

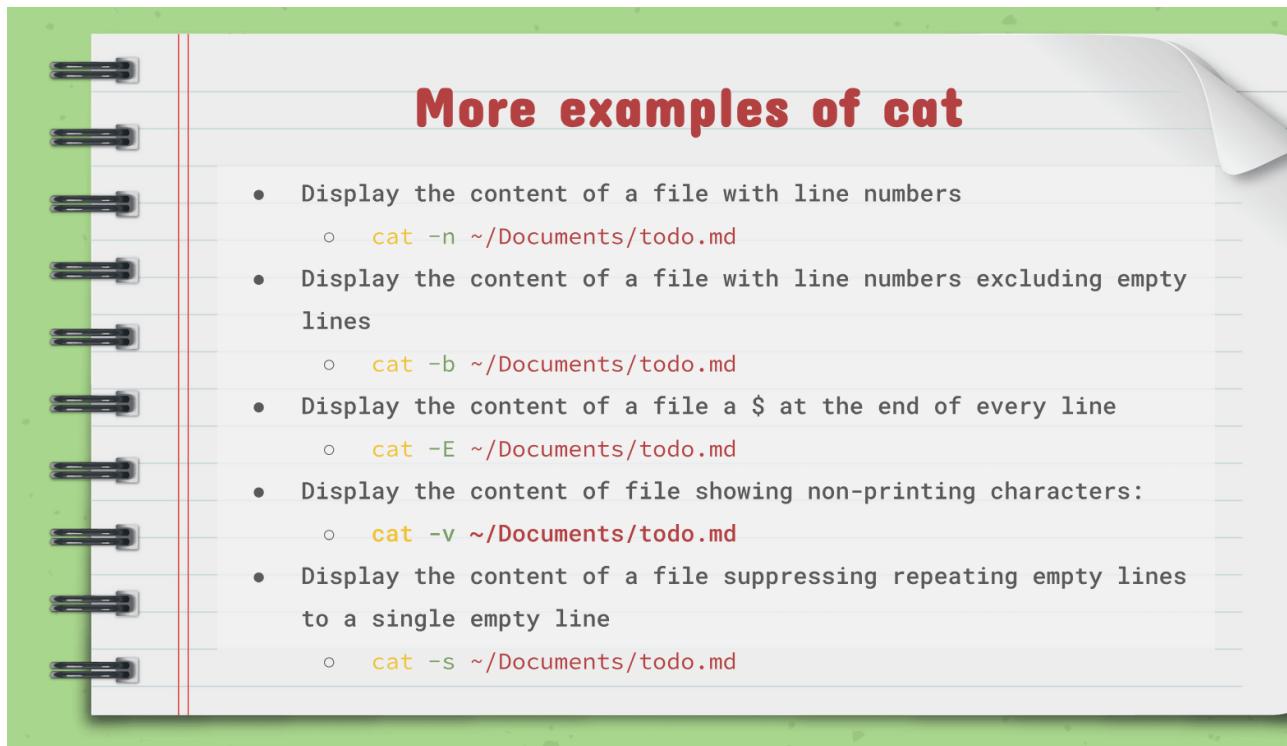
## Notes 7

# Handling Text Files

---

### The cat Command:

- Description:
  - The cat command is used for displaying the content of a file.
  - Cat is short for concatenate which is the command's intended use.
- Usage:
  - cat + option +file(s) to display
- Examples:
  - Display the content of a file located in the pwd; cat todo.lst
  - Display the content of a file using absolute path; cat ~/Documents/todo.lst
- More examples of cat:



