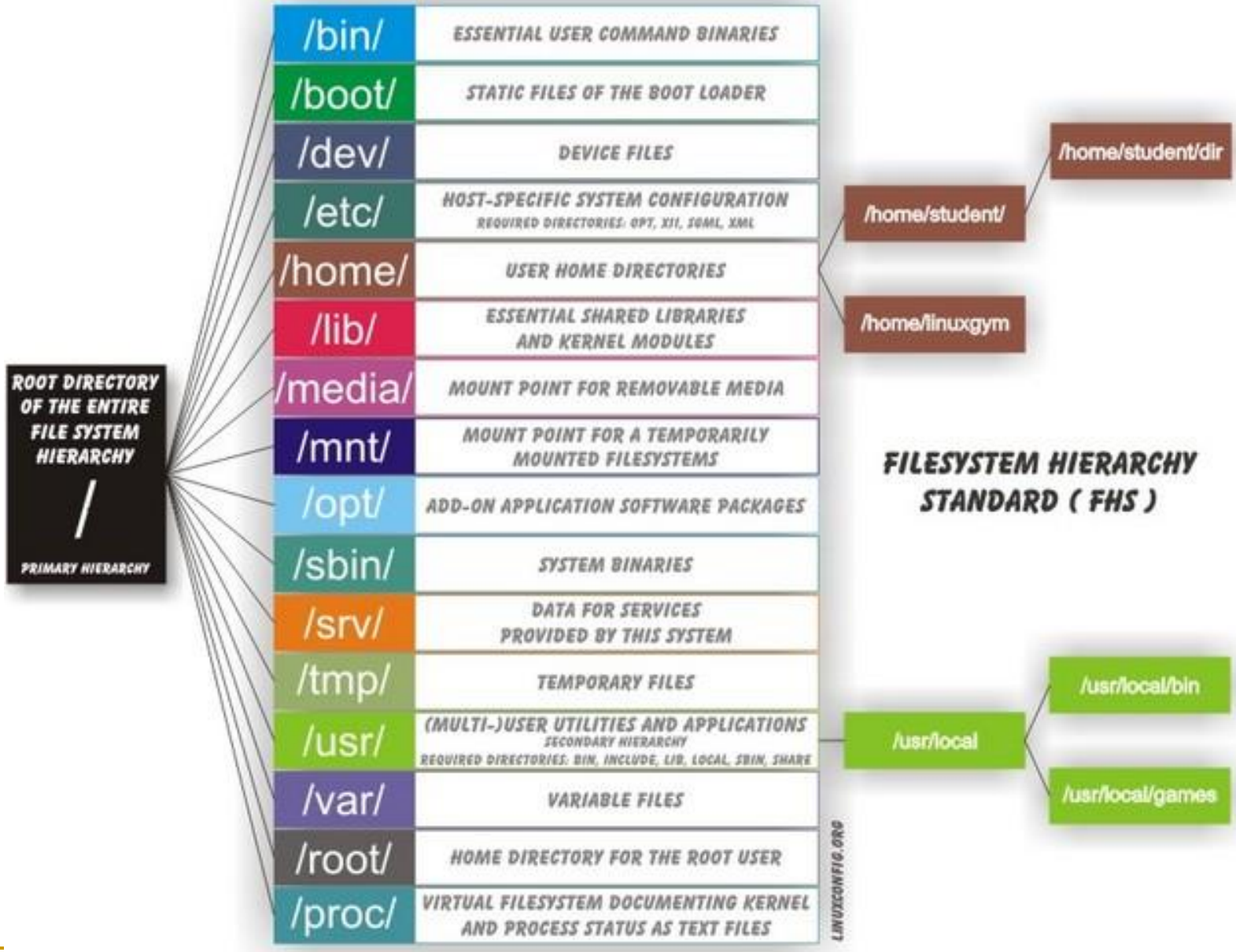
Linux file system ,Basic and Advance commands

Understanding the Linux File System

- The file system is organized in a hierarchical structure similar to an inverted tree
- Linux file structure files are grouped according to purpose. Ex: commands, data files, documentation.

Understanding the Standard Tree Structure

- The structure starts at the root level
  - Root is the name of the file at this basic level and it is denoted by the slash character (/)
- A directory is a file that can contain other files and directories
- A subdirectory is a directory within a directory
  - The subdirectory is considered the child of the parent directory


ROOT DIRECTORY OF THE ENTIRE FILE SYSTEM HIERARCHY
/
PRIMARY HIERARCHY
/bin/
ESSENTIAL USER COMMAND BINARIES
/boot/
STATIC FILES OF THE BOOT LOADER
/dev/
DEVICE FILES
/etc/
HOST-SPECIFIC SYSTEM CONFIGURATION
REQUIRED DIRECTORIES: OPT, X11, SGML, XML
/home/
USER HOME DIRECTORIES
/lib/
ESSENTIAL SHARED LIBRARIES AND KERNEL MODULES
/media/
MOUNT POINT FOR REMOVABLE MEDIA
/mnt/
MOUNT POINT FOR A TEMPORARILY MOUNTED FILESYSTEMS
/opt/
ADD-ON APPLICATION SOFTWARE PACKAGES
/sbin/
SYSTEM BINARIES
/srv/
DATA FOR SERVICES PROVIDED BY THIS SYSTEM
/tmp/
TEMPORARY FILES
/usr/
(MULTI-)USER UTILITIES AND APPLICATIONS
SECONDARY HIERARCHY
REQUIRED DIRECTORIES: BIN, INCLUDE, LIB, LOCAL, SBIN, SHARE
/var/
VARIABLE FILES
/root/
HOME DIRECTORY FOR THE ROOT USER
/proc/
VIRTUAL FILESYSTEM DOCUMENTING KERNEL AND PROCESS STATUS AS TEXT FILES
LINUXCONFIG.ORG
/home/student/
/home/student/dir
/home/linuxgym
/usr/local
/usr/local/bin
/usr/local/games
FILESYSTEM HIERARCHY STANDARD (FHS)

