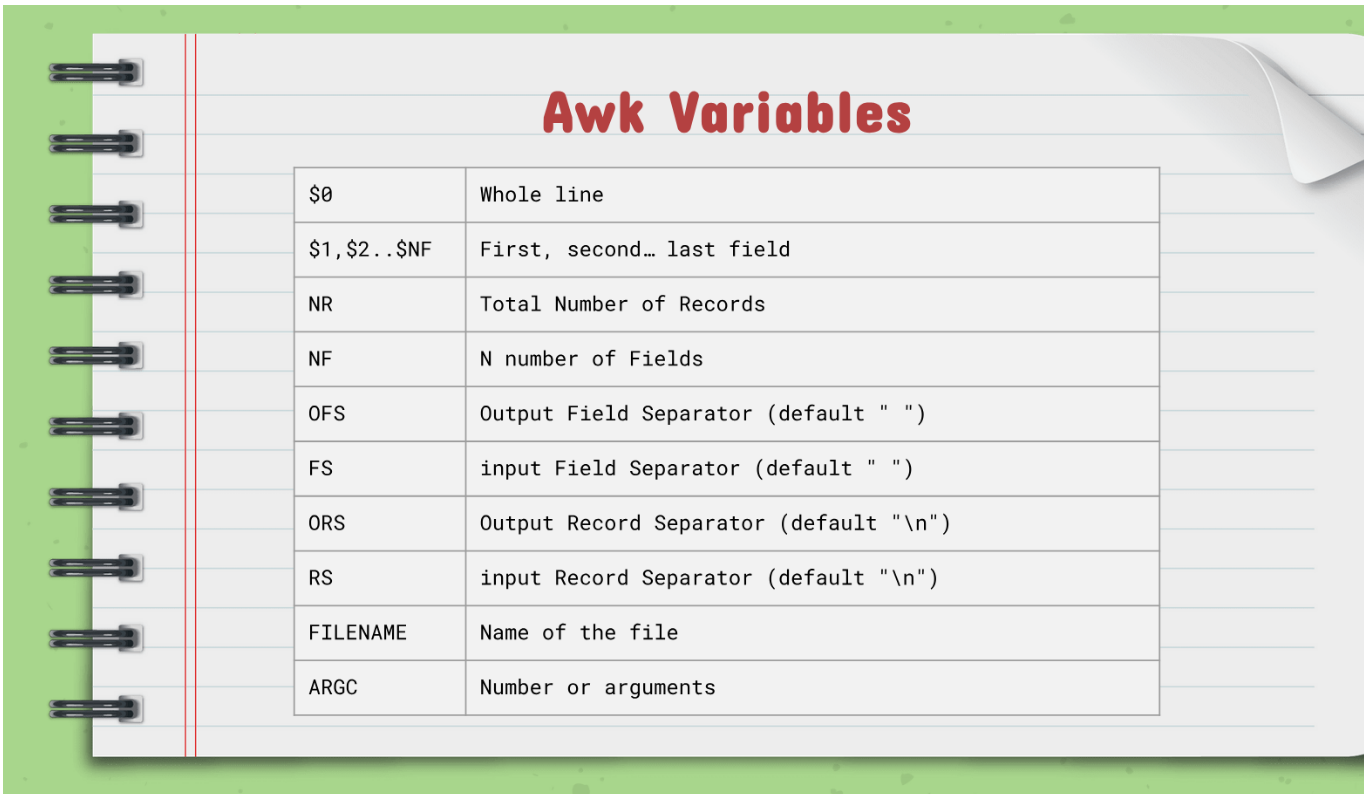
Notes 7

Handling Text Files Part #2

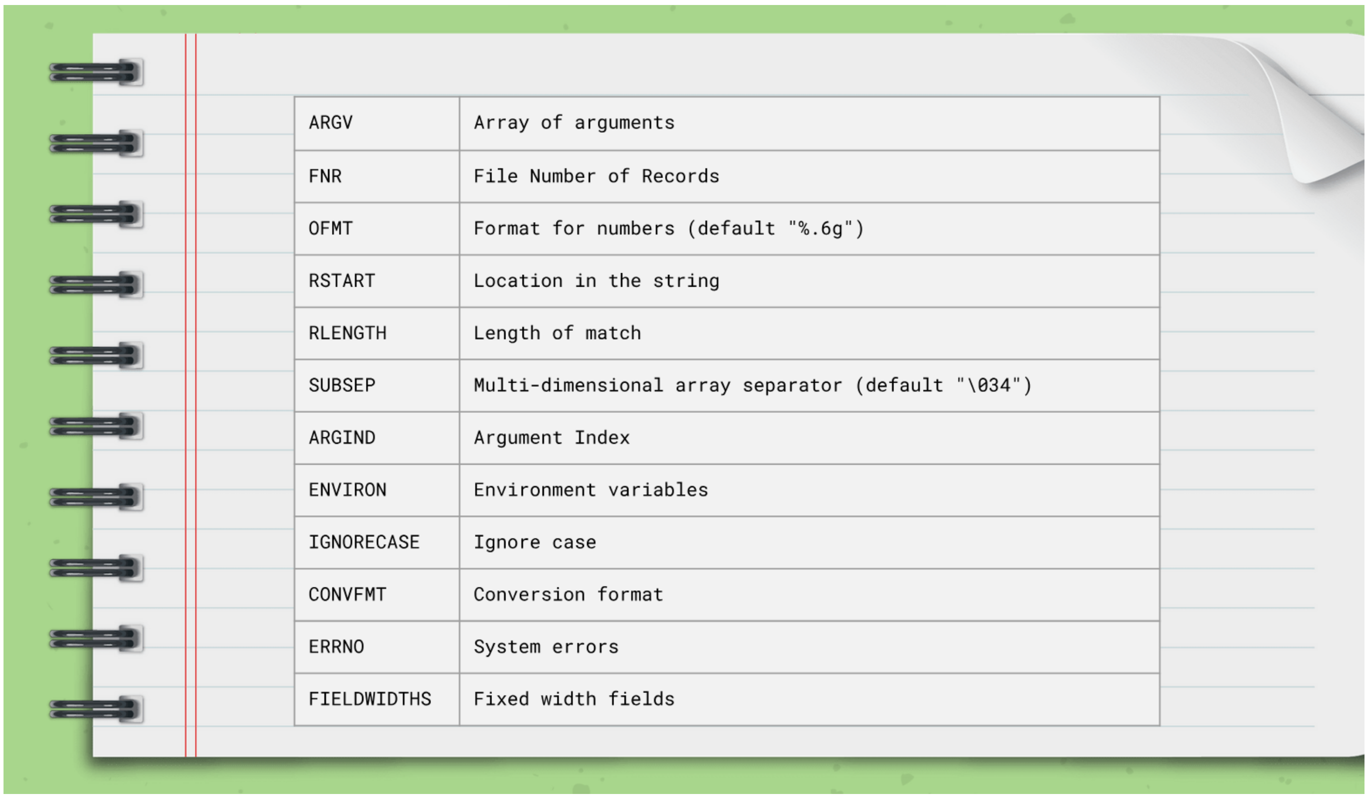
The awk Command:

- 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 Brain 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:

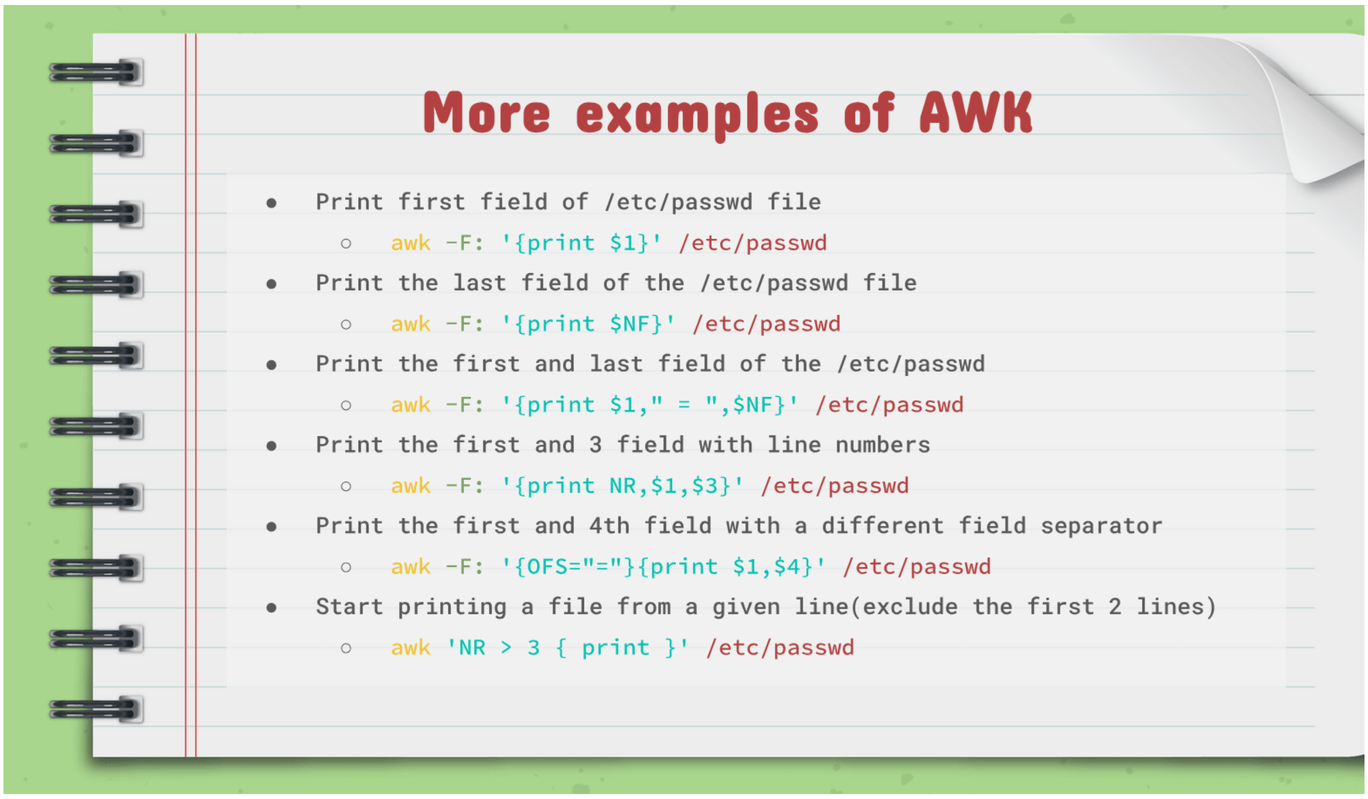


Awk Variables

$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 or arguments

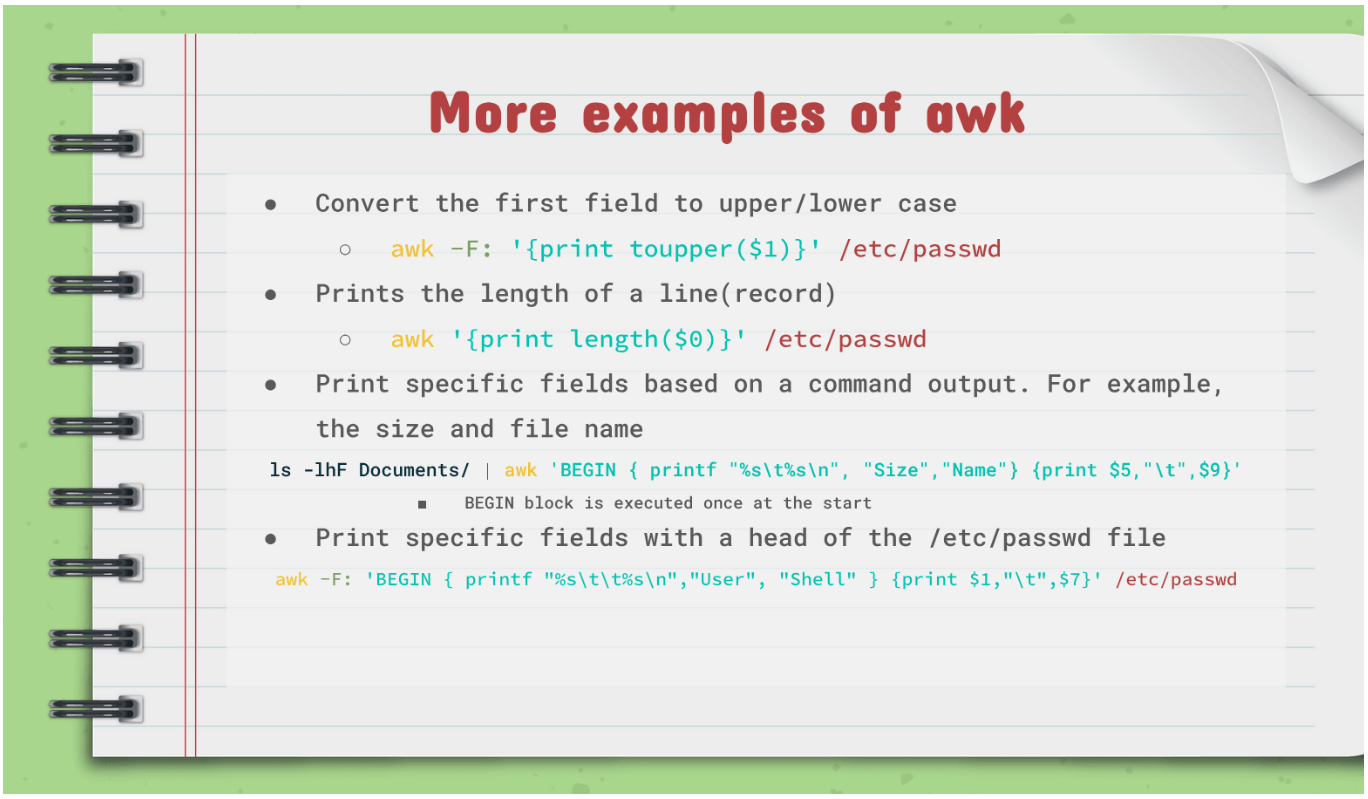


ARGV	Array of arguments
FNR	File Number of Records
OFMT	Format for numbers (default "%.6g")
RSTART	Location in the string
RLENGTH	Length of match
SUBSEP	Multi-dimensional array separator (default "\034")
ARGIND	Argument Index