---

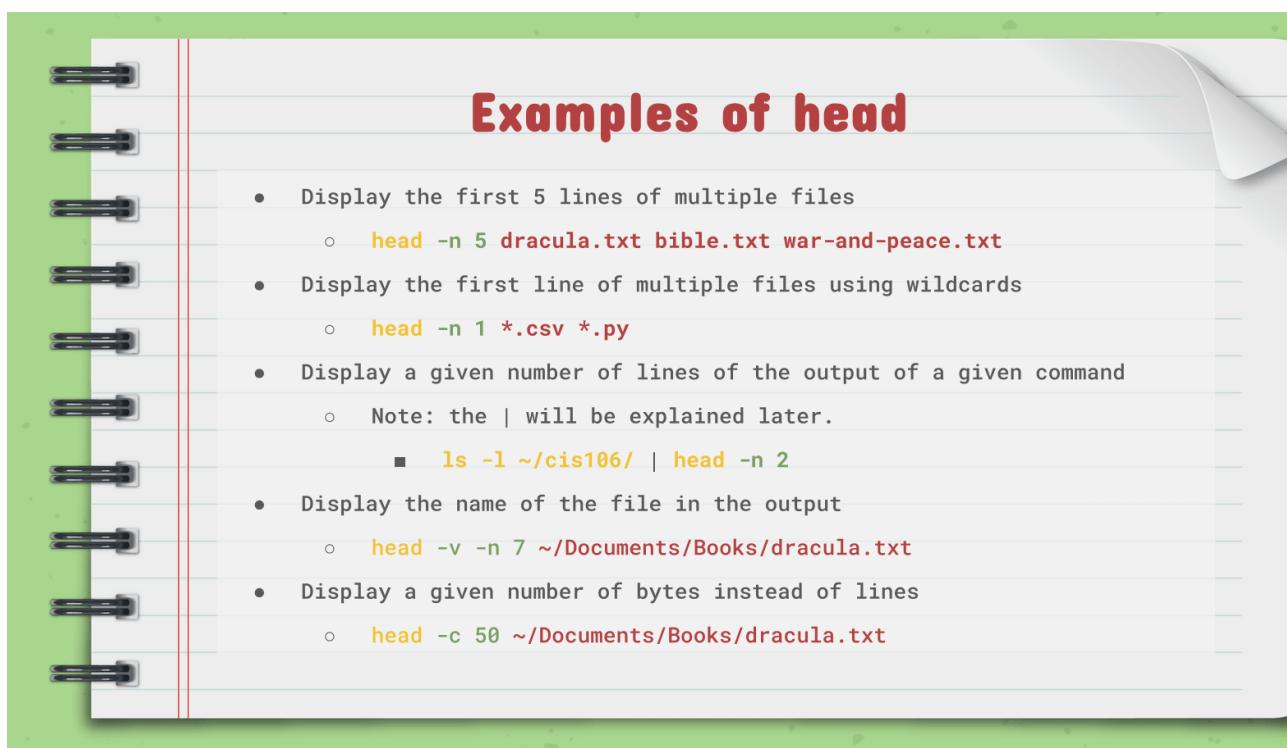
### The tac Command:

- Description:
  - The tac command is used for displaying the content of a file in reverse order.
  - Just like cat, tac concatenates files and displays the output of the concatenation.
- Usage:
  - tac + option + file(s) to display

- Examples:
    - Display the content of a file located in the pwd; tac todo.md
    - Display the content of a file using absolute path; tac ~/Documents/todo.md
- 

### The head Command:

- Description:
  - THe head command displays the top N number of lines of a given file.By default, it prints the first 10 lines. If more than one file name is provided then data from each file is preceded by its file name.
- Usage:
  - head + option + file(s)
- Examples:
  - Display the first 10 lines of a file; head ~/Documents/Book/dracula.txt
  - Display the first 5 lines of a file; head -5 ~/Documents/Book/dracula.txt
- More Examples of head:

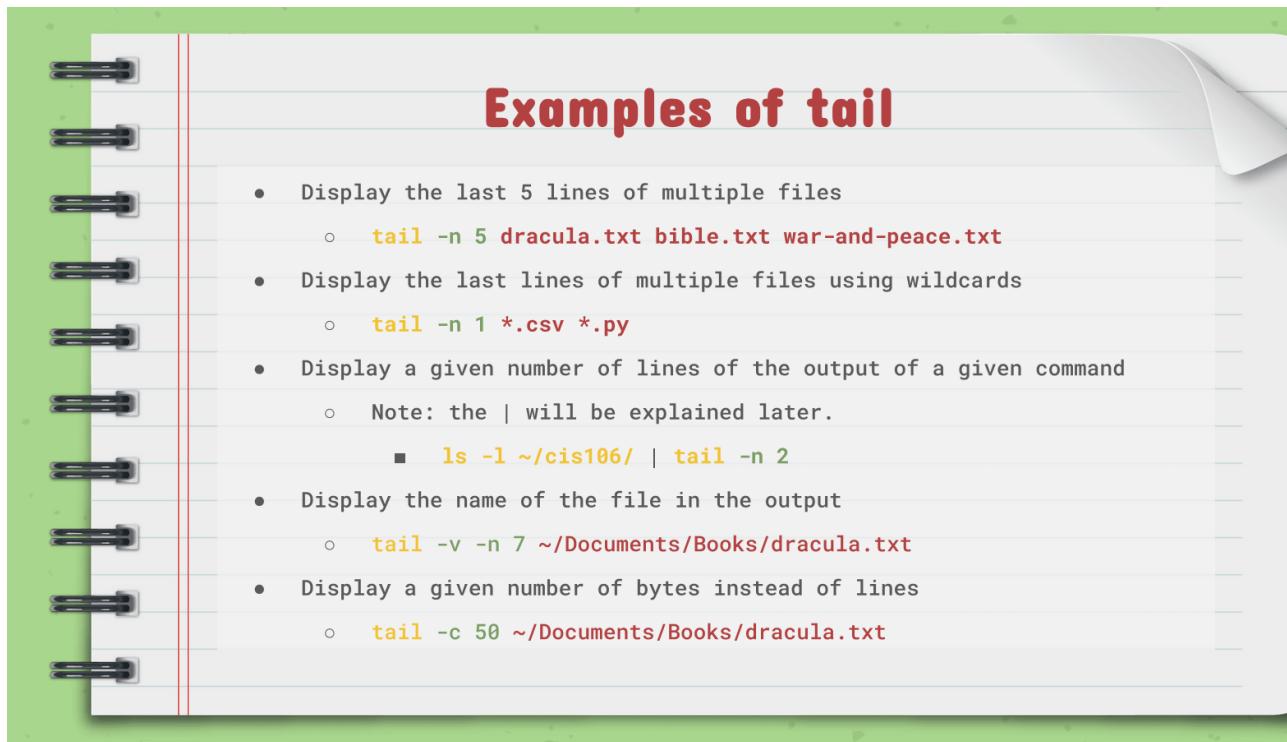


---

### The tail Command:

- Description:
  - The tail command displays the last N number of lines of a given file. By default, it prints the first 10 lines. If more than one file name is provided then data from each file is preceded by its file name.

- Usage:
  - tail + option + file
- Examples:
  - Display the last 10 lines of a file; tail ~/Documents/Book/dracula.txt
  - Display the last 5 lines of a file; tail -5 ~/Documents/Book/dracula.txt
- More Examples of tail:



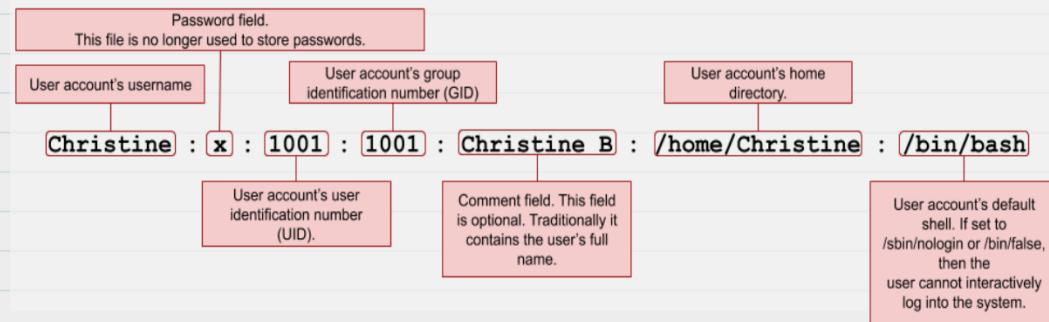
---

#### The cut Command:

- Description:
  - The cut command is used to extract a specific section of each line of a file and display it to the screen.
- Usage:
  - cut + option + file(s)
- Examples:
  - Display a list of all the users in your system; cut -d ':' -f1 /etc/passwd
  - Display a list of all the users in your system with their login shell; cut -d ':' -f1,7 /etc/passwd
- More Examples of cut:

## More on cut | The /etc/passwd file

- The /etc/passwd contains one line for each user account, with seven fields delimited by colons (:)
- Every time an account gets created, this file gets updated



## More on cut | explained

-d specifies the delimiter to use  
': ' is the delimiter  
-f1,7 specifies that for every line in /etc/passwd, the first and seventh field should be cut

A terminal window titled "Tilix: Terminal" shows the command:

```
cut -d ':' -f1,7 /etc/passwd
```

The output is:

```
root:/bin/bash
daemon:/usr/sbin/nologin
bin:/usr/sbin/nologin
```

Annotations explain the output:

- User names**: Points to the first column of the output.
- The login shell**: Points to the last column of the output.

