclean)
isofs                  20260   2 (autoclean)
vfat                   11156   0 (autoclean) (unused)
fat                    35952   0 (autoclean) [vfat]
ctc                    21200   1 (autoclean)
train01@reslx390:~ >
```

8.0 Lab Five Answers

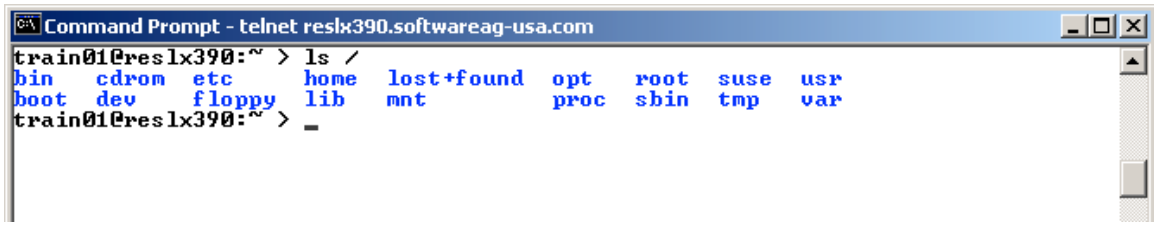
1. Explore the file system
   - `cd /`
   - `ls`
   - `cd ~`
   - `pwd`



```
train01@reslx390:~ > cd /
train01@reslx390:/ > ls
bin    cdrom  etc     home  lost+found  opt   root  suse  usr
boot   dev    floppy  lib   mnt         proc  sbin  tmp   var
train01@reslx390:/ > cd ~
train01@reslx390:~ > pwd
/home/train01
train01@reslx390:~ >
```

9.0 Lab Six Answers

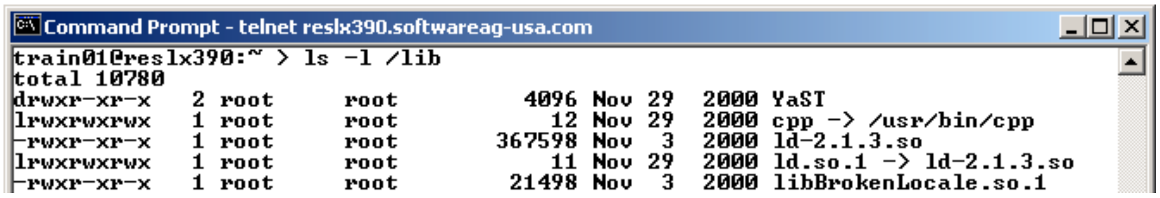
1. Explore your file system
   - Identify 1st level directories



Command Prompt - telnet reslx390.softwareag-usa.com

```
train01@reslx390:~ > ls /
bin    cdrom  etc     home  lost+found  opt   root  suse  usr
boot   dev    floppy  lib   mnt         proc  sbin  tmp   var
train01@reslx390:~ > _
```

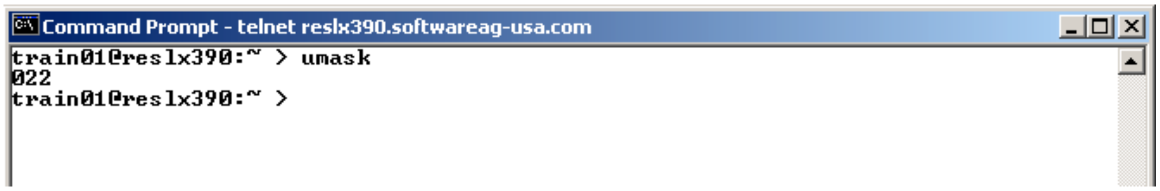
   - Locate a symbolic link



Command Prompt - telnet reslx390.softwareag-usa.com

```
train01@reslx390:~ > ls -l /lib
total 10780
drwxr-xr-x   2 root     root         4096 Nov 29  2000 YaST
lrwxrwxrwx   1 root     root           12 Nov 29  2000 cpp -> /usr/bin/cpp
-rwxr-xr-x   1 root     root       367598 Nov  3  2000 ld-2.1.3.so
lrwxrwxrwx   1 root     root           11 Nov 29  2000 ld.so.1 -> ld-2.1.3.so
-rwxr-xr-x   1 root     root        21498 Nov  3  2000 libBrokenLocale.so.1
```

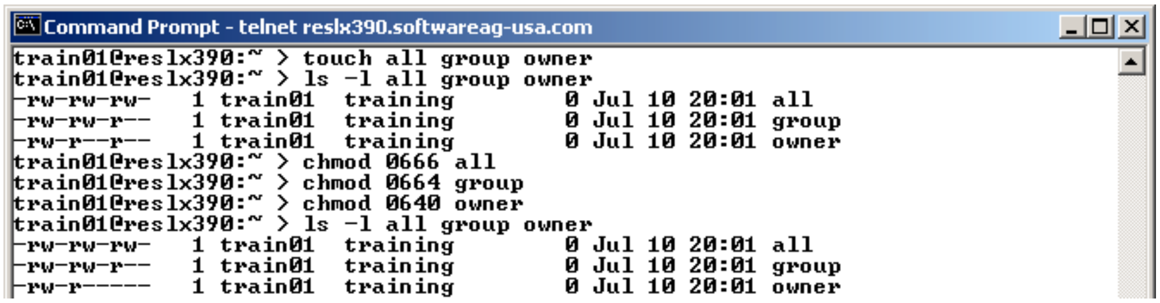
   - Use the umask command to display the current default



Command Prompt - telnet reslx390.softwareag-usa.com