ENVIRON	Environment variables
IGNORECASE	Ignore case
CONVFMT	Conversion format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

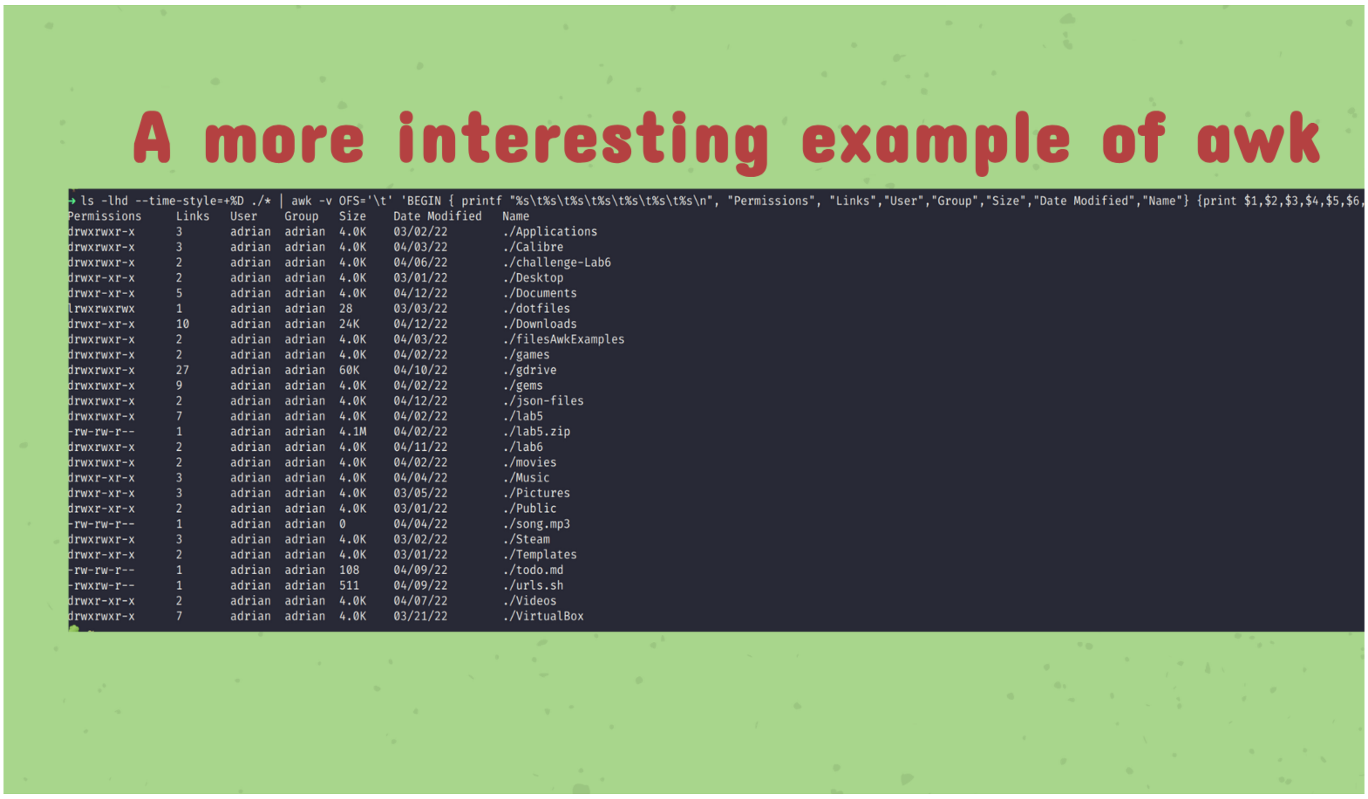
- Convert the first field to upper/lower case
    - `awk -F: '{print toupper($1)}' /etc/passwd`
- Prints the length of a line(record)
    - `awk '{print length($0)}' /etc/passwd`
- Print specific fields based on a command output. For example, the size and file name

```
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
```



