19CSE213 - Operating Systems- Lab Sheet 1

Linux Command Familiarization

- Open Terminal program on your Ubuntu machine
- If you have not installed Ubuntu on your machine, try the online terminal https://www.webminal.org/ (Register first and wait for two minutes to get it activated. Then click on the 'Terminal' menu)

Section 1. man pages

This section will explain the use of ***man*** pages (also called manual pages) on your Unix or Linux computer. You will learn the man command together with related command ***whatis***. Most Unix files and commands have pretty good man pages to explain their use. Man pages also come in handy when you are using multiple flavours of Unix or several Linux distributions since options and parameters sometimes vary.

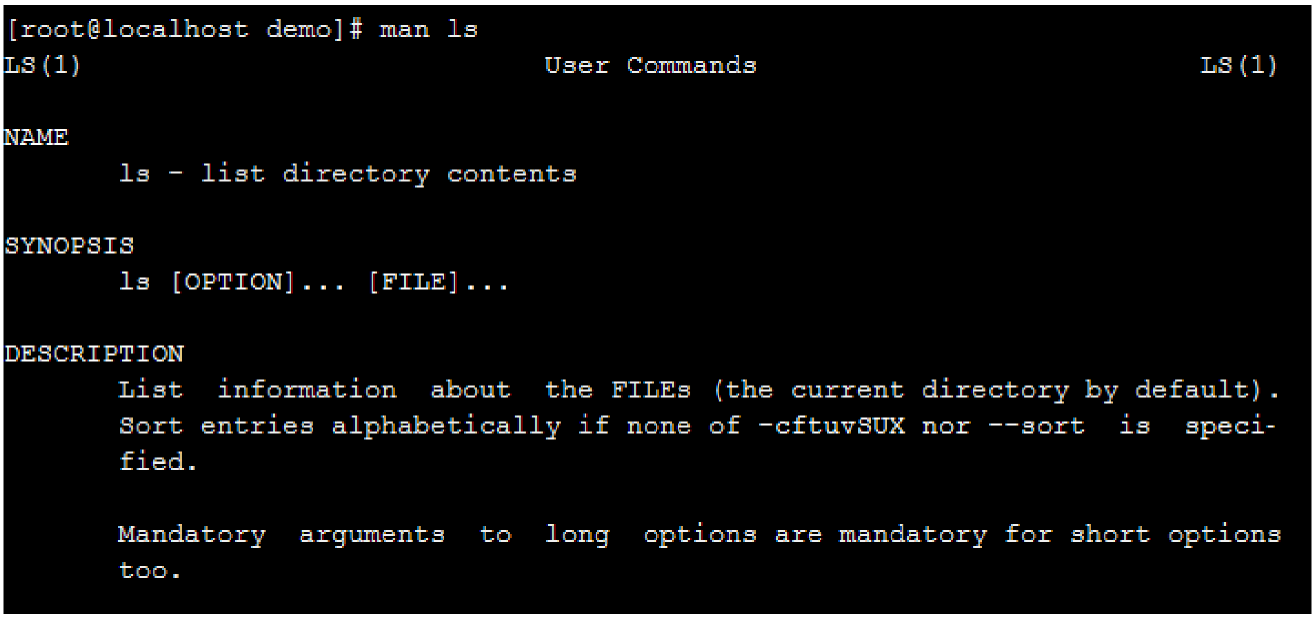
man command

Type ***man*** followed by a command (for which you want help) and start reading. Press q to quit the man page. Some man pages contain examples (near the end).

Webminal - Learn and Practise Linux online, Programming online

www.webminal.org

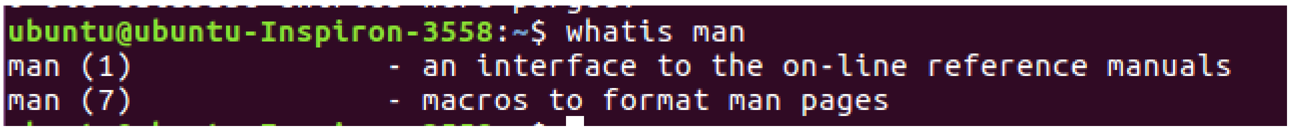
Free Features - Online Linux/bash terminal, no installation required. - Practice your Linux skills, including file system navigation, file management, scripting in bash/awk/sed and MySQL.



```
[root@localhost demo]# man ls
LS(1)                          User Commands                         LS(1)

NAME
       ls - list directory contents

SYNOPSIS
       ls [OPTION]... [FILE]...

DESCRIPTION
       List  information  about  the FILEs (the current directory by default).
       Sort entries alphabetically if none of -cftuvSUX nor --sort  is  speci-
       fied.

       Mandatory  arguments  to  long  options are mandatory for short options
       too.
```

whatis

To see just the description of a manual page, use ***whatis*** followed by a string.



