Notes 7

Explain how to use each of the following commands:

cat

Definition:

The cat command is used for displaying the content of a file.

Usage:

`cat` + `option` + `file to display`

Examples:

1. Display the content of the cereal.csv file: `cat cereal.csv`
2. Display the content of the cereal.csv file with line numbers: `cat -n cereal.csv`
3. Display the content of a file using absolute path: `cat ~/Documents/Csv/cereal.csv`
4. Display the content of the cereal.csv file showing non-printing characters: `cat -v cereal.csv`

---

tac

Definition:

The tac command is used for displaying the content of a file in reverse order.

Usage:

`tac` + `option` + `file to display`

Examples:

1. Display the content of the cereal.csv file in reverse: `tac cereal.csv`
2. Display the content of the cereal.csv file in reverse using absolute path: `tac ~/Documents/Csv/cereal.csv`

---

head

Definition:

The head command displays the top N number of lines of a given file. It prints the first 10 lines by default.

Usage:

`head` + `option` + `file(s)`

Examples:

1. Display the first 10 ines of the dracula.txt file: `head dracula.txt`
2. Display the first 5 lines of the dracula.txt file: `head -5 dracula.txt`
3. Display the first line of multiple files using wildcards: `head -n 1 *.csv *.py`
4. Display the first 5 lines of multiple files: `head -5 dracula.txt bible.txt war-and-peace.txt`

---

tail

Definition:

The tail command displays the last N number of lines of a given file. It prints the last 10 lines by default.

Usage:

`tail` + `option` + `file`

Examples:

1. Display the last 10 lines of the bible.txt file: `tail bible.txt`
2. Display the last 5 lines of the bible.txt file: `tail -5 bible.txt`
3. Display the last 5 lines of multiple files: `tail -n 5 dracula.txt bible.ttx war-and-peace.txt`
4. Display the last line of the bible.txt file: `tail -n1 bible.txt`

---

cut

Definition:

The cut command is used to extract a specific section of each line of a file and display it on the screen.

Usage:

`cut` + `option` + `file(s)`

Examples:

1. Display a list of all the users in your system: `cut -d ':' -f1 /etc/passwd`
2. Display a lis t of all the users in your system with their login shell: `cut -d ':' -f1,7 /etc/passwd`
3. Cut a range of bytes per line: `cut -b 1-5 username.txt`
4. Cut a file excluding a given field: `cut -d ',' --complement -s -f3 users.txt`

---

sort

Definition:

The sort command is used for sorting files, the sort command supports sorting: alphabetically, in reverse order, by number, and by month.

Usage:

`sort` + `option` + `file`

Examples:

1. Sort a file: `sort users.lst`
2. Sort a file in reverse order: `sort -r users.txt`
3. Sort by column number: `sort -k2 users.txt`
4. Sort a file with numeric data: `sort -n phones.txt`

---

wc

Definition:

The wc command is used for printing the number of lines, character and bytes in a file.

Usage:

`wc` + `option` + `file(s)`

Examples:

Display the number of characters in a file: `wc -m users.txt` Display the number of lines in a file: `wc -l users.txt` Display the number words in a file: `wc -w users.txt` Display the number of bytes in a file: `wc -c users.txt`

---

tr

Definition:

The tr command is used for translating or deleting characters from standard output.

Usage:

`Standard output` | `tr` + `option` + `set` + `set`

Examples:

1. Translate period with a comma: `cat file.txt | tr '.' ','`
2. Translate white space into a tab: `cat program.py | tr "[:space:]" '\t'`
3. Translate tabs into space: `cat file.py | tr -s "[:space:]" ''`

---

diff

Definition:

The diff command compares files and displays the differences between them.

Usage:

`diff` + `option` + `file1` + `file2`

Example:

1. Display the difference between two files: `diff cars.csv cars-backup.csv`
2. Display the difference between two files in a column format: `diff -y cars.csv cars-backup.csv`

---

grep

Definition:

Grep is used to search text in a given file. Grep works in a line by line basis.

Usage:

`grep` + `option` + `search criteria` + `file(s)`

Example:

1. Search any line that contains the word "dracula" in the dracula.txt file: `grep 'dracula' dracula.txt`
2. Search any line that contains the word "dracula" in the dracula.txt file regardless of case sensitivity: `grep -i 'dracula' dracula.txt`
3. Search and display only the matched string (pattern) in the dracula.txt file: `grep -o 'blood' dracula.txt`
4. Display how many lines contain the matched string in the dracula.txt file: `grep -c 'dracula' dracula.txt`