File System

1. / – Root
   - Every single file and directory starts from the root directory.
   - Only root user has write privilege under this directory.
   - Please note that /root is root user's home directory
2. /bin – User Binaries
   - Contains binary executables.
   - Common linux commands you need to use in single-user modes are located under this directory.
   - Commands used by all the users of the system are located here.
   - For example: ps, ls, ping, grep, cp.
3. /sbin – System Binaries
   - Just like /bin, /sbin also contains binary executables.
   - But, the linux commands located under this directory are used typically by system aministrator, for system maintenance purpose.
   - For example: iptables, reboot, fdisk, ifconfig, swapon

4. /etc – Configuration Files
   - Contains configuration files required by all programs.
   - This also contains startup and shutdown shell scripts used to start/stop individual programs.
   - For example: /etc/resolv.conf, /etc/logrotate.conf
5. /dev – Device Files
   - Contains device files.
   - These include terminal devices, usb, or any device attached to the system.
   - For example: /dev/tty1, /dev/usbmon0
6. /proc – Process Information
   - Contains information about system processes.
   - It contains runtime system info like: system memory, devices mounted, hardware configuration, etc.

7. /var – Variable Files

- var stands for variable files.
- Content of the files that are expected to grow can be found under this directory.
- This includes — system log files (/var/log); packages and database files (/var/lib); emails (/var/mail); print queues (/var/spool); lock files (/var/lock); temp files needed across reboots (/var/tmp);

8. /tmp – Temporary Files

- Directory that contains temporary files created by system and users.
- Files under this directory are deleted when system is rebooted.

9. /usr – all programs are installed here.

- /usr/local -The place for locally installed software and other files.

10. /home – Home Directories

- Home directories for all users to store their personal files.
- For example: /home/john, /home/nikita

11. /boot – Boot Loader Files
    - Contains boot loader related files.
    - Kernel initrd, vmlinuz, grub files are located under /boot
    - For example: initrd.img-2.6.32-24-generic, vmlinuz-2.6.32-24-generic
12. /lib – System Libraries
    - Contains library files that supports the binaries located under /bin and /sbin
    - Library filenames are either ld* or lib*.so.*
    - For example: ld-2.11.1.so, libncurses.so.5.7
13. /opt – Optional add-on Applications
    - opt stands for optional.
    - Contains add-on applications from individual vendors.
    - add-on applications should be installed under either /opt/ or /opt/ sub-directory.
    - Google chrome need to be installed separately.

14. /mnt – Mount Directory

- Temporary mount directory where sysadmins can mount filesystems.

15. /media – Removable Media Devices

- Temporary mount directory for removable devices.
- For examples, /media/cdrom for CD-ROM; /media/floppy for floppy drives; /media/cdrecorder for CD writer

16. /srv – Service Data

- Srv is a serve folder.
- It holds site specific data to be served by the system for protocols such as, ftp, rsync, www, cvs etc.

Logging

- You are now logged into the computer and will have a prompt that reflects the computer's name.
sammy@webapp:~$

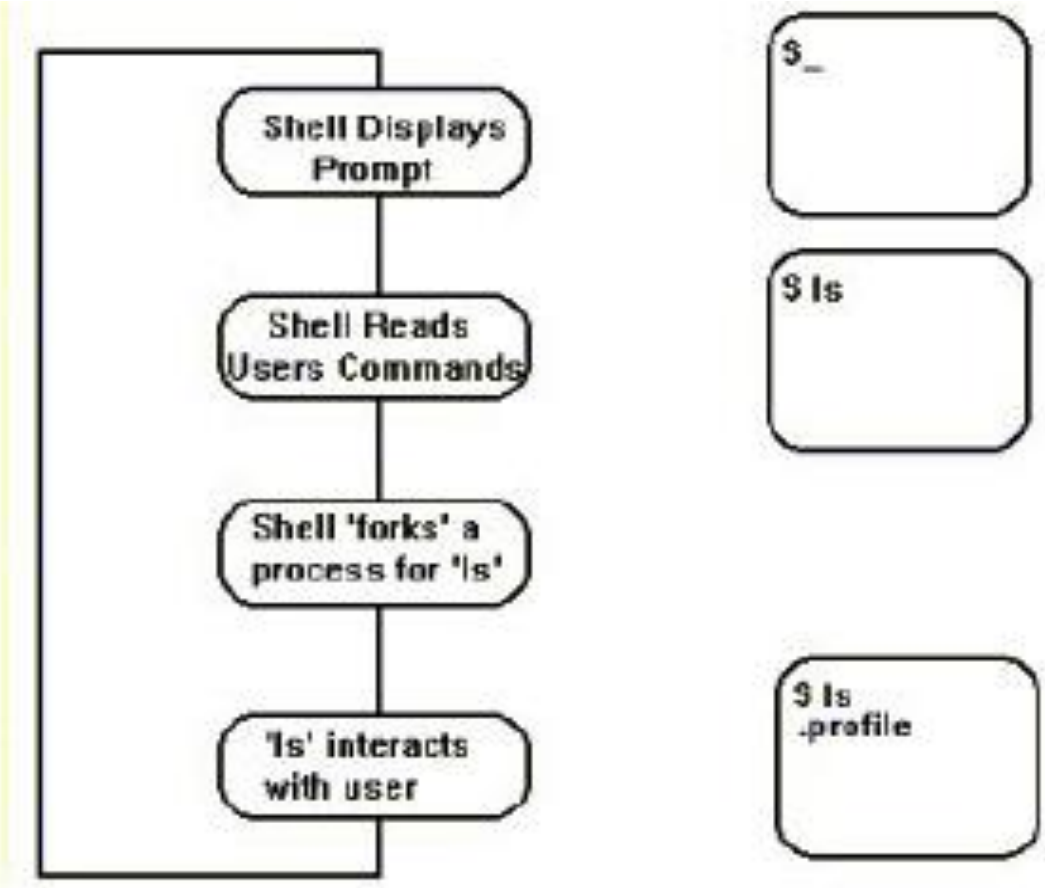
  - ✓ sammy: The *username* of the current user
  - ✓ webapp: The *hostname* of the server
  - ✓ ~: The *current directory*. In bash, which is the default shell, the ~, or tilde, is a special character that expands to the path of the current user's *home directory*; in this case, it represents /home/sammy
  - ✓ $: The prompt symbol. This denotes the end of the command prompt, after which the user's keyboard input will appear