```
train01@reslx390:~ > umask
022
train01@reslx390:~ >
```

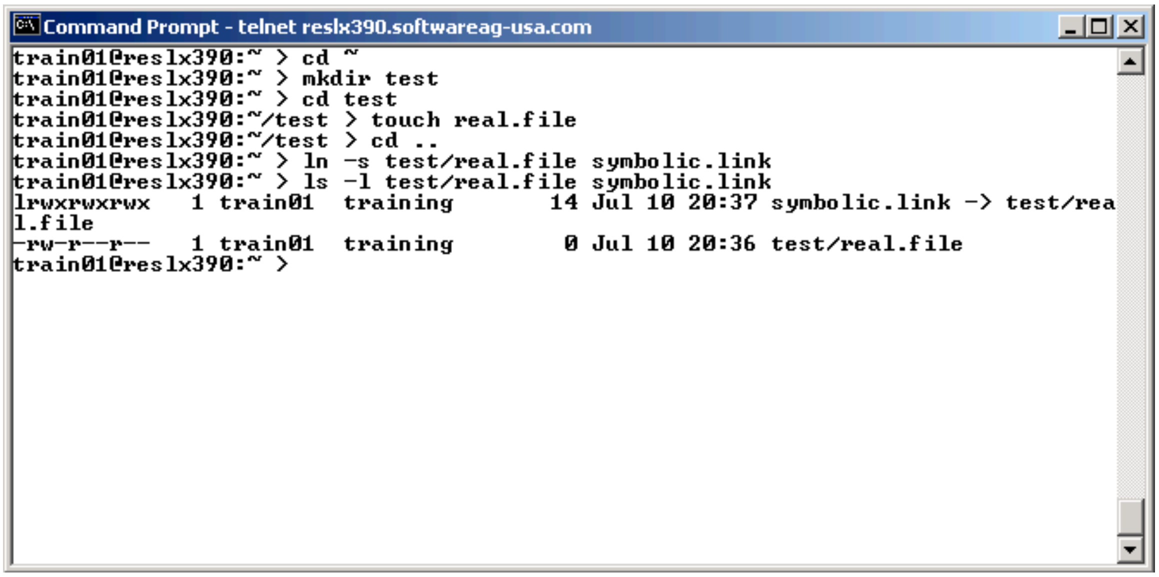
27. Create 3 files '`all`', '`group`', '`owner`'
   - `all` - give r/w permission to owner, group, and others
   - `group` - give r/w permission to owner, group, and r/o to others
   - `owner` - give r/w permission to owner, r/o to group, and none to others



Command Prompt - telnet reslx390.softwareag-usa.com

```
train01@reslx390:~ > touch all group owner
train01@reslx390:~ > ls -l all group owner
-rw-rw-rw-   1 train01  training        0 Jul 10 20:01 all
-rw-rw-r--   1 train01  training        0 Jul 10 20:01 group
-rw-r--r--   1 train01  training        0 Jul 10 20:01 owner
train01@reslx390:~ > chmod 0666 all
train01@reslx390:~ > chmod 0664 group
train01@reslx390:~ > chmod 0640 owner
train01@reslx390:~ > ls -l all group owner
-rw-rw-rw-   1 train01  training        0 Jul 10 20:01 all
-rw-rw-r--   1 train01  training        0 Jul 10 20:01 group
-rw-r-----   1 train01  training        0 Jul 10 20:01 owner
```

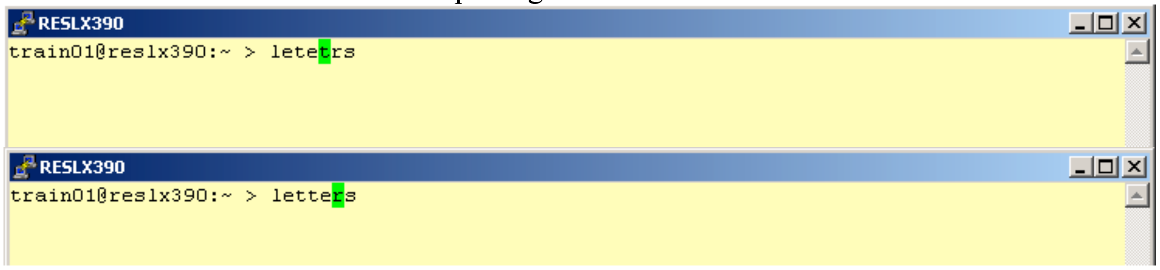
28. Create a directory ‘`test`’ under your home directory
    - Create a file ‘`real.file`’ in this directory
    - Create a symbolic link in your home directory to ‘`real.file`’ called ‘`symbolic.link`’


Command Prompt - telnet reslx390.softwareag-usa.com

