RedHat/Fedora command cheat sheet:

Packing, unpacking, and installing files:

gzip compress:
#*gzip* <filename.ext>

gzip extract:
#*gunzip* <filename.ext>

gzip retrieve information about file:
#*gzip -l* <filename.ext>

tar archive:
#*tar cfv* <archive name.ext> <file/folder to archive>

tar extract:
#*tar xvf* <filename.ext>

tar retrieve information about file:
#*tar tvf* <filename.tar>

gzip and tar compress in one command:
#*tar cvzf* <archive name.tar.gz> <file/folder to archive>

gzip and tar extract in one command:
#*tar xvzf* <filename.tar.gz>

install rpm files:
#*rpm -ivh* <rpm file name>

install rpm package even if already installed:
#*rpm -ivh --replacepkgs* <rpm file name>

uninstall rpm packages:
#*rpm -e* <rpm package name>

After uninstalling an rpm package, you will find that it still has config files and other bits strewn about. The *rm -rf* and *rm -f* commands are **VERY** powerful and can easily **render your linux installation unusable**. Be **very careful**. A 'reasonably safe' way (provided you pay attention and are careful) to clean the files up is:

1) Do a *find / -name "<filename>*" | more*
2) Check the output and make SURE it only includes the files you want to remove.
3) Do a *find / -name "<filename>*" | xargs rm -f*

---

tar basic functions and options:

function:
c To create a new archive
x To extract files from an archive
t To list the contents of an archive
r To append files to the end of an archive
u To update files that are newer than those in the archive
d To compare files in the archive to those in the filesystem

options:
f <filename> To specify that the tar file to be read or written is named <filename>
k To keep any existing files when extracting, i.e. don't delete the origional files
v To make tar show the files it is archiving or restoring (don't use in shell scripts)
z To specify that the data to be written to the tar file should be gzipped

---

Miscellaneous commands:

To download a web page(s):
#wget -m -r -l5

Checking open network ports:
#netstat -apn | more

Show file attributes and permissions:
#ll

Show all files in a directory (including hidden .<name> files
#*ls -A*

Show information about mounted volumes:
#df -h

Turn off all power management:
#xset -dpms

Load StartX setup routine:
#X86config

Display log file starting at the end:
#tail -f <file name>

Samba commands:
Add user/change password (user must have a unix account first):
#smbpasswd -a <username>

Apache commands:
.htpasswd file creation for Apache Directory security use:

To create a new .htpasswd file and add a user (will prompt for password):
#htpasswd -c /etc/httpd/conf/.htpasswd <name>
To create a new user in an existing .htpasswd file (will prompt for password):
#htpasswd /etc/httpd/conf/.htpasswd <name>

<Directory> security examples in httpd.conf (which use the .htpasswd file):

<Directory "/var/www/html/<directory>">
AuthType Basic
AuthName "Restricted Uploads"
AuthUserFile /etc/httpd/conf/.htpasswd
Require valid-user (means anyone in the .htpasswd file can access)
</Directory>

and

<Directory "/var/www/html/<directory>">
AuthType Basic
AuthName "Restricted file access"
AuthUserFile /etc/httpd/conf/.htpasswd
Require jjones (only jjones in the .htpasswd file has access)
</Directory>

RedHat/Fedora account creation:
To create a new user account:
#useradd <name>

To add/change a password:
#passwd <name> (will prompt for password twice)

To add a user to a group:
#usermod -G <groupname> <username>

Set owner of a file/folder:
#chown <user**.**group> <file/folder name>
#chown **-R** >user**.**group> <file/folder name> for recursive, i.e. apply changes to subfolders

Set permissions on a file/folder:
#*chmod* 777 <file/folder name> for full rights (dangerous!)
#*chmod* 775 <file/folder name> for full rights for user/group but no write/execute for 'other'
#*chmod* 765 <file/folder name> for full user rights, no write for 'group', and no write/execute for 'other'
Add **-R** for recursive, i.e. *chmod* **-R** 775 <file/folder name> to apply rights to subfolders

File permissions take the form of:

User	Group	Other
-RWX	RWX	RWX

The leading dash in the above table is for the type of data, **d** would be a directory and **-** indicates a file.

File permissions can be set using bits, as referenced above:

User			Group			Other		
read	write	execute	read	write	execute	read	write	execute
400	200	100	40	20	10	4	2	1

Another way to look at it would be to visualize -r--r--r-- and calculate it as:

400
40
4
= 444

~examples~
File permissions of -rwxrwxr-x would be:
400+200+100 plus 40+20+10 plus 4+1 (no write for Other) which equals 775
File permissions of -rwxr-xr-x would be:
400+200+100 plus 40+10 plus 4+1 (no write for Group or Other) which equals 755

So, *chmod -R 775 /var/www/html* means /html and subfolders have -rwxrwxr-x or full rights except 'Other', which doesn't have write permissions.

Making symbolic links:
#*ln -s* <location/filename> <name of symbolic link>

Example: *ln -s /var/www/html/homesite* homesite would create a link named homesite to /var/www/html/homesite which is a folder.

Sending Root system messages and logs to an email address:
Edit /etc/aliases
Un-remark (remove # symbol) from 'root:' and add the email address of the recipient.
Save file.
At the command prompt, type *newaliases* to update the database.

Enable daily yum updates:
Pre-FC6 setup:
chkconfig yum on
service yum start
Should see "Enabling nightly yum update: [OK]"

FC6 and later setup (yum-cron is a seperate package):
yum install yum-cron
chkconfig yum-cron on
service yum-cron start
Should see "Enabling nightly yum update: [OK]"

Other yum features:
List all available software:
#yum list

See if there are updated packages available:
#yum check-update

Update all installed packages that have a newer version available:
#yum update

Install specific package(s) (and its dependencies, if missing any):
#yum install <packagename>

Search all known packages entries (descriptions etc) for *<word>*
#yum search <word>

Show basic information about a package
#yum info *<packagename>*