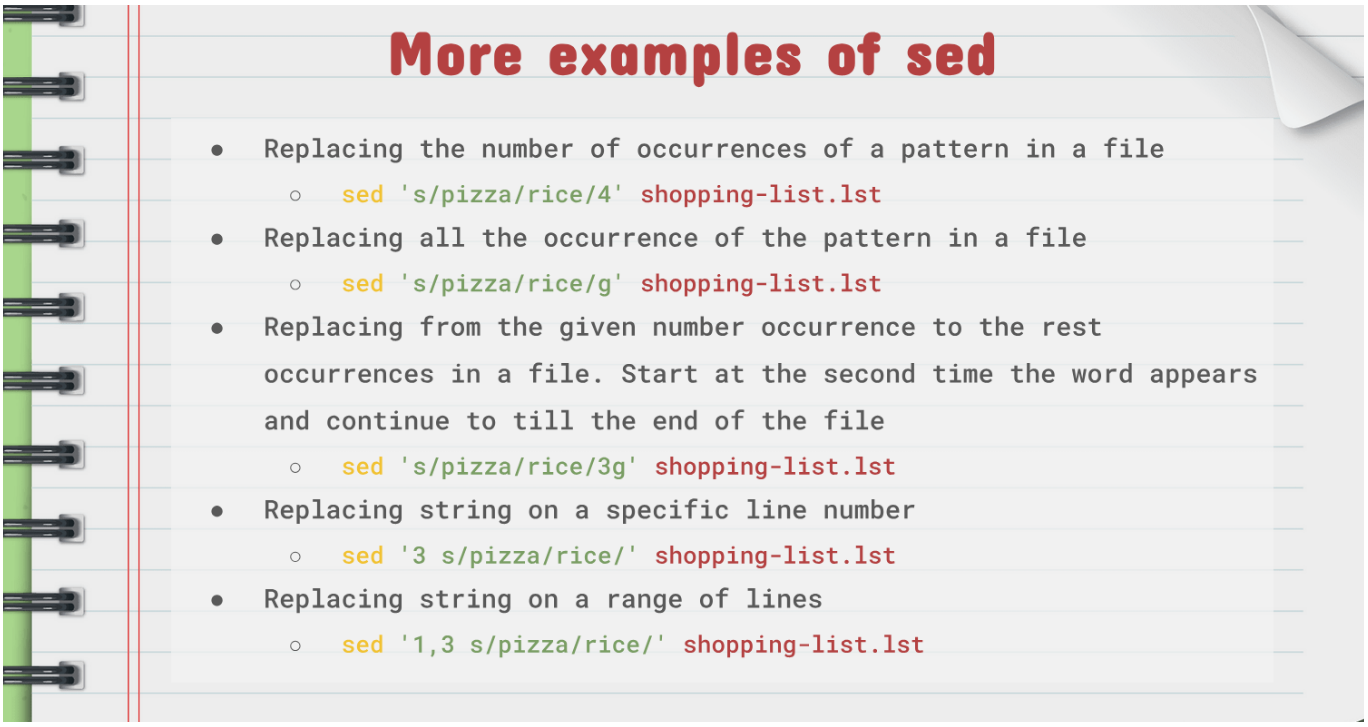
A more interesting example of awk

```
→ ls -lhd --time-style=+%D ./* | awk -v OFS='\t' 'BEGIN { printf "%s\t%s\t%s\t%s\t%s\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
drwxrwxr-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        ./filesAwkExamples
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
-rwxrw-r--      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
```

