Zarządzanie procesami.

W zeszycie wykonaj notatę z wykonania poniższych poleń.

Narzędzie **ps** pozwala na wyświetlenie listy wszystkich procesów, które są aktualnie wykonywane przez system. Narzędzie te pobiera informacje z systemu plików **/proc** W bazie tej każdy proces ma swoje odzwierciedlenie. Aby wyświetlić wszystkie możliwe parametry narzędzia wydaj polecenie:

ps --help all

Wydanie samego polecenia **ps** bez żadnego parametru uwidoczni nam tylko procesy skojarzone z danym terminalem (TTY - ang. TeleTYpewriter).

```
root@bolek-VirtualBox:~# ps
  PID TTY          TIME CMD
 2918 pts/0    00:00:00 su
 2919 pts/0    00:00:00 bash
```

Jak widać powyżej nie uzyskaliśmy zbyt wielu informacji. Podczas korzystania z narzędzia **ps** przedstawiane dane (rodzaj prezentowanych danych zależy od użytych przełączników) będą zgrupowane w tabeli. Poniżej zestawienie nagłówków tabeli oznaczające parametry procesu:

- **USER** - właściciel procesu,
- **PID** – Process ID - identyfikator danego procesu, używany do jego identyfikacji co ważne w tym samym czasie PID nie może być przydzielony dwóm różnym procesom,
- **%CPU** – procentowe użycie procesora
- **%MEM** –procentowe użycie pamięci
- **VSZ** – przydzielona pamięć wirtualna w kB,
- **RSS** – przydzielona pamięć fizyczna w kB,
- **TTY** - identyfikator terminala procesu. Zapis **?** oznacza brak skojarzenia z terminalem.
- **STAT** – status procesu. Oznaczenia statusu procesu:
  - **R** – proces jest aktualnie wykonywany (running),
  - **S** – proces uśpiony, oczekiwanie na jego uruchomienie (sleep),
  - **T** – proces zatrzymany (stopped, np. CTRL-Z)
  - **Z** – zombie process - proces niewłaściwie zamknięty przez proces nadrzędny (proces rodzica bądź proces macierzysty)
- **START** – czas uruchomienia procesu,

- **TIME** – użycie procesora wyrażona w czasie,
- **COMMAND** – nazwa komendy/procesu oraz jego parametry,
- **NI** – wartość priorytetu nice (o tym później).

1. Wyświetl wszystkie procesy, które są powiązane z terminalami użyj komendy: **ps a**

```
root@bolek-VirtualBox:~# ps a
  PID TTY      STAT   TIME COMMAND
  723 tty1     Ssl+   0:00 /usr/lib/gdm3/gdm-wayland-session gnome-session --aut
  727 tty1     Sl+    0:00 /usr/lib/gnome-session/gnome-session-binary --autosta
  736 tty1     Sl+    0:13 /usr/bin/gnome-shell
  752 tty1     S+     0:00 /usr/bin/Xwayland :1024 -rootless -terminate -core -l
  782 tty1     Sl     0:00 ibus-daemon --xim --panel disable
```

2. W systemie istnieją procesy, które w żaden sposób z TTY nie są powiązane, uzyskać informację o tych procesach użyj polecenia: **ps ax**

```
root@bolek-VirtualBox:~# ps a |more
  PID TTY      STAT   TIME COMMAND
  723 tty1     Ssl+   0:00 /usr/lib/gdm3/gdm-wayland-session gnome-session --aut
ostart /usr/share/gdm/greeter/autostart
  727 tty1     Sl+    0:00 /usr/lib/gnome-session/gnome-session-binary --autosta
rt /usr/share/gdm/greeter/autostart
  736 tty1     Sl+    0:13 /usr/bin/gnome-shell
  752 tty1     S+     0:00 /usr/bin/Xwayland :1024 -rootless -terminate -core -l
isten 4 -listen 5 -displayfd 6
  782 tty1     Sl     0:00 ibus-daemon --xim --panel disable
```

3. Użyj polecenia: **ps auxw** Dodanie przełącznika **u** spowoduje dodanie informacji na temat użytkownika, z którym dany proces jest skojarzony natomiast przełącznik **w** jest odpowiedzialny za wyświetlenie parametrów linii poleceń procesu (do polowy linii). Aby wyświetlić pełne dane niezależnie od długości skorzystaj z flagi **ww** (polecenie przyjmie postać: **ps auxww**)

```
root@bolek-VirtualBox:~# ps auxw |more
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root           1  0.0  0.4 154692  8452 ?        Ss   11:03   0:06 /sbin/init spla
sh
root           2  0.0  0.0      0     0 ?        S    11:03   0:00 [kthreadd]
root           4  0.0  0.0      0     0 ?        S<   11:03   0:00 [kworker/0:0H]
root           6  0.0  0.0      0     0 ?        S<   11:03   0:00 [mm_percpu_wq]
root           7  0.0  0.0      0     0 ?        S    11:03   0:03 [ksoftirqd/0]
root           8  0.0  0.0      0     0 ?        S    11:03   0:00 [rcu_sched]
root           9  0.0  0.0      0     0 ?        S    11:03   0:00 [rcu_bh]
root          10  0.0  0.0      0     0 ?        S    11:03   0:00 [migration/0]
```

4. Uzyskane wyniki procesów zawęź do konkretnego użytkownika (w przykładzie użytkownik bolek) skorzystaj z polecenia: **ps u -U <nazwa_użytkownika>**

```
root@bolek-VirtualBox:~# ps u -U bolek |more
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
bolek      926  0.0  0.4  80896  8528 ?        Ss   11:05   0:00 /lib/systemd/sy
stemd --user
bolek      927  0.0  0.1 188472  2324 ?        S    11:05   0:00 (sd-pam)
bolek      934  0.0  0.3 430092  8152 ?        Sl   11:05   0:00 /usr/bin/gnome-
keyring-daemon --daemonize --login
bolek      938  0.0  0.2 191944  5832 tty2     Ssl+ 11:05   0:00 /usr/lib/gdm3/g
```

5. Uzyskane wyniki posortuj według użycia procesora można skorzystać z polecenia: **axuk -pcpu** dozwolona jest również komenda: **ps aux --sort -pcpu** Sortowanie jest wykonane malejąco, aby uzyskać efekt odwrotny przed flagą **pcpu** zamiast znaku - (minus) użyj znak + (plus).

```
root@bolek-VirtualBox:~# ps axuk -pcpu |more
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
bolek     1027  9.9 23.3 3181368 477700 tty2   Sl+  11:05  26:09 /usr/bin/gnome-
shell
root        30  0.6  0.0      0     0 ?        S    11:03   1:41 [kworker/0:1]
bolek     1050  0.5  0.6 1170656 12616 ?       S<l  11:05   1:26 /usr/bin/pulsea
udio --start --log-target=syslog
bolek     2709  0.5  2.2 803184 45040 ?        Ssl  12:37   0:57 /usr/lib/gnome-
terminal/gnome-terminal-server
```

Opcji sortowania jest bardzo wiele, wszystko zależy od informacji jakie chcemy uzyskać wszystkie. Uruchomionych i działających procesów w systemie jest bardzo wiele.

6. Odszukaj tylko te, które w danej chwili nas interesują skorzystaj z poleceń przedstawionych na poniższym zrzucie. Przykład odnalezienia procesu przeglądarki Mozilla Firefox oraz edytora tekstu Libre Writer. Na zrzucie poniżej zostały ukazane również numer PID szukanych procesów - odpowiednio dla przeglądarki Mozilla Firefox - **3310** oraz Libre Writer - **3312** Numery tych procesów wykorzystamy za chwilkę przy opisie następnego narzędzia jakim jest program **top**.

```
root@bolek-VirtualBox:~# ps ax |grep firefox
 3310 pts/0    S+     0:00 grep --color=auto firefox
root@bolek-VirtualBox:~# ps ax |grep libre
 3312 pts/0    S+     0:00 grep --color=auto libre
```

Wiele z procesów w systemie Linux jak w każdym innym systemie jest uruchamiana przy starcie systemu inne są uaktywniane już przez samych użytkowników w momencie pracy z systemem i uruchamiania kolejnych zadań czy programów. Uruchomiony proces może być odpowiedzialny za

uruchamianie innych zadań, taki proces staje się wówczas procesem nadrzędnym bądź macierzystym dla procesów, które zostały uruchomione w wyniku jego działania. O procesach, które leżą niżej w hierarchii od procesu macierzystego mówimy, że są to procesy potomne. W systemie Linux takim najważniejszym procesem jest proces **init**, będący rodzicem wszystkich innych procesów. Proces ten ma zawsze przypisaną wartość PID równą 1.

