tail – wypisuje końcową część pliku.
Użycie:
    tail plik.txt
Użyteczne parametry:
-c *n* – wypisuje *n* ostatnich bajtów pliku
-n *n* – wypisuje ostatnie *n* linijek pliku.
Np.
    tail -n 10 system.log
wyświetli 10 ostatnich linijek pliku system.log

Gdy *n* ma dany + na początku, jest to ilość pomijanych od początku linii/bajtów.

W klasycznym Uniksie można spotkać implementacje nie przyjmujące notacji „n” – czyli np:

tail -5 /etc/passwd

head – jak tail, ale wypisuje pierwsze znaki bądź wiersze pliku, nie ostatnie.

W klasycznym Uniksie można również spotkać implementacje bez „-n”, i wóœczas działa to jak z „-n” w GNU, a klasyczna wersja może również tolerować -n.

Parametr -q dla head i tail wyłącza wyświetlanie nazw plików gdy program wypisuje zawartość wielu plików.

Narzędzi tych możemy używać w potoku, np.:
    cat pomiary.txt | cut -d , -f 2-4 | tail -n 20
Wyświetla ostatnie 20 pozycji z pliku pomiary.txt, pokazując tylko kolumny 2, 3 i 4, tekst w pliku oddzielony jest przecinkami.

W GNU można zastosować opcję **-c** która pozyskuje pewną ilość znaków, nie linijek.

less – przyjaźniejszy użytkownikowi program do czytania dłuższych plików
Użycie:
    less plik.txt
Albo w strumieniu:
    ls -al | less
W tym programie działają klawisze strzałek do przesuwania tekstu, Page Up/Page Down, Home/End.
Wyszukiwanie: klawisz / (szukanie w dół) lub ? (w górę), wprowadzenie tekstu, Enter. Do dopasowania znaków można używać wyrażeń regularnych np. takich jak w programie grep.
W związku z tym jeżeli chcemy znaleźć znak /, musimy go escape’ować znakiem \.
q – wyjście.
Użyteczne parametry:
    -N – wyświetla numery linii
    -S – gdy linia nie mieści się na ekranie, nie jest zawijana na kolejną linię terminala.

tr jak **transliterate –** zamienia jedne znaki na inne. Np.

`echo "aaabbb" | tr 'ab' 'cd'` da w wyniku **`cccddd`**

Użyteczne przełączniki:
    -d jak **delete** – przyjmuje pojedynczy zbiór znaków, które to usuwa zupełnie z danych.
    -c jak **complementary** – odwraca zbiór, zazwyczaj do usunięcia, czyli podajemy znaki, które chcemy pozostawić.
W programie tr można zazwyczaj stosowac przedziały np. `tr -dc a-zA-Z0-9` pozostawi znaki alfanumeryczne, wszystko inne kasując.

ps – jak *process*, wyświetla aktualnie działające procesy w postaci listy. Domyślnie wyświetla procesy aktualnie zalogowanego użytkownika.
Użyteczne parametry:

-l – wyświetla wielokolumnową listę procesów z dodatkowymi informacjami.
-e – wyświetla wszystkie procesy
-f – lista mniej szczegółowa od -l, bardziej od samego ps.

Opis kolumn:
F – oznaczenia (flagi) procesu
UID – identyfikator właściciela procesu.
PID – identyfikator procesu. Tego identyfikatora używamy chcąc cokolwiek zrobić z procesem.
%C, C, CP – użycie CPU
PRI – Priorytet procesu odpowiadający przydziałowi CPU w danym momencie.
NI – Współczynnik „niceness" - ogólna istotność procesu (priorytet z reguły jest wartością bieżącą).
SZ – Użycie pamięci wirtualnej (pamięć operacyjna + biblioteki współdzielone + otwarte adresy).
RSS – właściwe użycie pamięci.
WCHAN – adres, uchwyt lub nazwa czynnika powodującego oczekiwanie procesu.
TT – konsola, na której działa proces
TIME – czas rozpoczęcia procesu
COMMAND – polecenie, które uruchomiło proces.

W GNU/Linuksie i innych dystrybucjach:

-a – pokazuje procesy wszystkich użytkowników
-u – pokazuje procesy należące do danego użytkownika, pokazuje kolumny właściciela procesu (USER) [wersja Mac/BSD]
-x – pokazuje procesy działające w tle, nie używające żadnego terminala [niestandardowe rozszerzenie, działa m.in. na niektórych Mac'ach i BSD].

Stąd ps -aux w Mac/BSD pokaże procesy wszystkich użytkowników, a w GNU/Linuksie procesy użytkownika "x".

