USO Cheat Sheet
Comenzi utile Linux

Primul ajutor

`man comandă` – afișează pagina de manual a comenzii
`whereis app` – afișează posibilele locuri în care se poate afla `app`
`which cmd` – afișează calea executabilului lui `cmd`
`apropos pattern` – afișează comenzi ce au în descriere *pattern*

Sistemul de fișiere

`/`	root directory
`/bin`	binary files
`/home`	users' homedirs
`/usr`	secondary filesystem
`/var`	variable data (cache, log etc.)
`/etc`	config files
`/boot`	bootloader & boot info
`/lib`	system library
`/dev`	hardware devices
`/root`	root's home

`cd [DIR]` – schimbă din directorul curent în DIR (dat ca argument) sau în home, dacă nu e dat nici un argument
`pwd` – afișează directorul curent

`ls -lah [FILE]` – listare lungă a tuturor fișierelor din directrul curent, dacă nu e dat nici un argument
`-l` long listing format
`-a` nu ignoră intrări care încep cu .
`-h` human readable (pentru dimensiuni, de exemplu)
`rm -rf dir` – șterge tot conținutul directorului `dir`
`-r` ștergere recursivă
`-f` forțează ștergerea

`cp file1 file2` – copiază `file1` în `file2`
`cp -r dir1 dir2` – copiază `dir1` în `dir2` și creează `dir2` dacă acesta nu există
`mv file1 file2` – mută `file1` în `file2` dacă `file2` e director sau redenumește `file1` în `file2`
`touch file` – creează sau actualizează `file`
`dd if=FIȘIER_INTRARE of=FIȘIER_IEȘIRE bs=DIMENSIUNE_BLOC count=NUMĂR_BLOCURI` – copiere și conversie la nivel de octeți

`ln -s file link` – creează link-ul simbolic `link` către fișierul `file`

`cat [FILE1] [FILE2] ...` – concatenează conținutul fișierelor date ca argument și afișează la ieșirea standard
`tail -f file` – afișează, în timp real, conținutul fișierului `file` începând cu primele 10 linii. Dacă este omis parametrul `-f` atunci vor fi afișate la ieșirea standard ultimele 10 linii.
`tail -n NR file` – afișează ultimele `NR` linii din `file`
`head -n NR file` – afișează primele `NR` linii din `file`

Căutare

`grep -n pattern file` – caută *pattern* în `file`
`-n` afișează linia la care se găsește *pattern*
`grep -R pattern dir` – caută după *pattern* în directorul `dir`
`-R` Căutare recursivă
`command | grep pattern` – caută în output-ul comenzii după *pattern*
`find dir -name pattern` – caută după fișiere ce conțin în numele lor *pattern* în directorul `dir`
`locate file` – afișează toate instanțele în sistem a fișierului `file`

Arhivare, comprimare

`tar -xzvf file.tar.gz` – extrage arhivă gzip
`-x` extract files from file.tar.gz
`-z` use gzip,gunzip
`-v` verbose mode - afișează fișierele dezarhivate
`-f` use archive file or device file.tar.gz
`tar -czvf file.tar.gz files` – creează o arhivă folosind gzip. Directoarele vor fi arhivate recursiv (toate fișierele din directoarele regăsite ca argument al comenzii se vor afla în arhivă)
`-c` extract files from file.tar.gz
Restul parametrilor au aceeași semnificație ca la dezarhivare
`zip file.zip files` – creează o arhivă zip cu fișierele date ca argument. Dacă se află directoare prin argumente, conținutul lor **NU** va fi inclus recursiv.
`zip -r tema1.zip tema1` – creează o arhivă zip cu director-ul tema1 în rădăcină și include recursiv toate fișierele din director.