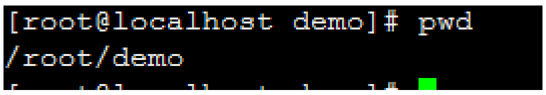
```
ubuntu@ubuntu-Inspiron-3558:~$ whatis man
man (1)              - an interface to the on-line reference manuals
man (7)              - macros to format man pages
```

Section 2: Working with directories

This module is a brief overview of the most common commands to work with directories: ***pwd***, ***cd, ls, mkdir*** and ***rmdir***. These commands are available on any Linux (or Unix) system.

pwd

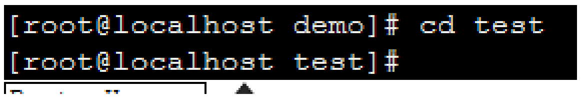
pwd command gives the full path or address of the current working directory. Go ahead, try it: Open a command line interface (also called a terminal, console) and type pwd. The tool displays your current directory.



```
[root@localhost demo]# pwd
/root/demo
```

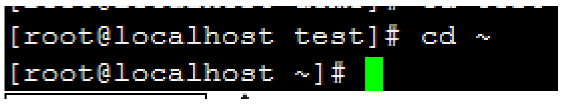
cd

You can change your current directory with the ***cd*** command (Change Directory).


```
[root@localhost demo]# cd test
[root@localhost test]#
```

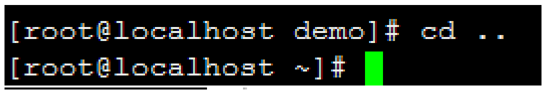
cd ~

The cd is also a shortcut to get back into your home directory. Just typing ***cd*** without a target directory, will put you in your home directory. Typing ***cd ~*** has the same effect.


```
[root@localhost test]# cd ~
[root@localhost ~]#
```

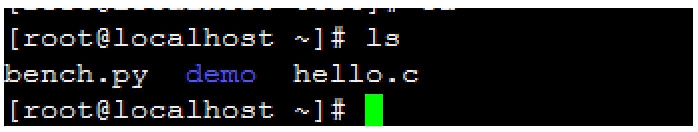
cd ..

To go to the parent directory (the one just above your current directory in the directory tree), type ***cd ..***


```
[root@localhost demo]# cd ..
[root@localhost ~]#
```

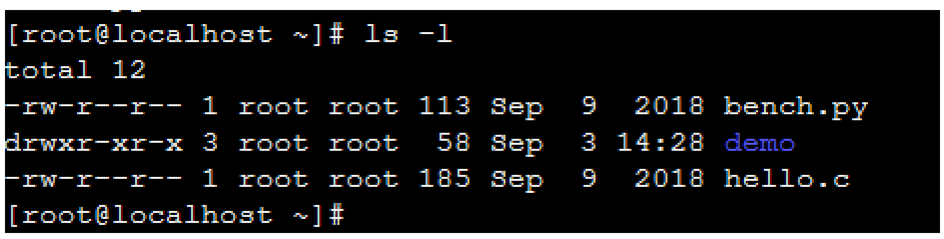
ls

You can list the contents of a directory with ls


```
[root@localhost ~]# ls
bench.py   demo   hello.c
[root@localhost ~]#
```

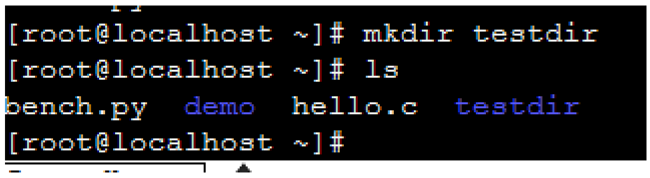
ls –l

Many times you will be using options with ls to display the contents of the directory in different formats or to display different parts of the directory. Typing just ls gives you a list of files in the directory. Typing ls -l (that is a letter L, not the number 1) gives you along listing. Go to the man page of ls and find out the different options of ls.


```
[root@localhost ~]# ls -l
total 12
-rw-r--r-- 1 root root 113 Sep  9  2018 bench.py
drwxr-xr-x 3 root root  58 Sep  3 14:28 demo
-rw-r--r-- 1 root root 185 Sep  9  2018 hello.c
[root@localhost ~]#
```

