Overview

- Introduction – Linux philosophy
- Command line basics – getting help
- The shell revisited: some features
- Navigating the file system
- File manipulation
- Text editing
- Various commands
- Archiving
- Groups, users, security
- Process control

Command line basics
Getting Help

Commands, Options, and Arguments

- A Linux command usually consists of 3 parts: the command itself, the command options, and its arguments.
  `command [OPTIONS] [ARG1] […ARGX]`
  - To execute the command, press enter.
  - When it runs, it may (or may not!) print output to the screen.
  - When completed, the command prompt is displayed again

Commands, Options, and Arguments

- **command** – the executable (program or package) that is to be run.
  - If you are running your own application, you must include either the full path or the relative path as part of the command.
    ```
    $ ./my_hello_world
    ```
  - Most commands that come packaged with the OS or are installed by the package manager (executables often located in /bin or /usr/bin) do not need the path because they have already been added to the environment variable `$PATH`.
  - Check with `whereis ls`

https://cvw.cac.cornell.edu/Linux/shells

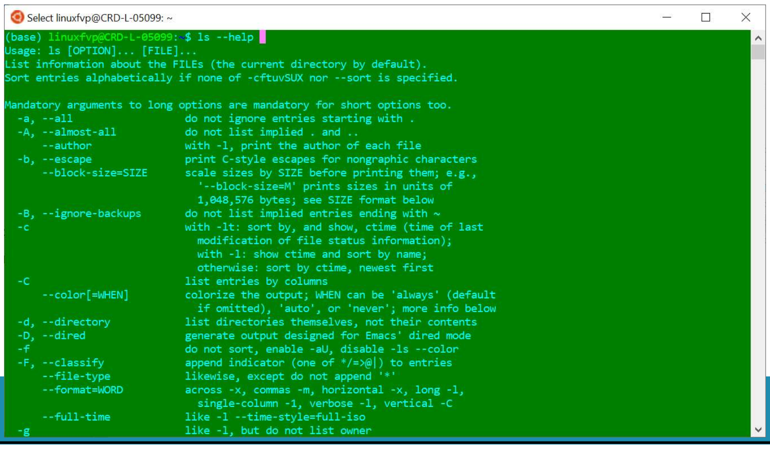
Commands, Options, and Arguments

- **option(s)** – (*flags*) optional arguments for the command that alter the behavior.
  - Start with - or -- (example: `-h` or `--help` for help).
  - Each command may have different options or no options at all.
  - Some options require an argument immediately following.
  - Explore options for commands by reading the Manual Pages.
- **argument(s)** – depend on the command and the flags selected.
  - Certain flags require an argument.
  - Filename arguments must include a path unless located in the current directory.

https://cvw.cac.cornell.edu/Linux/shells

Getting help: command built-in

- Help on most Linux commands is typically built into the command themself
- These flags usually look like "`-h`" or "`--help`".
- `$ ls --help`


Select linuxfvp@CRD-L-05099: ~

```
(base) linuxfvp@CRD-L-05099:~$ ls --help
Usage: ls [OPTION]... [FILE]...
List information about the FILEs (the current directory by default).
Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.

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 C-style escapes for nongraphic characters
      --block-size=SIZE      scale sizes by SIZE before printing them; e.g.,
                               '--block-size=M' prints sizes in units of
                               1,048,576 bytes; see SIZE format below
  -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, newest first
  -C                         list entries by columns
      --color[=WHEN]         colorize the output; WHEN can be 'always' (default
                               if omitted), 'auto', or 'never'; more info below
  -d, --directory            list directories themselves, not their contents
  -D, --dired                generate output designed for Emacs' dired mode
  -f                         do not sort, enable -aU, disable -ls --color
  -F, --classify             append indicator (one of */=>@|) to entries
      --file-type            likewise, except do not append '*'
      --format=WORD          across -x, commas -m, horizontal -x, long -l,
                               single-column -1, verbose -l, vertical -C
      --full-time            like -l --time-style=full-iso
  -g                         like -l, but do not list owner
```

Getting help: man pages

- Best source of information can be found in the online manual pages, "**man pages**" type "`man command`".

  ```
  $ man ls
  ```

- Tips:
  - To search for a particular word (e.g. file) within a man page, type "`/word`".
  - Use up/down arrows or the space bar to navigate through a man page.
  - To quit from a man page, type the "`q`" key.
  - If you do not remember the name of Linux command and you know a keyword relating to the command, search the man pages with the `–k`
    `apropos (man –k)` prints out a one-line summary of commands, based on a keyword search.