7. Przejrzyj całą hierarchię uruchomionych procesów i dodatkowo poznaj, który proces jest nadrzędny a który potomny skorzystaj z polecenia: **pstree**.

```
root@bolek-VirtualBox:~# pstree
systemd─┬─ModemManager───2*[{ModemManager}]
        ├─NetworkManager─┬─dhclient
        │                └─2*[{NetworkManager}]
        ├─accounts-daemon───2*[{accounts-daemon}]
        ├─acpid
        ├─avahi-daemon───avahi-daemon
        ├─colord───2*[{colord}]
        ├─cron
        ├─cups-browsed───2*[{cups-browsed}]
        ├─cupsd───dbus
        ├─dbus-daemon
        ├─fwupd───4*[{fwupd}]
        ├─gdm3─┬─gdm-session-wor─┬─gdm-wayland-ses─┬─gnome-session-b─┬─gnome-sh+
        │      │                 │                 │                 ├─gsd-a11y+
        │      │                 │                 │                 ├─gsd-a11y+
        │      │                 │                 │                 ├─gsd-clip+
        │      │                 │                 │                 ├─gsd-colo+
```

Narzędzie to dysponuje dodatkowymi przełącznikami, które pozwalają na definicję uzyskiwanych danych.

8. Wyświetl drzewo procesów danego użytkownika – wyświetl procesy użytkownika bolek

```
root@bolek-VirtualBox:~# pstree bolek |more
gdm-wayland-ses-+-gnome-session-b-+-deja-dup-monito---4*[{deja-dup-monito}]
                |                 |-gnome-shell-+-Xwayland
                |                 |             |-ibus-daemon-+-ibus-dconf---3*[
{ibus-dconf}]
                |                 |             |             |-ibus-engine-sim-
--2*[{ibus-engine-sim}]
                |                 |             |             `-2*[{ibus-daemon}
]
                |                 |             `-9*[{gnome-shell}]
                |                 |-gnome-software---3*[{gnome-software}]
                |                 |-gsd-a11y-keyboa---3*[{gsd-a11y-keyboa}]
                |                 |-gsd-a11y-settin---3*[{gsd-a11y-settin}]
                |                 |-gsd-clipboard---2*[{gsd-clipboard}]
                |                 |-gsd-color---3*[{gsd-color}]
                |                 |-gsd-datetime---3*[{gsd-datetime}]
                |                 |-gsd-disk-utilit---2*[{gsd-disk-utilit}]
                |                 |-gsd-housekeepin---3*[{gsd-housekeepin}]
```

9. Wykonaj budowanie drzewa od procesu z zdefiniowanym numerem PID – wyświetl proces sd-pam,

```
root@bolek-VirtualBox:~# pstree 926
systemd─┬─(sd-pam)
        ├─at-spi-bus-laun─┬─dbus-daemon
        │                 └─3*[{at-spi-bus-laun}]
        ├─at-spi2-registr───2*[{at-spi2-registr}]
        ├─dbus-daemon
```

10. Wykonaj budowanie drzewa do informacji o nazwie procesów dołącz informacje o numerach PID,

```
root@bolek-VirtualBox:~# pstree 926 -p
systemd(926)─┬─(sd-pam)(927)
             ├─at-spi-bus-laun(1037)─┬─dbus-daemon(1042)
             │                       ├─{at-spi-bus-laun}(1038)
             │                       ├─{at-spi-bus-laun}(1039)
             │                       └─{at-spi-bus-laun}(1040)
             ├─at-spi2-registr(1044)─┬─{at-spi2-registr}(1045)
             │                       └─{at-spi2-registr}(1046)
             ├─dbus-daemon(940)
             ├─dconf-service(1295)─┬─{dconf-service}(1305)
             │                     └─{dconf-service}(1306)
```

11. Użyj flagi **H** spowoduje podświetlenie procesu

```
root@bolek-VirtualBox:~# pstree -H 926 |more
systemd-+-ModemManager---2*[{ModemManager}]
        |-NetworkManager-+-dhclient
        |                `-2*[{NetworkManager}]
        |-accounts-daemon---2*[{accounts-daemon}]
        |-acpid
        |-avahi-daemon---avahi-daemon
        |-colord---2*[{colord}]
        |-cron
        |-cups-browsed---2*[{cups-browsed}]
        |-cupsd---dbus
        |-dbus-daemon
        |-fwupd---4*[{fwupd}]
        |-gdm3-+-gdm-session-wor-+-gdm-wayland-ses-+-gnome-session-b-+-gnome-she
ll-+-Xwayland
        |      |                 |                 |                 |
   |-ibus-daemon-+-ibus-dconf---3*[{ibus-dconf}]
        |      |                 |                 |                 |
   |             |-ibus-engine-sim---2*[{ibus-engin+
```

12. Spowoduj włączenie sortowanie według numerów PID.

```
root@bolek-VirtualBox:~# pstree -n -p
systemd(1)─┬─systemd-journal(219)
           ├─systemd-udevd(231)
           ├─systemd-timesyn(438)───{systemd-timesyn}(444)
           ├─accounts-daemon(526)─┬─{accounts-daemon}(562)
           │                      └─{accounts-daemon}(571)
           ├─ModemManager(527)─┬─{ModemManager}(564)
           │                   └─{ModemManager}(572)
           ├─dbus-daemon(528)
           ├─NetworkManager(573)─┬─{NetworkManager}(638)
           │                     ├─{NetworkManager}(642)
           │                     └─dhclient(671)
           ├─cupsd(574)───dbus(636)
           ├─udisksd(575)─┬─{udisksd}(622)
           │              ├─{udisksd}(624)
           │              ├─{udisksd}(634)
           │              └─{udisksd}(635)
           ├─avahi-daemon(576)───avahi-daemon(618)
           ├─systemd-logind(587)
           ├─rsyslogd(590)─┬─{rsyslogd}(628)
```

13. Dostępne są jeszcze inne przełączniki działające z narzędziem **pstree** aby je poznać skorzystaj z pomocy programu: **pstree –help** – **udokumentuj czynności**.

Zadaniem aplikacji **top** tak samo jak narzędzia **ps** oraz **pstree** jest przedstawienie nam aktualnego stanu systemu. Stan ten w przeciwieństwie do programu **ps** jest aktualizowany na bieżąco.
Uruchamiając program **top** na żywo mamy podgląd na stan uruchamianych i działających już procesów. Działanie narzędzia kontrolujemy poprzez zastosowanie odpowiednich przełączników, które dodawane są podczas uruchamiania narzędzia, mamy również możliwość definiowania stanu wyświetlanych informacji poprzez dobór odpowiednich skrótów.

14. Uruchom narzędzie **top** bez żadnych parametrów.

```
top - 15:41:31 up  4:38,  1 user,  load average: 0,12, 0,22, 0,21
Zadania:razem: 201, działających:   2, śpiących: 198, zatrzymanych:   1, zom
%CPU: 11,5 uż, 20,6 sy,  0,0 ni, 66,6 be,  0,7 io,  0,0 hi,  0,7 si,  0,0 sk
KiB RAM :  2045156 razem,   302376 wolne,  1140664 użyte,   602116 buf/cache
KiB Swap:   973820 razem,   973820 wolne,        0 użyte.   719436 dost. RAM

  PID UŻYTK.    PR  NI    WIRT    REZ    WSP S %CPU %PAM     CZAS+ KOMENDA
 1027 bolek     20   0 3181496 478228 107316 R 23,2 23,4  27:17.75 gnome-shell
 2709 bolek     20   0  803376  45268  34608 S  2,3  2,2   1:05.05 gnome-termi+
 3329 root      20   0   47788   4056   3312 R  1,0  0,2   0:00.09 top
    1 root      20   0  154692   8452   6324 S  0,7  0,4   0:06.42 systemd
   30 root      20   0       0      0      0 S  0,3  0,0   1:46.64 kworker/0:1
 1242 bolek     20   0  960032  59344  41668 S  0,3  2,9   0:04.22 nautilus-de+
 3314 root      20   0       0      0      0 S  0,3  0,0   0:00.04 kworker/u2:1
    2 root      20   0       0      0      0 S  0,0  0,0   0:00.05 kthreadd
    4 root       0 -20       0      0      0 S  0,0  0,0   0:00.00 kworker/0:0H
    6 root       0 -20       0      0      0 S  0,0  0,0   0:00.00 mm_percpu_wq
    7 root      20   0       0      0      0 S  0,0  0,0   0:03.94 ksoftirqd/0
    8 root      20   0       0      0      0 S  0,0  0,0   0:00.44 rcu_sched
    9 root      20   0       0      0      0 S  0,0  0,0   0:00.00 rcu_bh
   10 root      rt   0       0      0      0 S  0,0  0,0   0:00.00 migration/0
   11 root      rt   0       0      0      0 S  0,0  0,0   0:00.01 watchdog/0
   12 root      20   0       0      0      0 S  0,0  0,0   0:00.00 cpuhp/0
   13 root      20   0       0      0      0 S  0,0  0,0   0:00.00 kdevtmpfs
```

Sposób prezentacji danych uzyskanych dzięki narzędziu **top** możemy kontrolować w dwojaki sposób.

Pierwszy przykład będzie dotyczył wykorzystania numerów PID procesów programów Mozilla Firefox oraz Libre Writer. Numery PID procesów poznaliśmy dzięki narzędziu **ps**.

Aby mieć stały wgląd na to co dzieje się z uruchomionymi aplikacjami włączyć monitorowanie ich stanu poprzez wykorzystanie polecenia: **top -p <PID_procesu>** Po wywołaniu polecenia zostaną wyświetlone tylko informacje odnoszące się do zdefiniowanych numerów PID procesów.