W wielu systemach operacyjnych użytkownik nie ma uprawnień do jakichś parametrów ps'a. Teraz proszę zauważyć, że jeżeli np. łączymy się z naszego konta np. po FTP i przesyłamy w parametrze do polecenia swoje hasło, to to hasło widzą wszyscy użytkownicy, którzy mieli akurat okazję uruchomić sobie ps'a z kolumną COMMAND.

w – wyświetla kto jest zalogowany, kiedy, jak długo i na ilu terminalach.
User – nazwa użytkownika,
tty – terminal, na którym jest użytkownik zalogowany,
login@ - godzina zalogowania się użytkownika,
idle – czas ostatniej aktywności na terminalu,
JCPU – użycie CPU przez wszystkie procesy dołączone na ten terminal. Również zadania działające w tle. W systemach wielodostępnych niektóre konfiguracje podają tu użycie procesora przez samą usługę dostępu do serwera,
PCPU – Użycie CPU przez bieżący proces,
What – nazwa bieżącego procesu,

who – skrócona forma w

who am i – wyświetla nazwę użytkownika wykonującego polecenie, jego terminal, datę zalogowania, źródło połączenia. Sprawdzić także **whoami** i **logname,** wyświetlają tylko nazwę użytkownika.

Przy użyciu narzędzia **who** możemy wyciągnąć trochę technicznych informacji o systemie:
who -a – wyświetla listę zalogowanych użytkowników z plików tymczasowych i logów.
who -b – wyświetla datę uruchomienia systemu
Dodatkowo:
-u – wyświetla informacje dotyczące użytkowników aktualnie zalogowanych
-H - wyświetla nad wierszami nagłówki kolumn.

Większość z tych opcji powinna być (;-)) powyłączana w systemach wielodostępnych.

finger – wyświetla informacje o użytkowniku
Bez parametrów wyświetla informacje o bieżących użytkownikach. Z parametrem w postaci nazwy użytkownika wyświetla informacje o danym użytkowniku.
Wiele programów na raz w Uniksie
Pomimo obsługi przez konsolę, Unix jest systemem wielozadaniowym. Można mieć uruchomione wiele programów naraz, czy to działających, czy wstrzymanych. Wstrzymanie procesu odbywa się przez naciśnięcie klawisza Ctrl-Z. Od tej pory otrzymujemy znów konsolę, a proces nie działa.

kill – wysyła sygnał usunięcia do procesu.
Powszechne określenie „zabija proces" jest prawidłowe w większości przypadków. Parametrem jest opcjonalnie sygnał i numer PID procesu (np. kill -9 12345). Jednakże takie kończenie procesów może powodować, że po programie zostaną jakieś tymczasowe pliki, których ten nie miał okazji zapisać.

Niektóre procesy mogą inaczej reagować na nadchodzące sygnały. Przykładowo, program do kopiowania danych (a nieumiejętnie używany zniszczenia sobie danych na dysku) dd na sygnał USR1 wysyła na konsolę postęp kopiowania.

jobs – wyświetla listę wstrzymanych i działających w tle zadań. W nawiasach kwadratowych wyświetlane są numery zadania dla danego użytkownika. Te numery można podawać jako argumenty dla poleceń kill, bg czy fg poprzedzając je znakiem %. np. dla zadania [1] będzie to:
kill %1.

bg – jak *background*
Uruchamia ostatnio wstrzymany proces w tle. W przypadku gdy program realizuje stale jakąś funkcję będzie to robił, lecz w tle.
fg – jak *foreground*
Przywołuje ostatnio wstrzymany proces na konsolę. Zawieszenie procesu można zrealizować naciskając Ctrl-Z. Wracamy do konsoli i możemy kontynuować działania, a wstrzymany proces przywołujemy poleceniem fg.

Diagnostyka i rozpoznanie systemu:

top – program wyświetlający statystyki użycia systemu. Wyjście: Ctrl-C.

dmesg – wyprowadza ostatnie komunikaty z dziennika systemowego.

uname – informacje o tym jakiej wersji systemu używamy: Parametry:
-a – wyświetla pełną informację o systemie: Nazwę, wersję, architekturę.
-r – wyświetla wersję (wydanie) systemu operacyjnego.
-v – wersja systemu operacyjnego
-m (w starszych wersjach -i) – Typ architektury (procesor). Np. x86_64 – 64-bit PC, i686 – 32bit PC.

env – wyświetla listę i wartości zmiennych systemowych

Zadania:

Przy pomocy narzędzia less wyświetl listę plików w swoim katalogu i podkatalogach (drukowane poleceniem ls -lR). Odeślij listę do tła. Wyświetl bieżącą listę swoich procesów.

Bez przywoływania procesu less usuń go z pamięci.

Przy użyciu narzędzia **env, grep i cut** poleceniem w jednej linii wyświetl pierwszy element swojej ścieżki systemowej (zmienna `PATH=/pierwszy/element:/drugi/element:/trzeci/element`)