```
train01@reslx390:~ > cd ~
train01@reslx390:~ > mkdir test
train01@reslx390:~ > cd test
train01@reslx390:~/test > touch real.file
train01@reslx390:~/test > cd ..
train01@reslx390:~ > ln -s test/real.file symbolic.link
train01@reslx390:~ > ls -l test/real.file symbolic.link
lrwxrwxrwx   1 train01  training       14 Jul 10 20:37 symbolic.link -> test/rea
l.file
-rw-r--r--   1 train01  training        0 Jul 10 20:36 test/real.file
train01@reslx390:~ >
```

10.0 Lab Seven Answers

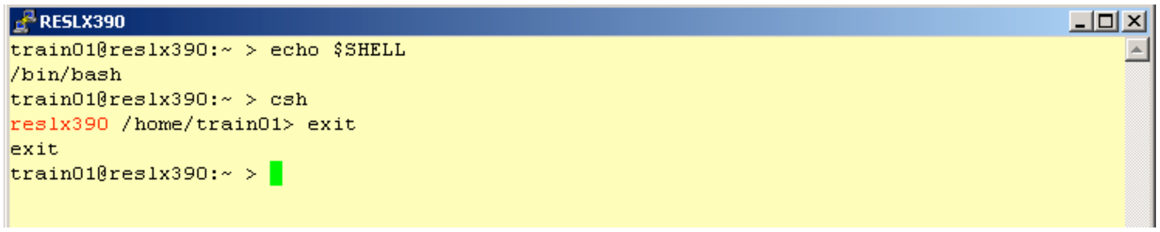
1. What shell are you using (see screen shot under "Invoke the C shell")
29. Editing the command line:
    - Scrolling through past commands: Use the up arrow and down arrow cursor keys
    - Inserting/deleting characters on command line: Use the <CTRL-U>, <DEL>, <INS> keys
    - Use the <CTRL-T> to correct spelling errors on the command line



