Part 1

Task 1

A process in Linux can have the following states:

- **Running**: The process is currently executing or in the running state.
- **Waiting (Blocked)**: The process is waiting for a certain event or resource to become available before it can proceed.
- **Stopped**: The process has been stopped, usually by a signal. It can be resumed later.
- **Zombie**: A terminated child process that has completed its execution but its parent process has not yet collected its exit status.
- **Terminated (Exited)**: The process has completed its execution and has either exited normally or due to an error.
- **Paging**: The process's memory pages are being transferred between the main memory and the secondary storage (e.g., swap space).
- **Deadlock**: A situation where multiple processes are unable to proceed because each is waiting for a resource held by another process.

Task 2

```
student@CsnKhai:~$ current_pid=$$
student@CsnKhai:~$ pstree -s $current_pid
init───sshd───sshd───sshd───bash───pstree
student@CsnKhai:~$ ps
  PID TTY          TIME CMD
  900 pts/0    00:00:00 bash
 1046 pts/0    00:00:00 ps
student@CsnKhai:~$
```

Task 3

The **/proc** filesystem in Linux is a virtual filesystem that provides a way to interact with kernel data structures and processes as if they were regular files. It doesn't represent physical files on the disk, but rather exposes kernel-related information and system statistics in a hierarchical directory structure.

Key features of the **/proc** filesystem include:

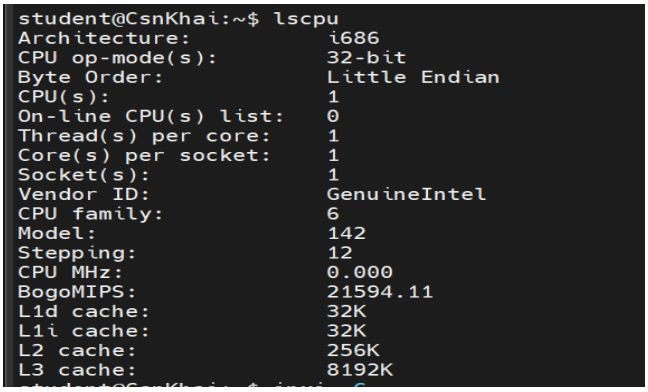
- **Process Information**: Each running process on the system is represented by a directory within **/proc**, named after its process ID (PID). Inside these directories, various files provide information about the process, such as its status, command-line arguments, environment variables, file descriptors, and more.
- **Kernel Information**: The **/proc** filesystem allows direct access to information about various aspects of the kernel, including kernel parameters, configuration, and statistics.
- **Hardware Information**: It provides details about hardware devices and configurations, including CPU and memory information.

- **System Statistics**: Various system statistics, such as memory usage, CPU usage, interrupts, filesystem statistics, and network statistics, can be accessed through **`/proc`** files.
- **Dynamic Updates**: The information provided by **`/proc`** files is dynamically updated as the system state changes, reflecting real-time data.
- **Configuration**: Some kernel parameters can be adjusted by writing to certain **`/proc`** files. This allows for runtime configuration changes without needing to restart the system.

The **`/proc`** filesystem is an essential tool for system administrators, developers, and users who want to gain insights into the inner workings of the Linux kernel, monitor system performance, and interact with processes and devices at a low level.