mkdir

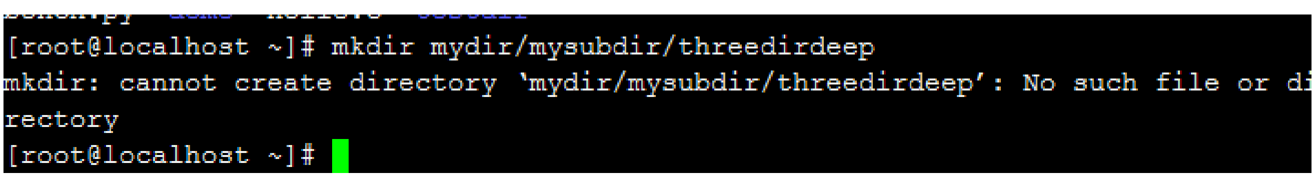
Walking around the Unix file tree is fun, but it is even more fun to create your own directories with mkdir. You have to give at least one parameter to mkdir, the name of the new directory to be created.



```
[root@localhost ~]# mkdir testdir
[root@localhost ~]# ls
bench.py  demo  hello.c  testdir
[root@localhost ~]#
```

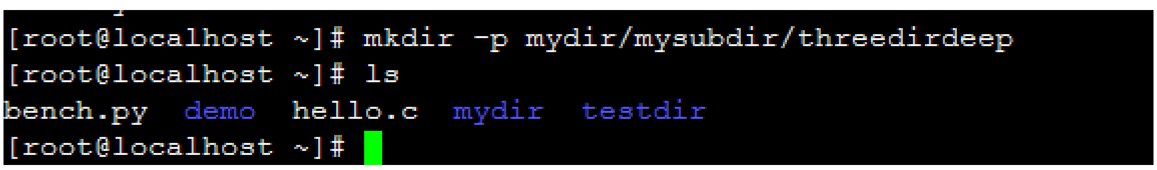
mkdir –p

The following command will fail, because the parent directory of t*hreedirsdeep* does not exist.



```
[root@localhost ~]# mkdir mydir/mysubdir/threedirdeep
mkdir: cannot create directory ‘mydir/mysubdir/threedirdeep’: No such file or di
rectory
[root@localhost ~]#
```

When given the option -p, mkdir will create parent directories as needed.



```
[root@localhost ~]# mkdir -p mydir/mysubdir/threedirdeep
[root@localhost ~]# ls
bench.py  demo  hello.c  mydir  testdir
[root@localhost ~]#
```

rmdir

When a directory is empty, you can use rmdir to remove the directory

rmdir –p

And similar to the mkdir -p option, you can use ***rmdir -p*** to recursively remove directories

Section 3: Working with files

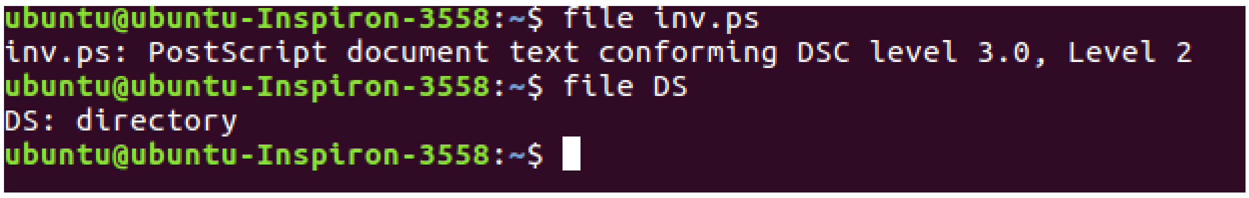
In this section we learn how to recognise, create, remove, copy and move files using commands like ***file, touch, rm, cp, mv*** and ***rename***.

All files are case sensitive

Files on Linux (or any Unix) are case sensitive. This means that FILE1 is different from file1, and /etc/hosts is different from /etc/Hosts (the latter one does not exist on a typical Linux computer).

file

The file utility determines the file type. Linux does not use extensions to determine the file type. The command line does not care whether a file ends in .txt or .pdf. As a system administrator, you should use the file command to determine the file type.

