Outline

- Introduction - history
- ➢ Command line basics – getting help
- File system
- Working with files and directories
- More file handling
- The shell revisited
- Monitoring resources

Command line basics
Getting Help

Commands, Options, and Arguments

- A Linux command normally consists of 3 parts: the **command itself**, the command **options**, and its **arguments**.
  ```
  command [OPTIONS prefixed with – or --] [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
- Check:
  - https://explainshell.com/
  - https://www.makeuseof.com/explain-shell-linux-man-pages-alternative/

Commands, Options, and Arguments

- **command** – the executable (program or package) that is to be run.
  - 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`
    - `echo $PATH`
  - If you are running your own application, you must include either the full path or the relative path as part of the command.
    ```
    (/path/to/executable/name_of_prog)
      $ ./my_hello_world
      $ ./a.out
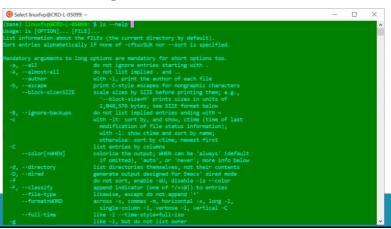    ```

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 a - 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 itself
- These flags usually look like "`-h`" or "`--help`".
- `$ ls --help`

# Getting help: man pages

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

    `$ man ls`

- Tips:
    - To search for a particular keyword 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` displays 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

# Useful 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.



# Command history related

| Command | Options | Common usage | What does it? |
|---|---|---|---|
| CTRL-c | | | Stop current command |
| CTRL-r | | reverse search in the command history | Recall the last command matching the characters you provide |
| !! | | | Repeat last command |
| !abc | | | Run last command starting with abc |
| !number | | | Retrieve command on line number in the history |

# history

- Check the maximum number of lines kept in history: `echo $HISTSIZE`
- Check the file containing the history: `echo $HISTFILE`
- when bash exits, it dumps the history in memory to a disk file.
  - history file is overwritten every time we exit a bash interactive login.
  - has implications when we have multiple bash sessions/terminals active because the HISTFILE would contain only the contents of the last exiting shell.
  - https://www.baeldung.com/linux/preserve-history-multiple-windows

# Copy/paste

- Shortcut:
  - Ctrl+Insert (copy text) and Shift+Insert (paste text)
  - Ctrl+shift+C (copy text) and Ctrl+shift+V (paste text)
- Use right-click of mouse button to copy and paste
- Use the middle mouse button
- Works on Ubuntu and many other Linux distributions
  Not all these methods may work, the keybindings for copy-pasting are dependent on the terminal

# information about users

- `$ who`
  Lists all the users logged on the system.
- `$ whoami`
  Tells what user I am logged as.
- `$ groups`
  Tells which groups I belong to.