Notes 6: Bash Shell

Explain how to use each of the following commands:

awk

Awk is a powerful text-processing tool in UnixLinux environments .it operates on a per -line basis, making it great ofr processing structured text file, such as CSV,tab- separated values,and more.also the awk command help you print a list of all the users in your system.

- example `awf -F ',' '{print $1}' data.txt"`*

this options specifies the fields separator as ",". it tells `awk` to treat`` "," as the delimiter bewtween fields.

---

sed

to use the 'sed' command in Linux, you can follow this structure sed 's/old_pattern/new_pattern/' it cna search, find and replace, insert, and deletion, by using SED you can edit files without opening them.

- example `sed -i 's/sed/awk/g' varios.txt`

  *This option tells `sed` to edit files in places `s/sed/awk/g`:this is the subsitution commandit searches for the stirng `sed` and replaces it wiht `awk`.the `g` flag stands for `global`,meaning it replaces all occurrences on each line.

---

less

the 'less' allow you to view text files, but unlike you to view text files,but unlike 'more', it supports both forward and backward navigation, as well searching with the file.

- Here's how you can use it:
  `less example.txt`
  Once you've opned the file wiht `less` ,you can navigate throught the content using the following `key:` Down Arrow or j: Move down one line at a time.
  Page Up or b: Move up one page at a time.
  Page Down or spacebar: Move down one page at a time.

---

alias

aliases are shortcuts or alternate names for names for commands.they are particulary useful for saving time and typing effort by defining commands.

- here's how you can use examples `alias 11='ls a-lF'`
- Aliases are particularly useful for simplifying commonly used commands. or they can be defined in your shell's configurations file (e.g., 'bashrc, .bash_profile, ,zhrc) to make them persistent across terminal sessions.

Explain with examples how to use

- *>
  The '>' symbol is a redirection operator used to redirect the ouput of a commands to a file or another command. it's a fundamental feature of the shell that allows you to manage commannd output effciently.
- Here"s how it work `command1 > output .txt | commannd2`
- `commadn > output .txt`
  The double greater than symbol `(>>)` appends the output to the end of the file instead of overwriting it.
  Redirect Input `(<):` The less than symbol (<) is used to redirect input from a file to a command.

- *>> symbol is a redirection operator used to append the output of a command to a file, rather than overwriting the file. It works similarly to the > operator, but instead of creating a new file or overwriting an existing one, it appends the output to the end of the file. Here's how it works: `command >> output.tx`
- `echo hello, world!" >> output.txt` this
- This command appends the text "Hello, world!" to the file output.txt. If output.txt already exists, the text will be added to the end of the file. If output.txt does not exist, it will be created and the text will be written to it.*

- I |. In Linux, the vertical bar symbol | is known as the pipe operator, and it's used for connecting multiple commands together, allowing the output of one command to serve as the input of another command. This is referred to as "piping" in Linux. here is a exmaple it work `command 1 | commands 2`
- `ifconfig`
- *In this command, "ifconfig" is used to display information about network interfaces on the system. The "I" in "ifconfig" stands for "interface", and the command is commonly used for network configuration and troubleshooting*