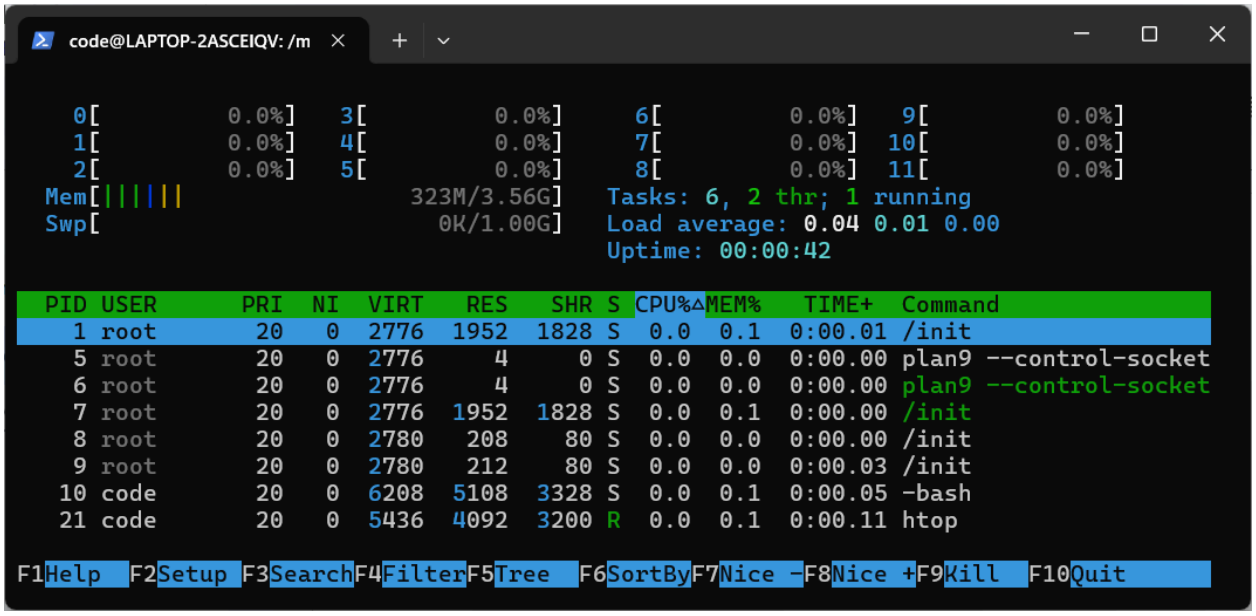
Tutorial 3 Command Line Tools

Htop

An interactive Linux command-line tool that shows running processes in real time.
After using the command “htop” we can see the results as this:



Just for your knowledge each terms means:

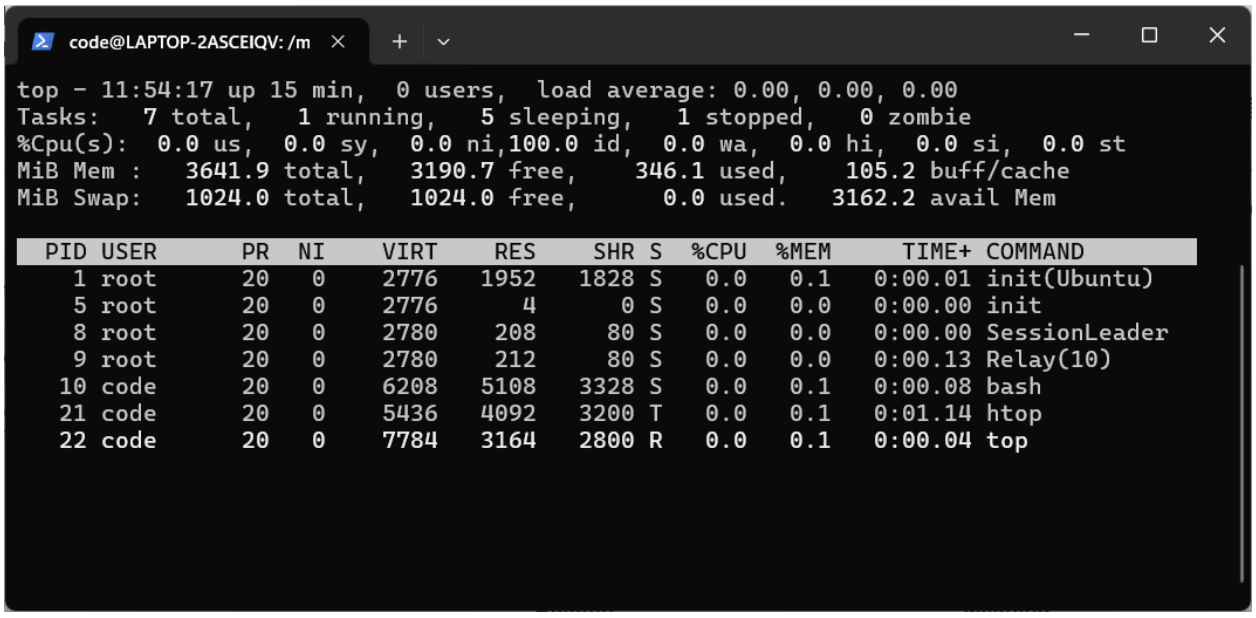
Field	Meaning
0,1,2,3,..	CPU cores usage (each bar shows activity per core)
Mem	Memory usage (used / total RAM)
Swp	Swap memory usage (used / total swap)
Tasks	Number of processes: total, threads, and how many are running
Load average	System load (last 1, 5, 15 minutes)
Uptime	Time since system boot

Column	Meaning
PID	Process ID (unique number for each process)
USER	The owner of the process

PRI	Priority of the process (lower = higher priority)
NI	Niceness value (affects priority)
VIRT	Virtual memory used by process (total address space)
RES	Resident memory (RAM actually used)
SHR	Shared memory used with other processes
S	Process state (R=Running, S=Sleeping, Z=Zombie, etc.)
CPU%	Percentage of CPU being used by the process
MEM%	Percentage of RAM being used
TIME+	Total CPU time consumed since process started
Command	The command/program that started the process

You can also use the "top" command to view running processes, but "htop" is preferred because it provides a clearer interface, easier navigation, and better visualization, making process management more user-friendly.

After running the top command

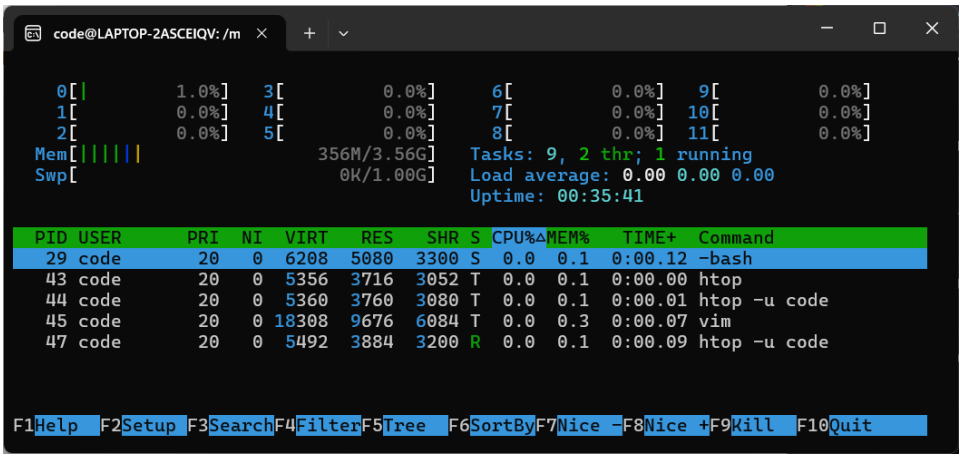


Some HTOP Commands
(https://www.geeksforgeeks.org/linux-unix/htop-command-in-linux-with-examples) :