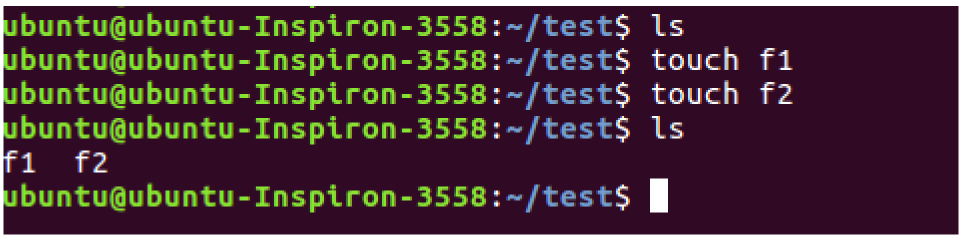


```
ubuntu@ubuntu-Inspiron-3558:~$ file inv.ps
inv.ps: PostScript document text conforming DSC level 3.0, Level 2
ubuntu@ubuntu-Inspiron-3558:~$ file DS
DS: directory
ubuntu@ubuntu-Inspiron-3558:~$ 
```

touch create an empty file

One easy way to create an empty file is with touch. This screenshot starts with an empty directory, creates two files with touch and the lists

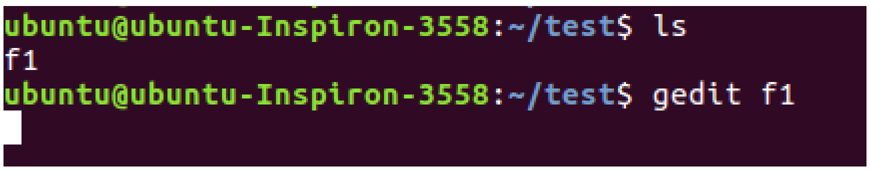
those files.



```
ubuntu@ubuntu-Inspiron-3558:~/test$ ls
ubuntu@ubuntu-Inspiron-3558:~/test$ touch f1
ubuntu@ubuntu-Inspiron-3558:~/test$ touch f2
ubuntu@ubuntu-Inspiron-3558:~/test$ ls
f1  f2
ubuntu@ubuntu-Inspiron-3558:~/test$ 
```

gedit

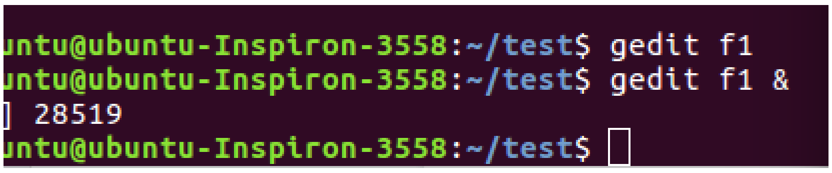
It is the general-purpose text editor. It is easy to use, with a clean and simple GUI. It includes tools for editing source code.



```
ubuntu@ubuntu-Inspiron-3558:~/test$ ls
f1
ubuntu@ubuntu-Inspiron-3558:~/test$ gedit f1
```

gedit can be used to edit an existing file or if the file with name f1 is not present then gedit will create a new file with name f1 and allows us to edit it.

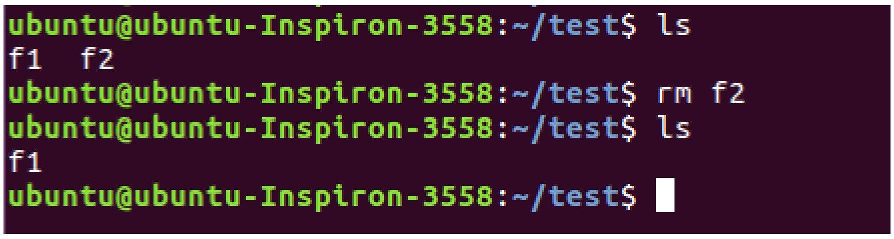
gedit filename & will allow the terminal and gedit window to execute simultaneously.



```
untu@ubuntu-Inspiron-3558:~/test$ gedit f1
untu@ubuntu-Inspiron-3558:~/test$ gedit f1 &
] 28519
untu@ubuntu-Inspiron-3558:~/test$ 
```

rm

used to remove files or directories. When you no longer need a file, use rm to remove it. Unlike some graphical user interfaces, the command line in general does not have a waste bin or trash can to recover files. When you use rm to remove a file, the file is gone. Therefore, be careful when removing files!