- At this point, you're ready to enter your first linux command
- exit—to sign off from the computer system. entering exit shuts down all programs

- Every unix command is a sequence of **letters, numbers** and **characters.** But there are no spaces.
- It is also case-sensitive. This means that *cat* and *Cat* are different commands.
- The prompt is displayed by a special program called the **shell**.
- **Shells accept** commands, and **run** those commands.
- They can also be programmed in their own language. These programs are called "**shell scripts**".

- When you first login, the prompt is displayed by bash, and you are running your first unix program, the bash shell.
- As long as you are logged in, the *bash shell* will constantly be running.

When a user logs on



Unix Commands

- Each command performs [variations of] a single task
  - "options" can be used to modify what a command does
  - different commands can be "glued together" to perform more complex tasks
- Syntax:

  command options arguments

  Examples:

Options can (usually) be combined together: these are equivalent

Command	Options	Arguments
pwd		
cd		/home/debray
ls	-a -l	
ls	-al	/usr/local

Unix Commands

- Each command performs [variations of] a single task
  - "options" can be used to modify what a command does
  - different commands can be "glued together" to perform more complex tasks
- Syntax:

  command options arguments

Not always required: may have default values

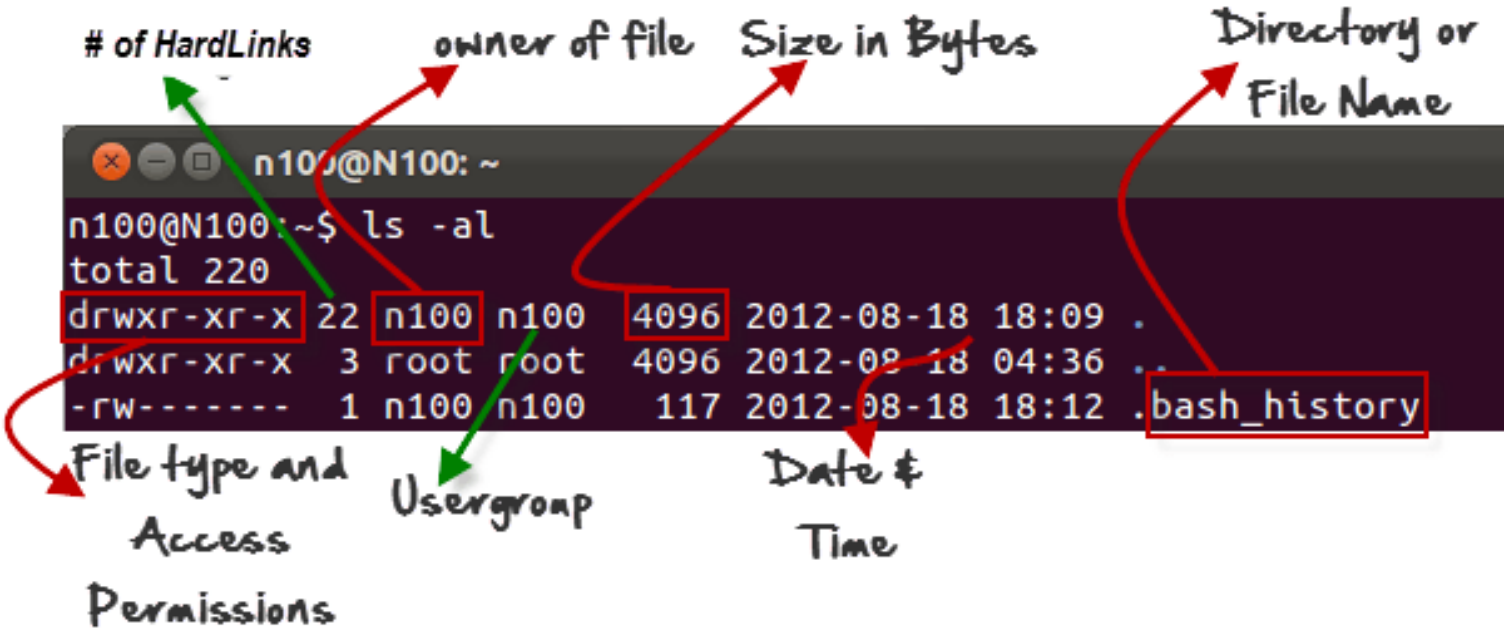
typical defaults:
- input: stdin
- output: stdout
- directory: current

Examples:

Command	Options	Argument
pwd		
cd		/home/debray
ls	-a -l	
ls	-al	/usr/local

defaults to current directory

'ls -al' gives detailed information of the files.



Help for commands

Figuring out which command to use

apropos *keyword*

man -k *keyword*

"searches a set of database files containing short descriptions of system commands for keywords"

Finding out about commands II

Figuring out how to use a command

man *command*

"displays the on-line manual pages"

- Provides information about command options, arguments, return values, bugs, etc.

