***C*hapter 2: Working with Files and Directories**

1. **Creating a directory (mkdir**) Page 15

**2.1 Creating a File** Page 15

Using cat >

Using touch

Using an editor

**2.2 Displaying the contents of files** Page 17

Concatenate: cat filename(s)

Copying files: cp

Recursive Copy cp –R

Rename (Move): mv

**2.3 Exercise** Page 21

Create a directory

Copy a file from my directory

Use of a variable “~”

cat; less; more

Some database terminology

ASCII chart and values (the xxd utility)

Comparing files (diff cmp)

Redirection (> and >>)

cut, paste, join, sort

Inter process communication (pipe |)

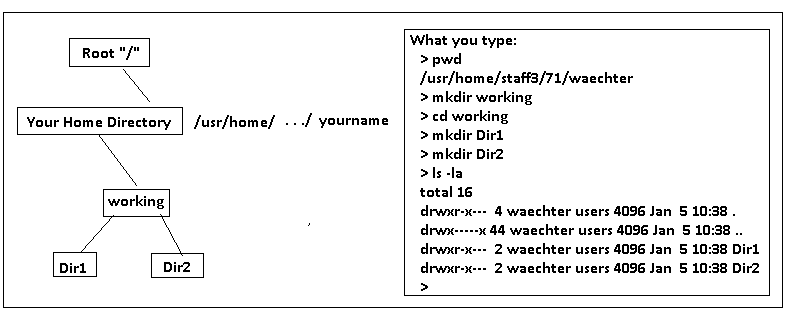
**2.4 Unix/Linux Commands** Page 27

<http://www.ss64.com/>**.**

**2.0 Creating a directory: (mkdir)**

The **mkdir** command creates a directory below your **current working directory**.

In this exercise we will create the following structure from your “HOME” directory. Call the two directories Dir1 and Dir2



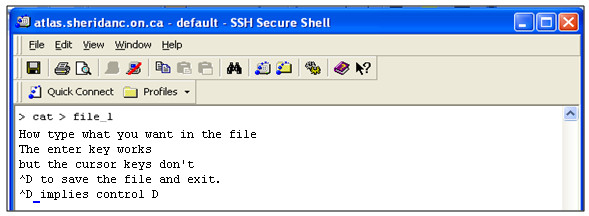
**2.1 Creating a File:** There are several ways to create a file:

**Using cat >**

**Move to your Dir1 directory: >cd Dir1**

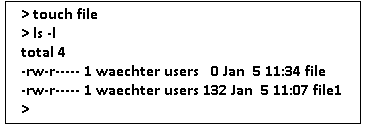
cat (concatenate) is a command that can be used to create a file directly from the keyboard. Simply type:

**cat >file1** and type the contents of the file you wish to create. Note the cursor keys don’t work so it not possible to edit the file using cat but you can create a small file directly from the console.



**Using touch**

The touch command will create an empty file and can also be used to update the time and date stamp of a file.

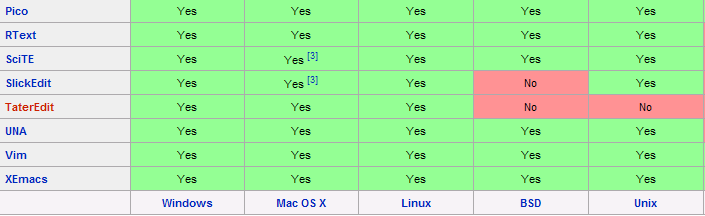


**Using an editor**

The most common way to create a text file is to use an editor. Many editors exist; see

<http://en.wikipedia.org/wiki/Comparison_of_text_editors> for a comparison of editors.

A partial listing is given here:



In this course we will use two common editors found on all unix/linux systems:

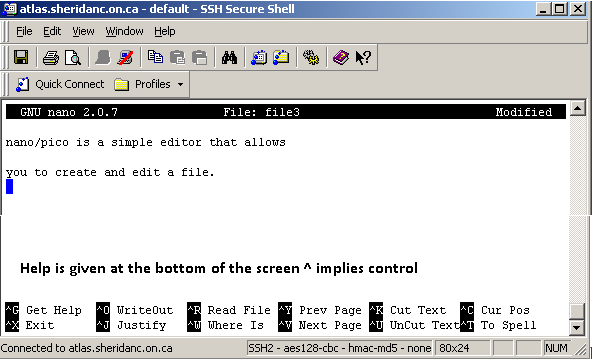
* pico (nano)
* vi (discussed later)

How many editors are compared in the above Wikipedia site? \_\_\_\_\_\_\_\_\_\_\_\_\_\_

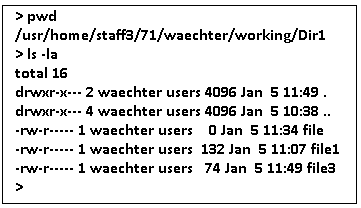
**Using pico (nano)**

Pico and nano are essentially the same editor. To create a file with either editor is intuitive.