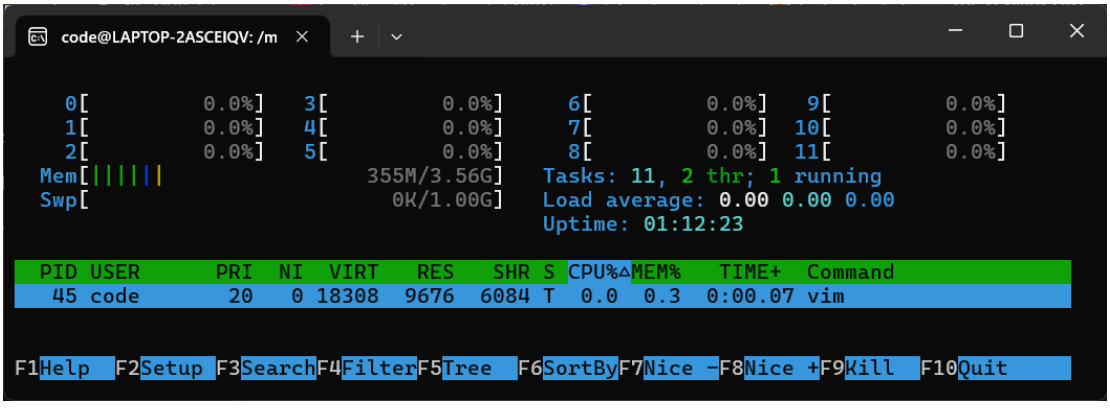
Option	Description
-d <delay>	Sets the delay between updates (in seconds).
-u <user>	Displays only the processes owned by the specified user.
-p <pid>	Shows only the process with the given process ID(s).
-s <column>	Sorts the process list by the specified column.
-t	Displays processes in a tree view under the commands column.
--no-color	Runs htop in monochrome mode, disabling colors.

1. **-u <user>** : In htop, using -u filters processes by the user name, so only processes owned by that user are displayed.
   eg :

```
code@LAPTOP-2ASCEIQV:/mnt/c/Users/Gautam Kumar/Downloads/OS$ htop -u code
```



2. **-p <PID>** : used to show only the given PIDs.
   Eg. My VIM process has a PID of 45. If I use the command "htop -p 45", it will display information only about that specific process.



nice/renice

Commands:

- nice [OPTION] [COMMAND [ARG]...]
- renice [-n] priority [-g|-p|-u] identifier..

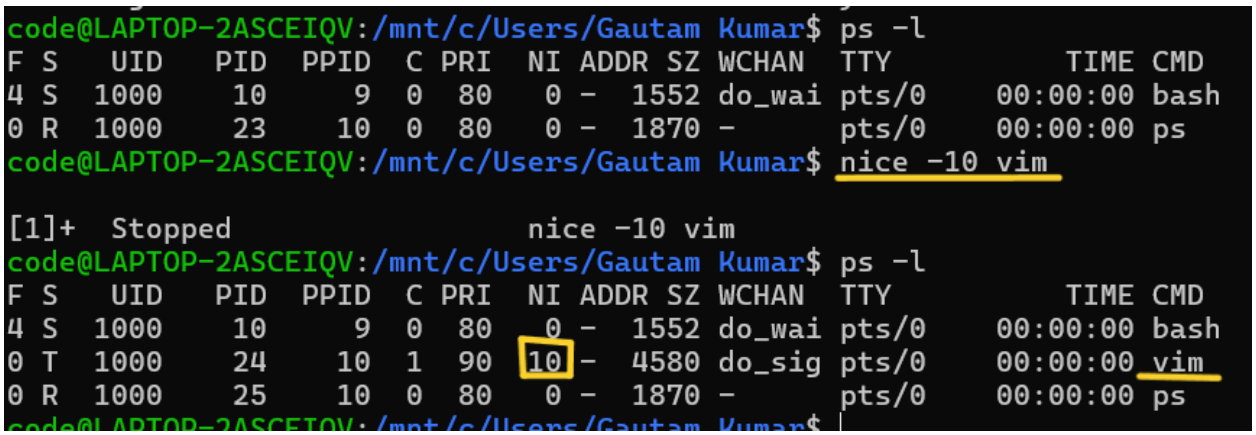
1. Nice :

E.g.: nice -10 vim or nice -n 10 vim

Explanation: When you use the `nice` command, the process you start will run with the priority you assign. In this case, we used a nice value of `10`, so the `vim` process will run with Lower priority.