15. Monitoruj stan dwóch aplikacji, użyj polecenia: **top -p 1027,3334** (numery PID oddzielamy przecinkami). Wywołanie komendy spowoduje wyświetlenie stanu procesu przeglądarki Firefox oraz stanu procesu edytora tekstu Writer.

```
root@bolek-VirtualBox:~# top -p 1027,3334

top - 15:47:36 up  4:44,  1 user,  load average: 3,08, 1,58, 0,75
Zadania:razem:   2, działających:   2, śpiących:   0, zatrzymanych:   0, zom
%CPU:  8,1 uż, 91,6 sy,  0,0 ni,  0,3 be,  0,0 io,  0,0 hi,  0,0 si,  0,0 sk
KiB RAM :  2045156 razem,   134548 wolne,  1363892 użyte,   546716 buf/cache
KiB Swap:   973820 razem,   969200 wolne,     4620 użyte.   563840 dost. RAM

  PID UŻYTK.    PR  NI    WIRT    REZ    WSP S %CPU %PAM     CZAS+ KOMENDA
 1027 bolek     20   0 3183672 480080 108356 R 49,3 23,5  28:53.11 gnome-shell
 3334 bolek     20   0 2142544 259196 127308 R 15,3 12,7   0:42.89 firefox
```

16. Włącz monitorowanie procesów konkretnego użytkownika, skorzystaj z flagi **-u** po której to podajemy **nazwę użytkownika**. Wyświetlenie działających procesów użytkownika **bolek** wywołamy po wydaniu komendy: **top -u bolek**

```
root@bolek-VirtualBox:~# top -u bolek

top - 15:49:01 up  4:45,  1 user,  load average: 3,50, 2,08, 1,00
Zadania:razem: 204, działających:   3, śpiących: 200, zatrzymanych:   1, zom
%CPU: 10,5 uż, 88,8 sy,  0,0 ni,  0,3 be,  0,0 io,  0,0 hi,  0,3 si,  0,0 sk
KiB RAM :  2045156 razem,   132068 wolne,  1366140 użyte,   546948 buf/cache
KiB Swap:   973820 razem,   969200 wolne,     4620 użyte.   561608 dost. RAM

  PID UŻYTK.    PR  NI    WIRT    REZ    WSP S %CPU %PAM     CZAS+ KOMENDA
 1027 bolek     20   0 3183672 480136 108356 R 46,4 23,5  29:33.83 gnome-shell
 3421 bolek     20   0 1692620 174800 104432 R 33,1  8,5   1:11.09 Web Content
 3334 bolek     20   0 2142548 259200 127308 S 15,9 12,7   0:56.28 firefox
 1034 bolek     20   0  262552  56156  46528 S  1,3  2,7   0:07.35 Xwayland
 2709 bolek     20   0  803376  44980  34312 S  0,7  2,2   1:09.02 gnome-termi+
  926 bolek     20   0   80896   8492   6596 S  0,0  0,4   0:00.69 systemd
  927 bolek     20   0  188472   2276      0 S  0,0  0,1   0:00.00 (sd-pam)
  934 bolek     20   0  430092   8148   7116 S  0,0  0,4   0:00.37 gnome-keyri+
```

Stan procesów zapisać do zewnętrznego pliku by później zająć się ich analizą do polecenia **top** należy dodać parametr **-b**.

17. Użyj polecenia nakazującego zapis stanu procesów użytkownika **bolek** do zewnętrznego pliku **procesy_user_bolek.txt** - polecenie: **top -b -n 3 -u bolek > procesy_user_bolek.txt** Użyty dodatkowy parametr **-n** nakazuje wykonanie zapisu trzech stanów.

```
root@bolek-VirtualBox:~# top -b -n 3 -u bolek > procesy_user_bolek.txt
root@bolek-VirtualBox:~# gedit procesy_user_bolek.txt
```

```
top - 15:53:25 up  4:50,  1 user,  load average: 3,85, 3,00, 1,65
Zadania:razem: 189, działających:   4, śpiących: 185, zatrzymanych:   0, zom
%CPU:  5,7 uż, 10,8 sy,  0,0 ni, 82,2 be,  0,7 io,  0,0 hi,  0,5 si,  0,0 sk
KiB RAM :  2045156 razem,   109156 wolne,  1376068 użyte,   559932 buf/cache
KiB Swap:   973820 razem,   969200 wolne,     4620 użyte.   562924 dost. RAM

  PID UŻYTK.    PR  NI    WIRT    REZ    WSP S %CPU %PAM     CZAS+ KOMENDA
 3334 bolek     20   0 2152860 275848 128104 R 35,3 13,5   1:48.51 firefox
 3421 bolek     20   0 1697740 180308 104432 R 29,4  8,8   2:36.89 Web Content
 1027 bolek     20   0 3184876 481460 107752 R 23,5 23,5  31:26.41 gnome-shell
  926 bolek     20   0   80896   8492   6596 S  0,0  0,4   0:00.69 systemd
  927 bolek     20   0  188472   2276      0 S  0,0  0,1   0:00.00 (sd-pam)
  934 bolek     20   0  430092   8148   7116 S  0,0  0,4   0:00.37 gnome-keyri+
  938 bolek     20   0  191944   5688   5168 S  0,0  0,3   0:00.02 gdm-wayland+
  940 bolek     20   0   49052   5808   3616 S  0,0  0,3   0:03.64 dbus-daemon
  942 bolek     20   0  738356  18128  15152 S  0,0  0,9   0:00.89 gnome-sessi+
  999 bolek     20   0  286096   7128   6112 S  0,0  0,3   0:00.29 gvfsd
 1004 bolek     20   0  431604   8052   7136 S  0,0  0,4   0:00.03 gvfsd-fuse
 1034 bolek     20   0  263040  56388  46572 S  0,0  2,8   0:10.62 Xwayland
 1037 bolek     20   0  367572   8424   7448 S  0,0  0,4   0:00.03 at-spi-bus-+
 1042 bolek     20   0   47572   4208   3620 S  0,0  0,2   0:00.16 dbus-daemon
 1044 bolek     20   0  220564   7132   6364 S  0,0  0,3   0:00.03 at-spi2-reg+
 1050 bolek      9 -11 1170656  12544   9488 S  0,0  0,6   1:33.17 pulseaudio
```

Dodatkowy przełącznik **-d** pozwala na zdefiniowanie czasu odświeżania i tak aby np. odświeżyć widok co 2 sekundy 10 razy skorzystaj z polecenia: **top -n 10 -d 2** Parametr **-i** odpowiedzialny jest za wyłączenie pokazywania nieaktywnych procesów zaś flaga **-H** umożliwia włączenia pokazywania wątków procesów.

Podczas pracy narzędzia **top** możliwe jest skorzystanie z skrótów, które pozwolą nam w sposób interakcyjny wpłynąć na wynik uzyskiwanych informacji. Co warto zaznaczyć podczas zabawy z programem **top** możemy spowodować, że system nasz zacznie pracować niestabilnie. Aby wykluczyć opcję takiego zachowania systemu narzędzie **top** możemy uruchomić w trybie bezpiecznym w którym to opcje potencjalnie niebezpieczne zostaną wyłączone. Aby uruchomić **top** w trybie secure wydaj polecenie: **top -s**

Podczas działania narzędzia **top** następujące skróty (klawisze) są aktywne:

h - wyświetla pomoc programu,

d (bądź s) - zmienia czas odświeżania,

spacja - natychmiastowe odświeżenie ekranu,

strzałki - przewijanie ekranu (dostępne są również klawisze: Page Up, Page Down, Home oraz End),

n - pozwala na zdefiniowanie ilości wyświetlanych procesów (0 - pokazuje wszystkie),

f - pozwala na zdefiniowanie wyświetlanych informacji oraz na sposobie ich reprezentacji. Po wciśnięciu **f** na nowo otwartym ekranie za pomocą **spacji** włączamy/wyłączamy określony parametr procesu zaś po wybraniu **prawej strzałki** dany element możemy przemieszczać definiując w ten sposób kolejność prezentowania danych (sortowanie wyłączamy **lewą strzałką**),

M - sortowanie procesów według zajmowanej pamięci,

N - sortowanie procesów według PID-u,

P - sortowanie procesów według użycia CPU (domyślne),

T - sortowanie procesów według czasu,

k - zabicie procesu. Po wciśnięciu klawisza zostaniemy poproszeni o podanie numeru PID interesującego nas procesu a w kroku kolejnym będziemy musieli określić sygnał jaki ma zostać wysłany by dany proces zamknąć. Domyślna wartość sygnału to 15, jeżeli użyjesz wartości 9 zabicie procesu będzie bardziej „brutalne”,

r - zmienia wartość priorytetu (nice) działającego procesu, ustalenie wartości poniżej 0 wymaga uruchomienia narzędzia przez użytkownika root (o priorytetach procesów dowiesz się więcej za chwilę),

W - zapisanie konfiguracji narzędzia **top**,

A - przełącza narzędzie **top** do widoku wielu okien (4) w każdym oknie możemy zdefiniować inne zasady wyświetlania informacji (okno wybieramy przy użyciu klawisza **g**).