Task 4

Or



Task 5

```
student@CsnKhai:~$ ps -ef
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 13:55 ?        00:00:00 /sbin/init
root         2     0  0 13:55 ?        00:00:00 [kthreadd]
root         3     2  0 13:55 ?        00:00:00 [ksoftirqd/0]
root         4     2  0 13:55 ?        00:00:00 [kworker/0:0]
root         5     2  0 13:55 ?        00:00:00 [kworker/0:0H]
root         7     2  0 13:55 ?        00:00:00 [rcu_sched]
root         8     2  0 13:55 ?        00:00:00 [rcu_bh]
root         9     2  0 13:55 ?        00:00:00 [migration/0]
root        10     2  0 13:55 ?        00:00:00 [watchdog/0]
root        11     2  0 13:55 ?        00:00:00 [khelper]
root        12     2  0 13:55 ?        00:00:00 [kdevtmpfs]
root        13     2  0 13:55 ?        00:00:00 [netns]
root        14     2  0 13:55 ?        00:00:00 [writeback]
root        15     2  0 13:55 ?        00:00:00 [kintegrityd]
root        16     2  0 13:55 ?        00:00:00 [bioset]
root        17     2  0 13:55 ?        00:00:00 [kworker/u3:0]
root        18     2  0 13:55 ?        00:00:00 [kblockd]
root        19     2  0 13:55 ?        00:00:00 [ata_sff]
root        20     2  0 13:55 ?        00:00:00 [khubd]
root        21     2  0 13:55 ?        00:00:00 [md]
root        22     2  0 13:55 ?        00:00:00 [devfreq_wq]
root        23     2  0 13:55 ?        00:00:13 [kworker/0:1]
root        25     2  0 13:55 ?        00:00:00 [khungtaskd]
root        26     2  0 13:55 ?        00:00:00 [kswapd0]
root        27     2  0 13:55 ?        00:00:00 [ksmd]
root        28     2  0 13:55 ?        00:00:00 [fsnotify_mark]
root        29     2  0 13:55 ?        00:00:00 [ecryptfs-kthrea]
root        30     2  0 13:55 ?        00:00:00 [crypto]
root        42     2  0 13:55 ?        00:00:00 [kthrotld]
root        44     2  0 13:55 ?        00:00:00 [kworker/u2:2]
root        45     2  0 13:55 ?        00:00:00 [scsi_eh_0]
root        46     2  0 13:55 ?        00:00:00 [scsi_eh_1]
root        67     2  0 13:55 ?        00:00:00 [deferwq]
root        68     2  0 13:55 ?        00:00:00 [charger_manager]
root       114     2  0 13:55 ?        00:00:00 [kworker/u3:1]
root       115     2  0 13:55 ?        00:00:00 [scsi_eh_2]
root       116     2  0 13:55 ?        00:00:00 [kpsmoused]
root       125     2  0 13:55 ?        00:00:00 [jbd2/sda1-8]
root       126     2  0 13:55 ?        00:00:00 [ext4-rsv-conver]
root       249     1  0 13:55 ?        00:00:00 upstart-udev-bridge --daemon
root       254     1  0 13:55 ?        00:00:00 /lib/systemd/systemd-udevd --daemon
message+   338     1  0 13:55 ?        00:00:00 dbus-daemon --system --fork
```

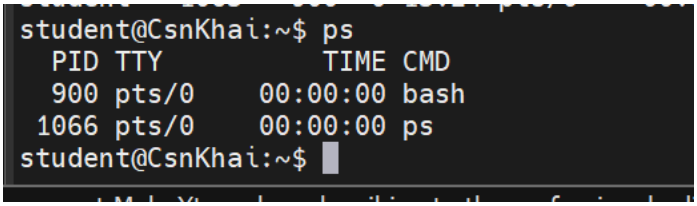
Task 6

Kernel processes and user processes are distinguished based on their roles and privileges in the operating system.

- **Kernel Processes:**
- Kernel processes are responsible for managing the core functions of the operating system.
- They have elevated privileges and direct access to hardware and system resources.
- They are essential for the proper functioning of the OS and are usually started during system boot.
- Examples of kernel processes include process schedulers, memory management, device drivers, etc.
- These processes typically have low process IDs (PIDs).
- **User Processes:**
- User processes are initiated by users or applications to perform specific tasks.
- They have limited access to system resources and cannot directly interact with hardware.
- User processes operate within user mode, which provides a layer of protection from interfering with the kernel.
- These processes are created by users, and they have higher process IDs (PIDs) compared to kernel processes.
- Examples of user processes include applications, command-line utilities, etc.

In general, user processes run in a controlled environment, isolated from the critical functions of the operating system managed by kernel processes.