## More examples of cut

- Cut a range of bytes per line
  - `cut -b 1-5 usernames.txt`
- Cut a file using a delimiter but changing the delimiter in the output.
  - `cut -d ':' -f1,7 --output-delimiter=' => '` `/etc/passwd`
- Cut a file excluding a given field
  - `cut -d ',' --complement -s -f3 users.txt`
- Cut the permissions from the output of ls
  - `ls -l | cut -d ' ' --complement -s -f1`

The paste Command:

## The paste command

- Description:
  - The paste command is used for joining files horizontally in columns
- Usage:
  - `paste + option + files`
- Basic Example:
  - Merge two files
    - `paste users.lst ip_address.lst`
  - Merge two files using a different delimiter
    - `paste -d ":" users1.lst ip_addresses.lst`

The sort Command:

# The sort command

- Description:
  - The sort command is used for sorting files. The sort command supports sorting: alphabetically, in reverse order, by number, and by month.
  - The sort command follows this order unless specified otherwise:
    - Lines starting with a number will appear before lines starting with a letter.
    - Lines starting with a letter that appears earlier in the alphabet will appear before lines starting with a letter that appears later in the alphabet.
    - Lines starting with a lowercase letter will appear before lines starting with the same letter in uppercase.
- Usage:
  - `sort + option + file`
- Basic Example:
  - Sort a file
    - `sort users.lst`

# More examples of sort

- Sort a file and save the output to a new file
  - `sort -o sorted.lst users.lst`
- Sort a file in reverse order
  - `sort -r users.txt`
- Sort by column number
  - `sort -k 2 users.txt`
- Sort a file with numeric data
  - `sort -n phones.txt`
- Check if a file is sorted
  - `sort -c sorted.lst`
- Sort and remove duplicate entries
  - `sort -u users.lst`

Note:  
Use the `-t` option  
to specify a  
delimiter.  
For example

`sort -t";" -k3`  
`cereal.csv`

The wc Command:

# The wc command

- **Description:**
  - The wc command is used for printing the number of lines, characters and bytes in a file
- **Usage:**
  - `wc + option + file(s)`
- **Basic Example:**
  - Display the number of characters in a file
    - `wc -m users.txt`
  - Display the number of lines in a file
    - `wc -l users.txt`
  - Display the number words in a file
    - `wc -w users.txt`

---

## The tr Command:

# The tr command

- **Description:**
  - The tr command is used for translating or deleting characters from standard output.
- **Usage:**
  - `Standard output | tr + option + set + set`
- **Basic Example:**
  - Translate one character to another (For example a period with a comma).
    - `cat file.txt | tr '.' ','`
  - Translate white space into tabs.
    - `cat program.py | tr "[\t\n ]" "\t"`
  - Translate tabs into space.
    - `cat file.py | tr -s "[\t\n ]" " "`

---

## The diff Command:

# The diff command

- Description:
  - The diff command compares files and displays the differences between them
- Usage:
  - `diff + option + file1 + file2`
- Basic Example:
  - Display the difference between two files
    - `diff cars.csv cars-backup.csv`
  - Display the difference between two files in a column format:
    - `diff -y cars.csv cars-backup.csv`

---

## The grep Command:

- Description:
  - Grep is used to search text in given file. Grep works line by line basis (it matches the search criteria in a line by line basis).
- Usage:
  - `grep + option + search criteria + file(s)`
- Example:
  - search any line that contains the word "dracula" in the given file; grep 'dracula'  
  `~/Documents/dracula.txt`
- More Examples of grep:

## Common options of grep

Option	Explanation
-i	Enables case insensitivity. (it will match regardless of case)
-n	Displays line number for every line matched
-E	Treats the pattern (search criteria) as an extended regular expression
-G	Treats the pattern (search criteria) as a basic regular expression
-v	Inverts the search (looks for the opposite of the given criteria)
-o	Only displays the matched string
-c	Search and display the total number of times a pattern is matched
-w	Matches only the given word (pattern) by itself.
-r, -R	Matches recursively. Useful for searching files in a given directory

