**Practical : 2**

**AIM: Process management**

1. Use ps to search for init process by name
2. What is the process id of init process?
3. Use the who am I command to determine the terminal name
4. Using your terminal name from above, use ps to find all process associates with your terminal.
5. What is the process id  of your shell
6. What is the parent process id of your shell
7. Start two instances of sleep 4432 in the background
8. Locate the process id of all the sleep command
9. Display those two sleep processes on top then quite top
10. Use the standard kill to kill one of the processes

**Theory:**

**Process Management:**

A process (or job) is the fundamental unit of work in an operating system. Process management includes creating and deleting processes and providing mechanisms for processes to communicate and synchronize with each other.

It handles operations by performing tasks like process scheduling and such as resource allocation.

* **ps command:**

**ps** displays information about a selection of the active processes.

**Syntax: ~$ ps**

It displays the following information in the output.

* **PID**: It shows the unique process ID.
* **TTY**: It shows the terminal type into which the user is logged in.
* **TIME**: It displays the total time that the process has been running.
* **CMD**: It displays the name of the command that launches the process. As we can notice in the output, the second process is started by the ps command itself.
* **Process id**: Each process has a unique identifier. The PID is used to specify the process to the operating system when an application makes a system call to signal, modify, or wait for the process.

* **whoami command**: It is basically the concatenation of the strings “who”, “am”, “i” as whoami. It displays the username of the current user when this command is invoked.

**Syntax:** ~$ whoami

* **ps command**: Command to check the process status (ps command

Item, Description, PID, Process ID, PPID, Parent process ID

Process id of shell: The parent process ID of your current context is exposed as an environment variable. To see the value, you can echo it out. For example, if you are SSH 'd into a Linux server,

the value of the $PPID environment variable will the process ID of the SSH process. The environment variable only outputs the process ID. Syntax: ~$ ps $ PPID

* **sleep command**: The **sleep** command suspends the calling process of the next command for a specified amount of time. This property is useful when the following command’s execution depends on the successful completion of a previous command.

# Process id of sleep command:

**Syntax:** ~$ sleep process I'd

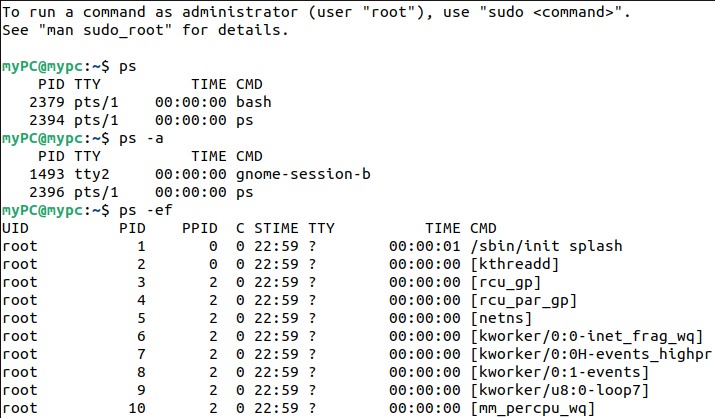
* **top command**: The top command is a well-known and most widely used utility to display dynamic real-time information about running processes in Linux and Unix-like operating systems. Usually, this command shows the summary information of the system and the list of processes or threads which are currently managed by the Linux Kernel.

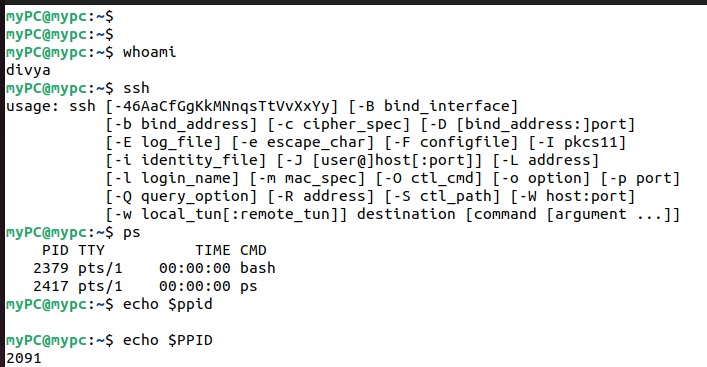
**Syntax**: ~$ top

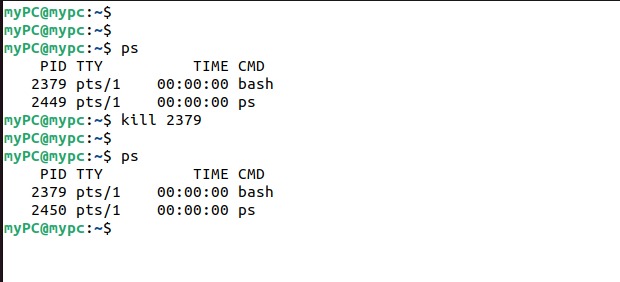
# kill command: The kill command transfers a signal towards a process in which further terminates the process. When the user does not describe any signal that is to be transferred with the kill command then the **TERM** signal (default) is transferred that will terminate the process. If the signal is not described, then it will default to **-15** (**-TERM**).

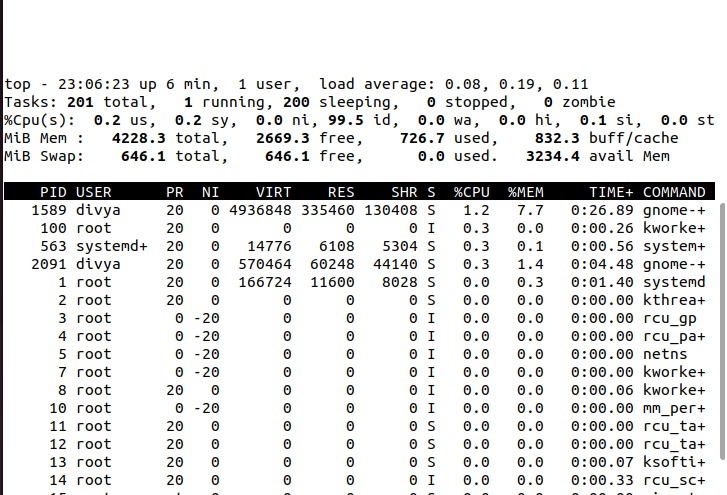
**Syntax:** ~$ kill

# Implementation:









**Conclusion**: Thus, we have successfully understood the process management and implemented all its topics.