Task 7

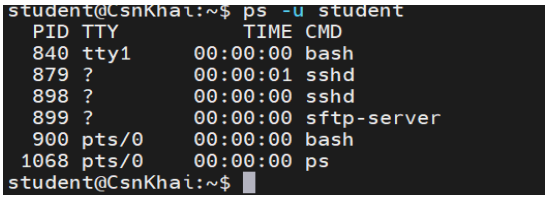


The main process statuses include:

- **Running (R)**: The process is active and executing on the CPU.
- **Sleeping (S)**: The process is waiting for an event or I/O and temporarily suspended from execution.
- **Waiting (D)**: The process is waiting for I/O without being able to quickly respond to signals.
- **Zombie (Z)**: A process that has finished its execution but its information is still present in the process table.
- **Stopped (T)**: The process has been stopped, for example, by a keyboard signal.
- **Dead (X)**: A process that has finished or been terminated but its information hasn't been removed from the process table yet.
- **Traced (t)**: The process is in debugging mode.
- **Paging (W)**: The process is waiting for loading/unloading from memory.
- **Idle (I)**: A special process representing a relatively inactive system state.
- **Unknown (U)**: The process status is unknown or cannot be determined.

Processes can transition between these states based on their tasks and interactions with the system.

Task 8



Task 9

```
student@CsnKhai:~$ ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.8   4152  2176 ?        Ss   13:55   0:00 /sbin/init
root         2  0.0  0.0      0     0 ?        S    13:55   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S    13:55   0:00 [ksoftirqd/0]
root         4  0.0  0.0      0     0 ?        S    13:55   0:00 [kworker/0:0]
root         5  0.0  0.0      0     0 ?        S<   13:55   0:00 [kworker/0:0H]
root         7  0.0  0.0      0     0 ?        S    13:55   0:00 [rcu_sched]
root         8  0.0  0.0      0     0 ?        S    13:55   0:00 [rcu_bh]
root         9  0.0  0.0      0     0 ?        S    13:55   0:00 [migration/0]
root        10  0.0  0.0      0     0 ?        S    13:55   0:00 [watchdog/0]
root        11  0.0  0.0      0     0 ?        S<   13:55   0:00 [khelper]
root        12  0.0  0.0      0     0 ?        S    13:55   0:00 [kdevtmpfs]
root        13  0.0  0.0      0     0 ?        S<   13:55   0:00 [netns]
root        14  0.0  0.0      0     0 ?        S<   13:55   0:00 [writeback]
root        15  0.0  0.0      0     0 ?        S<   13:55   0:00 [kintegrityd]
root        16  0.0  0.0      0     0 ?        S<   13:55   0:00 [bioset]
root        17  0.0  0.0      0     0 ?        S<   13:55   0:00 [kworker/u3:0]
root        18  0.0  0.0      0     0 ?        S<   13:55   0:00 [kblockd]
root        19  0.0  0.0      0     0 ?        S<   13:55   0:00 [ata_sff]
root        20  0.0  0.0      0     0 ?        S    13:55   0:00 [khubd]
root        21  0.0  0.0      0     0 ?        S<   13:55   0:00 [md]
root        22  0.0  0.0      0     0 ?        S<   13:55   0:00 [devfreq_wq]
root        23  0.2  0.0      0     0 ?        R    13:55   0:14 [kworker/0:1]
root        25  0.0  0.0      0     0 ?        S    13:55   0:00 [khungtaskd]
root        26  0.0  0.0      0     0 ?        S    13:55   0:00 [kswapd0]
root        27  0.0  0.0      0     0 ?        SN   13:55   0:00 [ksmd]
root        28  0.0  0.0      0     0 ?        S    13:55   0:00 [fsnotify_mark]
root        29  0.0  0.0      0     0 ?        S    13:55   0:00 [ecryptfs-kthrea]
root        30  0.0  0.0      0     0 ?        S<   13:55   0:00 [crypto]
root        42  0.0  0.0      0     0 ?        S<   13:55   0:00 [kthrotld]
root        44  0.0  0.0      0     0 ?        S    13:55   0:00 [kworker/u2:2]
root        45  0.0  0.0      0     0 ?        S    13:55   0:00 [scsi_eh_0]
root        46  0.0  0.0      0     0 ?        S    13:55   0:00 [scsi_eh_1]
root        67  0.0  0.0      0     0 ?        S<   13:55   0:00 [deferwq]
root        68  0.0  0.0      0     0 ?        S<   13:55   0:00 [charger_manager]
root       114  0.0  0.0      0     0 ?        S<   13:55   0:00 [kworker/u3:1]
root       115  0.0  0.0      0     0 ?        S    13:55   0:00 [scsi_eh_2]
root       116  0.0  0.0      0     0 ?        S<   13:55   0:00 [kpsmoused]
root       125  0.0  0.0      0     0 ?        S    13:55   0:00 [jbd2/sda1-8]
root       126  0.0  0.0      0     0 ?        S<   13:55   0:00 [ext4-rsv-conver]
root       249  0.0  0.2   3008   616 ?        S    13:55   0:00 upstart-udev-bridge --daemon
```

```
student@CsnKhai:~$ ps -ef
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 13:55 ?        00:00:00 /sbin/init
root         2     0  0 13:55 ?        00:00:00 [kthreadd]
root         3     2  0 13:55 ?        00:00:00 [ksoftirqd/0]
root         4     2  0 13:55 ?        00:00:00 [kworker/0:0]
root         5     2  0 13:55 ?        00:00:00 [kworker/0:0H]
root         7     2  0 13:55 ?        00:00:00 [rcu_sched]
root         8     2  0 13:55 ?        00:00:00 [rcu_bh]
root         9     2  0 13:55 ?        00:00:00 [migration/0]
root        10     2  0 13:55 ?        00:00:00 [watchdog/0]