**nano file3**  and start to type:



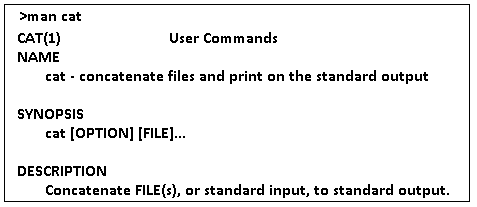
Our Dir1 directory should now appear as follows:



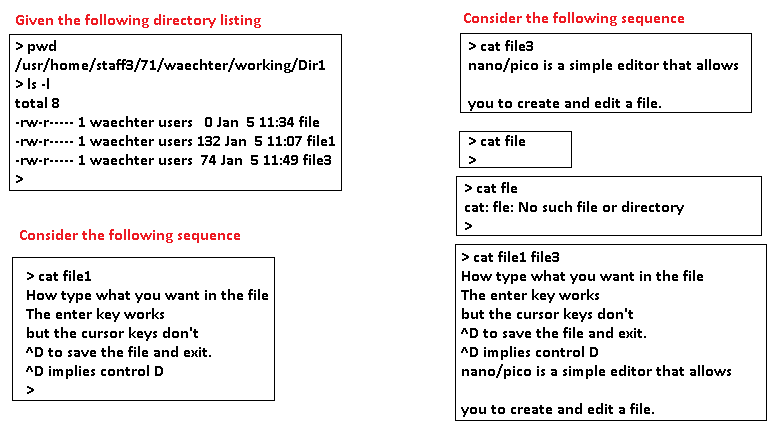
**2.2 Displaying the contents of files:**

There are several commands that allow us to display the contents of files. We first look at a command “cat” and later we look at other “pagers:

**Concatenate: cat filename(s)**

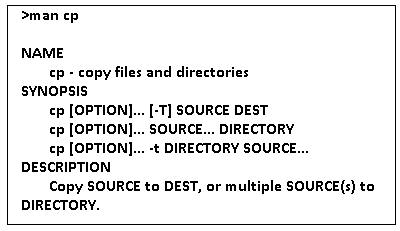
****

Some examples:

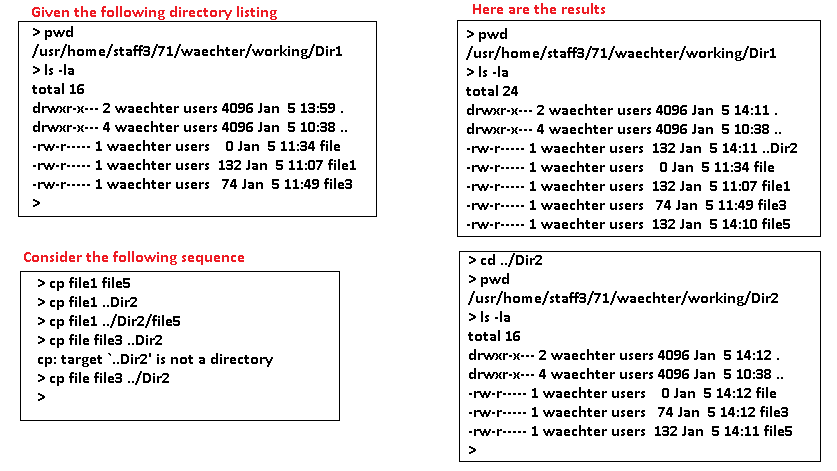


**Copying files:**

The command cp copies files. The general syntax is cp source destination.



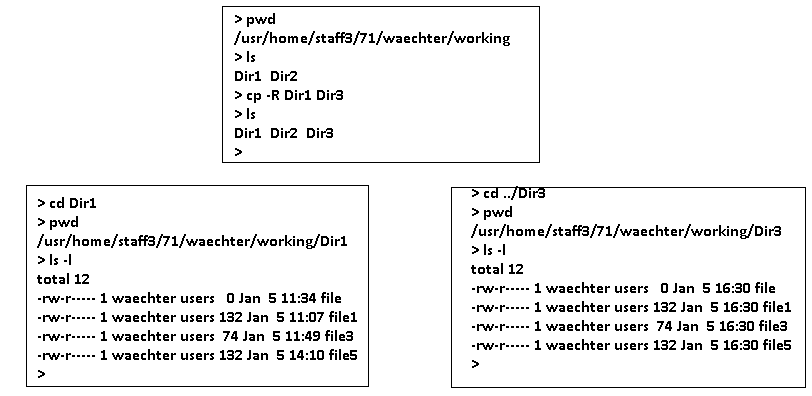
Some examples:



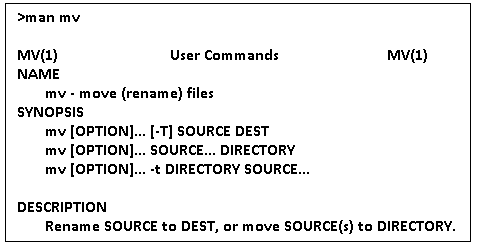
**Recursive Copy**

It is possible to recursively copy directories and all files and subdirectories. For example if we wanted to make a copy of our Dir1 directory we could execute the:

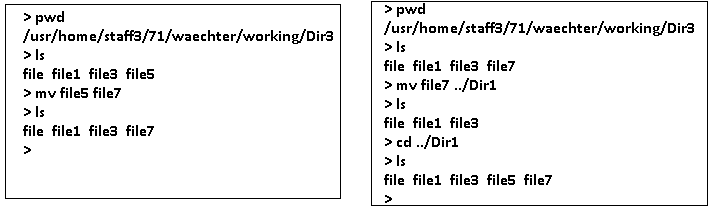
cp –R Dir1 Dir3 Assuming the CWD is the **working** directory:



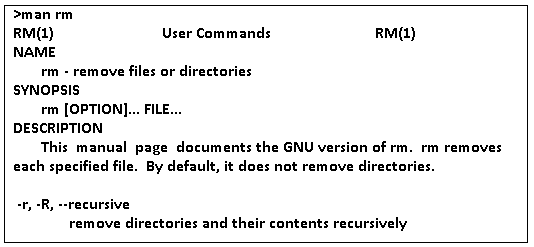
**Rename (Move):** The mv (move command) is used to rename a file or the move a file from one directory to another directory.



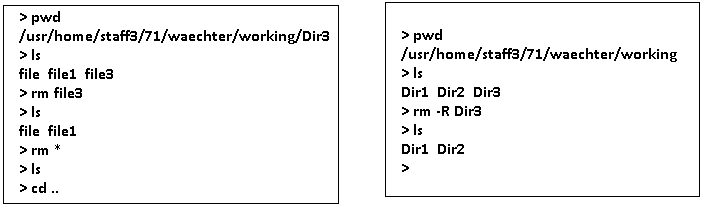
We will work from Dir3



**Deleting Files and Directories**: The **rm command** is used to remove files and directories.



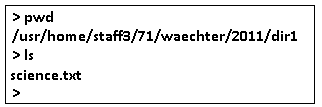
We will now work from the Dir2 directory.



**2.3 Exercise:**

In this exercise we need several files.

1. Create a directory “dir1” under your home directory. I created one under 2011 under my home directory**. /usr/home/staff3/71/waechter/2011/dir1**

****

1. Copy the science.txt file from my directory to your dir1 directory.

* First move to my directory.
* cd /usr/home/staff3/71/waechter/2011/dir1
* copy the science.txt file to your subdirectory.
* The simplest way to do this is:

cp science.txt ~/dir1

**Note: the use of the “~” key**

**>echo ~** More on the echo command later

1. What does the “~” key represent? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
2. Displaying the contents of a file. There are several ways to display the contents of text files. Several commands are given here:

**>cat science.txt >less science.txt >more science.txt**

Given the results of the following:

>cat science.txt | wc –l \_\_\_\_\_\_\_\_\_\_\_\_ >less science.txt|wc –l \_\_\_\_\_\_\_\_\_\_\_\_\_\_

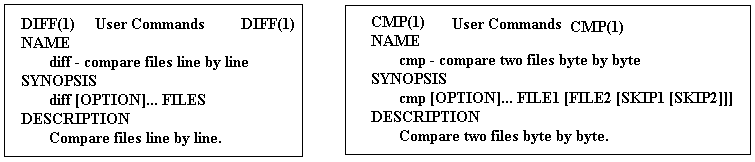
>more science.txt|wc –l \_\_\_\_\_\_\_\_\_\_\_ >head science.txt|wc –l \_\_\_\_\_\_\_\_\_\_\_\_\_

>head -5 science.txt| wc –l \_\_\_\_\_\_\_\_\_ >tail science.txt|wc –l \_\_\_\_\_\_\_\_\_\_\_\_\_\_

>tail -5 science.txt | wc –l \_\_\_\_\_\_\_\_\_\_

**Note: >less science.txt and >cat science.txt|less give the same output.**

1. Comparing files: There are two unix utilities that compare files: cmp and diff



How many **lines** of output are produced by executing each of the following commands?

>diff names names\_space \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

>diff names\_space names\_tab \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

>diff names names \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

>cmp names names\_space \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

>cmp names\_space names\_tab \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

>cmp names names \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Redirection: The use of **“> and >>”**. Perform the following tasks and answer the questions below.

