1.1 Listing files and directories

`ls` (list)

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, **`alovelace`**, and it is where your personal files and subdirectories are saved. To find out what is in your home directory, type:

```
>ls
```

The list command lists the contents of your current working directory. There may be no files visible in your home directory, in which case, the prompt will be returned. `ls` does not, in fact, 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 behavior of the command. There are system and online manual pages that tell you which options a particular command can take, and how each option modifies the behavior of the command.

1.2 Making Directories

`mkdir` (make directory)

We will now make a subdirectory in your home directory to hold the files you will be creating and using in the course of this guide. To make a subdirectory called **`unixfun`** in your current working directory type:

```
>mkdir unixfun
```

To see the directory you just created, type:

```
>ls
```

1.3 Changing to a different directory

`cd` (change directory)

The command **`cd 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.

To change to the directory you have just made, type

```
>cd unixfun
```

Now type **`ls`** to see the contents (which should be empty). Next create a sub-directory named `files` by typing:

```
>mkdir files
```

You should now how a sub-directory named `files` inside the directory `unixfun` which is in turn a sub-directory of your home directory.

1.4 Special directories

Still in the `unixfun` directory, type:

```
>ls -a
```

As you can see, in the `unixfun` directory (and in all other directories), there are two special directories called (.) and (..)

The current directory (.)

In UNIX, (.) means the current directory, so typing:

```
> cd .
```
(NOTE: there is a space between cd and the dot)

means stay where you are (the `unixfun` directory).

This may not seem very useful at first, but using (.) as the name of the current directory will save a lot of typing.

The parent directory (..)

(..) means the parent of the current directory, so typing:

```
>cd ..
```

will take you one directory up the hierarchy (back to your home directory in this case). Try it now.

Typing cd with no argument always returns you to your home directory. This is very useful if you are lost in the file system.

1.5 Pathnames

`pwd` (print working directory)

Pathnames enable you to work out where you are in relation to the whole file-system. For example, to find out the absolute pathname of your home-directory, type cd to get back to your home-directory and then type:

```
>pwd
```

The full pathname will look something like this:

```
/home/users/alovelace
```

which means that `alovelace` (your home directory) is in the `home` sub-directory, which is in the top-level **root directory** called "`/` " .

Understanding pathnames

First type `cd` to get back to your home-directory, then type:

```
>ls unixfun
```

to list the contents of your `unixfun` directory. Next, type:

```
>ls files
```

You will get a message similar to this:

```
files: No such file or directory
```

The reason is that `files` is not a sub-directory of 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 `files` directory from your home directory, you must type:

```
>ls unixfun/files
```

~ (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 ~/unixfun
```

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

1.6 Copying Files

cp (copy)

`cp file1 file2` is the command which makes a copy of `file1` in the current working directory and calls it

What we are going to do now, is to take a file stored in an open access area of the file system, and use the `cp` command to copy it to your `unixfun` directory.

First, change directories to your `unixfun` directory:

```
>cd ~/unixfun
```

Then type:

```
>cp ~mharmon/cs277/homeworks/prog1/activity.txt .
```

This command means to copy the file `activity.txt` to the current directory, keeping the name the same.

1.7 Moving Files

mv (move)

`mv file1 file2` moves (or renames) `file1` to `file2`

To move a file from one place to another, use the `mv` command. This has the effect of moving rather than copying the file, so you end up with only one file rather than two.
It can also be used to rename a file, by moving the file to the same directory, but giving it a different name.
We are now going to move the file `activities.txt` to your `files` directory.

First, change directories to your `unixfun` directory (can you remember how?). Then, inside the `unixfun` directory, type:

```
>mv activities.txt files/.
```

Type `ls` and `ls files` to see if it has worked.

1.8 Removing files and directories

`rm` (remove), `rmdir` (remove directory)

To delete (remove) a file, use the `rm` command. As an example, we are going to create a copy of the `activities.txt` file and then delete it.

Navigate to `~/unixfun/files` and then type:

```
>cp activities.txt temp.txt
>ls
>rm temp.txt
>ls
```

You can use the `rmdir` command to remove a directory (make sure it is empty first). Try to remove the `files` directory. You will not be able to since UNIX will not let you remove a non-empty directory.

1.9 Viewing contents of a file

`clear` (clear screen)

Before continuing, you may want to clear the terminal window of your previous commands. At the prompt type:

```
>clear
```

This will clear all text and leave you with the prompt at the top of the window.

`less`

The `less` command writes the contents of a file onto the screen one page at a time.

```
>less ~/unixfun/files/activities.txt
```

Press the space bar if you want to see another page, and type **q** if you want to quit reading.