CHAPTER 1
UNIX FOR NONPROGRAMMERS

The **man** command is used to display the manual entry associated with word entered as argument. The **-k** option is used displays a list of manual entries that contain entered **keyword**

```
man [chapter] word  #displays the manual entry associated with word
man -k keyword  #displays a list of manual entries that contain keyword
```

Note that in a command line all characters that follow up # to a new line are comment is comment

CREATING A FILE

Use editors vi, emacs, pico or the cat command

```
$ cat >myfile # $ is system prompt
Ali
Ahmet
Can
^D
$
```

note: ^D is end of file

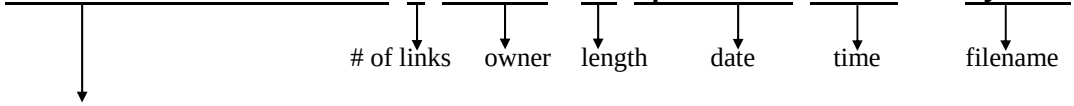
LISTING THE CONTENTS OF A DIRECTOTY : ls

```
ls –adglsR {filename}* {diectory name}*
```

note: * means zero or more, + means one or more

options
a : list also hidden files, i.e. the filenames starting with .
d : directories
g: include info about file group
l: long listing
R: recursively list the contents of subdirectories

```
$ ls
myfile
$ ls –l myfile
- r w - r - - r - -    1 halici  14 April  15 11:41    myfile
```



file type and permissions

-	r w -	r - -	r - -
file type	permissions for owner	permissions for group	permissions for others

LISTING A FILE: cat/more/page/head/tail

cat: concatanate
more, page: to display in parts without scroll
head: first n lines, for default n=10
tail: last n lines, for default n=10

```
$ cat myfile
Ali
Ahmet
Can
$ head -2
Ali
Ahmet
$ tail -2
Ahmet
Can
```

RENAMING A FILE : mv

```
mv –i oldFile newFile # renames oldFile as newFile
mv –i {file name}* directoryName # moves collection files to directory
mv –i oldDirect newDirect # moves files in oldDirect to newDirect
```

Note: -i prompts confirmation if newFileName already exists

```
$ mv  myfile  myNewFile
$ ls
myNewFile
$ cat myNewFile
Ali
Ahmet
Can
$
```

MAKING A DIRECTORY: mkdir

```
mkdir newDirectoryName

$ mkdir class
$ ls –l
-rw-r--r--  1 halici 14  April 15 11:41 myNewFile
drwxr-xr-x  2 halici 512 April 15 11:50 class/
$ mv myNewFile class
$ ls
class
$ ls class
myNewFile

$ ls –R
class
class:
  myNewFile
```

MOVING TO A DIRECTORY: cd, chdir

```
$ pwd  # print working directory
/home122/halici
$ cd class
$ pwd
/ home122/halici/class
```

COPYING A FILE : cp

```
cp –i oldFileName newFileName
cp –ir {file name}* directoryName
```

options:
i: confirm
r: recursively copy subdirectories

```
$ cp myNewFile mySecondFile
$ ls
myNewFile
mySecondFile
```

DELETING A DIRECTORY: rmdir

```
$ pwd
/ home122/halici/class
$ cd ..  # change to parent directory
$ pwd
/ home122/halici
$ ls
```

```
class
$ rmdir class
rmdir: class: directory not empty
```

An error message by the system is displayed. The directory is not deleted since it is not empty.

DELETING A FILE : rm

```
rm –fir {filename}*
```

f: inhibit error messages
i: inform each time
r: recursivey (if filename is a directory)

```
$ ls
class
$ ls class
myNewFile
mySecondFile
$rm class/* #remove all files in directory class
$ls class
$
```

All the files under the directory class are deleted, nothing remains to list by ls

PRINTING A FILE : lpr

```
$ cat >myclass
Ali
Amet
Can
^D
$ ls
myclass
$ cat myclass
Ali
Amet
Can
$ lpr myclass # send the content of the file class to printer
```

COUTING WORDS IN FILE: wc

```
wc -lwc {filename}*
```

options:
l: lines,
w:words,
c: characters

```
$ wc –l myclass
3
$ wc –w myclass
3
$ wc –c myclass
14
$wc myclass  #no option is used, this is equivalent to –lwc all together
3 3 14
↑ ↑  ↑
l w  c
```

FILE TYPES

-	regular file
d	directory file
b	buffered special file (such as disk drive)
c	unbuffered special file (such as disk terminal)
l	symbolic link
p	pipe
s	socket

FILE PERMISSIONS

r w -	r - -	r - -
user	group	others

	regular file	directory	special file
r read	The process may read the contents	The process can read the directory (i.e. list the names of the files that it contains)	The process may read from the file using the read( ) system call
w write	The process may change the contents	The process may add or remove files to/from the directory	The process may write to the file using the write( ) system call
x execute	The process may execute the file (which only makes sense if it is a program)	The process may access files in the directory or any of its subdirectories	No meaning

CHANGING FILE'S PERMISSIONS: chmod

chmod –R change{,change}* filename+

R: recursively change modes if filename is a directory

change :	cluster selection	operation	new permission
	u (user)	+ (add)	r (read)
	g (group)	- (remove)	w (write)
	o (others)	= (assign)	x (execute)
	a (all)		

Examples for change{,change}*

g+w	add group write permission
u-wx	remove user write and execute permissions
o+x	add others execute permission
u+w,g-r	add write permission for user and remove read permission from group
g=r	give group just read permission

```
$ ls –l myclass
-rw-r--r--  1 halici 14  April 15 12:05 myclass
$ chmod  o-r myclass # remove read permission from others
-rw-r-----  1 halici 14  April 15 12:05 myclass
```

The chmod utility allows you to specify the new permission setting of a file as an octal number

	user rwx	group rwx	others rwx
setting	rwx	r-x	---
binary	111	101	000
octal	7	5	0

```
$ chmod 750 myclass
$ ls –l myclass
-rwxr-x---  1 halici 14  April 15 12:05 myclass
```

Permission is set as desired

```
$cat >a
aaa
^D
$ chmod u-w a # remove write permission from user
$ ls –l a #see that it is removed
-r--r--r--  1 halici 4  April 15 12:10 a
$ rm a #delete the file a
$ ls
$
```

The file is removed ! Deleting a file depends on not on the file's write permission but the write permission of the directory that contains it (ie udating the content of the directory)

GROUPS

Suppose that I am a member of the group "**ee**"

```
$ ls –lg myfile # option g stands for listing also file's group
-rw-r--r--  1 halici 14 ee April 15 12:20 myfile
$ groups #list my group
ee
```

If I want to be added to a new group, say named "**cls**", I should request the system administrator to do it.

CHANGING FILE'S GROUP : chgrp

```
chgrp –R groupId {filename}*
```

R: recursively changes the group of the files in a directory

```
$ ls –lg myfile
-rw-r--r--  1 halici 14 ee April 15 12:20 myfile
$ chgrp cls myfile
$ ls –lg myfile
-rw-r--r--  1 halici 14 cls April 15 12:20 myfile
```