```
train01@reslx390:~ > letetrs
```

```
train01@reslx390:~ > letters
```

30. Try command completion:
    - Note what happens when you issue: `ls /etc/pro<TAB>`
31. Invoke the C shell (and then exit)

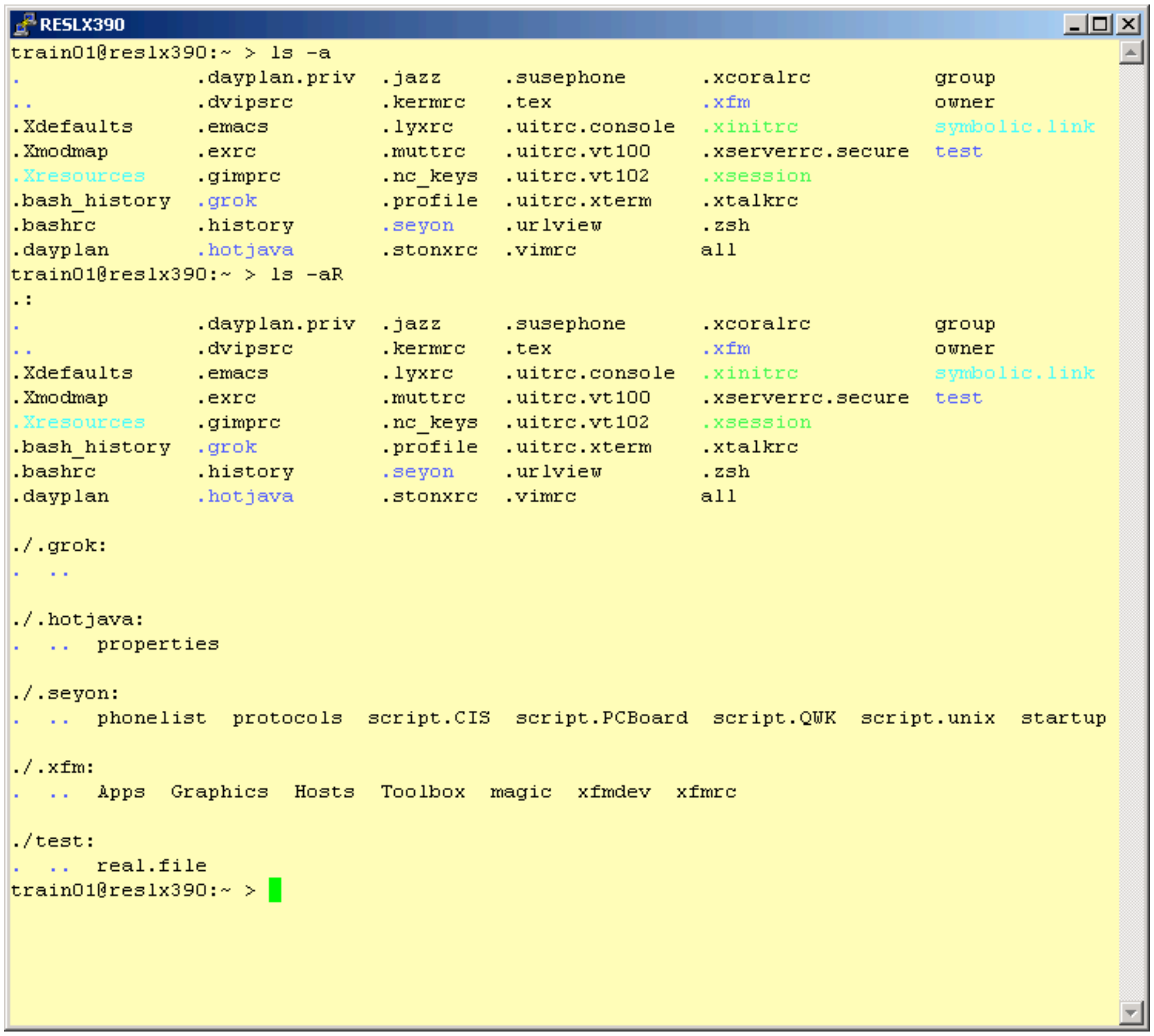