Wszystkie informacje pozyskane po uruchomieniu aplikacji **top** są zgrupowane w kolumnach.
Nad definicją nagłówków kolumn tabeli mamy ukazane informacje ogólne.

```
top - 15:59:33 up  4:56,  1 user,  load average: 3,03, 3,04, 2,13
Zadania:razem: 192, działających:   3, śpiących: 189, zatrzymanych:   0, zom
%CPU: 16,6 uż, 83,4 sy,  0,0 ni,  0,0 be,  0,0 io,  0,0 hi,  0,0 si,  0,0 sk
KiB RAM :  2045156 razem,   121132 wolne,  1393996 użyte,   530028 buf/cache
KiB Swap:   973820 razem,   969200 wolne,     4620 użyte.   537092 dost. RAM

  PID UŻYTK.    PR  NI    WIRT    REZ    WSP S %CPU %PAM     CZAS+ KOMENDA
 1027 bolek     20   0 3194192 490508 109168 R 44,9 24,0  33:36.03 gnome-shell
 3421 bolek     20   0 1706956 187064 104392 R 32,8  9,1   4:44.10 Web Content
 3334 bolek     20   0 2136644 274292 128452 S 19,0 13,4   2:48.04 firefox
 1034 bolek     20   0  263040  56572  46580 S  1,3  2,8   0:14.53 Xwayland
 2709 bolek     20   0  805820  47332  35744 S  1,3  2,3   1:13.46 gnome-termi+
 3675 root      20   0   47672   3832   3260 R  0,3  0,2   0:00.02 top
    1 root      20   0  154692   8404   6276 S  0,0  0,4   0:06.52 systemd
    2 root      20   0       0      0      0 S  0,0  0,0   0:00.05 kthreadd
    4 root       0 -20       0      0      0 S  0,0  0,0   0:00.00 kworker/0:0H
    6 root       0 -20       0      0      0 S  0,0  0,0   0:00.00 mm_percpu_wq
    7 root      20   0       0      0      0 S  0,0  0,0   0:04.93 ksoftirqd/0
    8 root      20   0       0      0      0 S  0,0  0,0   0:00.52 rcu_sched
    9 root      20   0       0      0      0 S  0,0  0,0   0:00.00 rcu_bh
   10 root      rt   0       0      0      0 S  0,0  0,0   0:00.00 migration/0
```

Wśród ukazanych danych odszukaj informację o:

a) Linia pierwsza: **top** - to informacja o aktualnej dacie i godzinie; czas uruchomienia systemu (up 14 min); liczbę użytkowników (2 users); średnie obciążenie (load average),

b) Linia druga: **Tasks** - to informacja o liczbie wszystkich zadań (160 total); zadań aktualnie uruchomionych (1 running); zadań uśpionych (159 sleeping); zadań zatrzymanych (0 stopped); zadań zombie - nieprawidłowo zamkniętych (0 zombie),

c) Linia trzecia: **%Cpu(s)** - to informacje odnoszące się do procesora: us - czas użytkownika; sy - czas systemu; ni - czas nice; id - czas bezczynności; wa - czas operacji wejścia/wyjścia; hi - czas obsługi przerywań sprzętowych; si - czas obsługi przerywań programowych,

d) Linia czwarta: **Mem** - to informacja o pamięci (jednostka kB): pamięć całkowita (total); pamięć wykorzystana (used); pamięć wolna (free); bufor (buffers),

e) Linia piąta: **Swap** - to informacja o pliku wymiany (jednostka kB): rozmiar całkowity (total); użycie (used); wolne (free); pamięć podręczna (cached Mem).

N nagłówki tabeli dostarczają nam danych (informacje te odnoszą się do konkretnego zadania; część informacji domyślnie jest wyłączona, aby włączyć ich wyświetlanie skorzystaj z klawisza f):

PID - identyfikator procesu,

PPID - identyfikator procesu rodzica,

USER - właściciel procesu/zadania,

UID - identyfikator właściciela procesu (0 oznacza użytkownika root),

PR - priorytet zadania,

NI - wartość parametru nice,

VIRT - wykorzystywany rozmiar pamięci wirtualnej,

RES - wykorzystywany rozmiar pamięci fizycznej,

SHR - wykorzystywany rozmiar pamięci współdzielonej,

S - stan procesu - poszczególne oznaczenia literowe (podobnie jak w przypadku narzędzia **ps**) oznaczają: S - proces uśpiony; Z - proces zombie; R - proces uruchomiony; T - proces zatrzymany,

%CPU - procent czasu procesora,

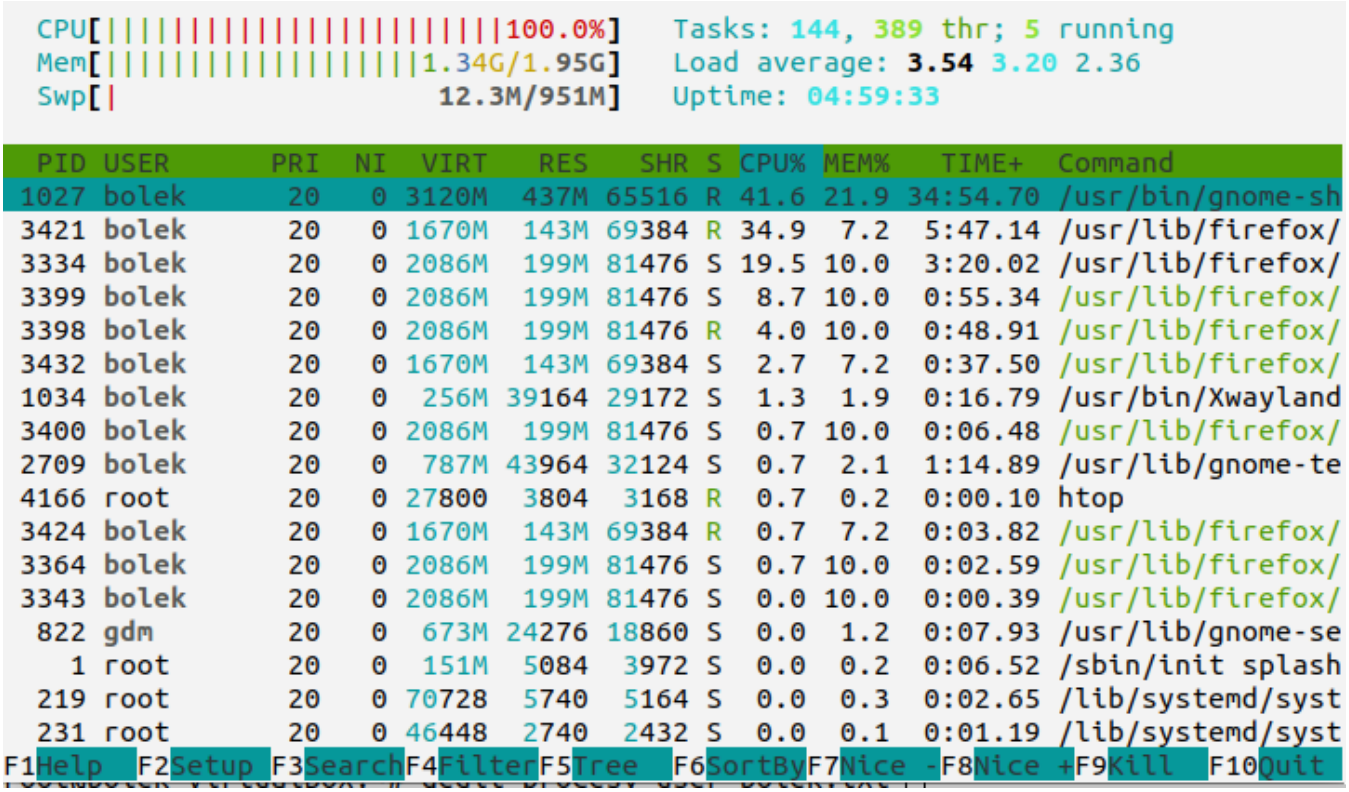
%MEM - procent wykorzystywanej przez proces pamięci fizycznej,

TIME+ - całkowity czas procesora poświęcony zadaniu,

COMMAND - polecenie uruchamiające dane zadanie bądź proces.

18. Użyj narzędzia **htop**. Program również w czasie rzeczywistym pokazuje nam stan uruchomionych procesów.

```
root@bolek-VirtualBox:~# apt-get install htop
```



Narzędzie to umożliwia nam sterowanie wyświetlanymi danymi za pomocą klawiszy funkcyjnych od F1 do F10. Program daje nam możliwość wyszukania danego procesu, użycia filtrów, ukazania drzewa procesów, sortowania, zmianę parametru nice a także zabicie danego procesu. Dodatkowo nad tabelą przedstawione są dane ogólne – tj.: użycie procesora; wykorzystanie pamięci i pliku wymiany; liczbę uruchomionych zadań; średnie obciążenie systemu po upływie 1, 5 i 15 minut; czas pracy systemu.