The sed Command

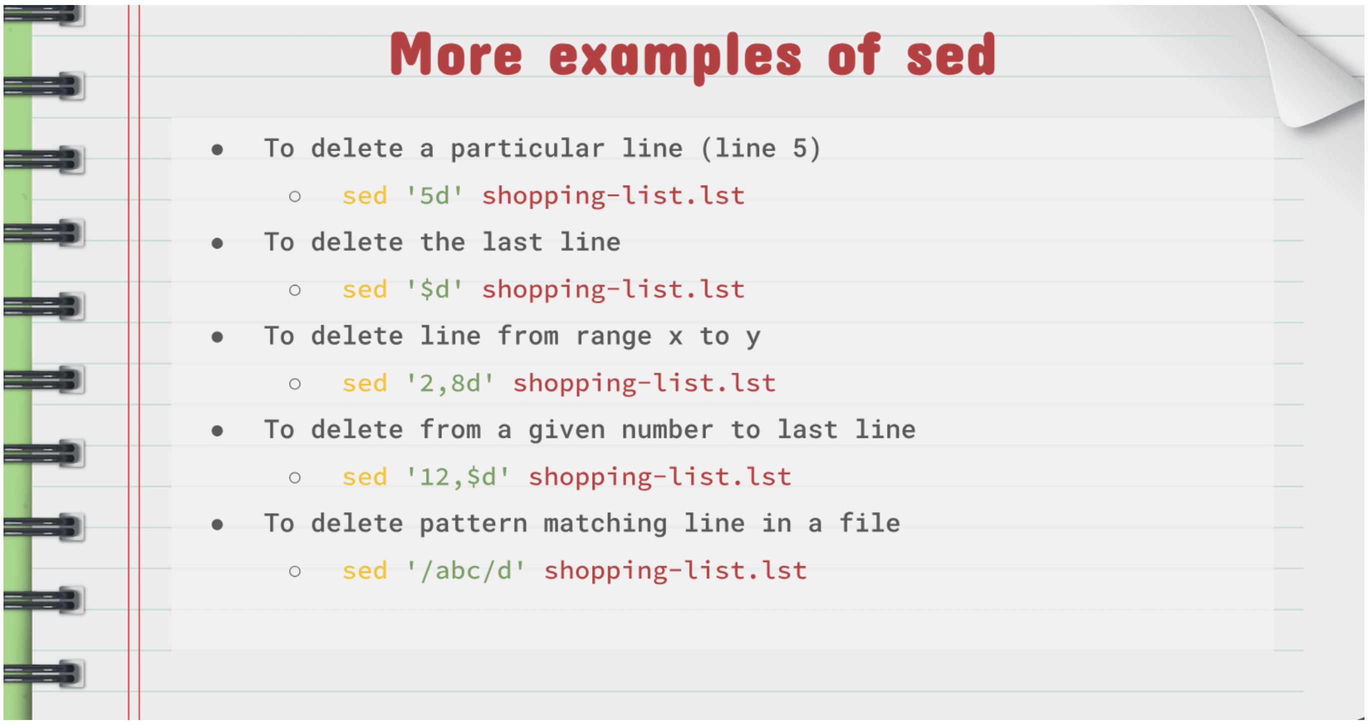
- Description:
    - SED is a stream editor that perform operations on files and standard output. For instance it can search, find and replace, inset, and deletion. By using SED you can edit files without opening them.
- Usage:
    - sed options + sed script + file

- Examples:
  - Replacing a string in given file (replace pizza for rice); sed 's/pizza/rice/' shopping-list.lst
- More Examples on SED:



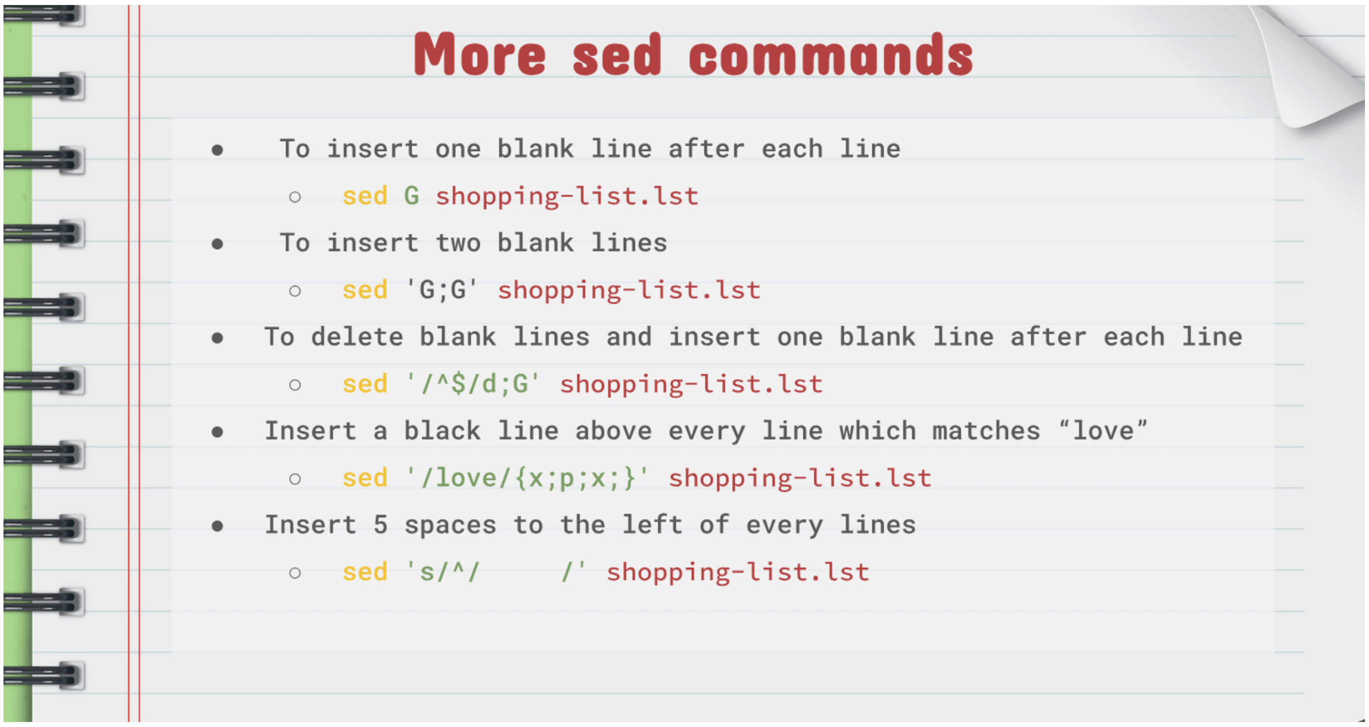
More examples of sed

- Replacing the number of occurrences of a pattern in a file
  - `sed 's/pizza/rice/4' shopping-list.lst`
- Replacing all the occurrence of the pattern in a file
  - `sed 's/pizza/rice/g' shopping-list.lst`
- Replacing from the given number occurrence to the rest occurrences in a file. Start at the second time the word appears and continue to till the end of the file
  - `sed 's/pizza/rice/3g' shopping-list.lst`
- Replacing string on a specific line number
  - `sed '3 s/pizza/rice/' shopping-list.lst`
- Replacing string on a range of lines
  - `sed '1,3 s/pizza/rice/' shopping-list.lst`