Example: “man ls”

```
hed: /home/debray
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.

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

       -a, --all
              do not ignore entries starting with .

       -A, --almost-all
              do not list implied . and ..

       --author
              with -l, print the author of each file

       -b, --escape
              print octal escapes for nongraphic characters

       --block-size=SIZE
              use SIZE-byte blocks

       -B, --ignore-backups
              do not list implied entries ending with ~

       -c     with -lt: sort by, and show, ctime (time of last modification of file status information) with -l:  show
              ctime and sort by name otherwise: sort by ctime

       -C     list entries by columns

       --color[=WHEN]
              control whether color is used to distinguish file types.  WHEN may be `never', `always', or `auto'
:
```

items within square brackets are optional

To exit
Press “q”

Example: “man man”

xterm

```
man(1)                                                          man(1)

NAME
       man - format and display the on-line manual pages

SYNOPSIS
       man [-acdfFhkKtwW] [--path] [-m system] [-p string] [-C config_file] [-M pathlist] [-P pager] [-B browser] [-H htmlpager]
       [-S section_list] [section] name ...

DESCRIPTION
       man formats and displays the on-line manual pages.  If you specify section, man only looks in that section of the manual.
       name  is  normally the name of the manual page, which is typically the name of a command, function, or file.  However, if
       name contains a slash (/) then man interprets it as a file specification, so that you can do  man  ./foo.5  or  even  man
       /cd/foo/bar.1.gz.

       See below for a description of where man looks for the manual page files.

MANUAL SECTIONS
       The standard sections of the manual include:

       1      User Commands

       2      System Calls

       3      C Library Functions

       4      Devices and Special Files

       5      File Formats and Conventions

       6      Games et. Al.

       7      Miscellanea

       8      System Administration tools and Deamons

       Distributions customize the manual section to their specifics, which often include additional sections.

OPTIONS
       -C  config_file
              Specify the configuration file to use; the default is /etc/man.config.  (See man.config(5).)
:
```

Logging In

- To log in to a Unix machine you can either:
  - sit at the console (the computer itself)
  - access remotely, via SSH, e.g.
- The system prompts you for your username and password.
- Usernames and passwords are case sensitive!

Password

- a password is a secret string that only the user knows (not even the system knows!)
- When you enter your password the system encrypts it and compares to a stored string.
- passwords should have at least 6 characters
- It's a good idea to mix case, include numbers and/or special characters (don't use anything that appears in a dictionary!)

Home Directory

- The user's personal directory. E.g.,
  - /home/student
  - /home/vit
- Your *current directory* when you log in
- `cd` (by itself) takes you home
- Location of many startup and customization files. E.g.:
  - .vimrc .bashrc .bash_profile .forward .plan .mozilla/ .elm/ .logout

Files and File Names

- A file is a basic unit of storage (usually storage on a disk).
- Every file has a name.
- Filenames are case-sensitive!
- File names can contain any characters (although some make it difficult to access the file) except the null character and the slash (/).

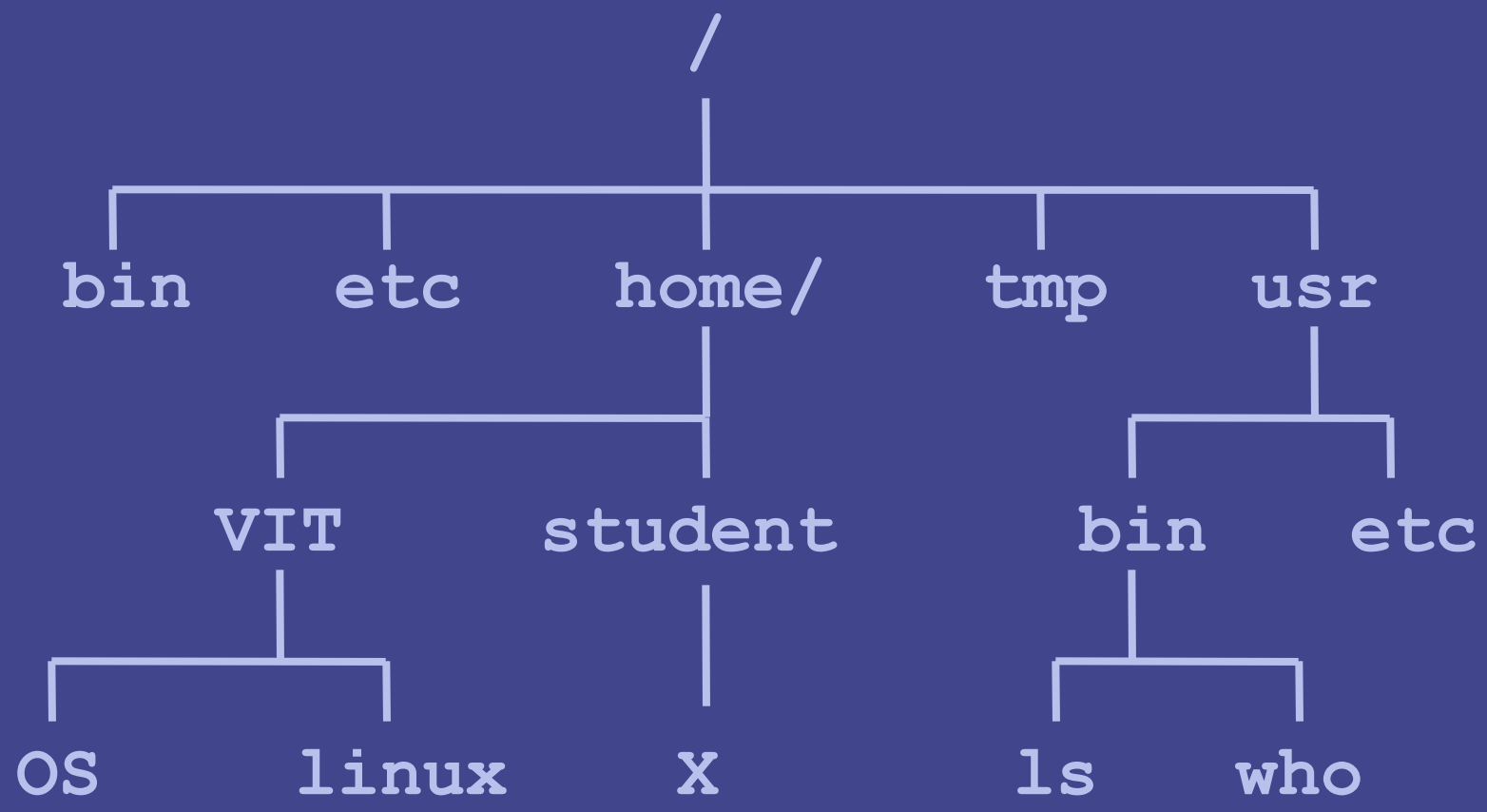
Directories

- A directory is a special kind of file - uses a directory to hold information about other files.
- We often think of a directory as a container that holds other files (or directories).
- A directory is the same idea as a *folder* on Windows.