Uruchomienie narzędzia poprzez dodanie w wierszu poleceń odpowiednich parametrów umożliwia nam wstępne zdefiniowanie uzyskiwanych informacji:

-d <sekunda> - określenie czasu odświeżania,

-u <użytkownik> - wyświetlenie procesów zdefiniowanego użytkownika,

-p <PID> - wyświetlenie zadania o określonej wartości numeru PID (gdy chcesz wyświetlić więcej zadań PID-y oddziel od siebie przecinkami),

-s <rodzaj_sortowania> - wyświetlenie danych posortowanych według ustalonej kolumny, pełna nazwa kolumn dostępna po wydaniu polecenia: **htop --sort-key=help**

Dużą zaletą narzędzia (niż omówionego programu **top**) jest w pełni interaktywny sposób zarządzania procesami za pomocą klawiszy funkcyjnych.

Innymi programami, którymi możemy posłużyć się celem obserwacji uruchomionych procesów są programy **ksysguard**, który jest dedykowany dla środowiska KDE oraz **gtop** przeznaczony dla środowiska GNOM.

W wpisie tym kilka razy pojawił się termin **nice**, parametr ten określa z jaki priorytetem jest uruchomiony dany proces. Zakres tego priorytetu (ang. niceness level) rozciąga się od -20 do 19, przy czym wartość 20 oznacza priorytet najwyższy zaś wartość 19 priorytet najniższy. Aby zmodyfikować wartość z jakim dane zadanie jest uruchamiane możemy skorzystać z narzędzia **nice**.

Użycie narzędzia jest bardzo proste wystarczy wydać polecenie: **nice <nazwa_zadania>**
Wydanie komendy **nice mcedit** spowoduje uruchomienie programu mcedit.

19. W pierwszej kolejności za pomocą polecenia: **ps auxw | grep mcedit** zidentyfikuj numer procesu PID uruchomionego programu mcedit. A następnie za pomocą komendy: **top -p <numer_PID> -n 1 -b** wyświetl dane dotyczące procesu o zadanym numerze PID (użyta flaga **-b** w narzędziu **top** spowoduje wyświetlenie informacji narzędzia **top** bez czyszczenia okna). Jak widać po poniższym zrzucie w poleceniu została użyta wartość 2754 ponieważ jest to PID zadania mcedit.

```
root@bolek-VirtualBox:~# ps auxw |grep mcedit
root      4168  0.0  0.0  15576  1024 pts/0    S+   16:04   0:00 grep --color=au
to mcedit
root@bolek-VirtualBox:~# top -p 2754 -n 1 -b
top - 16:04:36 up  5:01,  1 user,  load average: 3,85, 3,40, 2,53
Zadania:razem:   0, działających:   0, śpiących:   0, zatrzymanych:   0, zom
%CPU:  6,0 uż, 13,5 sy,  0,0 ni, 79,2 be,  0,7 io,  0,0 hi,  0,5 si,  0,0 sk
KiB RAM :  2045156 razem,   286200 wolne,  1381808 użyte,   377148 buf/cache
KiB Swap:   973820 razem,   961264 wolne,    12556 użyte.   480760 dost. RAM

  PID UŻYTK.    PR  NI    WIRT    REZ    WSP S %CPU %PAM     CZAS+ KOMENDA
```

Z zrzutu tego uzyskamy również informację o poszukiwanym parametrze nice. wartość parametru nice zadania mcedit wynosi **10**. Wartość ta jest domyślną wartością z jaką będą uruchamiane programy z wykorzystaniem narzędzia **nice**.

20. Zapisz pytanie i odpowiedz. Czy proces poznania wartości przypisanej do parametru nice musi być wykonany w ten sposób, czy nie można tego wykonać prościej? Odpowiedź – Można wykonać to

łatwiej przy użyciu tylko jednego narzędzia jakim jest **ps**. Tak więc ustalony parametr **nice** poznasz szybciej dołączając do narzędzia **ps** flagę **-l**

21. Wydaj polecenia: **ps -l a | grep mcedit** stosowna informacja zostanie zawarta w wyświetlonych danych. Jak można zauważyć wartość parametru **nice** poznana po wydaniu polecenia pokrywa się z tą poznaną dzięki narzędziu **top** i wynosi **10**.

```
root@bolek-VirtualBox:~# ps -l a | grep mcedit
0 S     0  4182  3585  0  80   0 -  3894 pipe_w pts/0      0:00 grep --color=aut
o mcedit
```

Użycie przełącznika **-l** w narzędziu **ps** pozwala na dołączenie do już wyświetlanych danych informacji dodatkowych (bardziej szczegółowych). Dodatkowe kolumny, które zostaną zawarte po użyciu flagi **-l** to informacje o:

UID - identyfikator właściciela procesu,

PPID - identyfikator procesu macierzystego,

PRI i **NI** - priorytety procesów,

WCHAN - pozwala sprawdzić jaką funkcję systemowy jądra wywołał proces.

Wywołanie narzędzia **nice** bez zdefiniowanie parametru procesu włącza dane zadanie z domyślnie zdefiniowaną wartością parametru nice ustawioną na 10. A co gdybyśmy dane zadanie chcieli uruchomić z wartością niższą bądź wyższą? A by wykonać to zadanie do składni polecenia wywołującego dany program musimy dołączyć parametr **-n** po którym to definiujemy wartość nice.

22. Użyj komendy: **nice -n -20 mcedit** spowoduje uruchomienie narzędzia mcedit z możliwie najwyższym do zdefiniowania priorytetem - 20.
23. Sprawdź efekt jak na zrzucie przedstawiono efekt wydanej komendy, jak widać sprawdzenie utwierdza w przekonaniu, że narzędzie mcedit zostało uruchomione z najwyższą wartością parametru nice.

```
root@bolek-VirtualBox:~# ps auxw |grep mcedit
root      4184  0.0  0.0  15576  1176 pts/0    S+   16:06   0:00 grep --color=au
to mcedit
root@bolek-VirtualBox:~# top -p 2499 -b -n 1
top - 16:07:26 up  5:04,  1 user,  load average: 4,41, 3,79, 2,83
Zadania:razem:   0, działających:   0, śpiących:   0, zatrzymanych:   0, zom
%CPU:  6,1 uż, 14,2 sy,  0,1 ni, 78,5 be,  0,7 io,  0,0 hi,  0,5 si,  0,0 sk
KiB RAM :  2045156 razem,   263264 wolne,  1402880 użyte,   379012 buf/cache
KiB Swap:   973820 razem,   961264 wolne,    12556 użyte.   459784 dost. RAM

  PID UŻYTK.    PR  NI    WIRT    REZ    WSP S %CPU %PAM     CZAS+ KOMENDA
root@bolek-VirtualBox:~# ps -l a |grep mcedit
0 S     0  4188  3585  0  80   0 -  3894 pipe_w pts/0      0:00 grep --color=aut
o mcedit
```

24. Zmień priorytet już uruchomionego zadania.

Aby wykonać to zadanie musimy skorzystać z polecenia: **renice <wartość_parametru_nice> <PID_procesu>** zmień wartość parametru nice z 20 na 10 uruchomionego zadania mcedit o numerze PID równym 1176 należy wydać polecenie: **renice +10 1176** Po wydaniu komendy sprawdźmy efekt jej wydania.

```
root@bolek-VirtualBox:~# renice +10 1176
1176 (process ID) old priority 0, new priority 10
root@bolek-VirtualBox:~# ps auxw |grep mcedit
root      4212  0.0  0.0  15576  1072 pts/0    S+   16:12   0:00 grep --color=au
to mcedit
root@bolek-VirtualBox:~# top -p 1176 -b -n 1
top - 16:31:22 up  5:28,  1 user,  load average: 2,59, 2,03, 2,44
Zadania:razem:   1, działających:   0, śpiących:   1, zatrzymanych:   0, zom
%CPU:  6,7 uż, 18,9 sy,  0,1 ni, 73,2 be,  0,6 io,  0,0 hi,  0,5 si,  0,0 sk
KiB RAM :  2045156 razem,   275076 wolne,  1365800 użyte,   404280 buf/cache
KiB Swap:   973820 razem,   961264 wolne,    12556 użyte.   518244 dost. RAM

  PID UŻYTK.    PR  NI    WIRT    REZ    WSP S %CPU %PAM     CZAS+ KOMENDA
 1176 bolek     30  10  368736   8704   7732 S  0,0  0,4   0:02.19 gdbus
root@bolek-VirtualBox:~# ps -l a | grep mcedit
0 S     0  4250  3585  0  80   0 -  3894 pipe_w pts/0      0:00 grep --color=aut
o mcedit
```

Jak można zauważyć powyżej wydanie polecenia **renice** zmieniło priorytet zadania mcedit.
Nowa wartość parametru nice wynosi 10.

