4. UNIX Navigation

This Lab deals with the commands used to navigate the UNIX file structure which we discussed in the last chapter.

Lab 4.1 - pwd (Print Working Directory)

When you first log in on a UNIX system, the *working* or *current* directory is your *home* directory. To find out where you are (i.e. what directory you are in) and display the absolute pathname of the current directory, type **pwd** (**P**rint **W**orking **D**irectory):

pwd
/home/user3332/

Lab 4.2 - ls (Listing the contents of a directory)

When you first login, your current working directory is your home directory. Your home directory has the same name as your user-name, for example, **user3992**, and it is where your personal files and subdirectories are saved.

To find out what is in your home directory, type

ls *(short for list)*

The **ls** command lists the contents (i.e. all your files, directories) of your current directory that you are in. By itself **ls** displays a short listing, containing only the names of the files. There may be no files visible in your home directory yet, in which case, the UNIX prompt will be returned. Alternatively, there may already be some files inserted by the System Administrator when your account was created.

ls does not cause all the files in your home directory to be listed, but only those ones whose name does not begin with a dot (**.**) Files beginning with a dot (**.**) are known as hidden files and usually contain important program configuration information. They are hidden because you should not change them unless you are very familiar with UNIX!!!

To list all files in your home directory including those whose names begin with a dot, type

```
ls -a
```

ls is an example of a command which can take options: **-a** is an example of an option. The options change the behaviour of the command. There are online manual pages (man command) that tell you which options a particular command can take, and how each option modifies the behaviour of the command.

The following will display detailed info about each file.

```
ls –l
```

You can combine options:

```
ls –lart
```

The following –d option will display a directory if it exists:

```
ls –d training
training/
```

To use a command, for example ls, on a file (or directory) not in the current working directory (the directory you are currently in), you must specify its full pathname, for example:

```
ls directory1/directorya
```

Home directories can also be referred to by the ~ (tilde) character. So for example typing:

```
ls ~user3022
```

will list the contents of user3022's home directory, no matter where you currently are in the file system.

Lab 4.3 – mkdir (Creating a directory)

The mkdir command is used to make directories. The basic syntax is **mkdir directory-name**. For example, create a directory called **training** within your current directory by typing:

```
mkdir training
ls -d training (checks to see it exists, you could also simply type ls)
training
```

The directory called training now exists. You can create more than one directory at the same time. Type the following:

```
mkdir dir1 dir2
ls                    // you will see that dir1 and dir2 exist
```

Lab 4.4 - cd (Changing Directories)

The command **cd < *change directory*>** means "change the current working directory to 'directory'. The current working directory may be thought of as the directory you are in, i.e. your current position in the file-system tree.
You should currently be in your home/userxxxx directory. You can check this by typing pwd. To change to the directory you have just created, type

```
cd training
pwd          // to check your location
```

Type **ls** to see the contents of the directory (which should be empty as you have just created it)

Lab 4.5 - Using .., /, -

To change back to your home directory type:

```
cd ..        // to go up a level
pwd          // to check where you are
/home/user3992/
```

where ".." stands for up one level.

To change to the root directory:

```
cd /
pwd
/
```

To change back to the directory that you last came from:

```
cd –
pwd
/home/user3992/
```

Lastly, you can return back to your home directory (from anywhere within the directory structure) by typing:

```
cd
```

Entering cd with no parameter returns you to your home directory no matter where you are. You can check to make sure that it worked by entering:

```
pwd
/home/user3992
```

Whatever it returns, the list should end in your username.

Lab 4.6 - Understanding pathnames

First type **cd** to get back to your home-directory.

Now, say you want to create a sub-directory called backups within the training directory you could type:

```
cd training    (to get you to your training directory)
```

create a directory in here called backups

```
mkdir backups
```

cd back to your home-directory, typing pwd to make sure you are in your home directory then type

```
ls training
```

This will give you a list of what is contained in the training directory which will be the directory backups. Now type

```
ls backups
```

You will get a message like this -

backups: No such file or directory

The reason is, **backups** is not in your current working directory. To use a command on a file (or directory) not in the current working directory (the directory you are currently in), you must either **cd** to the correct directory, or specify its full pathname. To list the contents of your **backups** directory, you must type

```
ls training/backups
```

~ (your home directory)

Home directories can also be referred to by the tilde **~** character. It can be used to specify paths starting at your home directory. So typing

```
ls ~/training
```

will list the contents of your training directory, no matter where you currently are in the file system.

Lab 4.7 - Creating full directory paths

If you enter the following command, it will create the newdir directory within the training directory:

```
mkdir training/newdir
```

If you enter the following command, you will get an error because the mkdir command by itself just creates one level at a time:

```
mkdir a/b/c/d
mkdir: Failed to make directory "a/b/c/d"; No such file or directory
```

Using the -**p** option with the mkdir command creates all of the directories & sub-directories:

```
mkdir –p a/b/c/d
```

```
ls –R a          // -R option displays everything under the directory recursively
b/
a/b:
c/
a/b/c:
d/
a/b/c/d:
```

Lab 4.8 – rmdir (Removing a directory)

The rmdir command removes a directory from the filesystem tree:

```
rmdir dir2
ls –d dir2              // dir2 is now deleted
dir2 not found
```

The rmdir when used on its own will not work unless the directories to be removed are completely empty, for example:

```
rmdir training          // this directory contains another directory called backups
rmdir: directory "training": Directory not empty
```

One method of removing all these sub-directories is to cd into each sub-directory & remove one level at a time:

```
cd a/b/c
rmdir d
cd ..
rmdir c
etc….
```

A more efficient method is to use the rm command with the -r option. The rm -r command will first remove the contents of the directory (i.e. files and/or sub-directories), and then will remove the directory itself, for example:

```
rm –r a
```

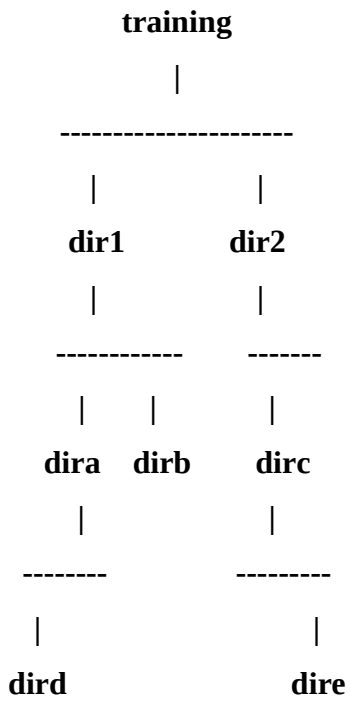
or to ask for confirmation before deleting:

```
rm –ir a
```

Warning: Be very careful when using the rm –r and rmdir commands that you do not delete your home directory. There is no UNDO in UNIX.

Exercise 4.1

Remove the original training directory you created and create a new training directory with the following structure in your UNIX account exactly as shown:



Go to your home directory and from there navigate into dird in one command and delete dirb from dird in one command.