## More examples of grep

- Search any line that contains the word 'dracula' regardless of the case
  - `grep -i 'dracula' ~/Documents/Books/dracula.txt`
- Search any line that contains the word dracula regardless of case and with number line
  - `grep -in 'dracula' ~/Documents/Books/dracula.txt`
- Search for all the lines that do not contain the word 'war'
  - `grep -v 'war' ~/Documents/Books/war-and-peace.txt`
- Search and display only the matched string (pattern)
  - `grep -o 'pride' ~/Documents/Books/war-and-peace.txt`
- Display how many lines contain the matched string
  - `grep -c 'dracula' ~/Documents/Books/dracula.txt`

## More examples of grep

- Search for a given strings inside files in a given directory.
  - `grep -IR 'conf' /etc/`
- Search and display the total number of times a given word appears in a file
  - `grep -wc '/bin/bash' /etc/passwd`
- The ^ (caret) symbol matches the empty string at the beginning of a line. Search for all the lines that start with a given word.
  - `grep -ni '^dracula' ~/Documents/Books/dracula.txt`
- Search for all the lines that ends with the string "nologin"
  - `grep -n 'nologin$' /etc/passwd`

## More examples of grep

- Search for all the lines that contain a single word in all the files in your system
  - `grep -nir '^linux$' /`
- The . (period) symbol is a meta-character that matches any single character. Search for all the lines that contain a word that starts with letter d has 4 characters after.
  - `grep -n '^d....' ~/Documents/Books/dracula.txt`
- Bracket expressions allows match a group of characters by enclosing them in brackets []. Match all the lines that contain a the words list, last, lost.
  - `grep -n 'l[aio]st' ~/Documents/Books/dracula.txt`

## More advance examples of grep

- Search for all the lines that start with a capital letter
  - `grep -n '^[A-Z]' ~/Documents/Books/war-and-peace.txt`
- Search for more than one word per line
  - `grep -Ew 'horror|love|scare' ~/Documents/Books/dracula.txt`
- Match only lines containing IPv4 addresses
  - `grep -E '[[digit:]]{1,3}\.[[digit:]]{1,3}\.[[digit:]]{1,3}\.[[digit:]]{1,3}' Documentation.txt`
- Search all lines that contain a character repeated 3 times
  - `grep -E "A{3}" file.txt`
- Search all lines that contain a phone number of the format 973-111-2222
  - `grep "[[:digit:]]\{3\}[-][[:digit:]]\{3\}[-][[:digit:]]\{4\}" contacts.csv`
- Display only the options of the of any command from its man page
  - `man ls | grep "^[[:space:]]*[[:punct:]]"`

### The awk Command:

- Description:
  - Awk is a scripting language used for processing and displaying text. Awk can work with a text file or from standard output. Awk was created in Bell labs during the 70s by Alfred Aho, Peter Weinberger, and Brian Kernighan and its name comes from its authors' initials. There are several implementations of Awk: nawk, mawk, gawk, and busybox.
  - Awk performs operations line by line.
- Usage:
  - `awk = option + {awk command} + file + file to save (optional)`
- Example:
  - print the first column of every line of a file; `awk '{print $1}' ~/Documents/Csv/cars.csv`
- More Examples on Awk:

## Awk Variables

\$0	Whole line
\$1,\$2..\$NF	First, second... last field
NR	Total Number of Records
NF	N number of Fields
OFS	Output Field Separator (default " ")
FS	input Field Separator (default " ")
ORS	Output Record Separator (default "\n")
RS	input Record Separator (default "\n")
FILENAME	Name of the file
ARGC	Number of arguments

ARGV	Array of arguments
FNR	File Number of Records
OFMT	Format for numbers (default "%.6g")
RSTART	Location in the string
RLENGTH	Length of match
SUBSEP	Multi-dimensional array separator (default "\034")
ARGIND	Argument Index
ENVIRON	Environment variables
IGNORECASE	Ignore case
CONVFMT	Conversion format
ERRNO	System errors
FIELDWIDTHS	Fixed width fields