```
train01@reslx390:~ > echo $SHELL
/bin/bash
train01@reslx390:~ > csh
reslx390 /home/train01> exit
exit
train01@reslx390:~ >
```

11.0 Lab Eight

1. Use the `ls -a` command to display directories

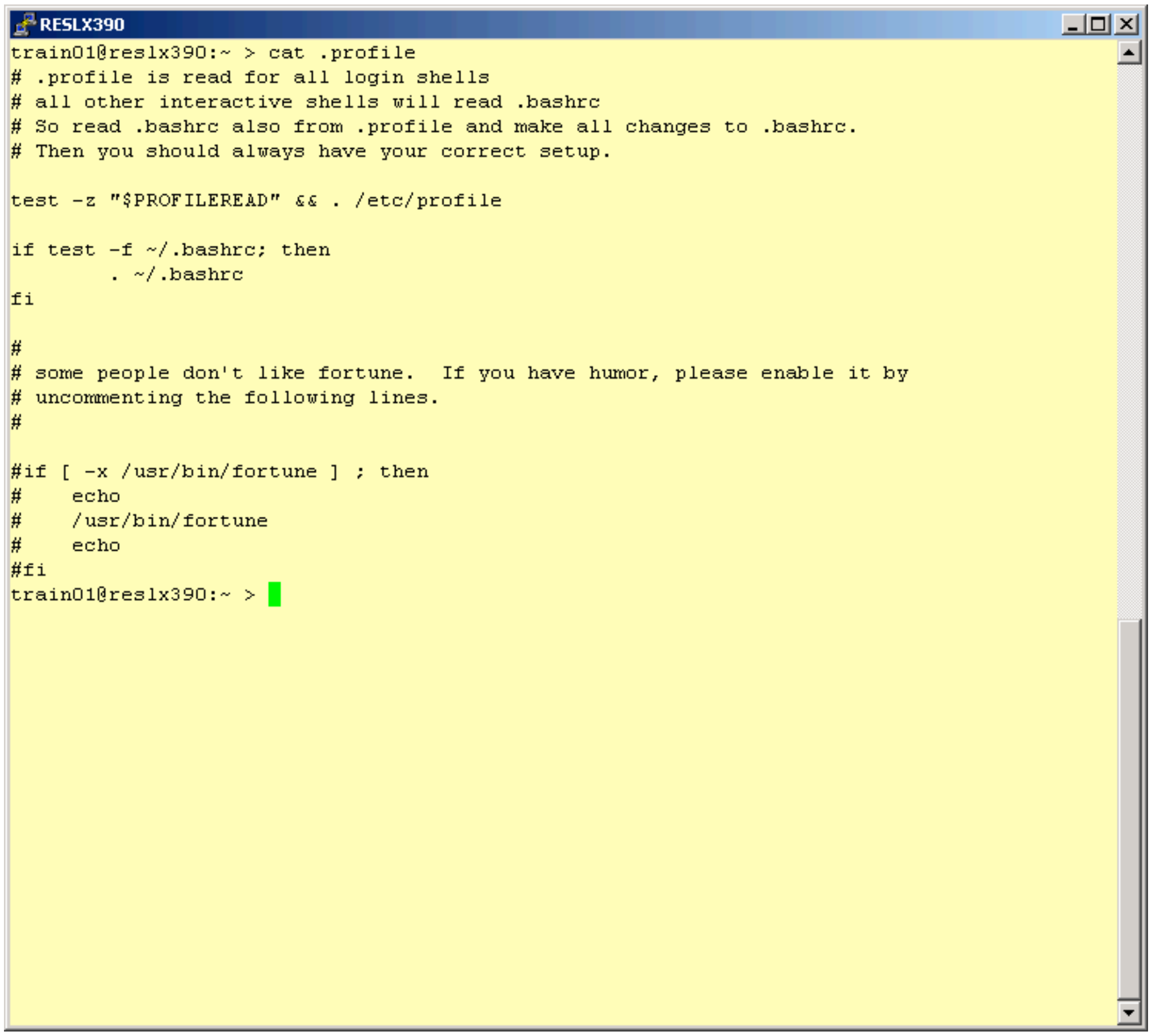
32. Use the `-R` option of `ls` to display down file tree



RESLX390

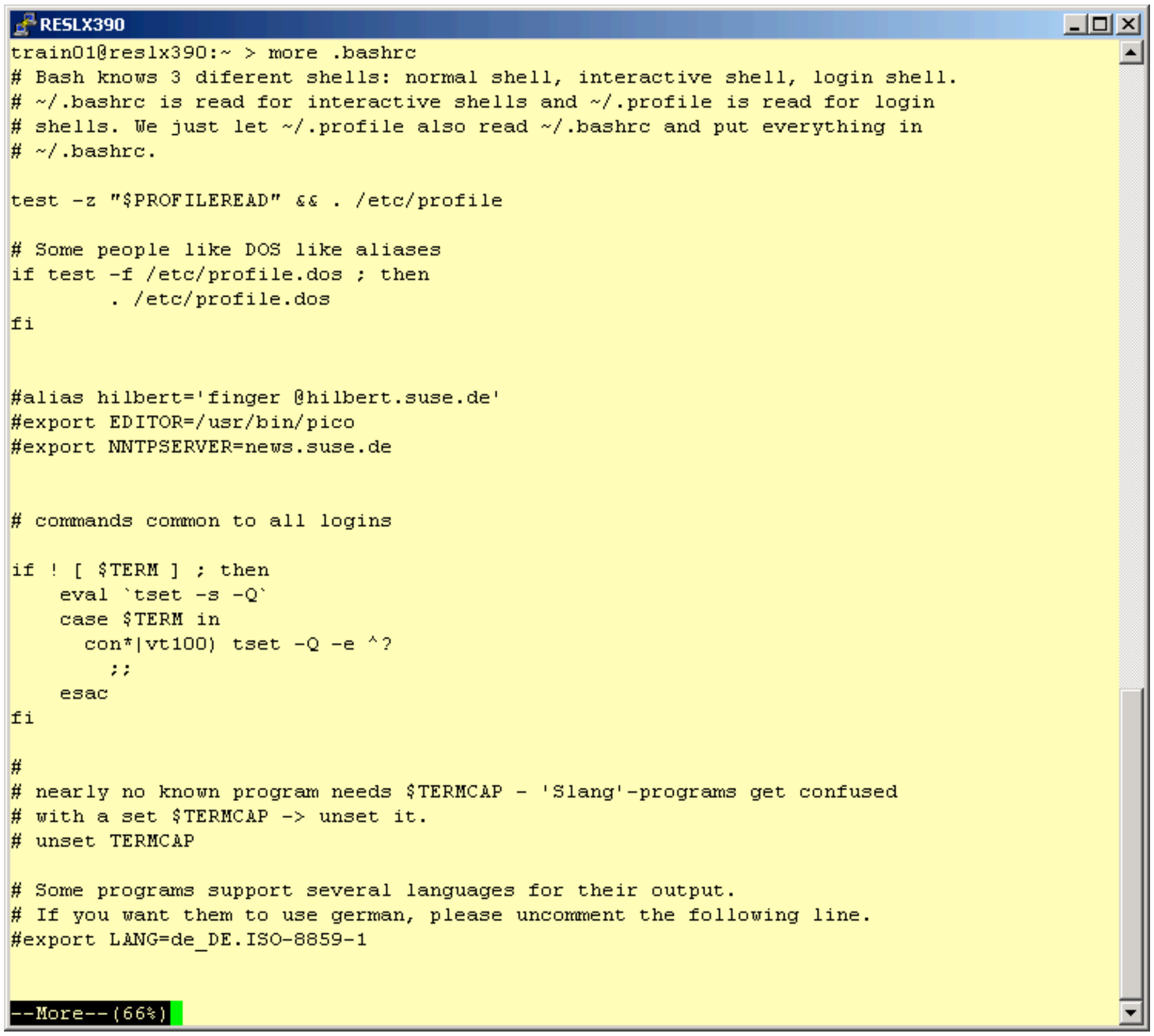
```
train01@reslx390:~ > ls -a
.                .dayplan.priv  .jazz      .susephone       .xcoralrc          group
..               .dvipsrc       .kermrc    .tex             .xfm               owner
.Xdefaults       .emacs         .lyxrc     .uitrc.console   .xinitrc           symbolic.link
.Xmodmap         .exrc          .muttrc    .uitrc.vt100     .xserverrc.secure  test
.Xresources      .gimprc        .nc_keys   .uitrc.vt102     .xsession
.bash_history    .grok          .profile   .uitrc.xterm     .xtalkrc
.bashrc          .history       .seyon     .urlview         .zsh
.dayplan         .hotjava       .stonxrc   .vimrc           all
train01@reslx390:~ > ls -aR
.:
.                .dayplan.priv  .jazz      .susephone       .xcoralrc          group
..               .dvipsrc       .kermrc    .tex             .xfm               owner
.Xdefaults       .emacs         .lyxrc     .uitrc.console   .xinitrc           symbolic.link
.Xmodmap         .exrc          .muttrc    .uitrc.vt100     .xserverrc.secure  test
.Xresources      .gimprc        .nc_keys   .uitrc.vt102     .xsession
.bash_history    .grok          .profile   .uitrc.xterm     .xtalkrc
.bashrc          .history       .seyon     .urlview         .zsh
.dayplan         .hotjava       .stonxrc   .vimrc           all