root        11     2  0 13:55 ?        00:00:00 [khelper]
root        12     2  0 13:55 ?        00:00:00 [kdevtmpfs]
root        13     2  0 13:55 ?        00:00:00 [netns]
root        14     2  0 13:55 ?        00:00:00 [writeback]
root        15     2  0 13:55 ?        00:00:00 [kintegrityd]
root        16     2  0 13:55 ?        00:00:00 [bioset]
root        17     2  0 13:55 ?        00:00:00 [kworker/u3:0]
root        18     2  0 13:55 ?        00:00:00 [kblockd]
root        19     2  0 13:55 ?        00:00:00 [ata_sff]
root        20     2  0 13:55 ?        00:00:00 [khubd]
root        21     2  0 13:55 ?        00:00:00 [md]
root        22     2  0 13:55 ?        00:00:00 [devfreq_wq]
root        23     2  0 13:55 ?        00:00:14 [kworker/0:1]
root        25     2  0 13:55 ?        00:00:00 [khungtaskd]
root        26     2  0 13:55 ?        00:00:00 [kswapd0]
root        27     2  0 13:55 ?        00:00:00 [ksmd]
root        28     2  0 13:55 ?        00:00:00 [fsnotify_mark]
root        29     2  0 13:55 ?        00:00:00 [ecryptfs-kthrea]
```

```
student@CsnKhai:~$ ps -e
  PID TTY          TIME CMD
    1 ?        00:00:00 init
    2 ?        00:00:00 kthreadd
    3 ?        00:00:00 ksoftirqd/0
    4 ?        00:00:00 kworker/0:0
    5 ?        00:00:00 kworker/0:0H
    7 ?        00:00:00 rcu_sched
    8 ?        00:00:00 rcu_bh
    9 ?        00:00:00 migration/0
   10 ?        00:00:00 watchdog/0
   11 ?        00:00:00 khelper
   12 ?        00:00:00 kdevtmpfs
   13 ?        00:00:00 netns
   14 ?        00:00:00 writeback
   15 ?        00:00:00 kintegrityd
   16 ?        00:00:00 bioset
   17 ?        00:00:00 kworker/u3:0
   18 ?        00:00:00 kblockd
   19 ?        00:00:00 ata_sff
   20 ?        00:00:00 khubd
   21 ?        00:00:00 md
   22 ?        00:00:00 devfreq_wq
   23 ?        00:00:14 kworker/0:1
   25 ?        00:00:00 khungtaskd
   26 ?        00:00:00 kswapd0
   27 ?        00:00:00 ksmd
   28 ?        00:00:00 fsnotify_mark
   29 ?        00:00:00 ecryptfs-kthrea
   30 ?        00:00:00 crypto
   42 ?        00:00:00 kthrotld
   44 ?        00:00:00 kworker/u2:2
   45 ?        00:00:00 scsi_eh_0
   46 ?        00:00:00 scsi_eh_1
   67 ?        00:00:00 deferwq
   68 ?        00:00:00 charger_manager
```

```
 1088 pts/0    00:00:00 ps
student@CsnKhai:~$ top
top - 15:38:46 up  1:43,  2 users,  load average: 0.00, 0.01, 0.05
Tasks:  65 total,   1 running,  64 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.7 sy,  0.0 ni, 98.7 id,  0.3 wa,  0.0 hi,  0.3 si,  0.0 st
KiB Mem:    247792 total,   130756 used,   117036 free,    13432 buffers
KiB Swap:        0 total,        0 used,        0 free.    89104 cached Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
    3 root      20   0       0      0      0 S  0.3  0.0   0:00.57 ksoftirqd/0
  879 student   20   0   11192   2608   1808 S  0.3  1.1   0:01.46 sshd
    1 root      20   0    4152   2176   1416 S  0.0  0.9   0:00.84 init
    2 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kthreadd
    4 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kworker/0:0
    5 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kworker/0:0H
    7 root      20   0       0      0      0 S  0.0  0.0   0:00.27 rcu_sched
    8 root      20   0       0      0      0 S  0.0  0.0   0:00.00 rcu_bh
    9 root      rt   0       0      0      0 S  0.0  0.0   0:00.00 migration/0
   10 root      rt   0       0      0      0 S  0.0  0.0   0:00.02 watchdog/0
   11 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 khelper
   12 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kdevtmpfs
   13 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 netns
   14 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 writeback
   15 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kintegrityd
   16 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 bioset
   17 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kworker/u3:0
   18 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kblockd
   19 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 ata_sff
   20 root      20   0       0      0      0 S  0.0  0.0   0:00.22 khubd
   21 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 md
   22 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 devfreq_wq
   23 root      20   0       0      0      0 S  0.0  0.0   0:14.88 kworker/0:1
   25 root      20   0       0      0      0 S  0.0  0.0   0:00.00 khungtaskd
   26 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kswapd0
   27 root      25   5       0      0      0 S  0.0  0.0   0:00.00 ksmd
   28 root      20   0       0      0      0 S  0.0  0.0   0:00.00 fsnotify_mark
e support MobaXterm by subscribing to the professional edition here:  https://mobaxterm.mobatek.net
```