25. Zmień parametru nice z 10 na -5 (pamiętaj, że zmiana wartości poniżej 0 wymaga praw użytkownika root).

```
root@bolek-VirtualBox:~# renice -5 1176
1176 (process ID) old priority 10, new priority -5
root@bolek-VirtualBox:~# top -p 1176 -b -n 1
top - 16:33:42 up  5:30,  1 user,  load average: 3,06, 2,40, 2,52
Zadania:razem:   1, działających:   0, śpiących:   1, zatrzymanych:   0, zom
%CPU:  6,7 uż, 19,4 sy,  0,1 ni, 72,7 be,  0,6 io,  0,0 hi,  0,5 si,  0,0 sk
KiB RAM :  2045156 razem,   269736 wolne,  1369216 użyte,   406204 buf/cache
KiB Swap:   973820 razem,   961264 wolne,    12556 użyte.   513256 dost. RAM

  PID UŻYTK.    PR  NI    WIRT    REZ    WSP S %CPU %PAM     CZAS+ KOMENDA
 1176 bolek     15  -5  368736   8704   7732 S  0,0  0,4   0:02.19 gdbus
```

Nie jest to jedyny sposób dokonania modyfikacji parametru nice. Narzędzia **top** oraz **htop** również oferują taką możliwość.

26. Zredukuj ilość danych ograniczmy się do wyświetlenia informacji dotyczących tylko zadania mcedit. Aby to wykonać wydajmy polecenie: **top -p 1176**
    Po wydaniu polecenia będziemy mieli wgląd w parametry tylko tego zadania.

Po uruchomieniu narzędzia **top** aby dokonać modyfikacji parametru nice wciskamy klawisz **r**.
Na postawione pytanie – o numer procesu PID klikamy **enter** (numer PID zadania mcedit jest ustawiony domyślnie) gdyby tak się nie stało wprowadzamy go ręcznie.

```
top - 16:37:58 up  5:34,  1 user,  load average: 3,31, 2,90, 2,69
Zadania:razem:   1, działających:   0, śpiących:   1, zatrzymanych:   0, zom
%CPU: 12,2 uż, 81,0 sy,  6,4 ni,  0,3 be,  0,0 io,  0,0 hi,  0,0 si,  0,0 sk
KiB RAM :  2045156 razem,   263544 wolne,  1374840 użyte,   406772 buf/cache
KiB Swap:   973820 razem,   961264 wolne,    12556 użyte.   507256 dost. RAM
Zmiana nice PID-u 5 na wartość
  PID UŻYTK.    PR  NI    WIRT    REZ    WSP S %CPU %PAM     CZAS+ KOMENDA
 1176 bolek     15  -5  368736   8704   7732 S  0,0  0,4   0:02.20 gdbus
```

Po podaniu informacji o numerze PID definiujemy wartość parametru nice na wartość 5.

27. Sprawdź stan parametru nice za pomocą znanych nam poleceń. Jak widać poniżej wartość nice wynosi 5.

```
root@bolek-VirtualBox:~# ps auxw |grep mcedit
root      4260  0.0  0.0  15576   976 pts/0    S+   16:40   0:00 grep --color=au
to mcedit
root@bolek-VirtualBox:~# top -p 1176 -b -n 1
top - 16:40:33 up  5:37,  1 user,  load average: 3,01, 2,93, 2,74
Zadania:razem:   1, działających:   0, śpiących:   1, zatrzymanych:   0, zom
%CPU:  6,9 uż, 20,7 sy,  0,1 ni, 71,2 be,  0,6 io,  0,0 hi,  0,5 si,  0,0 sk
KiB RAM :  2045156 razem,   257096 wolne,  1381288 użyte,   406772 buf/cache
KiB Swap:   973820 razem,   961264 wolne,    12556 użyte.   500808 dost. RAM

  PID UŻYTK.    PR  NI    WIRT    REZ    WSP S %CPU %PAM     CZAS+ KOMENDA
 1176 bolek     15  -5  368736   8704   7732 S  0,0  0,4   0:02.20 gdbus
root@bolek-VirtualBox:~# ps -l a | grep mcedit
0 S     0  4263  3585  0  80   0 -  3894 pipe_w pts/0      0:00 grep --color=aut
o mcedit
```

Zdarza się, że na wskutek nieprawidłowego działania programu musimy zakończyć jego proces. Z reguły uruchamiając jakieś zadanie w terminalu, aby je zakończyć należy posłużyć się kombinacją klawiszy Ctrl+C lecz w pewnych sytuacjach zabieg ten może okazać się niewystarczający. Wtedy musimy proces zakończyć ręcznie. Aby zakończyć zadanie należy skorzystać z polecenia:
kill <nazwa_bądź_numer_sygnału> PID

28. Z uruchomionymi zadaniami możemy porozumieć się poprzez wysłanie do nich odpowiedniego sygnału. Sygnały te są określone i zdefiniowane (nazwa + numer) a poznaj pełną ich listę do użycia z narzędziem **kill** poznamy po wydaniu polecenia: **kill -l**

```
root@bolek-VirtualBox:~# kill -l
 1) SIGHUP       2) SIGINT       3) SIGQUIT      4) SIGILL       5) SIGTRAP
 6) SIGABRT      7) SIGBUS       8) SIGFPE       9) SIGKILL     10) SIGUSR1
11) SIGSEGV     12) SIGUSR2     13) SIGPIPE     14) SIGALRM     15) SIGTERM
16) SIGSTKFLT   17) SIGCHLD     18) SIGCONT     19) SIGSTOP     20) SIGTSTP
21) SIGTTIN     22) SIGTTOU     23) SIGURG      24) SIGXCPU     25) SIGXFSZ
26) SIGVTALRM   27) SIGPROF     28) SIGWINCH    29) SIGIO       30) SIGPWR
31) SIGSYS      34) SIGRTMIN    35) SIGRTMIN+1  36) SIGRTMIN+2  37) SIGRTMIN+3
38) SIGRTMIN+4  39) SIGRTMIN+5  40) SIGRTMIN+6  41) SIGRTMIN+7  42) SIGRTMIN+8
43) SIGRTMIN+9  44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12 47) SIGRTMIN+13
48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14 51) SIGRTMAX-13 52) SIGRTMAX-12
53) SIGRTMAX-11 54) SIGRTMAX-10 55) SIGRTMAX-9  56) SIGRTMAX-8  57) SIGRTMAX-7
58) SIGRTMAX-6  59) SIGRTMAX-5  60) SIGRTMAX-4  61) SIGRTMAX-3  62) SIGRTMAX-2
63) SIGRTMAX-1  64) SIGRTMAX
```

Zapisz najczęściej używane sygnały:

sygnał 2 - SIGINT - przerwanie wykonania procesu wywołane wciśnięciem klawiszy Ctrl + C,

Sygnał 3 - SIGQUIT - przerwanie wykonania procesu wywołane wciśnięciem klawiszy Ctrl + \ sygnał ten dodatkowo nakazuje wykonanie obrazu pamięci procesu,

sygnał 9 - SIGKILL - zabicie procesu,

sygnał 15 - SIGTERM - miękkie zakończenie procesu, sygnał ten jest domyślnie zdefiniowany gdy nie wskażemy w poleceniu innego,

sygnał 19 - SIGSTOP - zatrzymanie wykonywania procesu wywołane wciśnięciem klawiszy Ctrl + Z, tak zatrzymany proces możemy następnie wznowić wykorzystując do tego narzędzie **fg** bądź **bg**.

29. Zabji dany proces, wydaj polecenie: **kill <PID>** poniżej przykład zabicia procesu przeglądarki Firefox.

```
root@bolek-VirtualBox:~# pgrep -l firefox
3334 firefox
root@bolek-VirtualBox:~# kill 3334
root@bolek-VirtualBox:~#
```

Do zidentyfikowania numeru PID przeglądarki zostało wykorzystane narzędzie **pgrep**.
Narzędzie **pgrep** zostało wywołane z parametrem **-l** który jest odpowiedzialny za wyświetlenie pełnej nazwy procesu. Narzędzie to świetnie sprawdza się w sytuacji, w której znamy nazwę uruchomionego procesu. Wywołanie polecenie bez flagi **-l** spowoduje wyświetlenie samej wartości numeru procesu.

30. Wyślij określony sygnał do procesu w poleceniu **kill** zdefiniuj jego wartość. Wartość ta może być podana w sposób numeryczny bądź słowny. Aby nakazać wysłanie sygnału SIGKILL, zabijając w ten sposób proces możemy skorzystać z polecenia przedstawionego na poniższym zrzucie (tradycyjnie już zabijamy proces przeglądarki).

```
root@bolek-VirtualBox:~# pgrep -l firefox
4557 firefox
root@bolek-VirtualBox:~# kill -9 4557
root@bolek-VirtualBox:~#
```

Aby zakończyć wszystkie procesy o danej nazwie należy skorzystać z polecenia:
killall <nazwa_procesu>.

Narzędzie to posiada kilka przydatnych przełączników:

-u <użytkownik> <proces> - zabicie wszystkich procesów danego użytkownika, brak definicji procesu spowoduje zabicie wszystkich procesów zdefiniowanego użytkownika,

-o <czas> - zabicie procesów starszych niż zdefiniowany czas,

-y <czas> - zabicie procesów młodszych niż zdefiniowany czas. Czas określamy za pomocą następujących zmiennych: **s** dla sekund; **m** dla minut; **h** dla godzin; **d** dla dni; **w** dla tygodni; **M** dla miesięcy; **y** dla lat,

-s <sygnał> - określenie wysyłanego sygnału zamiast domyślnego SIGTERM,

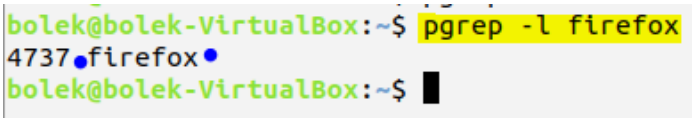
-I - brak rozróżniania wielkości liter dla definiowanych procesów.