Gestiunea utilizatorilor

`sudo` – rulează o comandă ca root
`whoami` – afișează utilizatorul curent
`who` – afișează utilizatorii logați
`w` – afișează utilizatorii logați și activitățile lor
`finger student` – afișează informații despre utilizatorul `student`
`passwd` – modifică parola user-ului curent (dacă nu e dat nici un argument) sau a user-ului dat ca argument

`chown user file -R` – schimbă utilizatorul proprietar (owner) al lui `file`.
`-R` dacă `file` e un director se poate folosi acest argument pentru a schimba recursiv owner-ul tuturor fișierelor din director
`chgrp group file -R` – schimbă grupul lui `file`. Analog `chown`
`chmod octal file` - schimbă permisiunile lui `file` în format octal astfel: Formatul octal are 3 cifre (permisiunile pentru user, group, others), ce pot fi maxim 7, și se combină prin suma următoarelor cifre:
`0` nici un drept
`1` execuție (x)
`2` scriere (w)
`4` citire (r)
Exemplu: `chmod 755 file` – `rwx` pentru owner, `rx` pentru group și others. `man chmod` pentru detalii complete

Procese, semnale

`ps` – afișează procesele shell-ului curent
`ps -ef` – afișează toate procesele și detalii (full-format listing) despre acestea
`top, htop` – Linux task manager

`kill -l` – afișează toate semnalele
`kill pid` – trimite semnalul `SIGTERM (15)` procesului cu id-ul `pid` (închide procesul)
`kill -9 pid` – trimite semnalul `SIGKILL (9)` procesului cu id-ul `pid` (forțează distrugerea procesului)
`killall proc` – *omoară* toate procesele numite *proc*

`bg` – trece un proces din stopped în running în background
`fg` – trece un proces în foreground
`&` – lansează un proces în background running

Informații hardware

`cat /proc/cpuinfo` – informații despre procesor/CPU al sistemului
`cat /proc/meminfo` – informații despre memoria sistemului
`free` – informații despre memoria totală, utilizată la momentul curent, cache, swap etc.
`lspci` – afișează componetele periferice (PCI)
`lsusb` – afișează device-urile USB
`uname -a` – afișează informații despre kernel
`df` – afișează *disk usage* al sistemului de fișiere
`du -hs dir` – afișează dimensiunea pe disk (totală) a directorului/fișierului `dir`
`dmesg` – afișează mesaje de la kernel (exemplu: module inserate/șterse, device-uri USB inserate etc.)

Configurare rețea

`ifconfig` – afișează informații despre toate interfețele de rețea din sistem.
`ip address show` – afișează toate interațele de rețea și adresele lor ip
`ip route show` – afișează tabelele de rutare ale interfețelor
`arp -a, ip neighbour show` – vizualizarea tabelei ARP

`ifconfig eth0 192.168.60.13 netmask 255.255.255.0` – configurează **temporar** intefața de rețea `eth0` cu adresa IP `192.168.60.13` și masca de rețea 24.
`dhclient eth0` – configurează **temporar** dinamic (DHCP) interața `eth0`
`/etc/network/interfaces` – fișierul pentru configurări permanente ale interațelor de rețea
`ifup, ifdown` – pornește, respectiv oprește, o interață
`ping host` – testează conectivitatea trimițând mesaje de tip ICMP lui `host`

Servicii rețea

`ssh user@host` – conectare remote la `host` cu contul `user`
`ssh -p PORT_NO user@host` – conectare remote pe portul `PORT_NO`
`ssh-keygen` – generare cheii de autentificare
`ssh-copy-id` – instalarea cheii publice pe mașina remote

`wget file` – descarcă `file`
`wget -c file` – continuă o descărcare oprită
`host hostname` – determină adresa IP a numelui `hostname` (DNS lookup)

`netstat -tlnp` - informații despre subsistemul de rețea. Fără nici un parametru va afișa lista de conexiuni deschise.
`-t` afișează doar conexiuni ce folosesc protocolul TCP. Pentru UDP folosiți `-u`
`-l` afișează doar porturile pe care o stație ascultă
`-n` afișare numerică în loc de a încerca să determine nume
`-p` afișarea programului (numele executabilului) ce ascultă pe port. E nevoie de drept de root pentru aceasta

USO Cheat Sheet
Shell Scripting

Citire. Afișare. Înlănțuire comenzi

`read a` – Citește variabila `a` de la intrarea standard

`echo -ne "Hello, Bash \n!"` – afișare text.
`-n` nu va pune un trailing end of line, care este pus implicit
`-e` permite interpretarea backslash escapes (ca în C la printf)
`;` – secvențierea comenzilor. **Exemplu**: `echo "StarCraft II"; echo "Wings of Liberty"`
`\` – Un backslash la finalul liniei semnifică faptul că linia se continuă pe rândul următor.
`&&, ||` – execută un al doilea proces doar dacă primul s-a încheiat cu success, respectiv eroare. **Exemple:** `true && echo "Success"` `false || echo "Fail"`

