$$

Q1(a

The output of the current working directory is **/usr/bin** and I get this as oppose to just **/bin/** because it is a symbolic link.

%%

Q1(b

I. The command **ls ??????????????** displays files whose names are of length 15 characters.

dvd+rw-booktype* glib-gettextize* gtkdoc-scangobj* libpng12-config*

dvd-ram-control* gnome-perfmeter* intltool-update* run-with-aspell*

gdmXnestchooser* gnome-printinfo* libIDL-config-2* tsoljds-tstripe*

glib-genmarshal* gst-inspect-0.8* libpng10-config* tsoljdslabel-ui*

II. The command **ls ?z*** displays all files that have z as the second letter.

7z* bzegrep* bzmore* gzfgrep* gznew*

7za* bzfgrep* gzcat* gzforce* tzselect*

7zr* bzgrep* gzcmp* gzgrep*

bzcat* bzip2* gzdiff* gzip*

bzcmp* bzip2recover* gzegrep* gzless*

bzdiff* bzless* gzexe* gzmore*

III. The command **ls *[ij]** displays the files that end with either an i or a j.

fmli* gtkdoc-scanobj* native2ascii@ vi*

gtkdoc-scangobj* idlj@ tsoljdslabel-ui* xdtosj@

IV. The command **ls –d [A-Z]*** displays the files that begin with an uppercase letter.

CC@ ControlPanel@ HtmlConverter@

CCadmin@ DBMirror.pl* X11/

V. The command **ls –d [ABCEFGHIJKLNOPQRSTUVWXYZ]** displays the files that begin with an uppercase letter other than D or M.

$$

Q2(a

The commands **cd $HOME** and **mkdir public_html** make a new directory with that name in the home directory.

%%

Q2(b

The command **ls -l** command displays the permissions for the public_html directory along with the other sibling directories.

%%

Q2(c

The command **cd public_html** changes the current working directory to public_html.

%%

Q2(d

The command **touch abc** creates a file called abc under the public_html directory.

%%

Q2(e

The command **cd ..** changes the directory to the parent directory (one level up).

%%

Q2(f

The command **chmod 300 public_html** changes the permissions of public_html to 300.

%%

Q2(g

I can see the information about the abc file because the permissions for that file are different from the permissions of the parent directory. We can't see any information about public_html because in the last question we took the read permission away from us (the owner).

%%

Q2(h

The minimum we should do to set all sub-direcotries to the same permissions would be using command **chmod -R 700 public_html** to set the public_html directory and file abc to rwx for owner.

$$$

Q3(a

The **ls –a –r .*rc** command dipslays all the hidden files in reverse lexicographical whose names end in rc.

.twmrc* .tcshrc* .mwmrc* .dmrc* .cshrc*

%%

Q3(b

The **finger ebachme** command shows me the following information about myself:

Login name: ebachme In real life: Eric James Bachmeier

Directory: /gaul/s1/student/2013/ebachme Shell: /local/tcsh

On since Sep 22 11:52:52 on pts/1 from nexus-3.wireless.uwo.ca

No unread mail

No Plan.

%%

Q3(c

I created a plan called .plan by using the command **vi .plan**.

%%

Q3(d

The command line **chmod 444 .plan** will give the read permissions to everyone.

%%

Q3(e

The **finger ebachme** command gives me this:

Login name: ebachme In real life: Eric James Bachmeier

Directory: /gaul/s1/student/2013/ebachme Shell: /local/tcsh

On since Sep 22 11:52:52 on pts/1 from nexus-3.wireless.uwo.ca

No unread mail

Plan:

I plan to finish this assignment before tomorrow night.

%%

Q3(f

The command **chmod 600 .plan** changes the permission of .plan to rw for owner only.

%%

Q3(g

After changing the permissions of .plan and using the **finger** command again, my plan was not displayed. It appears there needs to be read permissions for at least owner and other to view the plan.

%%

Q3(h

The difference between the outputs of (b), (e), and (g) is that before I created a plan, there was No Plan being displayed in that line. After I created a plan it has the plan previewed in that line. When the permissions were changed for part (g) the Plan: heading was there but the actual file info was not outputted to the screen.

$$

Q4(a

To create a directory in my home directory called Working-Area I use the command **mkdir Working-Area** after ensuring I was in my home dir (**cd**).

%%

Q4(b

I used the command **mkdir Working-Area/Dir1** to create the sub-directory Dir1 and the command **touch Working-Area/{File1}** to create a new file called File1.