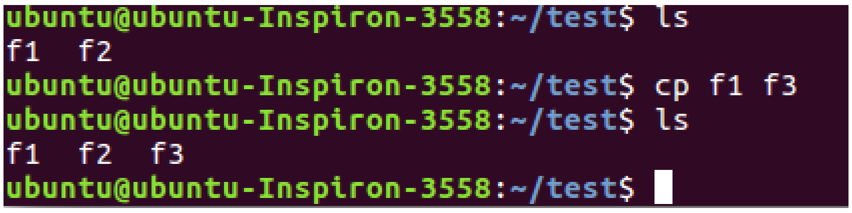


```
ubuntu@ubuntu-Inspiron-3558:~/test$ ls
f1  f2
ubuntu@ubuntu-Inspiron-3558:~/test$ rm f2
ubuntu@ubuntu-Inspiron-3558:~/test$ ls
f1
ubuntu@ubuntu-Inspiron-3558:~/test$ 
```

(Read **man page** of rm and find how it can be used for removing non empty directories)

cp -copy one file.

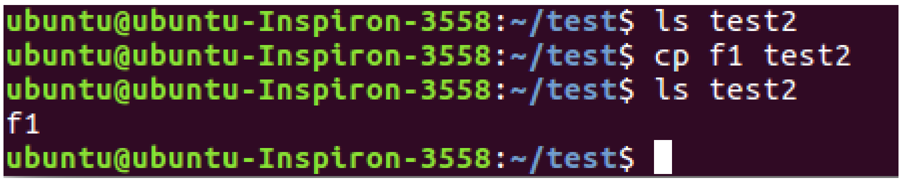
To copy a file, use cp with a source and a target argument.



```
ubuntu@ubuntu-Inspiron-3558:~/test$ ls
f1  f2
ubuntu@ubuntu-Inspiron-3558:~/test$ cp f1 f3
ubuntu@ubuntu-Inspiron-3558:~/test$ ls
f1  f2  f3
ubuntu@ubuntu-Inspiron-3558:~/test$ 
```

Copy to another directory

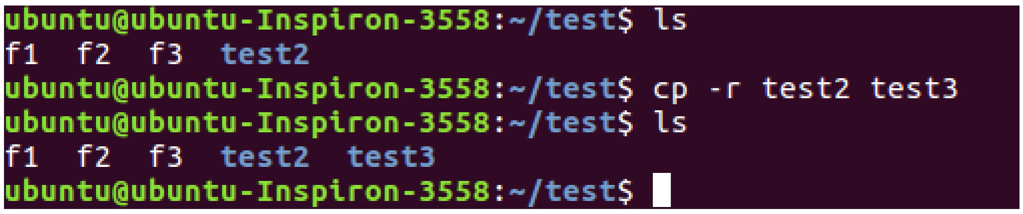
If the target is a directory, then the source files are copied to that target directory.



```
ubuntu@ubuntu-Inspiron-3558:~/test$ ls test2
ubuntu@ubuntu-Inspiron-3558:~/test$ cp f1 test2
ubuntu@ubuntu-Inspiron-3558:~/test$ ls test2
f1
ubuntu@ubuntu-Inspiron-3558:~/test$ 
```

cp -r

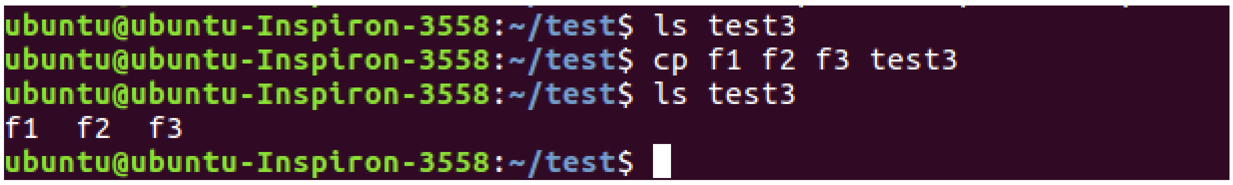
To copy complete directories, use cp -r (the -r option forces recursive copying of all files in all subdirectories).



```
ubuntu@ubuntu-Inspiron-3558:~/test$ ls
f1  f2  f3  test2
ubuntu@ubuntu-Inspiron-3558:~/test$ cp -r test2 test3
ubuntu@ubuntu-Inspiron-3558:~/test$ ls
f1  f2  f3  test2  test3
ubuntu@ubuntu-Inspiron-3558:~/test$
```