More about File Names

- Review: every file has a name (at least one).
- Each file *in the same directory* must have a unique name.
- Files that are in different directories can have the same name.

The Filesystem (eg)



Pathnames

- The *pathname* of a file includes the file name and the name of the directory that holds the file, and the name of the directory that holds the directory that holds the file, and the name of the … up to the root
- The pathname of every file in a given *filesystem* is unique.
- To create a pathname you start at the root (so you start with "/"), then follow the path down the hierarchy (including each directory name) and you end with the filename.
- In between every directory name you put a "/".

Relative Pathnames

- Prefixed w/the current directory, $PWD
- So, relative to the current working directory

```
$ cd /home/VIT
$ pwd
/home/VIT
$ ls linux/Syllabus
linux/Syllabus
$ ls X
ls: X: No such file or directory
$ ls /home/student/X
/home/student/X
```

Special Relative paths…

- . – The current directory
- .. – The *parent* directory

```
$ pwd
/home/VIT
$ ls ./os
./os
$ ls ../student
X
```

Commands for Traversing Filesystem

- `ls` – lists contents of a directory
  - `-a` – all files
  - `-l` – long listing
- `pwd` – print working (current) directory
- `cd` – change directory
  - w/out argument, takes you home

Listing files (ls)

- It shows the files /directories in your current directory.
- **OPTIONS:**

  -l Lists all the files, directories and their mode, Number of links, owner of the file, file size, Modified date and time and filename.

  -t Lists in order of last modification time.

  -a Lists all entries including hidden files.

  -d Lists directory files instead of contents.

  -p Puts slash at the end of each directories.

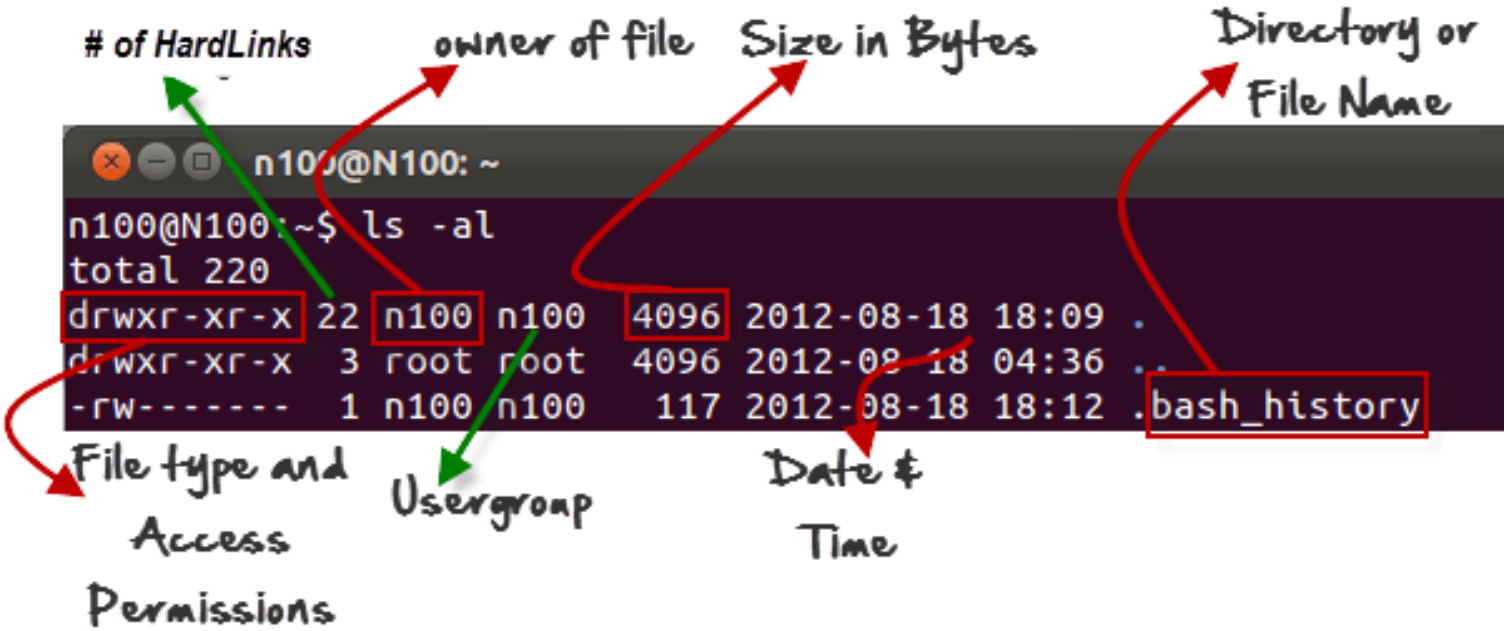
  -u List in order of last access time.

  -i Display inode information.

  -ltr List files order by date.

  -lSr List files order by file size.

'ls -al' gives detailed information of the files.



Basic Commands (cont)

1.date: To display current date & time of the system.

2. cal :To display calendar of current month.

3. who:List who is currently logged on to the system.