Task 10

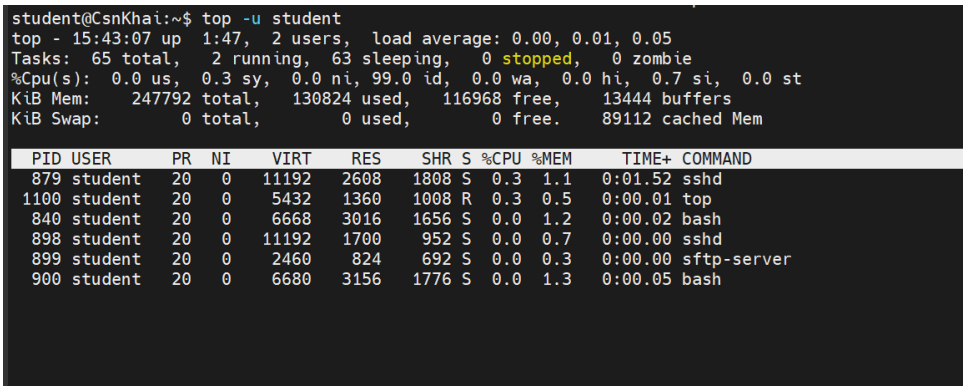
The **top** command provides a dynamic, real-time view of the system's processes and resource utilization. When you run the **top** command in the terminal, it displays a continuously updating list

of processes, along with various system-level statistics. The key information displayed by the **top** command includes:

- **Header Information:** The top of the display shows system-level information, including the current time, system uptime, number of logged-in users, load averages, and other general information.
- **Process Listing:** Below the header, the main portion of the display lists the running processes. Each process is represented by a row of information containing details such as:
- Process ID (PID)
- User who owns the process (USER)
- Percentage of CPU utilization (%CPU)
- Percentage of memory utilization (%MEM)
- Resident Set Size (RSS) - memory usage in kilobytes
- Virtual Memory Size (VIRT) - total virtual memory used
- Command being executed (COMMAND)
- **System Statistics:** Below the process list, there are system-wide statistics showing:
- CPU usage breakdown by user, system, and idle
- Memory usage details, including total, used, free, and cached memory
- Swap usage details
- Load average over 1, 5, and 15 minutes
- **Interactive Commands:** While **top** is running, you can use various keyboard shortcuts to interact with the display. For example, you can change the sorting order of processes, renice a process, filter processes, and more.