copy multiple files to directory

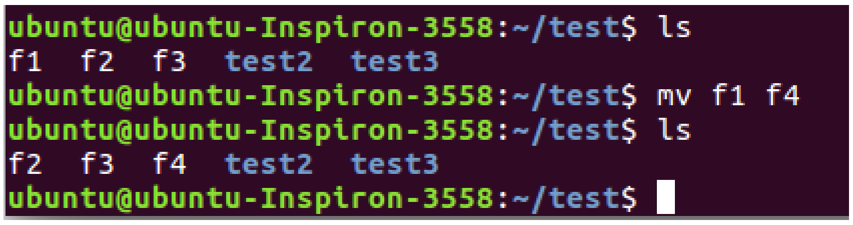
You can also use cp to copy multiple files into a directory. In this case, the last argument must be a directory.



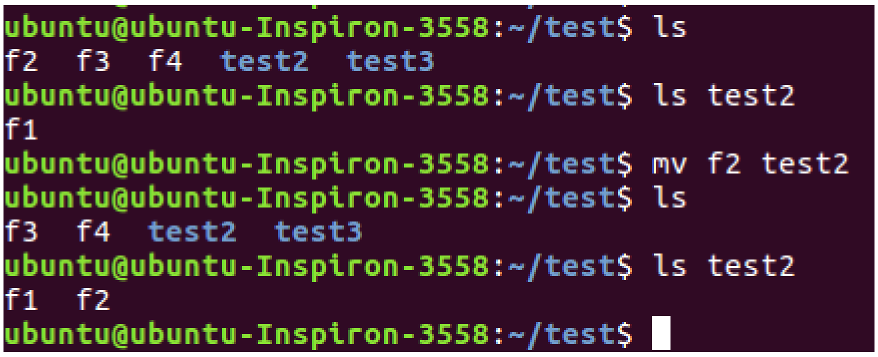
```
ubuntu@ubuntu-Inspiron-3558:~/test$ ls test3
ubuntu@ubuntu-Inspiron-3558:~/test$ cp f1 f2 f3 test3
ubuntu@ubuntu-Inspiron-3558:~/test$ ls test3
f1  f2  f3
ubuntu@ubuntu-Inspiron-3558:~/test$
```

mv

- Rename files with mv - Use mv to rename a file or to move the file to another directory.

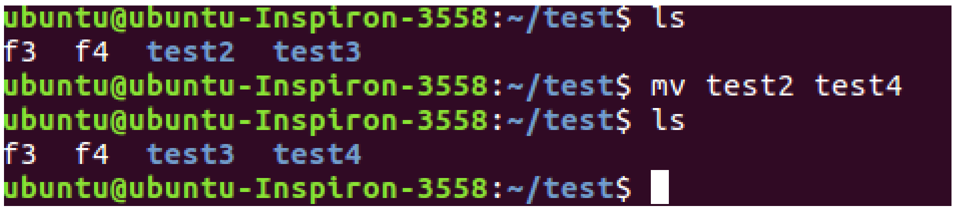


```
ubuntu@ubuntu-Inspiron-3558:~/test$ ls
f1  f2  f3  test2  test3
ubuntu@ubuntu-Inspiron-3558:~/test$ mv f1 f4
ubuntu@ubuntu-Inspiron-3558:~/test$ ls
f2  f3  f4  test2  test3
ubuntu@ubuntu-Inspiron-3558:~/test$
```



```
ubuntu@ubuntu-Inspiron-3558:~/test$ ls
f2  f3  f4  test2  test3
ubuntu@ubuntu-Inspiron-3558:~/test$ ls test2
f1
ubuntu@ubuntu-Inspiron-3558:~/test$ mv f2 test2
ubuntu@ubuntu-Inspiron-3558:~/test$ ls
f3  f4  test2  test3
ubuntu@ubuntu-Inspiron-3558:~/test$ ls test2
f1  f2
ubuntu@ubuntu-Inspiron-3558:~/test$
```

- Rename directories with mv - The same mv command can be used to rename directories