More examples of sed

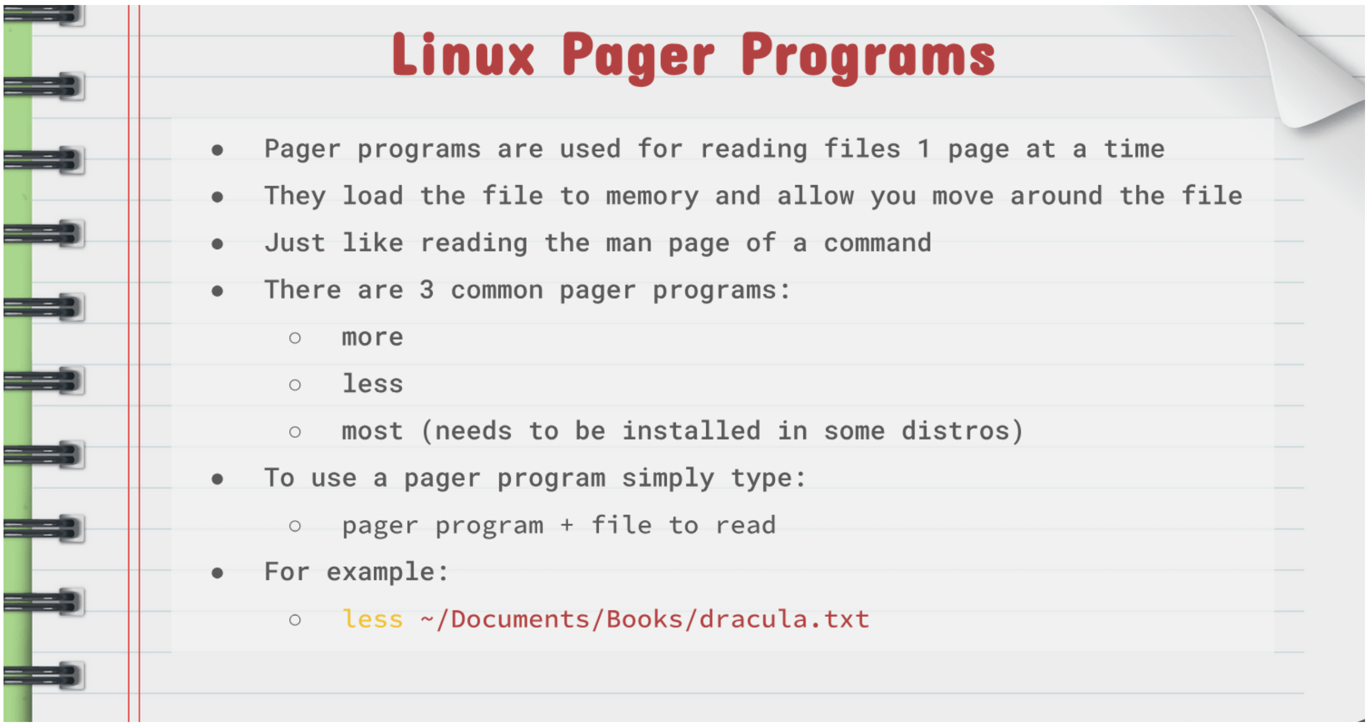
- To delete a particular line (line 5)
  - `sed '5d' shopping-list.lst`
- To delete the last line
  - `sed '$d' shopping-list.lst`
- To delete line from range x to y
  - `sed '2,8d' shopping-list.lst`
- To delete from a given number to last line
  - `sed '12,$d' shopping-list.lst`
- To delete pattern matching line in a file
  - `sed '/abc/d' shopping-list.lst`



More sed commands

- To insert one blank line after each line
  - `sed G shopping-list.lst`
- To insert two blank lines
  - `sed 'G;G' shopping-list.lst`
- To delete blank lines and insert one blank line after each line
  - `sed '/^$/d;G' shopping-list.lst`
- Insert a black line above every line which matches “love”
  - `sed '/love/{x;p;x;}' shopping-list.lst`
- Insert 5 spaces to the left of every lines
  - `sed 's/^/     /' shopping-list.lst`