The **top** command provides a real-time snapshot of system activity and resource utilization, making it a valuable tool for monitoring and troubleshooting system performance. To exit the **top** command, you can simply press the "q" key.

Task 11

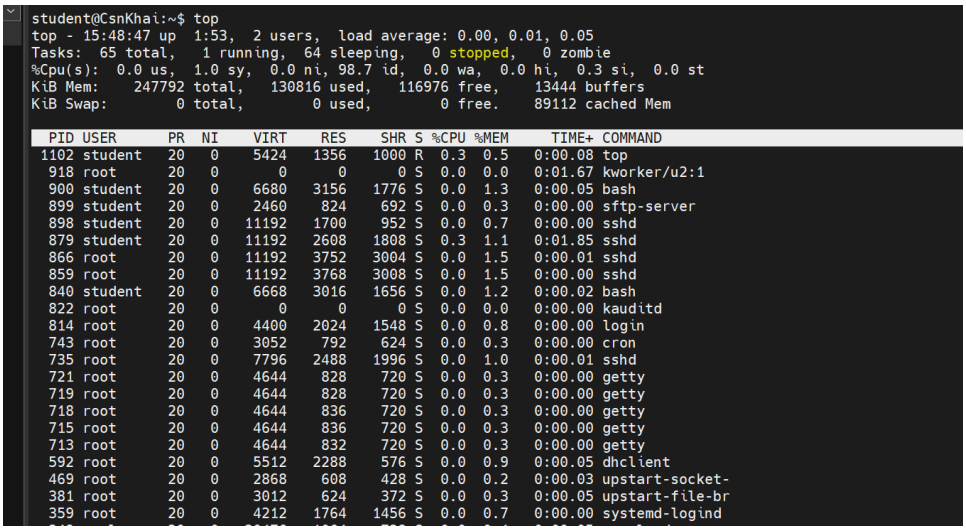


- Task 12

  q: This command quits the **top** program and exits.
- **k**: This command allows you to send a signal to a selected process. After pressing **k**, you'll be prompted to enter the Process ID (PID) of the process you want to send a signal to, and then you can enter the signal number (e.g., 15 for SIGTERM).
- **r**: This command changes the priority of a process. After pressing **r**, you'll be prompted to enter the PID of the process and then the new priority value.

- **c**: This command toggles between showing command names and full command lines in the process list.
- **i**: This command toggles the display of idle processes.
- **W**: This command saves your current **top** configuration to a configuration file.
- **f**: This command allows you to select which fields are displayed in the process list. It provides an interactive menu to customize the display.
- **o**: This command allows you to sort the process list based on a specific field. For example, pressing **o** followed by **%CPU** will sort the processes by CPU usage.
- **H**: This command toggles the highlighting of running tasks.
- These are just a few examples of the interactive commands that **top** offers. You can access the full list of commands and their descriptions by pressing the **h** key while in the **top** interface. This will display the help screen with detailed information about all available commands.