4. Whoami:Report what user you are logged on as.

5. echo :Echo a string (or list of arguments) to the terminal

6. bc:To perform mathematical operations

7. clear:To clear the screen

8. alias : Used to tailor commands

Ex alias erase=rm Generate an employee report using AWK programming.

alias grep=”grep -i”

alias cp=“cp –i”

9. man <cmd name>: To get help for any command

10. passwd: To change the passward

11. exit: To logout from the terminal

passwd

- With the passwd command, you can change the password associated with your individual account name.

- For example,

```
sariyer:~> passwd
Changing password for dag.
Old password:
New passwd:
Retype new passwd:
sariyer:~>
```

Basic Commands (cont)

File & Directory Related Commands

1. cp <fromfile> <tofile>: Copy from the <fromfile> to the <tofile>
2. mv <fromfile> <tofile> : Move/rename the <fromfile> to the <tofile>
3. rm <file>:Remove the file named <file>
4. mkdir <newdir>:Make a new directory called <newdir>
5. rmdir <dir>:Remove an (empty) directory
6. cd <dir> :Change the current working directory to dir
7. pwd : Print (display) the working directory
8. cat > <file> :To create new file and save it by pressing ^d
9. cat >> <file>: To append contents into existing file
10. cat <file>:To see the contents of existing file
11. more <file>:Paging out the contents of file

Basic Commands (cont)

12. file <file>:To check the type of file
13. wc <file>:To count lines,words,charaters of file
14. cmp <file1> <file2>:To compare two files
15. comm <file1> <file2>:To display common values between two files
16. diff <file1> <file2>:To convert one file to another
17. gzip <file>:To compress the file
18. gunzip <file>:To unzip the contents of zipped file
19. ls :List the files in the current working directory
20. ls <dir>:List all files & directories in given directory
21. ln <fromfile><tofile>: Creates a symbolic link to a file

Advance commands

1. pr <file> :Paginating the file

   Ex: pr –h “test” –d –n fname

2. head <file>:Display first 10 lines of file

   Ex: head –n -3 fname

3. tail <file> :To display last 10 lines of file

   Ex: tail -3 fname ; tail –c 100 fname

4. cut <file> :Splitting file vertically

   Ex: cut –c 2-10,12-14 fname

   cut –d “|” –f 2,4 fname

5. paste <file1> <file2> :To combine two file vertically rather than horizontally

   Ex: paste –d “|” fname1 fname2

6. sort <file>:To sort file in order by field wise

   Ex: sort –t”|” –k 2 fname

   sort –r fname

Advance commands

7. uniq <file> :Locate repeated & nonrepeated lines

   Ex: uniq fname; uniq –d fname

8. tr ch1 ch2 < <file1>:To translate occurrence of ch1 by ch2

   Ex: tr ‘|’ ‘+’ < fname1

9. tee: read from standard input and write to standard output and files

   Ex: ls *.txt | wc -l | tee count.txt

Some of the things these commands manipulate:

- The time stamp: Each file has three dates associated with it. These are creation time, last modification time and last access time.
- The owner: the owner of files
- The group: the group of users
- The permissions: read, write, execute permissions of files. The permissions tell unix who can access what file, or change it, or, in the case of programs, execute it. Each of these permissions can be toggled separately for the owner, the group, and all the other users.

File Time Attributes

- Time Attributes:
  - when the file was last changed `ls -l`
  - sort by modification time `ls -lt`

Types of Users

- **user(owner): he is the user who create the file.**
- **Group: They are the users who belong to the same group that the owner of the file belongs.**
- **Others : they are any other users in the system.**
- And there is a super user **(the root)** is the administrator of the computer system which have access privileges to all files.
- The login name for the super user is root and user id is 0.

Utilities for Manipulating file attributes

- **chmod** change file permissions
- **chown** change file owner
- **chgrp** change file group
- **umask** user file creation mode mask
- only owner or super-user can change file attributes
- upon creation, default permissions given to file modified by process **umask** value

File Owners

- Each file is owned by a user.
- You can find out the username of the file's owner with the `-l` or `-o` option to `ls`:

```
[jjohnson@ws44 winter]$ ls -l
total 24
drwxr-xr-x    7 jjohnson users    80 Jan  3  2005 cs265/
-rw-------    1 jjohnson users  8258 Jan  3  2005 cs265.html
-rw-r--r--    1 jjohnson users  8261 Jan  3  2005 cs265.html~
```

Getting more information about files…

```
drwxrwsr-x  23 gmt      icon      4096 2009-11-24 13:17 icon
-rw-r--r--   1 jharriso jharriso  3599 2009-12-22 16:41 index.html
drwxrwsr-x   5 gmt      officweb  4096 2008-08-05 11:20 intranet
drwx------   2 root     root      4096 1996-06-07 09:32 lost+found
```

owner group size last-modified time file name

no. of hard links

access permissions

file type

–		normal file
d		directory
l	(*ell*)	symbolic link

File access permissions

```
drwxrwsr-x   23 gmt      icon      4096 2009-11-24 13:17 icon
-rw-r--r--    1 jharriso jharriso  3599 2009-12-22 16:41 index.html
drwxrwsr-x    5 gmt      officweb  4096 2008-08-05 11:20 intranet
drwx------    2 root     root      4096 1996-06-07 09:32 lost+found
```

access permissions for others (**o**)

access permissions for group (**g**)

access permissions for owner (**u**)

- **r** read
- **w** write
- **x** execute (executable file)
  enter (directory)
- **–** no permission

File Permissions

- Each file has a set of permissions that control who can mess with the file.
- There are three types of permissions:
  - read abbreviated **`r`**
  - write abbreviated **`w`**
  - execute abbreviated **`x`**
- There are 3 sets of permission:
  1. user
  2. group
  3. other (the world, everybody else)