./.grok:
.  ..

./.hotjava:
.  ..  properties

./.seyon:
.  ..  phonelist  protocols  script.CIS  script.PCBoard  script.QWK  script.unix  startup

./.xfm:
.  ..  Apps  Graphics  Hosts  Toolbox  magic  xfmdev  xfmrc

./test:
.  ..  real.file
train01@reslx390:~ >
```

33. Use `cat` to display a file



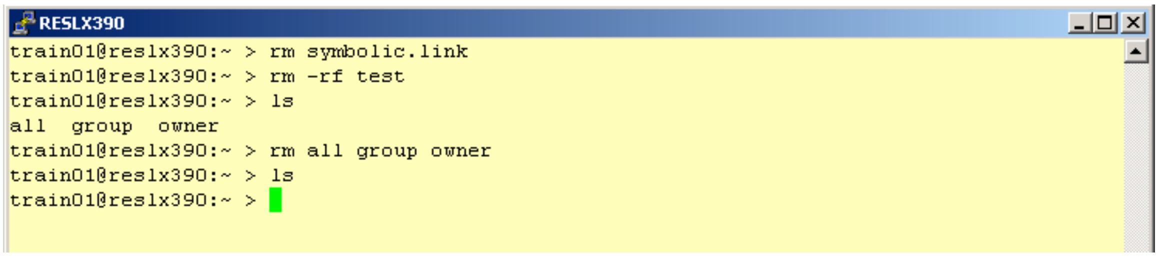
```
train01@reslx390:~ > cat .profile
# .profile is read for all login shells
# all other interactive shells will read .bashrc
# So read .bashrc also from .profile and make all changes to .bashrc.
# Then you should always have your correct setup.

test -z "$PROFILEREAD" && . /etc/profile

if test -f ~/.bashrc; then
        . ~/.bashrc
fi

#
# some people don't like fortune.  If you have humor, please enable it by
# uncommenting the following lines.
#

#if [ -x /usr/bin/fortune ] ; then
#    echo
#    /usr/bin/fortune
#    echo
#fi
train01@reslx390:~ > 
```

34. Use `more` to display a file one page at a time



```
train01@reslx390:~ > more .bashrc
# Bash knows 3 diferent shells: normal shell, interactive shell, login shell.
# ~/.bashrc is read for interactive shells and ~/.profile is read for login
# shells. We just let ~/.profile also read ~/.bashrc and put everything in
# ~/.bashrc.