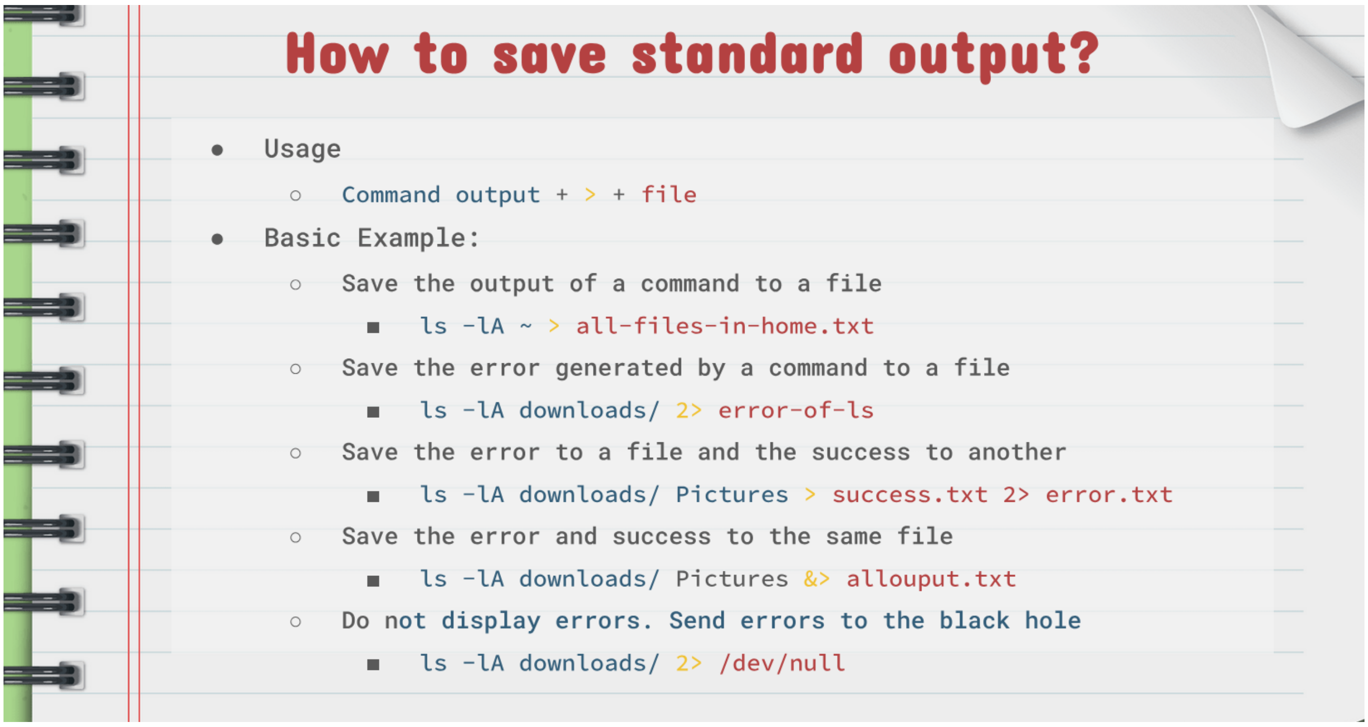
Less Command



Linux Pager Programs

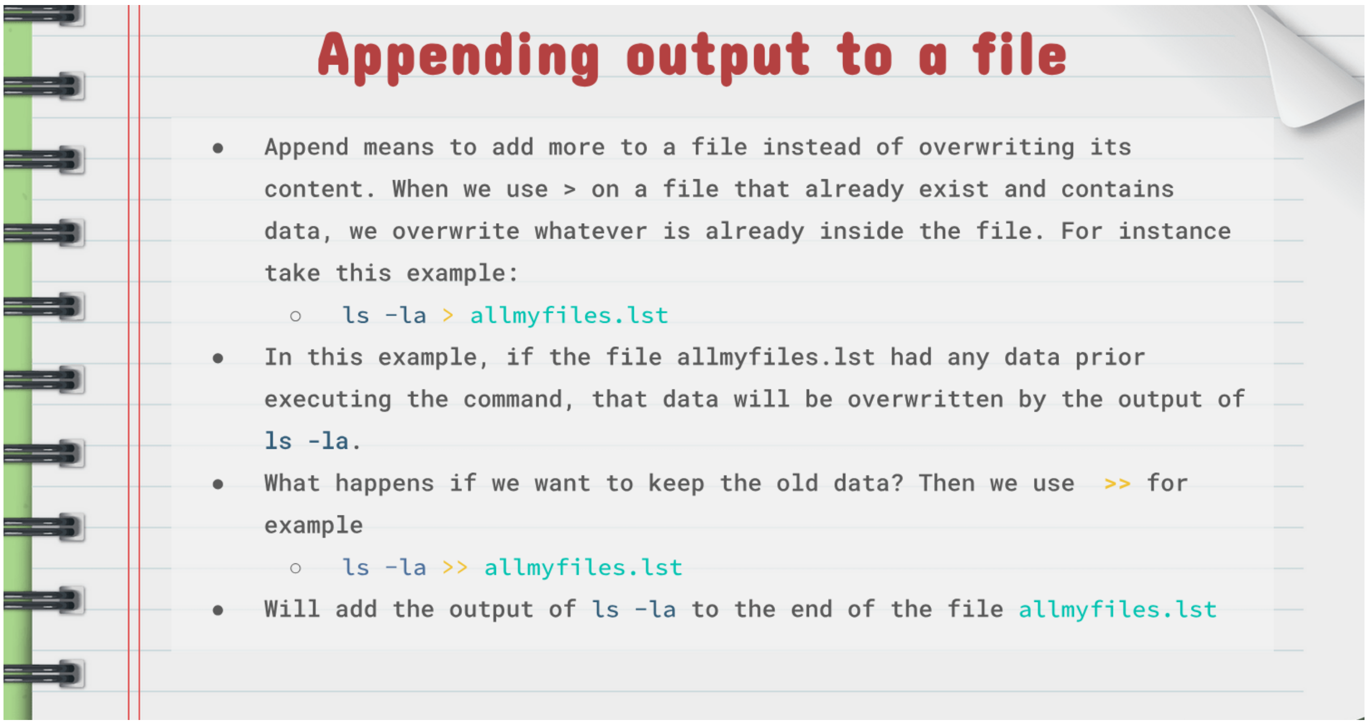
- Pager programs are used for reading files 1 page at a time
- They load the file to memory and allow you move around the file
- Just like reading the man page of a command
- There are 3 common pager programs:
  - more
  - less
  - most (needs to be installed in some distros)
- To use a pager program simply type:
  - pager program + file to read
- For example:
  - `less ~/Documents/Books/dracula.txt`

