**UNIX COMMAND LINE AND BASIC OPERATIONS**

UNIX is an operating system that was developed in the 1960s and is used widely today. One of the key features of UNIX is its command line interface, which allows users to interact with the system by typing commands into a terminal. Here are some basic UNIX command line operations:

1. **Navigation**

The UNIX command line allows you to navigate through your system's file structure. The most basic navigation commands are:

* **cd** (change directory): This command allows you to change your current directory. For example, if you want to change to the home directory, you would type "cd ~" (without quotes).
* **ls** (list): This command lists the files and directories in the current directory. For example, "ls" will list all the files and directories in the current directory.
* **pwd** (print working directory): This command shows the current directory you are in.

1. **Creating, Moving, and Removing Files and Directories**

The UNIX command line also allows you to create, move, and remove files and directories. Here are some basic commands:

* **mkdir** (make directory): This command creates a new directory. For example, "mkdir new\_directory" will create a new directory named "new\_directory" in the current directory.
* **touch**: This command creates a new file. For example, "touch new\_file.txt" will create a new file named "new\_file.txt" in the current directory.
* **mv** (move): This command moves a file or directory from one location to another. For example, "mv file.txt ~/Documents/" will move the file "file.txt" to the Documents directory in your home directory.
* **cp** (copy): This command copies a file or directory from one location to another. For example, "cp file.txt ~/Documents/" will copy the file "file.txt" to the Documents directory in your home directory.
* **rm** (remove): This command deletes a file or directory. For example, "rm file.txt" will delete the file "file.txt" in the current directory.

1. **File Manipulation**

The UNIX command line also allows you to manipulate the contents of files. Here are some basic commands:

* **cat**: This command displays the contents of a file. For example, "cat file.txt" will display the contents of the file "file.txt".
* **echo**: This command outputs text to the terminal. For example, "echo 'Hello World!'" will output "Hello World!" to the terminal.
* **less**: This command allows you to view the contents of a file one page at a time. For example, "less file.txt" will display the contents of the file "file.txt" one page at a time.
* **more**: This command line allows you to view the contents of a file one screenful at a time. It is particularly useful for viewing large files or files with long lines. When you run the more command, it displays the first screenful of the file, and then waits for you to press a key before displaying the next screenful. You can use the spacebar to advance to the next screenful, or you can use the enter key to advance one line at a time. For example, "more file.txt" will display the first screenful of the file "file.txt".
* **grep**: This command line allows you to search for a specific pattern in a file or a set of files. It is particularly useful for finding lines that contain a specific word or phrase. When you run the grep command, you specify the pattern you want to search for, and the file or files you want to search. The grep command will then search for the pattern in the specified file or files, and display any lines that contain the pattern. For example, "grep example file.txt" will display any lines in the file "file.txt" that contain the word "example".

1. **Process Management**

The UNIX command line also allows you to manage running processes on your system. Here are some basic commands:

* **ps**: This command lists the processes running on your system.
* **top**: This command displays a dynamic view of the processes running on your system.
* **kill**: This command terminates a running process. For example, "kill 1234" will terminate the process with the ID 1234

**BASH:**

Bash is a shell, which is a command-line interpreter that provides a way for users to interact with their operating system. Bash is the default shell on most Linux and Unix-based systems, including macOS. Bash is an acronym for Bourne-Again SHell, which is a reference to the original Unix shell, the Bourne shell.

Bash provides a command-line interface (CLI) for users to execute commands and scripts. It interprets the commands entered by the user, executes them, and provides output based on the results. Bash supports many features that make it a powerful tool for managing your system, including command-line editing, history, aliases, and job control.

Some of the key features of Bash include:

* **Command-Line Editing**: Bash provides several keyboard shortcuts that allow you to edit and manipulate commands that you have entered on the command line. For example, you can use the arrow keys to navigate through your command history, or you can use the backspace key to delete characters in your current command.
* **History**: Bash keeps a history of the commands you have entered on the command line. You can use the up and down arrow keys to navigate through your command history, or you can use the history command to view a list of your previous commands.
* **Aliases**: Bash allows you to create aliases, which are shortcuts for longer commands. For example, you could create an alias called ll that executes the ls -l command, making it easier to remember and use.
* **Job Control**: Bash allows you to manage multiple processes running in the background. You can start a process in the background using the & symbol, and you can use commands like jobs, fg, and bg to manage these processes.

Bash also supports scripting, which allows you to create and run scripts that automate tasks on your system. Bash scripts are text files that contain a series of commands that Bash executes when the script is run. Bash scripts can be used for a variety of tasks, such as automating backups, managing system configurations, or performing repetitive tasks.

**PIPES:**

In the UNIX command line, a pipe is a mechanism for connecting the output of one command to the input of another command, allowing you to chain commands together and create more powerful and flexible commands.

The pipe symbol | is used to create a pipe between two commands. For example, let's say you want to search for a specific pattern in a file using the grep command, and then count the number of lines that contain that pattern using the wc command. You can use a pipe to connect the output of the grep command to the input of the wc command like this:

grep "pattern" file.txt | wc –l

This command will search for the pattern "pattern" in the file file.txt, and then pass the output to the wc command. The wc command will then count the number of lines in the output that contain the pattern, and display the result.

Pipes are a powerful and flexible feature of the UNIX command line, allowing you to create complex commands by chaining together simpler commands. They are particularly useful for working with large datasets or for automating repetitive tasks.