Narzędziem pozwalającym nam na zabicie procesu bez znajomości jego PID-u lecz z wykorzystaniem nazwy procesu jest program **pkill**. Zabicie danego programu odbywa się po wydaniu komendy:
pkill <nazwa_procesu>

Zabicie danego procesu możemy również wykonać za pomocą omówionych narzędzi **top** oraz **htop**.

31. Wykonaj ls /proc oraz pgrep – l firefox celem poznania numeru PID przeglądarki Firefox.

```
root@bolek-VirtualBox:~# ls /proc
1     1139  1343  231   4737  717  831        consoles     modules
10    1140  139   24    4797  723  837        cpuinfo      mounts
1004  1141  14    25    4846  725  839        crypto       mtrr
1027  1142  1482  259   526   727  843        devices      net
1034  1151  15    26    527   736  844        diskstats    pagetypeinfo
1037  1155  1505  27    528   741  847        dma          partitions
1042  1157  1555  2709  573   752  848        driver       sched_debug
1044  1164  156   28    574   757  849        execdomains  schedstat
1050  1165  157   293   575   76   850        fb           scsi
1060  1166  158   30    576   762  862        filesystems  self
1061  1167  159   3180  587   765  869        fs           slabinfo
1065  1169  16    3196  590   769  885        interrupts   softirqs
1067  1170  1608  32    593   77   9          iomem        stat
107   12    163   3209  594   770  905        ioports      swaps
1083  1230  1638  33    596   782  923        irq          sys
1089  1231  1696  34    6     785  926        kallsyms     sysrq-trigger
1094  1238  17    3572  603   788  927        kcore        sysvipc
1099  1239  1705  3584  618   8    934        keys         thread-self
11    1242  18    3585  619   802  938        key-users    timer_list
1103  1249  185   4     636   803  940        kmsg         tty
1107  1255  186   4195  646   804  942        kpagecgroup  uptime
1111  1262  19    4224  652   806  999        kpagecount   version
1123  1287  2     425   662   808  acpi       kpageflags   version_signature
```



W systemie Linux występuje katalog /proc W katalogu tym zawarte są informacje o działających procesach. Reprezentacja danego procesu jest realizowana poprzez zapis katalogu. Nazwa katalogu odpowiada numerowi PID procesu. W danym katalogu są umieszczone pliki, które opisują status danego procesu, np.: plik cmdline zawiera polecenie uruchamiające dany proces a plik status informacje o stanie procesu. Proces przeglądarki Firefox o numerze PID 4737 ma swoje odzwierciedlenie w folderze /proc w formie katalogu o nazwie 4737.

System plików **/proc** jest całkowicie wirtualny co oznacza, że nie jest on nigdzie zapisany na dysku a istnieje tylko w pamięci. Za utworzenie i zapis informacji odpowiada jądro systemu a jego głównym zadaniem jest dostarczenie informacji o systemie.

Poniżej przedstawiam listę co ważniejszych pozycji znajdujących się w **/proc**.

/proc/cpuinfo - wykaz informacji o procesorze: model, wydajność, itp.,

/proc/consoles - informacja o terminalach,

/proc/crypto - zabezpieczenia kryptograficzne,

/proc/devices - lista urządzeń,

/proc/filesystems - informacja o dostępnych systemach plików,

/proc/interrupts - lista przerwań wykorzystywanych przez system,

/proc/ioports - porty wejścia/wyjścia,

/proc/kcore - obraz pamięci systemu,

/proc/meminfo - wykaz użycia pamięci fizycznej i swapu.

/proc/modules - lista modułów wykorzystywana przez system,

/proc/net - informacje o protokołach sieciowych,

/proc/partitions - lista partycji,

/proc/stat -statystyki systemu,

/proc/uptime - czas pracy systemu,

/proc/version - wersja jądra,

/proc/vmstat - statystyki użycia pamięci wirtualnej.

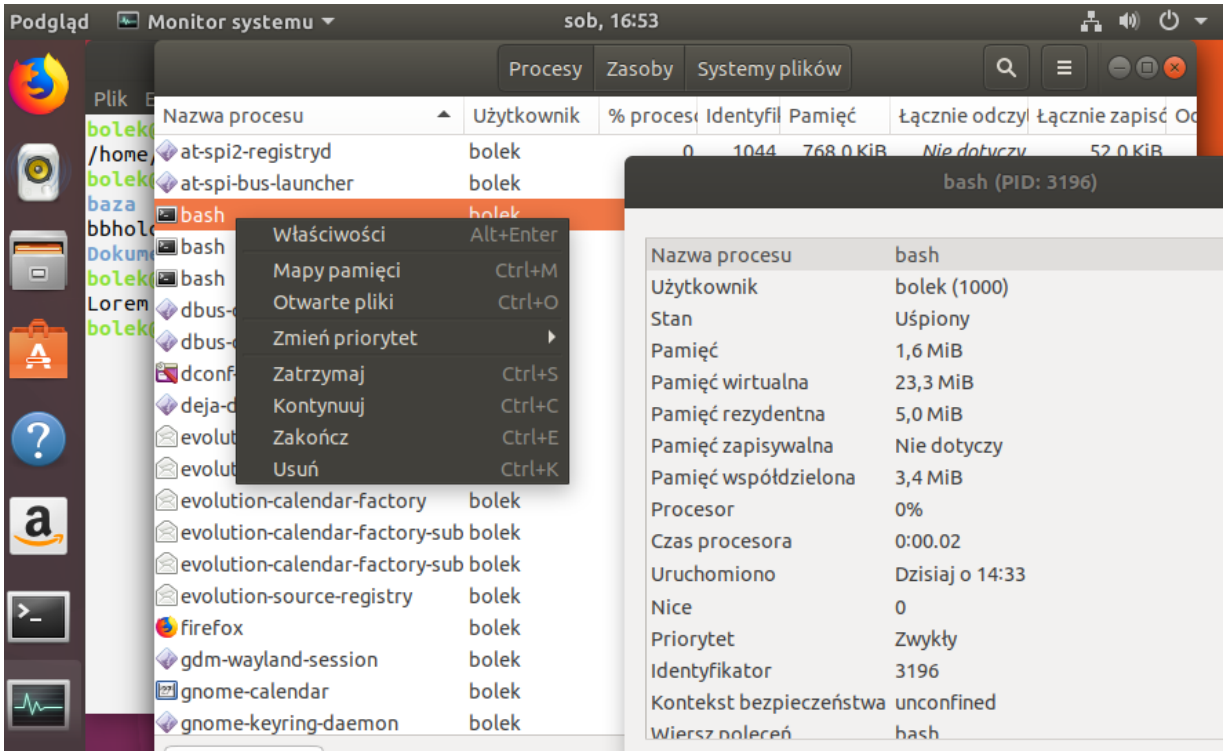
Informacje zawarte w tych plikach można odczytywać np. przy pomocy narzędzia **cat** lecz nie należy ich edytować gdyż może to doprowadzić do niestabilnej pracy systemu.

32. Użyj narzędzia **cat** celem uzyskania informacji o wykorzystaniu pamięci.

```
root@bolek-VirtualBox:~# cat /proc/meminfo
MemTotal:        2045156 kB
MemFree:          179892 kB
MemAvailable:     541604 kB
Buffers:           17764 kB
Cached:           442708 kB
SwapCached:         3944 kB
Active:          1206108 kB
Inactive:         490684 kB
Active(anon):     867900 kB
Inactive(anon):   354620 kB
Active(file):     338208 kB
Inactive(file):   136064 kB
Unevictable:          16 kB
Mlocked:              16 kB
SwapTotal:        973820 kB
SwapFree:         961520 kB
Dirty:                12 kB
Writeback:             0 kB
AnonPages:       1232896 kB
Mapped:           219088 kB
Shmem:             38352 kB
Slab:              68444 kB
```

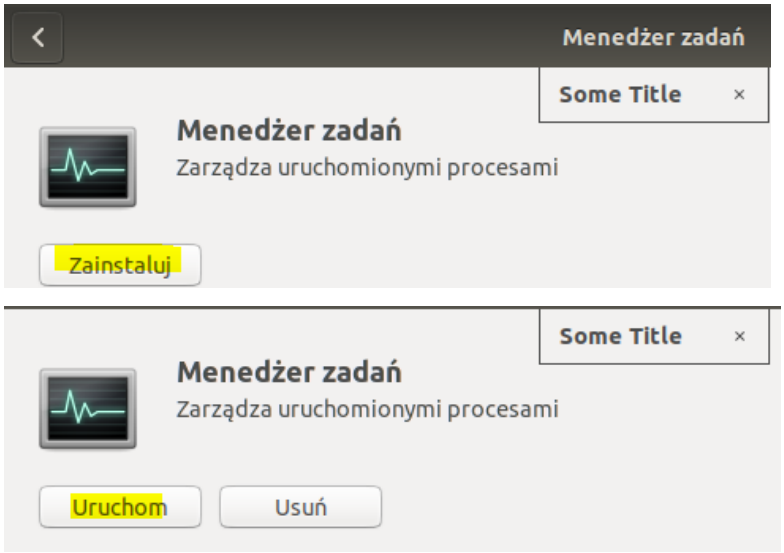
Szereg narzędzi korzysta z informacji zawartych w **/proc**. Przeglądając poszczególne pliki możemy uzyskać bardzo wiele informacji o naszym systemie choć co trzeba przyznać forma prezentacji tych danych jest czasem mało atrakcyjna.