`ls -l` and permissions

-rwxrwxrwx

User Group Others

Type of file:
- – plain file
d – directory
b-block device file
c- character device file
l – symbolic link
p- pipe
s- socket

rwx

- Files:
  - **r** – allowed to read.
  - **w** – allowed to write
  - **x** – allowed to execute
- Directories:
  - **r** – allowed to see the names of the file.
  - **w** – allowed to add and remove files.
  - **x** – allowed to enter the directory

Access Types

TABLE 8.2 Possible Access Permission Values for a File for a User, Their Octal Equivalents, and Their Meanings

r	w	x	Octal Digit for Permission	Meaning
0	0	0	0	No permission
0	0	1	1	Execute-only permission
0	1	0	2	Write-only permission
0	1	1	3	Write and execute permissions
1	0	0	4	Read-only permission
1	0	1	5	Read and execute permissions
1	1	0	6	Read and write permissions
1	1	1	7	Read, write, and execute permissions

Changing Permissions

- The **`chmod`** command changes the permissions associated with a file or directory.
- There are a number of forms of chmod, this is the simplest:

```
chmod mode file
```

chmod – numeric modes

- Consider permission for each set of users (user, group, other) as a 3-bit #
  - r – 4
  - w – 2
  - x – 1
- A permission (mode) for all 3 classes is a 3-digit octal #
  - 755 – rwxr-xr-x
  - 644 – rw-r—r--
  - 700 – rwx------

Changing file access privileges (cont…)

TABLE 8.4 Values for Symbolic Mode Components

Who	Operator	Privilege
u User	+ Add privilege	r Read bit
g Group	– Remove privilege	w Write bit
o Other	= Set privilege	x Execute/search bit
a All		u User's current privileges
ugo All		g Group's current privileges
		o Others' current privileges
		l Locking privilege bit
		s Sets user or group ID mode bit
		t Sticky bit

chmod - examples

```
$ chmod 700 CS571
$ ls -o Personal
drwx------  10 kschmidt 4096 Dec 19 2004 CS571/
$ chmod 755 public_html
$ chmod 644 public_html/index.html
$ ls -ao public_html
drwxr-xr-x  16 kschmidt 4096 Jan  8 10:15 .
drwx--x--x  92 kschmidt 8192 Jan  8 13:36 ..
-rw-r--r--   5 kschmidt  151 Nov 16 19:18 index.html
$ chmod 644 .plan
$ ls -o .plan
-rw-r--r--   5 kschmidt  151 Nov 16 19:18 .plan
```

`chmod` – symbolic modes

- Can be used to set, add, or remove permissions
- Mode has the following form:

`[ugoa] [+-=] [rwx]`

  - u – user g – group o – other a – all
  - + add permission - remove permission = set permission

chmod examples

```
$ ls -al foo
-rwxrwx--x   1 hollingd grads foo

$ chmod g-wx foo
$ ls -al foo
-rwxr----x   1 hollingd grads foo

$ chmod u-r .
$ ls
ls: .: Permission denied
```

Chown and chgrp: Changing File Ownership

- There are two commands meant to manipulate the ownership of a file or directory – chown and chgrp.
- They can be used only by the owner of the file. Here’s the syntax for both:

chown options new_user file(s)

chgrp options new_group file(s)

Chown and chgrp: Changing File Ownership

- chown (change ownership) takes the new user's user-id as argument followed by one or more files to change the file ownership.
- The chgrp (change group) command changes the group owner of a file.
- Both chown and chgrp also work with the –R option to perform their operations in a recursive manner.
- The super user can change every file attribute.

```
# ls -lart tmpfile
-rw-r--r-- 1 himanshu family 0 2012-05-22
20:03 tmpfile
# chown root tmpfile
# ls -l tmpfile
-rw-r--r-- 1 root family 0 2012-05-22
20:03 tmpfile
```

Change the group

```
# ls -l tmpfile
-rw-r--r-- 1 himanshu family 0 2012-05-22 20:03 tmpfile
# chown :friends tmpfile
# ls -l tmpfile
-rw-r--r-- 1 himanshu friends 0 2012-05-22 20:03 tmpfile
```

Change both owner and the group

```
# ls -l tmpfile
-rw-r--r-- 1 root family 0 2012-05-22 20:03 tmpfile
# chown himanshu:friends tmpfile
# ls -l tmpfile
-rw-r--r-- 1 himanshu friends 0 2012-05-22 20:03
tmpfile
```

Using chown command on symbolic link file

```
 ls -l tmpfile_symlnk
lrwxrwxrwx 1 himanshu family 7 2012-05-22 20:03 tmpfile_symlnk -> tmpfile

 # chown root:friends tmpfile_symlnk
 # ls -l tmpfile_symlnk
 lrwxrwxrwx 1 himanshu family 7 2012-05-22 20:03 tmpfile_symlnk -> tmpfile
 # ls -l tmpfile
 -rw-r--r-- 1 root friends 0 2012-05-22 20:03 tmpfile
```

- When the chown command was issued on symbolic link to change the owner as well as the group then its the referent of the symbolic link ie 'tmpfile' whose owner and group got changed.
- This is the default behavior of the chown command.
- Also, there exists a flag '–dereference' for the same.

Using chown command to forcefully change the owner/group of symbolic file.

