Question 1

awk

Description: * Is a scripting language used for processing and displaying text * Syntax/Formula: * `awk + option + [awk command] + file + file to save (optional)` * Examples: * Print the first column of every line of a file: * `awk '{print $1} ~/Downloads/games/minecraft.exe` * Print the last field of the /Downloads/game/minecraft.exe * `awk -F '[print $NF]' ~/Downloads/games/minecraft.exe` * Print the first and 3 field with line numbers: * `awk -F: '{print NR. $1, $3}' /books/mystery`

cat

```
* Description:
  * Used for displaying the content of a file
* Syntax/Formula:
    * `cat + option + files(s) to display`
* Examples:
  * How to see the content of a file:
    * `cat /etc/passwd`
  * How to see the content of a file with line numbers
    * `cat -n /etc`
  * How to see the content of a file with ending line character
    * `cat -E file`
```

cp

- Description:
  - Used copies files/directories from a source to a destination
  - Syntax/Formula:
    - `cp + files to copy + destination`
  - Examples:
    - Copy files to a directory:
      - `cp games/minecraft.exe michael/gamefiles`
    - Copying directories using -r option:
      - `cp -r ~/games/minecraft.exe ~/michael/`
    - Copying contents of a directory to another directory
      - `cp Downloads/wallpapers/* ~/Pictures/`

cut

Description: * Used to extract a specific section of each line of a file and display it to the screen * Syntax/Formula: * `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 /etc/passwd '` * To create a file with a space in its name: * `cut -d -f1,6 /etc/passwd`

grep

Description: * Used to search text in given file * Syntax/Formula: * `grep + option + search criteria + file(s)` * Examples: * Search any line that contains the word "potato": * `grep 'potato' ~/Documents/food.txt` * Search for a given strings inside files in a given directory: * `grep -IR 'drac' /books/` * Search and match only the word: * `grep -o dracula ~/Documents/Books/dracula.txt`

head

- Description: * Displays the top N number of lines of a given file
  - Syntax/Formula:
    - `head + option + files(s)`
  - Examples:
    - Display the first 10 lines of a file:
      - `head dracula.txt`
    - Display the first 20 lines of a file:
      - `head -20 michael/list.txt`

ls

- Description: * Used for listing the content of a given directory or the file/directory itself
  - Syntax/Formula:
    - `ls + option + directory to list`
  - Examples:
    - List the content of the present working directory:
      - `ls`
    - List all the files inside a given directory:
      - `ls -a ~/Pictures`
    - Long list all the files inside a given directory:
      - `ls -l ~/Pictures`

man

- Description: * Are documentation files that describe Linux Shell commands, executable programs,etc * `man + command`
  - Examples:
    - To view the manual of the ls command:
      - `man ls`
    - Open the man page of the passwd command
      - `man passwd`
    - Searches for a man page for a given word or regular expression or phrase.
      - `man -k file`

mkdir

- Description:
  - Used for creating a single directory or multiple directories
  - Syntax/Formula:

        - `mkdir + the name of the directory`
    - Examples:
        - Creating a directory in the present working directory:
            - `mkdir michael`
        - Creating a directory in a different directory using absolute path
            - `mkdir ~/michael/mypictures`
        - Creating multiple directories
            - `mkdir michael/mypictures michael/documents`

mv

- Description:
    - Used to move and rename directories
    - Syntax/Formula:
        - `mv + source + destination`
    - Examples:
        - To move a file from a directory to another relative path:
            - `mv Pictures/beach.png Michael/`
        - How to see the content of a file with line numbers
            - `mv Pictures/beach png games/ ~/Documents/`
        - To rename a file
            - `mv Pictures/beach.png Pictures/playa.png`

tac

- Description: * Used for displaying the content of a file in reverse order
    - Syntax/Formula:
        - `tac + option + file(s) to display`
    - Examples:
        - Display the content of a file located in the pwd:
            - `tac list.txt`
            - `tac michael.txt`
        - Display the content of a file using absolute path
            - `tac ~/Documents/list.txt`

tail

Description: * Used for displaying the last N number of lines of a given file * Syntax/Formula: * `tail + option + file` * Examples: * Display the last 10 lines of a file: * `tail dracula.txt` * Display the last 5 lines of a file * `touch michael.txt dog.txt` * To create a file with a space in its name: * `touch michael's gym plan.txt`

touch

- Description:
    - Used for creating files
    - Syntax/Formula:
        - `touch + file name`