%%

Q4(c

After changing the current working directory to Working-Area I use the command **mkdir Dir1/Dir3 Dir1/Dir4** to create these two new sub-directories.

%%

Q4(d

The command **touch Dir1/Dir3/{File3}** creates a regular file under the Dir3 directory.

%%

Q4(e

After changing the working directory to Dir4 (**cd Dir4**) I use the command **touch File4},{File5},{File6.**

%%

Q4(f

To create Dir2 as a symbolic link I first change the working directory to **cd Working-Area** and then use the command **ln -s Dir1/Dir4 Dir2.** In this case, Dir1/Dir4 directory is the source.

%%

Q4(g

The command **chmod 700 Working-Area** sets the permissions all to user and none for other or group users. I did this command from my home directory (**cd $HOME**).

%%

Q4(h

The command **chmod 750 Working-Area/Dir1/Dir3** sets the permissions to all for owner and only read and execute for group users.

%%

Q4(i

The command **chmod 755 Working-Area/Dir1/Dir3/File3** sets the permissions to all for owner and only read and execute for other and group users.

%%

Q4(j

The command **chmod 511 Working-Area/Dir1/Dir4/File5** sets the permissions to read and execute for owner and only execute for other and group users.

$$

Q5(a

After returning to my home directory (**cd $HOME**) I use the command **cat > letter.txt** to create a file and enter the numbers 01 through 12 on the following command lines.

%%

Q5(b

The command **more letter.txt** displays the contents of the text file one page at a time.

01

02

03

04

05

06

07

08

09

10

11

12

%%

Q5(c

The command **tail -3 ~/letter.txt** displays the last three lines of the text file.

10

11

12

The command **tail +3 ~/letter.txt** displays the text file, starting at the third line.

03

04

05

06

07

08

09

10

11

12

%%

Q5(d

The command **head -3 ~/letter.txt** displays the first three lines of the text file.

01

02

03

The command **head +3 ~/letter.txt** displays every line before the third last line.

01

02

03

04

05

06

07

08

09

10

%%

Q5(e

The Unix command **who | tee ~/letter.txt | wc -l** writes the output of the who command to letter.txt and outputs the number of lines in the updated file to the screen.

15

%%

Q5(f

The Unix command to show the calendar for November 1955 would be **cal 11 1955.**

November 1955

S	M	Tu	W	Th	F	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

%%

Q5(g

The difference between **cat < letter.txt** and **cat letter.txt** is that the cat command with < opens the file through the shell rather than the cat command.

%%

Q5(h

The command **echo cat** displays the word cat as standard output to the screen.

cat

The command **cat echo** tries to open echo as a file and display its contents, but the file does not exist.

cat: cannot open echo

$$

Q6(a

The command **cp –R courses/* $HOME** would copy all the files and sub-directories under courses to the home directory. This method of commands does this copying recursively.

%%

Q6(b

The command **cp –R ~/courses/* ~** would copy all the files and sub-directories even I we wasn’t in the home directory.

%%

Q6(c

The command **chmod –R 700 ~/courses** would change the permissions of all files and directories under courses/ to rwx for owner only.

$$

Q7(a

In Unix, an absolute pathname specifies the location of a file starting from the root.

Example: student/ebachme/Documents/testFile.txt

A relative pathname is a directory relating to the present working directory.

Example: If we are already in the /student/ebachme/ directory we can make the current working directory the Downloads folder by using relative path **cd Downloads.**

%%

Q7(b

The ./ directory is the current working directory.

%%

Q7(c

The ../ directory is the parent directory.

%%

Q7(d

The ~/ directory is the home directory.

%%

Q7(e

The **rmdir** command could have given an error message because there may be hidden files in the directory that is trying to be deleted.

%%

Q7(f

This command could have failed because:

1. The file1 may not be in the current directory.

2. The directory backup/file1.bak may be relative and the user may be working from the wrong directory.

3. The two arguments should be switched so that the file copies to the directory. You cannot copy a directory to a file.

%%

Q7(g

The Unix command **cp ~/.login /dev/pts/7** seems to get login

information from the .login hidden directory and copy it to my terminal window.

%%

Q7(h

I. A value of 000 implies that everyone has all permissions.

II. A value of 002 implies that the other (anyone else) users have read and execute permissions but the owner and group users have all permissions.

$$