# Final Assignment

# Question 1

## awk

**Description** `awk` is a scripting language used for processing and displaying text. Awk can work from standard output or a text file.

**Syntax:** `awk` + `options` + `{awk command}` + `file` + `file to save (optional)`

**Examples** Printing the first column of every line of a file- `awk '{print $1}' ~/Documents/games.txt` Printing the first field of /etc/passwd- `awk -F: '{print $1}' /etc/passwd` Printing the first and last field of /etc/passwd- `awk -F: '{print $1, " = ", NF}' /etc/passwd`

## cat

**Description** `cat` is the command used for displaying the content of a file.

**Syntax:** `cat` + `options` + `file(s) to display`

**Examples** Displaying the content of a file in the present working directory- `cat groceries.txt` Displaying the content of a file with line numbers- `cat -n groceries.txt` Displaying the content of a file with line numbers without empty lines- `cat -b groceries.txt`

## tac

**Description** `tac` displays the content of a file in reverse order.

**Syntax:** `tac` + `options` + `file(s) to display`

**Examples** Displaying the content of a file in the present working directory in reverse order- `tac groceries.txt` Displaying the content of a file using absolute path in reverse order- `tac ~/Documents/groceries.txt` Displaying the content of a file with line numbers without empty lines in reverse order- `tac -b groceries.txt`

## cp

**Description** `cp` is the command used for copying files/directories from their source to a destination.

**Syntax:** `cp` + `files to copy` + `destination`

**Examples** Copying Documents/wallpapers.zip to the Pictures directory- `cp Documents/wallpapers.zip Pictures/` Copying the Pictures directory to the Documents directory- `cp -r ~/Pictures/photos ~/Documents/` Copying the content of a directory to another directory- `cp Downloads/wallpapers/* ~/Pictures/

## mv

**Description** `mv` moves and renames a file/directory

**Syntax:** `mv` + `source` + `destination`

**Examples** Moving a file from a directory to another using relative path- `mv Downloads/rock.py Documents/` Moving and renaming a file at the same time- `mv Downloads/rock.py Documents/stone.py` Renaming a file- `mv snap pans`

## cut

**Description** `cut` cuts a specific section of each line of a file and displays it on screen.

**Syntax:** `cut` + `option` + `files`

**Examples** Displays a list of all the users in the system- `cut -d ':' -f1 /etc/passwd` Cuts a range of bytes per line- `cut -b 1-5 usernames.txt` Displays a list of all the users in the system, including their login shell- `cut -d ':' -f1,7 /etc/passwd`

## grep

**Description** `grep` searches for text in a file. It searches on a line by line basis.

**Syntax:** `grep` + `option` + `search criteria` + `file(s)`

**Examples** Searches for the word "book" in a file- `grep 'book' ~/Documents/book.txt` Searches for the word "book" in a file with case insensitivity- `grep -i 'book' ~/Documents/book.txt` Searches for all the lines in a file that exclude the word "book"- `grep -v 'book' ~/Documents/book.txt`

## head

**Description** `head` displays the first number of lines of a file. By default it prints the first 10 lines of a file.

**Syntax:** `head` + `option` + `file(s)`

**Examples** Displays the first 10 lines of book.txt- `head ~/Documents/book.txt` Displays the first 15 lines of book.txt- `head -15 ~/Documents/book.txt` Displays the first line of book.txt- `head -1 ~/Documents/book.txt`

## tail

**Description** `tail` displays the last number of lines of a file. By default it print the last 10 lines of a file

**Syntax:** `tail` + `option` + `file(s)`

**Examples** Displays the last 10 lines of book.txt- `tail ~/Documents/book.txt` Displays the last 15 lines of book.txt- `tail -15 ~/Documents/book.txt` Displays the last line of book.txt- `tail -1 ~/Documents/book.txt`

## mkdir

**Description** `mkdir` is used for creating either a single directory or multiple directories.

**Syntax:** `mkdir` + `name of directory`

**Examples** Creating the "Bread" directory in pwd- `mkdir Bread` Creating the "Bread" and "Pan" directories in pwd- `mkdir Bread Pan` Creating the "Bread" directory in a different directory using absolute path- `mkdir ~/Documents/Bread`