If you want to give a higher priority, you need to write the command as:

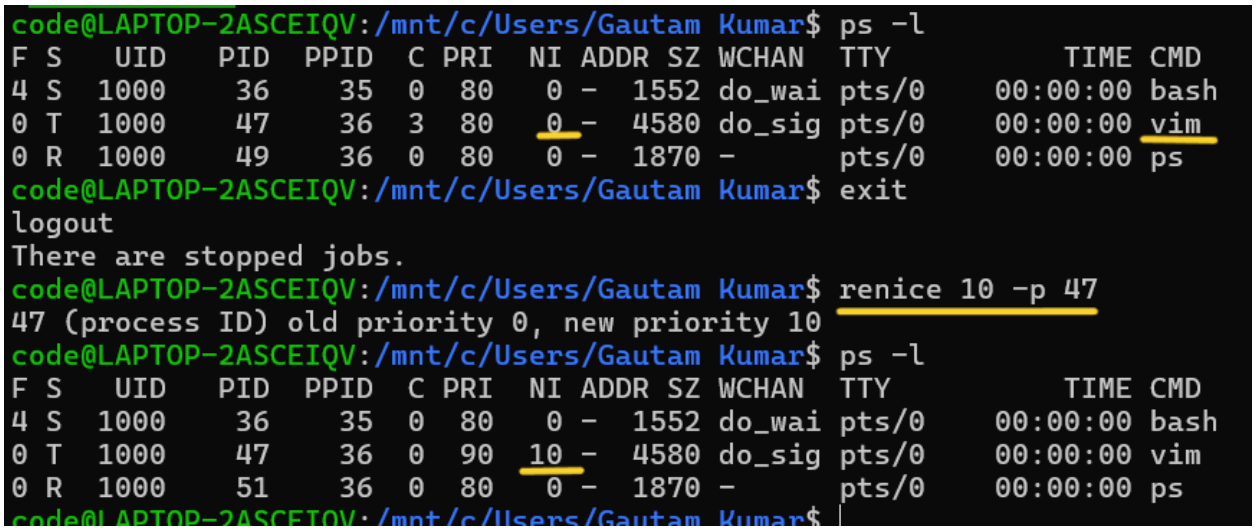
```
nice -n -10 vim
```



2. Renice:

E.g.: `renice 10 -p 47`

Explanation: When you use the `renice` command, you can change the priority (nice value) of an already running process. In this case, we set the nice value to `10` for the process with PID (-p) `47`, so it will run with **lower priority**.



If you want to give a higher priority, you need to write the command as:

```
sudo renice -10 -p 47
```

```
code@LAPTOP-2ASCEIQV:/mnt/c/Users/Gautam Kumar$ renice -10 -p 47
renice: failed to set priority for 47 (process ID): Permission denied
code@LAPTOP-2ASCEIQV:/mnt/c/Users/Gautam Kumar$ sudo renice -10 -p 47
[sudo] password for code:
47 (process ID) old priority 10, new priority -10
code@LAPTOP-2ASCEIQV:/mnt/c/Users/Gautam Kumar$ ps -l
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S  1000    36    35  0  80   0 -  1552 do_wai pts/0    00:00:00 bash
0 T  1000    47    36  0  70 -10 -  4580 do_sig pts/0    00:00:00 vim
0 R  1000    56    36  0  80   0 -  1870 -      pts/0    00:00:00 ps
```

Note: `nice` is used **when starting** a process, while `renice` is used **to change the priority of an already running process**. And we use `sudo` to change a process to a higher priority because lowering the nice value, e.g., `-10`, requires superuser privileges.

