React-Less

1. `cat` **(Concatenate)**

- Displays the contents of a file
- Example: `cat filename.txt`

2. `head` **and** `tail`

- Display the first or last few lines of a file
  - `head`: displays the first 10 lines by default, but can be customized with `-n` option (e.g., `head -5 filename.txt`)
  - `tail`: displays the last 10 lines by default, but can be customized with `-n` option (e.g., `tail -3 filename.txt`)

3. `sed` **(Stream Editor)**

- Edits text in a file using regular expressions
- Example: `sed 's/old_text/new_text/g' filename.txt` (replaces "old_text" with "new_text")

4. `grep` **(Global Regular Expression Print)**

- Searches for a pattern within a file and prints the matching lines
- Example: `grep keyword filename.txt`

5. `cut`

- Extracts specific fields from a file based on delimiters (e.g., space, comma)
- Example: `cut -d ',' -f 1 filename.csv` (extracts first field using comma as delimiter)

6. `sort` **and** `uniq`

- Sorts the lines of a file alphabetically or numerically
  - `sort`: sorts in ascending order by default; use `-r` option for descending order (e.g., `sort -r filename.txt`)
  - `uniq`: removes duplicate lines from a sorted file

7. `awk`

- A powerful tool for text processing that can perform actions based on conditions
- Example: `awk '{print $1}' filename.txt` (prints the first field of each line)

These are just a few examples, but they cover some of the most common use cases for text processing in Linux.

Example Use Case: Suppose you have a file named `students.csv` with the following contents:

```
 John,20,Male
Jane,21,Female
Bob,19,Male
Alice,22,Female
```

You can use the above commands to perform various operations on this file. For instance:

- Display the first three lines: `head -3 students.csv`
- Replace "Male" with "Boy": `sed 's/Male/Boy/g' students.csv`
- Sort the file by age in descending order: `sort -r -k 2 students.csv`

Curated by Brajesh Kumar