## touch

**Description** `touch` is used to create files

**Syntax:** `touch` + `name of file`

**Examples** Creating the "hamburger.txt" file in pwd- `touch hamburger.txt` Creating the "hamburger.txt" and "cheeseburger.docx" files in pwd- `touch hamburger.txt cheeseburger.docx` Creating a file a space in its name- `touch "ham burger.txt"`

## ls

**Description** `ls` displays all of the files inside a given directory. It will display all of the files in the pwd if a directory is not specified.

**Syntax:** `ls` + `directory to display`

**Examples** Displays the files in the pwd- `ls` Displays the files in the pwd sorted by file size- `ls -S` Long listing the files in the `~/Documents` directory with human readable file size without the group nor owner and sorted by file size- `ls -lhgGS ~/Documents`

## man

**Description**

`man` opens the manual page of a command that explains what it does. To exit the man page, press the letter `q`.

**Syntax:** `man` + `command`

**Examples** Opens the manual page of a command- `man ls` Showing all of the available manual pages of a command- `man -a ls` Searching for a man page for a given word- `man -k file`

## tr

**Description** `tr` translates or deletes characters from standard output.

**Syntax:** `standard output| tr` + `option` + `set` + `set`

**Examples** Translating one character to another- `cat games.txt| tr '.' 'D'` Translating spaces into tabs- `cat games.txt| tr "[:space:]" '\t'` Translating tabs to space- `cat games.txt| tr -s "[:space:]" ' '`

## tree

**Description** `tree` displays a tree of all the files and directories in a given directory.

**Syntax:** `tree` + `directory to display`

**Examples** Displays a tree of the pwd- `tree` Displays a tree of a directory using relative path- `Documents/Books` Displays a tree of a directory using absolute path- `~/Documents/Books`

# Question 2

**How to work with multiple terminals open?**

To open another terminal, press `Ctrl` +`shift` + `t` or in Tilux, you can right click on the terminal and choose between a horizontal or vertical split. You can click on each of the terminals to work on a specific one. For example, you can open the man page of a command on one terminal and execute commands on another terminal.

**How to work with manual pages?**

You can open the man page of a command by inputting `man` + `command`. To look for a specific man page of a command, input `man` + `page number` + `command`. To search for a man page for a given word or phrase, input `man -k` + `word or phrase`.

**How to parse (search) for specific words in the manual page**

You can use grep and output redirection to search for a specific word in the man page. For example: `man ls | grep "human"`

**How to redirect output (> and |)**

To save output, input `command output` + `>` + `file`. To redirect the standard output of a file to the standard input of another, input `command_1 | command_2`.

**How to append the output of a command to a file**

To append the output of a command to a file, input `command output` + `>` + `file`. To save the output to a file and keep the old data, input `command output` + `>>` + `file`.

**How to use wildcards**

The `*` wildcard matches everything and nothing and matches any number of characters.

For example: `ls *.docx` lists all files starting with .docx

The `?` wildcard matches precisely one character.

For example: `ls .??*` lists all hidden characters by matching all files that start with a dot or two dots and have any character after it.

The `[]` wildcard matches a range of characters. Using an exclamation mark reverses the match to not show those characters.

For example: `ls f[aeiou]*` lists all the files with vowels after the letter f `ls f[!aeiou]*` lists all the files without vowels after the letter f

**For copying and moving multiple files at the same time**

To copy or move multiple files at the same time, you use the `mv` or `cp` commands + the `*` wildcard to move or copy multiple files at the same time.

**How to use brace expansion** To use brace expansion input `command` + `{}`. Brace expansion can be used to create multiple files in a single command.

For example: `touch file{A..Z}.txt` creates multiple .txt files starting from A to Z.

**For creating entire directory structures in a single command** To create entire directory structures in a single command, input `mkdir -p file1/{file2,file2}/{file3,file3}`