Chrt

(https://www.geeksforgeeks.org/linux-unix/chrt-command-in-linux-with-examples)

- chrt [options] priority command argument ...
- chrt [options] -p [priority] PID

1. To see the maximum and minimum priority of a scheduling policy, we use:

chrt -m

```
code@LAPTOP-2ASCEIQV:/mnt/c/Users/Gautam Kumar$ chrt -m
SCHED_OTHER min/max priority    : 0/0
SCHED_FIFO min/max priority     : 1/99
SCHED_RR min/max priority       : 1/99
SCHED_BATCH min/max priority    : 0/0
SCHED_IDLE min/max priority     : 0/0
SCHED_DEADLINE min/max priority : 0/0
```

You can see that the priorities of `SCHED_FIFO` and `SCHED_RR` are available in the range 1–99, while the others show `0/0`, which means they do not use priority values.

2. chrt -p <PID of the process> To see the scheduling policy of a process eg:

chrt -p 47

```
code@LAPTOP-2ASCEIQV:/mnt/c/Users/Gautam Kumar$ ps -l
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S  1000    36    35  0  80   0 -  1552 do_wai pts/0    00:00:00 bash
0 T  1000    47    36  0  70 -10 -  4580 do_sig pts/0    00:00:00 vim
0 R  1000    71    36  0  80   0 -  1870 -      pts/0    00:00:00 ps
code@LAPTOP-2ASCEIQV:/mnt/c/Users/Gautam Kumar$ chrt -p 47
pid 47's current scheduling policy: SCHED_OTHER
pid 47's current scheduling priority: 0
code@LAPTOP-2ASCEIQV:/mnt/c/Users/Gautam Kumar$ chrt -p 36
pid 36's current scheduling policy: SCHED_OTHER
pid 36's current scheduling priority: 0
```

chrt -p 36

3. To change the scheduling policy or priority of a process, we use:

`sudo chrt -<policy> -p <priority> <PID>` (for only FIFO and RR priority is 1/99)

Here are various scheduling policies, and we use the following options with chrt to set them:

- -f : **SCHED_FIFO** (First-In-First-Out)
- -b : **SCHED_BATCH**
- -i : **SCHED_IDLE**
- -r : **SCHED_RR** (Round Robin)
- -o : **SCHED_OTHER** (default policy)

Eg 1. **sudo chrt -b -p 0 122**

```
code@LAPTOP-2ASCEIQV:/mnt/c/Users/Gautam Kumar$ ps -l
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S  1000   111   110  0  80   0 -  1552 do_wai pts/0    00:00:00 bash
0 T  1000   122   111  0  80   0 -  4580 do_sig pts/0    00:00:00 vim
0 R  1000   125   111  0  80   0 -  1870 -      pts/0    00:00:00 ps
code@LAPTOP-2ASCEIQV:/mnt/c/Users/Gautam Kumar$ chrt -p 122
pid 122's current scheduling policy: SCHED_OTHER
pid 122's current scheduling priority: 0
code@LAPTOP-2ASCEIQV:/mnt/c/Users/Gautam Kumar$ sudo chrt -b -p 0 122
[sudo] password for code:
code@LAPTOP-2ASCEIQV:/mnt/c/Users/Gautam Kumar$ chrt -p 122
pid 122's current scheduling policy: SCHED_BATCH
pid 122's current scheduling priority: 0
```

Eg 2. **sudo chrt -f -p 3 122** (Note: Since the scheduling policy is **FIFO**, the priority must be set between **1 and 99**.)

```
code@LAPTOP-2ASCEIQV:/mnt/c/Users/Gautam Kumar$ chrt -p 122
pid 122's current scheduling policy: SCHED_BATCH
pid 122's current scheduling priority: 0
code@LAPTOP-2ASCEIQV:/mnt/c/Users/Gautam Kumar$ sudo chrt -f -p 3 122
code@LAPTOP-2ASCEIQV:/mnt/c/Users/Gautam Kumar$ chrt -p 122
pid 122's current scheduling policy: SCHED_FIFO
pid 122's current scheduling priority: 3
```

C Library Functions:

The C program demonstrates Linux process scheduling functions. It retrieves a process's scheduling policy and priority using sched_getscheduler and sched_getparam. Then, it attempts to change the process's priority (sched_setparam) and scheduling policy to real-time FIFO (sched_setscheduler). By default, processes run under SCHED_OTHER with priority 0. Changing to real-time policies requires root privileges, while priority changes only apply to FIFO or RR policies.