## More examples of AWK

- Print first field of /etc/passwd file
  - awk -F: '{print \$1}' /etc/passwd
- Print the last field of the /etc/passwd file
  - awk -F: '{print \$NF}' /etc/passwd
- Print the first and last field of the /etc/passwd
  - awk -F: '{print \$1," = ",\$NF}' /etc/passwd
- Print the first and 3 field with line numbers
  - awk -F: '{print NR,\$1,\$3}' /etc/passwd
- Print the first and 4th field with a different field separator
  - awk -F: '{OFS="="}{print \$1,\$4}' /etc/passwd
- Start printing a file from a given line(exclude the first 2 lines)
  - awk 'NR > 3 { print }' /etc/passwd

## More examples of awk

- Convert the first field to upper/lower case
  - awk -F: '{print toupper(\$1)}' /etc/passwd
- Prints the length of a line(record)
  - awk '{print length(\$0)}' /etc/passwd
- Print specific fields based on a command output. For example, the size and file name

```
ls -lhF Documents/ | awk 'BEGIN { printf "%s\t%s\n", "Size", "Name"} {print $5, "\t", $9}'  
    ■ BEGIN block is executed once at the start
```
- Print specific fields with a head of the /etc/passwd file

```
awk -F: 'BEGIN { printf "%s\t\t%s\n", "User", "Shell" } {print $1, "\t", $7}' /etc/passwd
```

# A more interesting example of awk

```
* ls -lhd --time-style=%D ./* | awk -v OFS='t' 'BEGIN { printf "%s\t%s\t%s\t%s\t%s\n", "Permissions", "Links", "User", "Group", "Size" } { print $1,$2,$3,$4,$6 }' | awk -v OFS='t' 'BEGIN { printf "%s\t%s\t%s\t%s\t%s\t%s\n", "Permissions", "Links", "User", "Group", "Size", "Date Modified", "Name" } { print $1,$2,$3,$4,$5,$6 }'
```

Permissions	Links	User	Group	Size	Date Modified	Name
drwxrwxr-x	3	adrian	adrian	4.0K	03/02/22	./Applications
drwxrwxr-x	3	adrian	adrian	4.0K	04/03/22	./Calibre
drwxr-xr-x	2	adrian	adrian	4.0K	04/06/22	./challenge-Lab6
drwxr-xr-x	2	adrian	adrian	4.0K	03/01/22	./Desktop
drwxr-xr-x	5	adrian	adrian	4.0K	04/12/22	./Documents
lrwxrwxrwx	1	adrian	adrian	28	03/03/22	./dotfiles
drwxr-xr-x	10	adrian	adrian	24K	04/12/22	./Downloads
drwxrwxr-x	2	adrian	adrian	4.0K	04/03/22	./fileAwkExamples
drwxrwxr-x	2	adrian	adrian	4.0K	04/02/22	./games
drwxrwxr-x	27	adrian	adrian	60K	04/10/22	./gdrive
drwxrwxr-x	9	adrian	adrian	4.0K	04/02/22	./gems
drwxrwxr-x	2	adrian	adrian	4.0K	04/12/22	./json-files
drwxrwxr-x	7	adrian	adrian	4.0K	04/02/22	./lab5
-rw-rw-r--	1	adrian	adrian	4.1M	04/02/22	./lab5.zip
drwxrwxr-x	2	adrian	adrian	4.0K	04/11/22	./lab6
drwxrwxr-x	2	adrian	adrian	4.0K	04/02/22	./movies
drwxr-xr-x	3	adrian	adrian	4.0K	04/04/22	./Music
drwxr-xr-x	3	adrian	adrian	4.0K	03/05/22	./Pictures
drwxr-xr-x	2	adrian	adrian	4.0K	03/01/22	./Public
-rw-rw-r--	1	adrian	adrian	0	04/04/22	./song.mp3
drwxrwxr-x	3	adrian	adrian	4.0K	03/02/22	./Steam
drwxr-xr-x	2	adrian	adrian	4.0K	03/01/22	./Templates
-rw-rw-r--	1	adrian	adrian	108	04/09/22	./todo.md
-rwxr-wr--	1	adrian	adrian	511	04/09/22	./urls.sh
drwxr-xr-x	2	adrian	adrian	4.0K	04/07/22	./Videos
drwxrwxr-x	7	adrian	adrian	4.0K	03/21/22	./VirtualBox