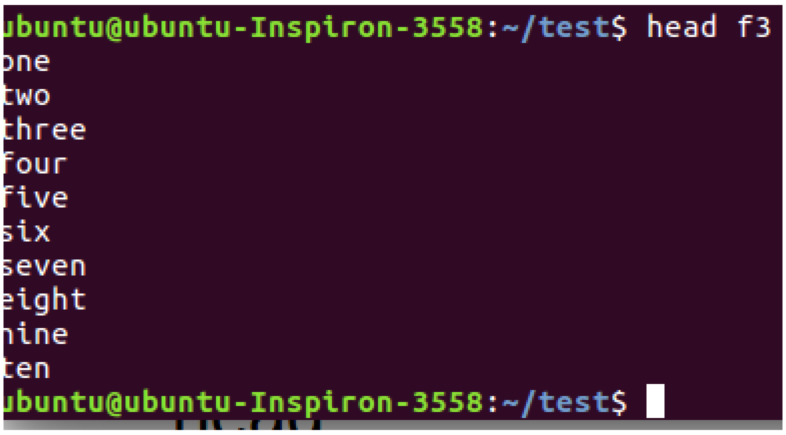
```
ubuntu@ubuntu-Inspiron-3558:~/test$ ls
f3  f4  test2  test3
ubuntu@ubuntu-Inspiron-3558:~/test$ mv test2 test4
ubuntu@ubuntu-Inspiron-3558:~/test$ ls
f3  f4  test3  test4
ubuntu@ubuntu-Inspiron-3558:~/test$
```

Section 4 Working with file contents

In this section we will look at the contents of text files with head, tail, cat, tac, more, less and strings. We will also get a glimpse of the possibilities of tools like cat on the command line.

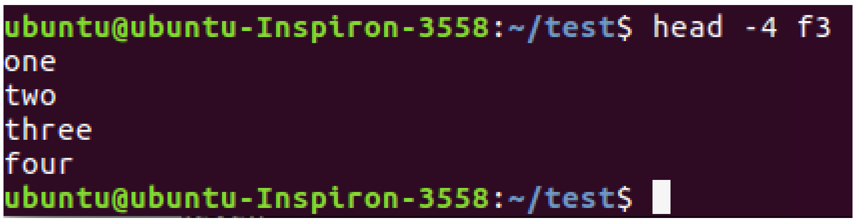
head

You can use head to display the first ten lines of a file.



```
ubuntu@ubuntu-Inspiron-3558:~/test$ head f3
one
two
three
four
five
six
seven
eight
nine
ten
ubuntu@ubuntu-Inspiron-3558:~/test$
```

The head command can also display the first n lines of a file.



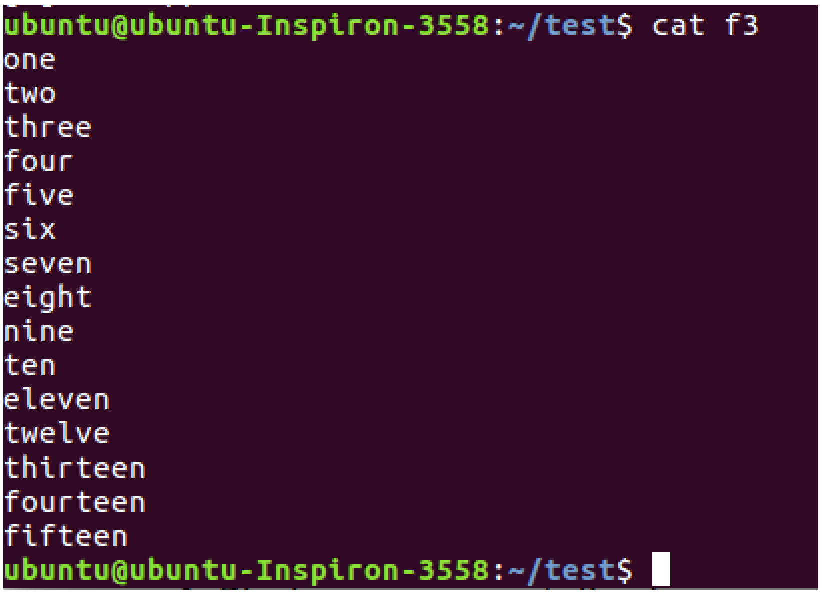
```
ubuntu@ubuntu-Inspiron-3558:~/test$ head -4 f3
one
two
three
four
ubuntu@ubuntu-Inspiron-3558:~/test$
```

tail

Similar to head, the tail command will display the last ten lines of a file.