test -z "$PROFILEREAD" && . /etc/profile

# Some people like DOS like aliases
if test -f /etc/profile.dos ; then
        . /etc/profile.dos
fi


#alias hilbert='finger @hilbert.suse.de'
#export EDITOR=/usr/bin/pico
#export NNTPSERVER=news.suse.de


# commands common to all logins

if ! [ $TERM ] ; then
    eval `tset -s -Q`
    case $TERM in
      con*|vt100) tset -Q -e ^?
        ;;
    esac
fi

#
# nearly no known program needs $TERMCAP - 'Slang'-programs get confused
# with a set $TERMCAP -> unset it.
# unset TERMCAP

# Some programs support several languages for their output.
# If you want them to use german, please uncomment the following line.
#export LANG=de_DE.ISO-8859-1


--More--(66%)
```

35. Erase the link ‘`symbolic.link`’, erase the ‘`test`’ directory and its contents, then erase the ‘`all`’, ‘`group`’, and ‘`owner`’ files.



```
train01@reslx390:~ > rm symbolic.link
train01@reslx390:~ > rm -rf test
train01@reslx390:~ > ls
all  group  owner
train01@reslx390:~ > rm all group owner
train01@reslx390:~ > ls
train01@reslx390:~ >
```

12.0 Appendix C. How to Speak 'Strine

(Courtesy of www.twi.ch & http://australia-online.com/diction.html)

If you want to pass for a native of Australia try speaking slightly nasally, shortening any word of more than two syllables and then adding a vowel to the end of it, making anything you can into a diminutive (even the Hell's Angels can become mere bikies) and peppering your speech with as many expletives as possible.

12.1 Common Words

Arvo

Afternoon. “The Sarvo” means this afternoon, as in “Seeya the sarvo”. On Xmas morning a lot of people go to the beach to test out their new prezzies. But by the early arvo, they're at home stuffing themselves with Chrissie din-dins!

Avagoodweegend

Classic Aussie farewell comparable to American TGIF, basically means “Have a good weekend!”

Bend the Elbow

To have a drink - pretty well self-explanatory!

Bickie

Rhymes with “sticky”. Literally means a biscuit, but Aussie bickies are more like American cookies, and American biscuits are more like Australian scones (pronounced like the “Fonz”!)... go figure!

Bloody

Universal epithet: the great Australian adjective. Used to emphasize any point or story. Hence “bloody beauty”(bewdy!) or “bloody horrible” or even “absa-bloody-lutely”!

Bludger

Lazy bastard, definitely an insult in Oz. Originally thought to be someone who lives off the earnings of a call girl. In conversation, the verb 'to bludge' is most commonly used like the US 'to bum' a cigarette.

Bob's Yer Uncle

“Everything is OK” or “Everything's Sweet” or “Going according to plan”. Similar phrase includes: “She's apples!”... Bob may refer to Australia's long-serving Prime Minister , Sir Robert “Bob” Menzies.

Bon-Bons

Christmas Crackers. Special party favours which are essential on the Christmas table. Shaped like big lollies, their contents always include a corny joke or riddle, small plastic toy and a paper party hat (which must be worn by all who attend Chrissie dinner, even guests)

Bonzer

Pronounced “bonza” - grouse, great, excellent.

Boxing Day

December 26th. Public holiday and traditional outdoor barbie day. Major Aussie sporting events kick off on this day including the 'Sydney to Hobart' and the 'Melbourne Test”.

Chew the Fat

To talk, engage in pleasant conversation, to have a chinwag.

Chook

Chicken. Often served barbecued at fancy turns (parties). If your hostess is befuddled and/or overcome by trying to do too many things at once, one might say she was “running around like a chook with its head cut-off!”

Chrissie

Christmas. By now you probably realize that Aussies like to shorten any words they can by adding an “o” or “ie” or “y”. No bloke named Christopher would be called Chrissie, probably 'Chrisso' or 'Toffa'.

Crack a Tinnie

Means to open up a can of beer major pastime during Aussie silly season.

Dial

Face. If some says to put a 'smile on your dial' it basically means to cheer up, she'll be right, mate.

Dunny

The toilet, W.C., or bathroom. If someone busting to know where the dunny is, tell 'em to “follow their nose to the thunderbox”.

Esky

Portable icebox or cooler - it's always a good idea to have one in the boot (trunk) of your car stocked with some cold ones (ice cold tubes) just in case the party's bar runs dry.

Fair Dinkum

Kosher, the real thing - as in “Fair Dinkum Aussie” (true blue Australian original). Often used by itself as a rhetorical question to express astonishment verging on disbelief ... “Fair Dinkum, mate?” (you' ve got to be kidding, haven't you?)

Full as a Goog

Completely filled with food (and drink). A 'Goog' is an egg (sometimes called a “googie egg”).

G'day

Universal greeting, used anytime day or night, but never as a farewell. Pronounced “gud-eye”, usually followed by “mate” (mite) or a typically strung-together “howyagoinallright” (= how are you today, feeling pretty good?)

Good Onya

Omnipresent term of approval, sometimes ironic, offering various degrees of heartfelt congratulations depending on inflection. Indispensable during Aussie small talk - substitute “really, oh yeh, aha, etc.”

Good Tucker

Excellent food. After pigging out at Chrissie lunch, it's polite to tell your hosts how good the tucker was.

Grouse

Rhymes with “house” and means outstanding, tremendous. Can be applied universally to all things social ... “grouse birds (women), grouse band, in fact, grouse bloody gay and hearty (great party!)”

Happy as Larry

Fortunate, lucky. Who “Larry” is may forever be lost at the bottom of the Katherine Gorge.

Holls

Vacations or 'holidays'. Since most Aussies get at least 4 weeks 'holls' every year they usually take 2 or 3 of them at Chrissie which is our biggest family get together time (like US Thanksgiving).

Hooroo

Pronounced “who-roo”... means “see ya later”, make sure you don't say g'day when meaning goodbye - it's a dead giveaway you're not a true blue Aussie battler!

Laughing Gear

Mouth. Common phrase is “Wrap your laughing gear around this one” i.e. Have a drink!

Lolly

A sweet or candy. But to “Do Your Lolly” means to get agitated and angry, similar to “Spit the dummie”

Mate

Friend, associate, or anyone you can't remember the name of

Melbourne Test

Game of cricket played by Australia's national cricko team versus visiting country usually starting on Boxing day. The game lasts for up to 6 days, and is watched religiously on the TV (like a 5 day Superbowl)

Ocker

Pronounced “ocka” - Typical uncultivated Australian, similar to American “redneck”.

Paralytic

Extremely drunk. Not good form too early on at a bash (party) especially if you end up having an “up & under” or “chunder” (puking or throwing-up while inebriated).

Plonk

Wine. Never used to describe the other main alcoholic beverage at an Australian social occasion - beer, i.e. the golden nectar, throat charmer, ice cold tube, etc.

Poets Day

Friday. Stands for P*ss off early, tomorrow’s Saturday.

Prezzie

A present or gift. If you've been a good little vegemite you'll probably get lots of bonza prezzies.

Pull your head In!

Use sparingly, since this equates a rather annoyed “shut up & mind your own business”. Only say this to the host if you know you're leaving (or off like a bride's nightie).

Raw Prawn

Not necessarily an uncooked shrimp! If someone says “Don't come the Raw Prawn with me, mate” it basically means “Don't try to fool me or rip me off” or “Rack off Noddy, you're being a tad offensive”.

Rels/Relos

Relatives, The family members you probably only see every Christmas!

Ripper

Pronounced “rippa” means beaut, tippy-tops, grouse - that bloke named “Jack” in the old Dart (England) was certainly not ripper!

Sheila

Archaic term now only found in Paul Hogan movies

Shout

To shout means to buy the next round (of drinks usually), so if someone says "It's your shout, mate" don't get vocal, just buy a couple of tinnies (cans of beer) and remain sociable, the next few drinks are someone else's responsibility!

Silly Season

Traditional summer holiday period, kicking off in December and running through to our national holiday Australia Day, January 26th (similar to the US July 4th).

Spit The Dummie

A "dummie" is Australian for a child's pacifier. Your Hostess will not usually have cause to spit the dummie (completely lose her cool or go ballistic) if you and your mates can act like proper toffs (refined gentlemen) and enjoy the soiree!

Starters

Australian for Hors D'oeuvres or "appetizers". Aussies call their appetizers "entrees". Also your first drink of the day, hence the ubiquitous question heard throughout the Silly Season: "What's for Starters?" Also commonly called a "Heart Starter".

Strewth

Pronounced "sta-ruth". A general exclamation of disbelief or shock: i.e. "Strewth, would ya hava go at that, then?!" (My goodness, can you believe what we're seeing!?)

Sydney to Hobart

World famous Australian ocean Yacht race that commences on Boxing Day in Sydney Harbour boats from all around the world race down the East Coast across Bass Strait to our Apple Isle, "Tassie" (Tasmania).

The Go

The "rage" or current trendy thing. The latest trend in clothing or whatever is described as "all the go!"

Whinge

Rhymes with "hinge" as in door! Means to complain incessantly or to "belly ache" (= "whine"). Whingers are not fun to have around and definitely not likely to be asked back again to the next party. If you must whinge, keep it amongst your good mates!