    ```
    $ man -k control
    ```

Getting help: info pages

- Info pages are similar to man page, but instead of being displayed on one long scrolling screen, they are presented in shorter segments with links to other pieces of information.
- Access with the "`info`" command

  ```
  $ info ls
  ```

- Tips:
  - To quit from a info page, type the "`q`" key.
  - Type "h" to get more help on the info, or `info –help`
  - Use the arrow keys to browse through the text
  - Move the cursor on a line starting with an asterisk, containing the keyword about which you want info, then hit Enter.
  - Use the P and N keys to go to the previous or next subject.
  - The space bar will move you one page further

Getting help

- bash has a built-in help facility available for each of the shell *builtins*.
- Get an overview of the builtins: `help -d`

```
$ help pwd
$ help ls
```

- `whatis` displays a very brief description of a command

```
$ whatis pwd
```

- `whereis` locates the binary, source and manual files for the specified command
- `type` display information about the command type

More on the command line

1. Linux systems are case (and space) sensitive.
   - `MyFile` is not same as `myfile`
2. There is no "recycle bin" or "trash can" when working in the command line environment. There might be one for GUI.
   When files are deleted on the command line, they instantly disappear forever.
3. You should always practice new commands on a test case. This minimizes the chances of an accident that can take down an important system

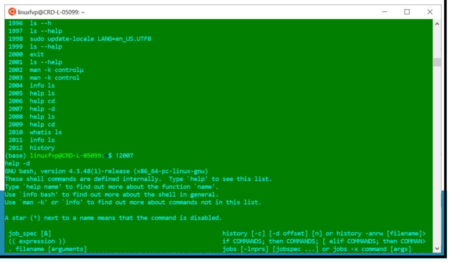
The Shell revisited: features

Auto-Completion

- Have the shell automatically complete commands or file paths.
- Activated using the **<TAB>** key on most systems
- examples
  - `$ whe<TAB>`
  - `$ whereis`
  - `$ ls -l /etc/en<TAB>`
  - `$ ls -l /etc/environment`
- When more than one match is found, the shell will display all matching results (use **<TAB>** twice)
  - `$ ls -l /etc/host<TAB>`

Command history / Arrow Up

- Previously executed commands can be recalled by using the **Up Arrow** key on the keyboard.
- Most Linux distributions remember the last five hundred commands by default.
- Display commands that have recently been executed
  - The `history` command displays a user's command line history.
  - You can execute a previous command using `![NUM]` where NUM is the line number in history you want to recall.



```
 1996  ls --h
 1997  ls --help
 1998  sudo update-locale LANG=en_US.UTF8
 1999  ls --help
 2000  exit
 2001  ls --help
 2002  man -k controlµ
 2003  man -k control
 2004  info ls
 2005  help ls
 2006  help cd
 2007  help -d
 2008  help ls
 2009  help cd
 2010  whatis ls
 2011  info ls
 2012  history
(base) linuxfvp@CRD-L-05099:~$ !2007
help -d
GNU bash, version 4.3.48(1)-release (x86_64-pc-linux-gnu)
These shell commands are defined internally.  Type `help' to see this list.
Type `help name' to find out more about the function `name'.
Use `info bash' to find out more about the shell in general.
Use `man -k' or `info' to find out more about commands not in this list.

A star (*) next to a name means that the command is disabled.

 job_spec [&]                                                history [-c] [-d offset] [n] or history -anrw [filename]>
 (( expression ))                                            if COMMANDS; then COMMANDS; [ elif COMMANDS; then COMMAN>
 . filename [arguments]                                      jobs [-lnprs] [jobspec ...] or jobs -x command [args]
```

Globbing: use wildcard

Wildcard	Function
*	Matches 0 or more characters
?	Matches 1 character
[abc]	Matches one of the characters listed
[a-c]	Matches one character in the range
[!abc]	Matches any character not listed
[!a-c]	Matches any character not listed in the range
{tacos,nachos}	Matches one word in the list

```
$ ls -l /etc/host*
$ ls -l /etc/hosts.{allow,deny}
$ ls -l /etc/hosts.[!a]*
$ ls -l /etc/host?
```

Displaying file contents

Several ways of displaying the contents of files.

- `$ cat file1`
  displays the contents of the given file.
- `$ cat file1 file2 file3 ...` (concatenate)
  Concatenates and outputs the contents of the given files.
- `$ more file1`
  Display the output of a command or text file one page at a time.
  - Can also jump to the first occurrence of a keyword (`/` command).