Task 13



```
KiB Swap:        0 total,        0 used,        0 free.    89112 cached Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
  859 root      20   0   11192   3768   3008 S  0.0  1.5   0:00.00 sshd
  866 root      20   0   11192   3752   3004 S  0.0  1.5   0:00.01 sshd
  900 student   20   0    6680   3156   1776 S  0.0  1.3   0:00.05 bash
  840 student   20   0    6668   3016   1656 S  0.0  1.2   0:00.02 bash
  879 student   20   0   11192   2608   1808 S  0.0  1.1   0:01.92 sshd
  735 root      20   0    7796   2488   1996 S  0.0  1.0   0:00.01 sshd
  592 root      20   0    5512   2288    576 S  0.0  0.9   0:00.05 dhclient
    1 root      20   0    4152   2176   1416 S  0.0  0.9   0:00.84 init
  814 root      20   0    4400   2024   1548 S  0.0  0.8   0:00.00 login
  359 root      20   0    4212   1764   1456 S  0.0  0.7   0:00.00 systemd-logind
  898 student   20   0   11192   1700    952 S  0.0  0.7   0:00.00 sshd
  254 root      20   0   12036   1404    980 S  0.0  0.6   0:00.06 systemd-udevd
 1102 student   20   0    5424   1356   1000 R  0.0  0.5   0:00.12 top
  349 syslog    20   0   30476   1064    728 S  0.0  0.4   0:00.05 rsyslogd
  338 message+  20   0    4236    984    704 S  0.0  0.4   0:00.06 dbus-daemon
  715 root      20   0    4644    836    720 S  0.0  0.3   0:00.00 getty
  718 root      20   0    4644    836    720 S  0.0  0.3   0:00.00 getty
  713 root      20   0    4644    832    720 S  0.0  0.3   0:00.00 getty
  719 root      20   0    4644    828    720 S  0.0  0.3   0:00.00 getty
  721 root      20   0    4644    828    720 S  0.0  0.3   0:00.00 getty
  899 student   20   0    2460    824    692 S  0.0  0.3   0:00.00 sftp-server
  743 root      20   0    3052    792    624 S  0.0  0.3   0:00.00 cron
  381 root      20   0    3012    624    372 S  0.0  0.3   0:00.05 upstart-file-br
  249 root      20   0    3008    616    468 S  0.0  0.2   0:00.11 upstart-udev-br
  469 root      20   0    2868    608    428 S  0.0  0.2   0:00.03 upstart-socket-
    2 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kthreadd
    3 root      20   0       0      0      0 S  0.3  0.0   0:00.66 ksoftirqd/0
    4 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kworker/0:0
    5 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kworker/0:0H
```

Task 14

```
student@CsnKhai:~$ sudo nice -n -10 khelper
[sudo] password for student:
```

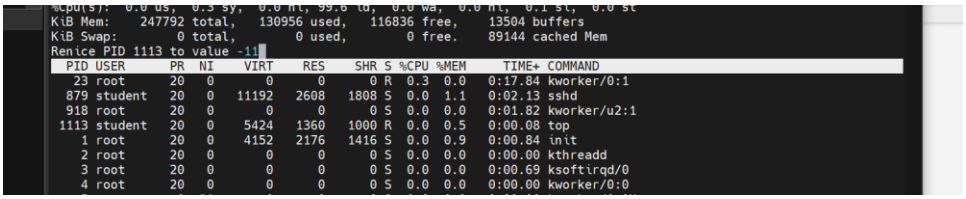
Task 15

Yes, you can change the priority of a process using the **top** command:

- Start the **top** command by typing **top** in your terminal.
- Inside the **top** interface, you will see a list of running processes. Find the process for which you want to change the priority.
- Use the arrow keys to highlight the process you want to adjust the priority for.
- Press the **r** key on your keyboard. This will bring up a prompt asking for the new priority value.
- Enter the new priority value you want to assign to the process and press Enter.

```
KiB Mem:    247792 total,   130956 used,   116836 free,    13504 buffers
KiB Swap:        0 total,        0 used,        0 free.    89144 cached Mem
PID to renice [default pid = 23]
  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
   23 root      20   0       0      0      0 S  0.3  0.0   0:17.73 kworker/0:1
  879 student   20   0   11192   2608   1808 S  0.3  1.1   0:02.12 sshd
 1113 student   20   0    5424   1360   1000 R  0.3  0.5   0:00.07 top
    1 root      20   0    4152   2176   1416 S  0.0  0.9   0:00.84 init
    2 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kthreadd
    3 root      20   0       0      0      0 S  0.0  0.0   0:00.69 ksoftirqd/0
```

```
%Cpu(s):  0.0 us,  0.7 sy,  0.0 ni, 99.3 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem:    247792 total,   130956 used,   116836 free,    13504 buffers
KiB Swap:        0 total,        0 used,        0 free.    89144 cached Mem
Renice PID 1113 to value
  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
   23 root      20   0       0      0      0 S  0.3  0.0   0:17.73 kworker/0:1
  879 student   20   0   11192   2608   1808 S  0.3  1.1   0:02.12 sshd
 1113 student   20   0    5424   1360   1000 R  0.3  0.5   0:00.07 top
    1 root      20   0    4152   2176   1416 S  0.0  0.9   0:00.84 init
    2 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kthreadd
    3 root      20   0       0      0      0 S  0.0  0.0   0:00.69 ksoftirqd/0
```