    - Examples:
        - To create a file called michael:
            - `touch michael`
        - To create several files
            - `touch michael.txt dog.txt`
        - To create a file with a space in its name:
            - `touch michael's gym plan.txt`

tr

Description: * Used for translating or deleting characters from standard * Syntax/Formula: * `standard output | tr + option + set + set` * Examples: * Translate one character to another: * `cat file.txt | tr '.' ','` * Translate white space into tabs: * `cat program.py | tr "[:space"] '/t` * Translate tabs into space: * `cat file.py | tr -s "[:space:]" ' '`

tree

- Description: * Displays directory paths and files in each subdirectory
    - Syntax/Formula:
        - `touch + file name`
    - Examples:
        - displays path
            - `tree Downloads` | |---minecraft.exe | |---roblox.exe | |---pacman.exe`

touch

- Description:
    - Displays directory paths and files in each subdirectory
    - Syntax/Formula:
        - `touch + file name`
    - Examples:
        - To create a file called michael:
            - `touch michael`
        - To create several files
            - `touch michael.txt dog.txt`
        - To create a file with a space in its name:
            - `touch michael's gym plan.txt`

Question 2

- How to work with multiple terminals open? Working with multiple terminals open can be useful when you need to perform different tasks simultaneously or keep various processes running. Opening multiple terminals and organizing terminal windows can make switch between them easily.
- How to work with manual pages? Working with manual pages, also known as man pages, is a convenient way to access documentation and information about commands, utilities, and system functions in Unix-like operating systems.

- How to parse (search) for specific words in the manual page Open the manual page by using the command man. Then enter 'man ls' in the terminal which will allow you to go into search mode.
- How to redirect output (> and |) By using the greater-than symbol (>) and the pipe symbol (|), you can redirect the output of a command to a file or send it as input to another command. These techniques are valuable for saving or processing command output efficiently.
- How to append the output of a command to a file
    - To append the output of a command to a file, you can use the output redirection operator >>. This operator appends the output to the specified file instead of overwriting it. Here's how you can do it:
- How to use wildcards Wildcards are special characters used in command-line interfaces to represent patterns of filenames or arguments. They are useful for selecting multiple files or directories that match a pattern.

1. Asterick(*): THe asterisk represent any sequence of characters, inclufing no characters at all. It is used to march multiple characters in a filename or argument. Ex. ls *.txt > matches all files ending with ".txt" in the current directory
2. Question mark (?): The question mark matches any single character. It is useful when you want to specify a single character in a filename or argument. EX. ls file?.txt > matches files with "file" followed by any single character and ending with ".txt".
3. Brackets ([]):Brackets are used to match any single character from a specific set of characters. Ex. ls [abc]*.txt > matches files starting with either "a", "b", or "c" and ending with ".txt".
4. Curly braces ({}): Curly braces allow you to specify multiple options or combinations. This is known as brace expansion Ex. mv {old,new}_file.txt > renames files "old_file.txt" to "new_file.txt"
   cp file{1,2,3}.txt destination/ > copies files "file1.txt", "file2.txt", and "file3.txt" to the destination directory

- How to use brace expansion
    - Brace expansion is a feature in many Unix-like shells that allows you to generate multiple options or combinations using curly braces ({}). It's a powerful tool for quickly creating lists of filenames, arguments, or any other sequences of characters.