Displaying file contents

- `$ less file1`
  - Does more than more.
  - Doesn't read the whole file before starting.
  - Supports backward movement in the file (? command).
  - Search with `/`, next (`n` or `N`)
  - Press `q` to exit
- `$ display file1`
  Displays graphical file (simple image) (needs imagemagick)

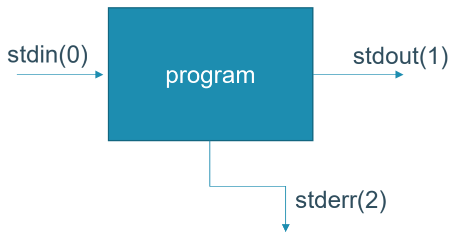
The head and tail commands

- `$ head [-<n>] <file>`
  Displays the first <n> lines (or 10 by default) of the given file.
  Doesn't have to open the whole file to do this!
- `$ tail [-<n>] <file>`
  Displays the last <n> lines (or 10 by default) of the given file.
  No need to load the whole file in RAM! Very useful for huge files.
- `$ tail -f <file>` (follow)
  Displays the last 10 lines of the given file and continues to display new lines when they are appended to the file.
  Very useful to follow the changes in a log file, for example.

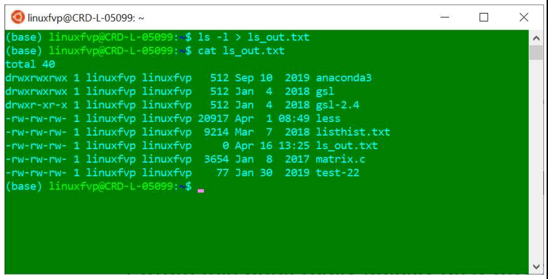
Input / output

- Inputs and outputs of a program are called streams in Linux.
- **stdin** (**st**andar**d in**put) – stream data going into a program. By default, this is input from the keyboard.
- **stdout** (**st**andar**d out**put) – output stream where data is written out by a program. By default, this output is sent to the screen.
- **stderr** (**st**andar**d err**or) – another output stream (independent of stdout) where programs output error messages. By default, error output is sent to the screen.

https://cvw.cac.cornell.edu/Linux/io



Redirection



```
(base) linuxfvp@CRD-L-05099:~$ ls -l > ls_out.txt
(base) linuxfvp@CRD-L-05099:~$ cat ls_out.txt
total 40
drwxrwxrwx 1 linuxfvp linuxfvp   512 Sep 10  2019 anaconda3
drwxrwxrwx 1 linuxfvp linuxfvp   512 Jan  4  2018 gsl
drwxr-xr-x 1 linuxfvp linuxfvp   512 Jan  4  2018 gsl-2.4
-rw-rw-rw- 1 linuxfvp linuxfvp 20917 Apr  1 08:49 less
-rw-rw-rw- 1 linuxfvp linuxfvp  9214 Mar  7  2018 listhist.txt
-rw-rw-rw- 1 linuxfvp linuxfvp     0 Apr 16 13:25 ls_out.txt
-rw-rw-rw- 1 linuxfvp linuxfvp  3654 Jan  8  2017 matrix.c
-rw-rw-rw- 1 linuxfvp linuxfvp    77 Jan 30  2019 test-22
(base) linuxfvp@CRD-L-05099:~$
```

- Redirection is all about files
- Output redirect out from screen to a file
  - This can be done with the redirection operator >
    `$ ls -l > ls_out.txt`
  - Redirection will create the named file if it doesn't exist, or else overwrite the existing file.
- Append with the operator >>
  - append instead of rewriting the file
    `$ ls -l >> ls_out.txt`
- Input redirection
  - Input can also be given to a command from a file instead of typing it in the shell by using the redirection operator <
    `$ sort < tabel.dat`

Redirection

- Error Redirection
  - Normal of standard output (stdout), will not affect stderr (separate stream).
  - Use the redirection operator 2>
    `$ ls non-existing* 2> ls_err.txt`
- Trash any data
  - `/dev/null` is a special file that is used to trash any data that is redirected to it. Any output that is sent to /dev/null is discarded.
    `$ ls > /dev/null`

Pipes

- Piping is all about processes
- Pipes (also referred to as pipelines) can be used to direct the output of one command to the input of another.
  The Shell arranges it so that the standard output of one command is fed to another command
- use the **|** key on the keyboard
- `$ ls -l | less`
  - the output of the ls command is piped into the less program
  - compare with `$ ls -l > less` (the output of the ls command is saved in a file with the name less)

Redirection

- Redirect and Save Output
  - When redirecting output to a file, nothing is shown on the screen.
  - To have the output go to both a file and the screen, use the `tee` command:
    ```
    $ ls  -al | tee ls_out
    ```

Hands-on 2