More Examples of >, >>, |



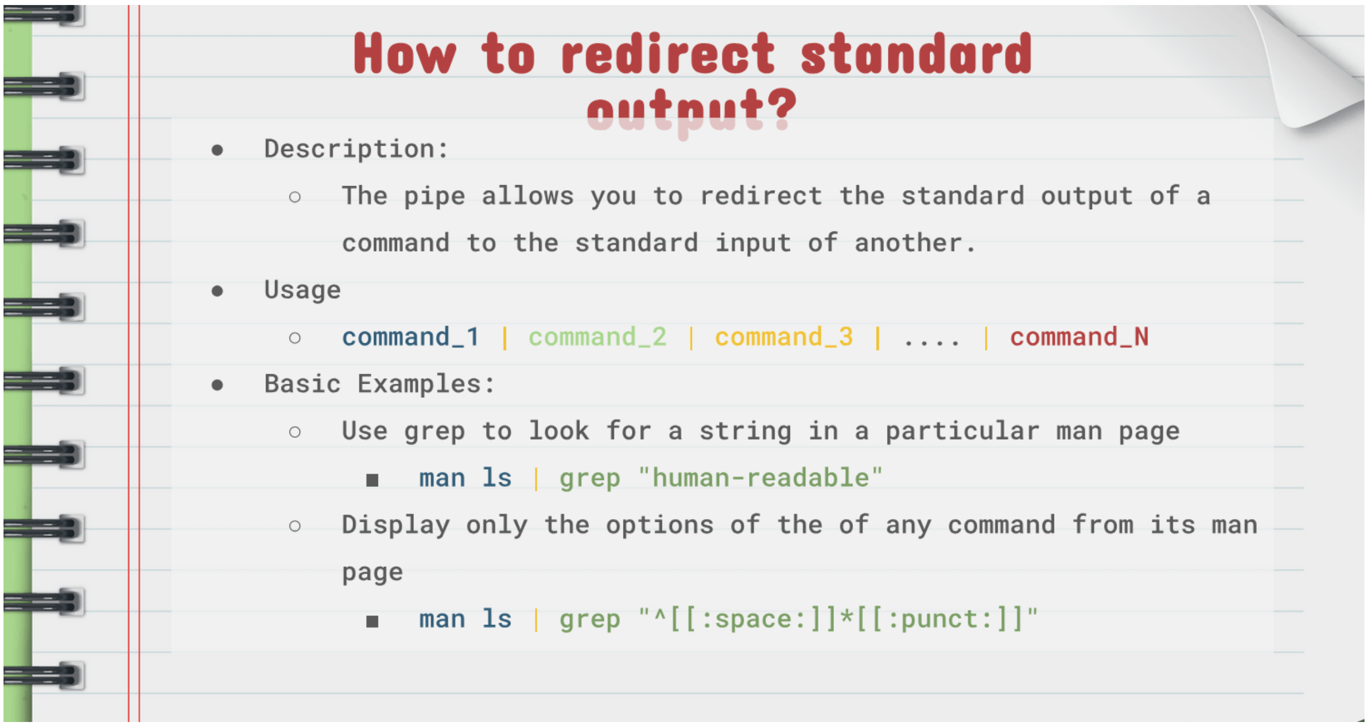
How to save standard output?

- Usage
  - `Command output + > + file`
- Basic Example:
  - Save the output of a command to a file
    - `ls -lA ~ > all-files-in-home.txt`
  - Save the error generated by a command to a file
    - `ls -lA downloads/ 2> error-of-ls`
  - Save the error to a file and the success to another
    - `ls -lA downloads/ Pictures > success.txt 2> error.txt`
  - Save the error and success to the same file
    - `ls -lA downloads/ Pictures &> allouput.txt`
  - **Do not display errors. Send errors to the black hole**
    - `ls -lA downloads/ 2> /dev/null`



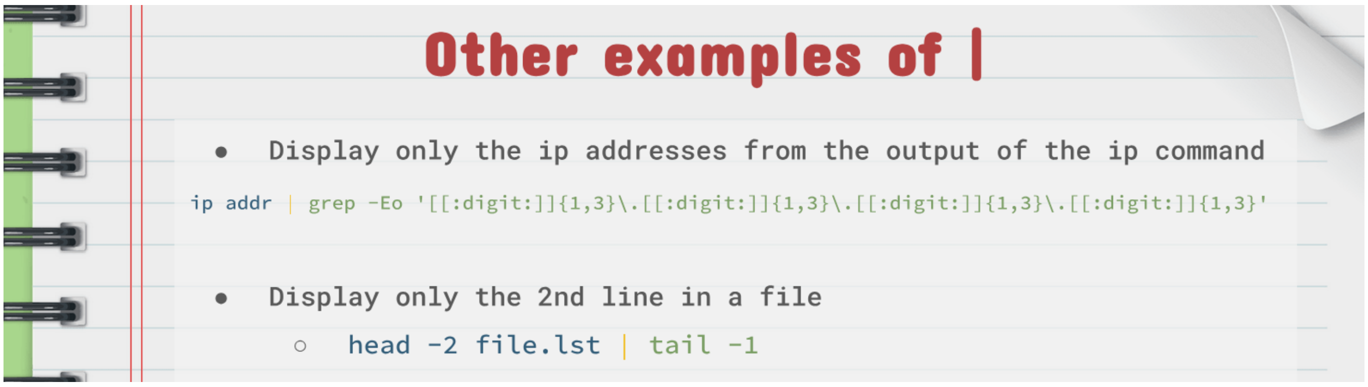
Appending output to a file

- Append means to add more to a file instead of overwriting its content. When we use > on a file that already exist and contains data, we overwrite whatever is already inside the file. For instance take this example:
  - `ls -la > allmyfiles.lst`
- In this example, if the file allmyfiles.lst had any data prior executing the command, that data will be overwritten by the output of **ls -la**.
- What happens if we want to keep the old data? Then we use `>>` for example
  - `ls -la >> allmyfiles.lst`
- Will add the output of `ls -la` to the end of the file `allmyfiles.lst`



How to redirect standard output?

- Description:
  - The pipe allows you to redirect the standard output of a command to the standard input of another.
- Usage
  - `command_1 | command_2 | command_3 | .... | command_N`
- Basic Examples:
  - Use grep to look for a string in a particular man page
    - `man ls | grep "human-readable"`
  - Display only the options of the of any command from its man page
    - `man ls | grep "^[[:space:]]*[[:punct:]]"`



Other examples of |

- Display only the ip addresses from the output of the ip command

```
ip addr | grep -Eo '[[:digit:]]{1,3}\.[[:digit:]]{1,3}\.[[:digit:]]{1,3}\.[[:digit:]]{1,3}'
```

- Display only the 2nd line in a file
  - `head -2 file.lst | tail -1`