cat

The cat command is one of the most universal tools, yet all it does is copy standard input to standard output. In combination with the shell this can be very powerful and diverse. Some examples will give a glimpse into the possibilities. The first example is simple, you can use cat to display a file on the screen or terminal. If the file is longer than the screen, it will scroll to the end.



```
ubuntu@ubuntu-Inspiron-3558:~/test$ cat f3
one
two
three
four
five
six
seven
eight
nine
ten
eleven
twelve
thirteen
fourteen
fifteen
ubuntu@ubuntu-Inspiron-3558:~/test$
```

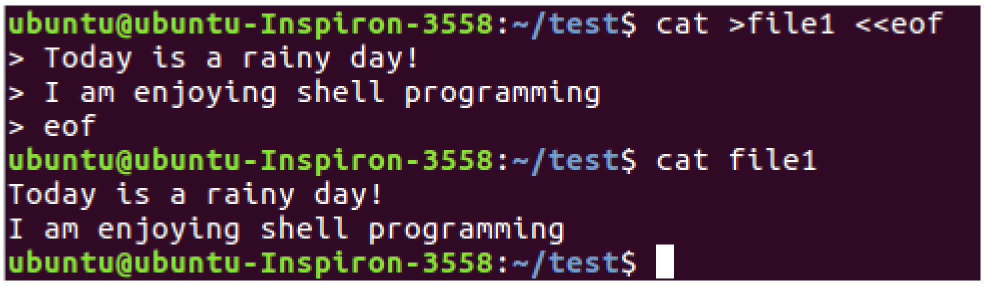
Create files

You can use cat to create flat text files. Type the cat > fn command as shown in the screenshot below. Then type one or more lines, finishing each line with the enter key. After the last line, type and hold the Control (Ctrl) key and press d.

The Ctrl d key combination will send an EOF (End of File) to the running process ending the cat command.

Custom end marker

You can choose an end marker for cat with << as is shown in this screenshot. This construction is called a here directive and will end the cat command.



```
ubuntu@ubuntu-Inspiron-3558:~/test$ cat >file1 <<eof
> Today is a rainy day!
> I am enjoying shell programming
> eof
ubuntu@ubuntu-Inspiron-3558:~/test$ cat file1
Today is a rainy day!
I am enjoying shell programming
ubuntu@ubuntu-Inspiron-3558:~/test$
```

copy files

In the third example, you will see that cat can be used to copy files.

concatenate

cat is short for concatenate. One of the basic uses of cat is to concatenate files into a bigger (or complete) file.

tac

This example will show you the purpose of tac (cat backwards).

Lab Exercise

1.Display the path of your current directory.

2. Make a new directory named ***main***.

3. Now go to the directory ***main***.

4. Make the directories in the following hierarchy using a single command.

    Dir1 -> Dir 2 -> Dir3

5. Print the path of the current directory.

6. Go to ***Dir3*** using a single command.

7. Create a new file ***demo1***, type and save the following contents,

    This is my first file in shell.

    I can edit this file!!!

8. Create a new file ***demo2***, type and save the following contents,

    Hi !!! This is the second file.

    I am doing shell commands.

    I can edit this file!!!

9. Display the contents of file ***demo1*** in terminal.

10. List the files and folders present in ***Dir3***.

11. Go to ***Dir 2***.

12. Go to your home directory.

13. Stay where you are, and list the contents of ***Dir3***.

14. List all the files (including hidden files) in your home directory.

15. Create a new file **test1**, type and save the contents into your file.

I am working with linux shell.

Good bye

16. Copy the contents of **test1** to **test2** in the same directory.
17. Rename **test2** as **test3**.
18. Determine the file type of **test3**.
19. Move the file **test3** to the directory Dir3.
20. Create a file **count**, with content one to twenty in words with one line having only one number using a single command.
21. Copy the file **count** to **count2** using cat command.
22. Create another file **count3** with numbers twenty one to twenty five (in five lines).
23. Concatenate the contents of files **count2** and **count3** and write it into the file **countfinal**.
24. Remove the files **demo1** and **demo2** in directory Dir3.

25.Go to Dir2 and remove the subdirectory Dir3.

26. Come back to your home folder and remove Dir2.
27. Display first 10 lines of the file **countfinal** in terminal.
28. Display last 10 lines of the file **countfinal** in terminal.
29. Display first 5 lines of the file **countfinal** in terminal.
30. Display last 4 lines of the file **countfinal** in terminal.
31. Display the contents of the file **countfinal** in the inverted form.(last line first and first line last)