Using flag '-h', you can forcefully change the owner or group of a symbolic link as shown below.

```
# ls -l tmpfile_symlnk
lrwxrwxrwx 1 himanshu family 7 2012-05-22 20:03 tmpfile_symlnk -> tmpfile
# chown -h root:friends tmpfile_symlnk
# ls -l tmpfile_symlnk
lrwxrwxrwx 1 root friends 7 2012-05-22 20:03 tmpfile_symlnk -> tmpfile
```

umask: Default File Permissions

- The default permissions are inherited by files and directories created by all users:
  - rw-rw-rw- (octal 666) for regular files
  - rwxrwxrwx (octal 777) for directories
- However, these are not the permissions you see. This default is transformed by subtracting the user mask from it to remove one or more permissions.
- This mask is evaluated by using umask:

$ umask

```
student@unix-13:~$
umask
0022
student@unix-13:~$
```

umask: Default File Permissions

$ umask

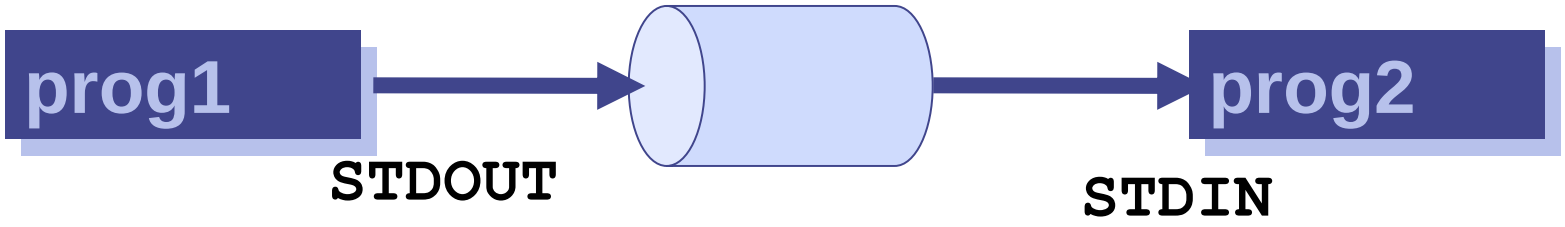
77

- This is an octal number, and subtracting this value from the file default yields 666 – 077 = 600.
- This represents the default permissions (rw-------) when you create a file.
- The default directory permissions are set (rwx------) when a directory is created.

Filter and Redirection:

Pipes – connecting processes

- A pipe is a holder for a stream of data.
- A pipe can be used to hold the output of one program and feed it to the input of another.



Asking for a pipe

- Separate 2 commands with the "|" character.
- The shell does all the work!

```
ls -1 | sort
ls -1 | sort > sortedlist
ls -1 | sort | head > top.ten
```

filters

- Programs that read some input (but don't change it), perform a simple transformation on it, and write some output (to stdout)
- Some common filters…
  - `wc` – word count (line count, character count)
  - `tr` – translate
  - `grep`, `egrep` – search files using regular expressions
  - `sort` – sorts files by line (lexically or numerically)
  - `cut` – select portions of a line
  - `uniq` – Removes identical adjacent lines
  - `head`, `tail` – displays first (last) n lines of a file

pipes and combining filters

- Connect the output of one command to the input of another command to obtain a composition of filters

- who | wc -l
- ls | sort -f
- ls -s | sort -n
- ls -l | sort -nr -k4
- ls -l | grep ‘^d’

• pr

- pr <file> :Paginating the file

Ex pr –h “test” –d –n fname

```
admin36@soft36:~$ pr -n -h "trial" test

2014-01-29 17:07                    trial                    Page 1

    1 This is test file
    2 hello vit
```

• sort

- sort <file>:To sort file in order by field wise
- Ex sort –t”|” –k 2 fname
- sort –r fname

• cut

- cut <file> :Splitting file vertically
- Ex cut –c 2-10,12-14 fname
- a. cut –d “|” –f 2,4 fname

paste

- paste <file1> <file2> :To combine two file vertically rather than horizontally
- Ex paste –d “|” fname1 fname2

head

- head <file>:Display first 10 lines of file
- Ex head –n -3 fname

• tail

- tail <file> :To display last 10 lines of file
- Ex tail -3 fname ;
- tail –c 100 fname

• tee

- tee: read from standard input and write to standard output and files
- Ex. ls *.txt | wc -l | tee count.txt

uniq

- uniq <file> :Locate repeated & nonrepeated lines
- Ex uniq fname; uniq –d fname

- tr

- The “translate characters” command operates on standard input–it doesn’t accept a filename as a parameter.
- Instead, it’s two parameters are arbitrary strings.
- It replaces all occurences of string1 in the input string2.
- In addition to relatively simple commands such as tr frog toad, tr can accept more complicated commands.

tr *string1 string2*

```
admin40@admin40-OptiPlex-360:~$ cat
>test
this is test file
admin40@admin40-OptiPlex-360:~$ tr '[a-z]'
'[A-Z]' <test
THIS IS TEST FILE
admin40@admin40-OptiPlex-360:~$
```

Redirection

1. Standard input redirection: It is used to redirect standard input.

   Ex. cat < fname

2. Standard output redirection : It is used to redirect standard output.

   Ex cat >fname

3. Standard error redirection: It is used to redirect standard error.

   Ex cat fname 2>Errorfile

Grep: Global Regular Expression Print

- Searching and pattern matching tools
- Searches files for one or more pattern arguments. It does plain string, basic regular expression, and extended regular expression searching
- Following are some of the options for grep

  -i ignore case for matching

  -v doesn’t display lines matching expression

  -n display line numbers along of occurrences

  -c counting number of occurrences

  -l display list of file names

  -e exp for matching

  -f file take patterns from file

  -E treat pattern as an extended reg. exp

  -F matches multiple fixed strings (fgrep)

Grep: Global Regular Expression Print

- grep -ow "yourstring“
- grep -w "search_word" text_file
- grep -P '^(tomcat!?)' tst1.txt

Programming Tools and Utilities Available under Linux

- **Text Editors**
  - Xemacs
  - Emacs
  - Pico
  - vi
- **Compilers**
  - C compiler - gcc
  - C++ compiler - g++
  - Java compiler & Java Virtual Machine - javac & java
- **Debuggers**
  - C / C++ debugger - gdb
- **Interpreters**
  - Perl - perl
  - Tcl/Tk - tcl & wish
- **Miscellaneous**
  - Web Browsers - Mozilla, Netscape, Firefox, and Lynx (lynx is text based)
  - Instant Messengers - Gaim
  - Email - Netscape is there, but we will learn Pine

Try following commnads

- compgen
- fg
- Bg
- Zcat
- du
- df
- touch
- which