>cat names >ex1 The > key redirects the output to a file.

>cat names >>ex1 the >> keys are used to append to an existing file.

How many lines are now in the ex1 file? \_\_\_\_\_\_\_\_\_\_\_ use wc –l

> cat names names\_tab >> ex1 How many lines are now in the ex1 file?

\_\_\_\_\_\_\_\_\_\_\_ use wc –l

1. **The cut command:** Give the output for each of the following: Give at least one record for each.

The cut command takes a vertical slice of a file, printing only the specified columns or fields.

Content of the company.data file:

406378:Sales:Itorre:Jan

031762:Marketing:Nasium:Jim

636496:Research:Ancholie:Mel

396082:Sales:Jucacion:Ed

If you want to print just columns 1 to 6 of each line (the employee serial numbers), use the -c1-6 flag, as in this command:

cut -c1-6 company.data

406378

031762

636496

396082

And since this file obviously has fields delimited by colons, we can pick out just the last names by specifying the -d: and -f 3 flags, like this:

cut -d: -f 3 company.data

Itorre

Nasium

Ancholie

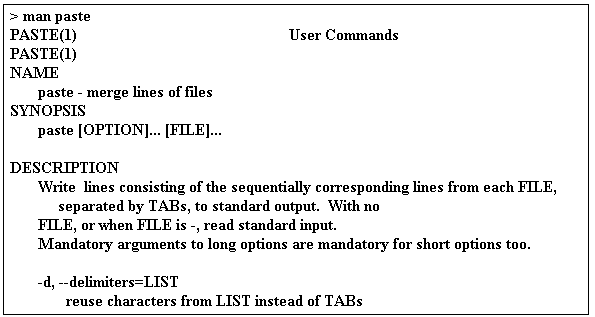
Jucacion

>cut -b1,4 names >cut -b1,4 names >cut -b 9-15 names\_space

>cut –b 9-15 names\_tab >cut –b 19-16 names >cut –d: -f2 names

>cut –f1 names\_space >cut –f1 names\_tab >cut –f1 names

What is the default field delimiter when using cut? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **The paste command: paste - merge lines of files: >ex2**

**file1:**

**Jack Wallen**

**Jessica Wallen**

**Johnny Wallen**

**Jeri Wallen**

**The contents of file2 will look like:**

**123-45-6789**

**234-56-7890**

**345-67-8901**

**456-78-9012**

**by running the command paste file1 file2 > file3 and then viewing the file, the contents of file3 will look like Table A.**

**Table A**

**Jack Wallen 123-45-6789**

**Jessica Wallen 234-56-7890**

**Johnny Wallen 345-67-8901**

**Jeri Wallen 456-78-9012**

1. Give the output for each of the following: Give at least one record for each.

>paste names names\_space > ex2

>cat ex2 >cut –f1 ex2 >cut –f2 ex2

>paste –d: names names\_space > ex3

>cat ex3 >cut –f1 ex3 >cut –f2 ex3

1. **The join command: (join lines of two files on a common field)**

**The join command is like the paste command—only a bit more intelligent. The join command takes two files and merges their columns—as long as both files share a common field.**

**% join file1 file2 > file3**

**If the content of file1 is:**

**1 Barbara**

**2 Peter**

**3 Stan**

**4 Marie**

**and file2 is:**

**2 Dog**

**4 Car**

**7 Cat**

**the resulting file3 would be:**

**2 Peter Dog**

**4 Marie Car**

Execute this command: **>join -t: names products > ex4**

How many records in the ex4 file? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **The sort command: The sort command, as its name implies, will sort data according to the users’ needs.**

**The sort command does have a number of useful switches. The sort switches include:**

**-d Sorts only blanks and alphanumeric characters**

**-f Ignores case**

**-I Ignores nonprinting characters**

**-M Sorts by month**

**-n Sorts numerically**

**-r Reverses the sort order**

**-k Starts at a user-defined position and ends at a user-defined position**

**ls -al | sort -n -k 5**

**This results in the following ls command sorted output, which as you can see, is a directory listing, sorted by file size (the 5th column):**



Give the output of each of the following:

**>**sort –k 2 names\_tab >sort –k 2 –t: names

1. **Using inter process communication: “pipe |”**

The output from one process and be used as input to another process. We have already seen this with the **ls –l|wc –l command.**

Now consider:

>cut –d: -f 2 names | sort

>cut -d: -f 2 names | sort | wc -l

**2.4 Unix/Linux Commands**

There are more unix commands than we could possibly cover in this course. A nice site that gives the bash commands is: <http://www.ss64.com/>**.** In this course we will look at the commands necessary for a user to be productive in a Unix/Linux environment and some of the basic administrative commands.