Task 16

```
student@CsnKhai:~$ kill -15 23
-bash: kill: (23) - Operation not permitted
student@CsnKhai:~$ kill -9 23
-bash: kill: (23) - Operation not permitted
student@CsnKhai:~$
```

```
-bash: kill: (23) - Operation not permitted
student@CsnKhai:~$ sudo kill 15 23
student@CsnKhai:~$
```

Commonly used signals include:

- **SIGTERM** (15): Termination signal. This is the default signal sent by **kill** and allows the process to perform cleanup tasks before exiting.
- **SIGKILL** (9): Kill signal. This signal forcefully terminates the process without giving it a chance to perform any cleanup.
- **SIGINT** (2): Interrupt signal. This is typically generated when you press Ctrl+C in the terminal.
- **SIGSTOP** (19): Stop signal. This signal pauses the process, and it can be resumed using the **SIGCONT** signal.
- **SIGCONT** (18): Continue signal. This signal resumes a process that has been paused by **SIGSTOP**.

Task 17

```
y
y
y
y

^Z[1]+  Stopped                 yes
student@CsnKhai:~$ jobs
[1]+  Stopped                 yes
student@CsnKhai:~$
```

```
^Z[1]+  Stopped                 yes
student@CsnKhai:~$ jobs
[1]+  Stopped                 yes
student@CsnKhai:~$ sleep 100
^Z
[2]+  Stopped                 sleep 100
student@CsnKhai:~$ jobs
[1]-  Stopped                 yes
[2]+  Stopped                 sleep 100
student@CsnKhai:~$
```

```
student@CsnKhai:~$ jobs
[1]-  Stopped                 yes
[2]+  Stopped                 sleep 100
student@CsnKhai:~$ bg %1
```

```
student@CsnKhai:~$ jobs
student@CsnKhai:~$ sleep 1000
^Z
[1]+  Stopped                 sleep 1000
student@CsnKhai:~$ jobs
[1]+  Stopped                 sleep 1000
student@CsnKhai:~$ bg %1
[1]+ sleep 1000 &
student@CsnKhai:~$ jobs
[1]+  Running                 sleep 1000 &
student@CsnKhai:~$ fg %1
sleep 1000
^Z
[1]+  Stopped                 sleep 1000
student@CsnKhai:~$ jobs
[1]+  Stopped                 sleep 1000
student@CsnKhai:~$
```

```
student@CsnKhai:~$ nohup sleep 3600 &
[2] 1031
student@CsnKhai:~$ nohup: ignoring input and appending output to 'nohup.out'
jobs
[1]+  Stopped                 sleep 1000
[2]-  Running                 nohup sleep 3600 &
student@CsnKhai:~$ bg %1
[1]+ sleep 1000 &
student@CsnKhai:~$ jobs
[1]-  Running                 sleep 1000 &
[2]+  Running                 nohup sleep 3600 &
student@CsnKhai:~$ fg %1
sleep 1000
^Z
[1]+  Stopped                 sleep 1000
student@CsnKhai:~$ jobs
[1]+  Stopped                 sleep 1000
[2]-  Running                 nohup sleep 3600 &
student@CsnKhai:~$ fg
sleep 1000
^Z
[1]+  Stopped                 sleep 1000
student@CsnKhai:~$ jobs
[1]+  Stopped                 sleep 1000
[2]-  Running                 nohup sleep 3600 &
student@CsnKhai:~$ fg %2
nohup sleep 3600
^Z
[2]+  Stopped                 nohup sleep 3600
student@CsnKhai:~$ jobs
[1]-  Stopped                 sleep 1000
[2]+  Stopped                 nohup sleep 3600
student@CsnKhai:~$
```

Part 2

Task 4

```
student@CsnKhai:~$ ssh student@192.168.0.108
The authenticity of host '192.168.0.108 (192.168.0.108)' can't be established.
ECDSA key fingerprint is d6:eb:2b:a9:bd:63:86:af:31:2f:bb:01:b5:14:63:ab.
Are you sure you want to continue connecting (yes/no)? y
Please type 'yes' or 'no': yes
Warning: Permanently added '192.168.0.108' (ECDSA) to the list of known hosts.
student@192.168.0.108's password:
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.13.0-63-generic i686)

 * Documentation:  https://help.ubuntu.com/
New release '16.04.7 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Fri Aug 25 07:19:48 2023 from 192.168.0.103
student@CsnKhai:~$
```