**Chapter 4: 21, 23 ,25, 26**

**4.21**

You have a number of programs in $HOME/progs which are called by other programs. You have now decided to move these programs to $HOME/internet/progs. How can you ensure that users don't notice this change?

In this case you could use a symbolic link in $HOME/progs that points to $HOME/internet/progs. That way when the programs are called they are simply pointed to by the symbolic link.

**4.23.**

Explain how ls obtains the

1. filename
   1. The filename stored within the directory itself, therefore the kernel displays that listing when ls is run
2. name of owner
   1. The name of the owner of the file is stored in the inode of the file. The kernel gets the inode number from the directory, and then searches that inode when ls is run
3. name of group owner when displaying the listing
   1. The name of the group owner is stored in the inode as well, so when ls is run, the inode number is read and then the kernel follows the pointer to find the name of the group owner

**4.25**

The owner can change all attributes of a file on a BSD-based system. Explain whether the statement is true or false. Is there any attribute that can only be changed by the superuser?

This statement is false. On a BSD-based system only a system administrator is able to change the ownership of a file itself, this answers the following question in the case that the file ownership is the only attribute that can be changed by a superuser.

**4.26**

What are the three time stamps maintained in the inode, and how do you display two of them for the file foo?

The three time stamps maintained in the inode are when the inode itself was last modified, when the file content was last modified, and when the file was last accessed.

In order to display the time it was last modified you can used the ls -l command, whereas for time of last modification you can use the ls -lu command

**Chapter 6: 3, 5, 6, 7, 10, 11**

**3.**

Devise wild-card patterns to match the following filenames:

1. foo1, foo2, and Foo5
   1. [Ff]oo[125]
2. quit.c, quit.o, quit.h
   1. quit.[coh]
3. watch.htm, watch.HTML, Watch.html
   1. [wW]atch.[hH][tT][mM][lL]
4. all filenames that begin with a dot and end with a .swp
   1. .\*.swp

**5.**

How do you remove from the current directory all ordinary files that

1. are hidden
   1. rm .\*
2. begin and end with #
   1. rm #\*#
3. have numerals as the first three characters
   1. rm [0-9][0-9][0-9]\*
4. have single-character extensions
   1. rm \*.?

These commands will work in all shells including the c-shell since we aren't using !

**6.**

Devise wild-card patterns to match all filenames comprising at least three characters

1. where the first character is numeric and the last character is not alphabetic
   1. [0-9]\*[!a-zA-Z]
2. not beginning with a dot
   1. [!.]\*
3. containing 2004 as an embedded string except at the beginning or the end
   1. {!2004}\*{!2004}

**7.**

Explain what these wild-card patterns match:

1. [A-z]????\*
   1. This will match a string beginning with either an upper case letter, [, \, ], ^, \_, `, or a lower-case letter follwed by at least 4 characters
2. \*[0-9]\*
   1. This will match a string that has at least one digit in the middle
3. \*[!0-9]
   1. This will match a string that ends with a non-digit
4. \*.[!s][!h]
   1. This will match a string that begins with any amount of character followed by a . with any character that is not an s, followed by any character that is not an h

**10.**

Explain the significance of single and double quoting, including when one is preferred to the other. What are the two consequences of using double quotes.

Any expression contained within single quotes will negate all special characters for all shells except for c-shell. With double quotes will turn off all special characters except the $ and \.

The consequences of using double quotes is the variables will still be substituted in the comman and the escape character will still effect any following character

**11.**

When will wc < chap0[1-5] work? How can you remove chap0[1-5] if you have a file of that name?

wc < chap0[1-5] will only work if there exists the files chap01, chap02, chap03, chap04 or chap05.

If you wish to remove a file of that name you have to use the command

rm “chap0[1-5]”