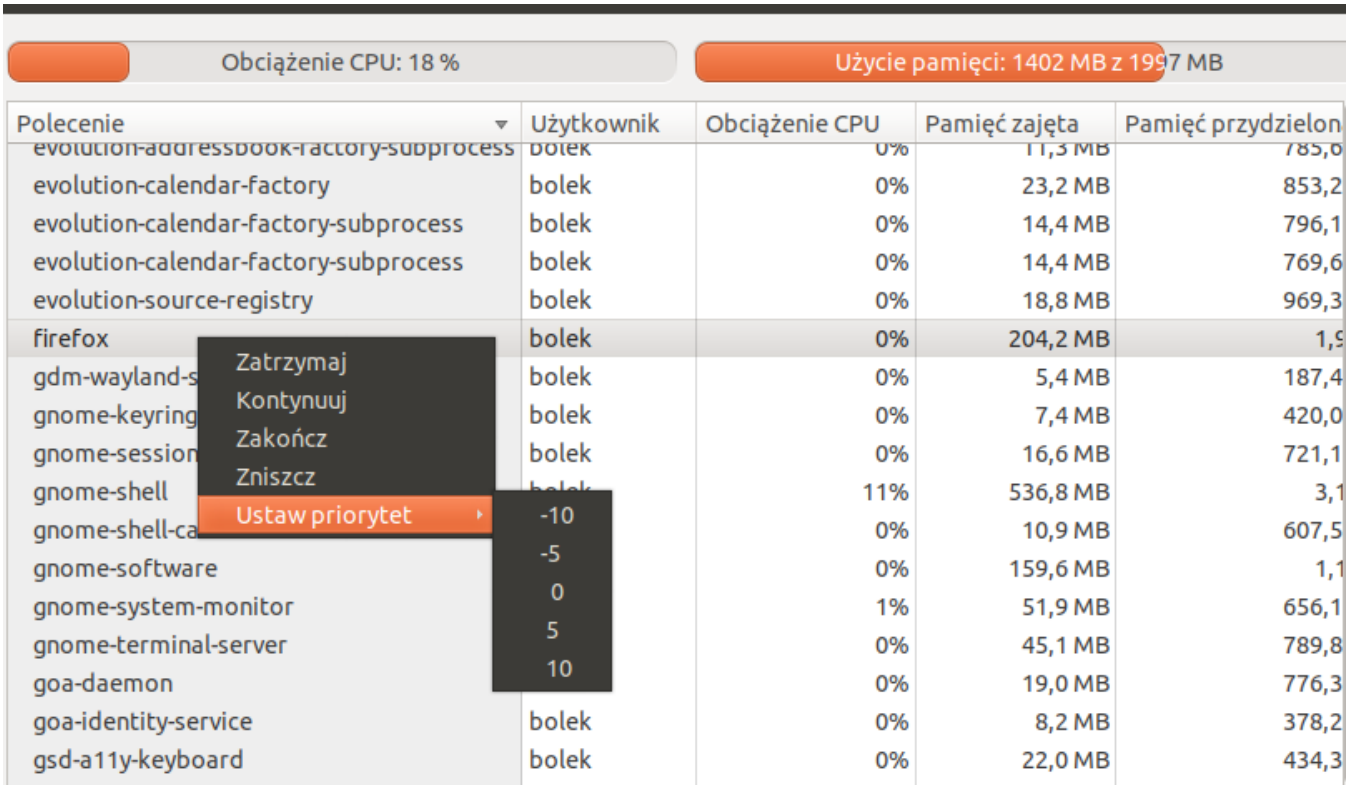
33. Użyj zarządzania procesami w trybie graficznym. **gnome-system monitor** (by wywołać program w terminalu należy wpisać jego nazwę).



Monitor systemu pozwala nam na: zatrzymanie i wznowienie danego procesu, zatrzymanie procesu, zmianę jego priorytetu a także na uzyskanie podstawowych informacji o danym zadaniu (dostępne po kliknięciu na **Właściwości**).

34. Zainstaluj i uruchom program graficznym o podobnej funkcjonalności co systemowy Monitor systemu jest: **lxtask**





35. Uruchom narzędzie to **w**, wydanie komendy pozwoli nam przejrzeć procesy każdej powłoki wszystkich zalogowanych użytkowników.

```
root@bolek-VirtualBox:~# w
 17:00:37 up  5:57,  1 user,  load average: 0,96, 1,14, 1,63
UŻYTK.   TTY      Z                ZAL.OD   BEZCZ. JCPU   PCPU CO
bolek    tty2     /dev/tty2        11:05    3:56m 52:53   3.69s lxtask
```

Narzędzie to podaje informacje:

górny pierwszy wiersz (od lewej) - aktualny czas; czas pracy systemu; liczbę zalogowanych użytkowników oraz średnie obciążenie systemu - trzy wartości: ostatnia minuta, ostatnie pięć minut, ostatnie piętnaście minut,

kolumny to informacja o: użytkowniku; identyfikatorze terminala; komputerze, z którego nastąpiło logowanie; czasie w którym nastąpiło logowanie; czasie bezczynności; czasie zużycia procesora (JCPU oraz PCPU) oraz uruchomionym zadaniu.

36. Uruchom polecenie **free** poda nam zużycie pamięci (jednostka kB).

```
root@bolek-VirtualBox:~# free
               razem        użyte       wolne    dzielone   buf/cache    dostępne
Pamięć:      2045156      1385236       175808       39836       484112       486100
Wymiana:      973820        66304       907516
```

Opcje:

-s <sekunda> - ciągłe działanie programu, odświeżanie o ustaloną wartość,

-b - zmiana jednostki na bajty,

-m - zmiana jednostki na megabajty,

-t - podsumowanie.

Zadaniem narzędzia **fuser** jest wyświetlenie nazw procesów, które korzystają z podanych plików bądź katalogów.

37. Wywołaj narzędzie z parametrem **-v** nakazującym wyświetlenie bardziej szczegółowych informacji i podaniu lokacji uzyskamy listę procesów korzystających z tejże lokacji. Poniżej w przykładzie ukazano procesy korzystające z katalogu domowego użytkownika bolek.

```
root@bolek-VirtualBox:~# fuser -v .
                     UŻYTKOWNIK  PID DOSTĘP POLECENIE
/root:               root       3585 ..c.. bash
root@bolek-VirtualBox:~# su bolek
bolek@bolek-VirtualBox:/root$ fuser -v .
                     UŻYTKOWNIK  PID DOSTĘP POLECENIE
/root:               bolek      5921 ..c.. bash
```

Uzyskane dane dostarczają informacji o użytkowniku, PID procesu, nazwie procesu a także rodzaju uzyskanego dostępu. Rodzaj ustanowionego dostępu jest określany za pomocą symboli literowych:

c - katalog bieżący,

e - uruchomiony plik wykonywalny,

f - otwarty plik, pomijane w domyślnym trybie wyświetlania,

r - katalog root,

m - mmap-owany plik lub biblioteka dzielona.

Wywołanie narzędzia bez flagi **-v** wyświetli tylko numer procesów PID i rodzaj ustanowionego dostępu.

38. Sprawdź dostępności serwera FTP za pomocą fuser.

```
root@bolek-VirtualBox:~# fuser -v -n tcp 21
```

39. Użyj program fuser do zabicia procesów odnoszących się np. do danego katalogu.
Wydanie polecenia: **fuser -v -k -i .** spowoduje zabicie wszystkich procesów odnoszących się do katalogu domowego użytkownika, przy czym dołączenie flagi **-i** powoduje, że jesteśmy pytani o potwierdzenie wykonania wyłączenia każdego zadania z osobna. Pominięcie parametru **-i** spowoduje automatyczne zabicie wszystkich procesów.

```
root@bolek-VirtualBox:~# fuser -v -k -i .
                     UŻYTKOWNIK  PID DOSTĘP POLECENIE
/root:               root       3585 ..c.. bash
                     root       5920 ..c.. su
                     bolek      5921 ..c.. bash
                     root       5934 ..c.. su
                     root       5935 ..c.. bash
Zabić proces 3585? (y/N) n
Zabić proces 5920? (y/N) n
Zabić proces 5921? (y/N) n
Zabić proces 5934? (y/N) n
Zabić proces 5935? (y/N) n
```

Ćwiczenie na podstawie:

http://slow7.pl/item/104-dogadac-sie-z-linuxem-zarzadzanie-procesami