Caractere speciale Bash

- operatori
- redirectare: >, <, &>, >>, <<
- secvențiere, înlănțuire: ;, ||, &&, |, &
- expandare: $
- comentare: #
- citare (escaping): ', ", \
- separare: blank (spațiu)
- globbing: ?, *, [,], {, }

Rularea unui script Bash

`source script.sh, . script.sh` – execută comenzile din script ca și cum ar fi fost introduse de la tastatură
`Bash script.sh` – rulează `script.sh` în alt shell Bash creat
`./script` – rulează script folosind interpretorul dat în prima linie prin shebang (`#!`). Exemplu de linie shebang: `#!/usr/bin/env python`. **Atenție!** Trebuie să avem drepturi de execuție pe script!

Variabile Bash. Variabile speciale

`NUME=VALOARE` – definire variabilă în Bash. **NU** lăsați spații!
`export NUME=VALOARE` – configurare variabilă ca variabilă de mediu (exportare)

`$?` – valoarea de retur a ultimei comenzi
`$!` – PID-ul ultimului proces (job) lansat în background
`$_` – ultimul argument al ultimei comenzi
`$#` – Numărul de parametri transmiși scriptului (echivalent `argc` în C)
`$0` – Numele scriptului (echivalent `argv[0]` în C) `$1, $2 ...` – Primul, al doilea argument etc. (echivalent `argv[1], argv[2]` în C)

`IFS` – Internal Field Separator. Variabila determină modul în care Bash recunoaște câmpuri sau limitele cuvintelor când interpretează șiruri de caractere. **Exemplu:** `var1="a+b+c"; IFS=+; echo $var1`

Filtre text

`cut -d DELIMITATOR -f LISTĂ_CÂMPURI file` – selectare coloane de text din fiecare linie a fișierului `file` pe baza `DELIMITATOR` (implicit e TAB) și alege să afișeze doar câmpurile din `LISTĂ_CÂMPURI`. **Exemplu:** `cut -f 1,4 -d ':' < /etc/passwd`

`wc -l file` – determină câte linii are `file`
`wc -w file` – determină numărul de cuvinte din `file`
`wc -c file` – determină numărul de octeți ai lui `file`

`sort -n file` – sortare numerică
`sort -r file` – reverse sort
`sort -u file, sort file | uniq` – cu unicizare
`sort -k 3 file` – sortează în funcție de coloana 3

tr, sed, awk

`tr -s '\n' < file` – șterge liniile goale din `file` și afișează
`tr -s 'A-Za-z0-9 ' < file` – șterge caracterele alfanumerice și spații duplicat din `file` și afișează
`tr -d -c 'A-Za-z0-9' < /dev/urandom | head -c 10` – generator de parole de 10 caractere
`sed 's/old/new/g' file` – înlocuiește toate aparițiile `old` cu `new` în fișierul `file` și afișează la ieșirea standard
`sed '1-10s/old/new/g' file` – la fel ca mai sus, doar că pentru primele 10 linii
`sed 's/[ \t]*$//g' file` – șterge *trailing whitespace* de la sfârșitul fiecărei linii din `file` și afișează la ieșirea standard
`sed 's/\t/ /g' file` – înlocuiește TAB cu 4 spații în fiecare linie din `file`
`awk ' { t = $1; $1 = $2; $2 = t; print; } ' file` – interschimbă primele două coloane din `file` și afișează la ieșirea standard

Instrucțiune decizională

```
if condiție1
then
    instrucțiuni1
elif condiție2
then
    instrucțiuni2
else
    alte_instrucțiuni
fi
```

`test expresion` – comandă de verificare a valorii de retur a `expression` Pentru a compara numere folosim:
`-eq` equal
`-ne` not equal
`-gt` greater than
`-ge` greater or equal
`-lt` less than
`-le` less or equal

Pentru a compara șiruri folosim:
`-n str` lungimea lui str este diferită de 0
`-z str` lungimea lui str este 0
`s1 = s2` șirurile s1 și s2 sunt egale

Se poate folosi și construcția `[ ... ]` (**atenție la spații, trebuie să existe! Exemple:**

```
test $a -lt 3
[ $a -lt 3 ]


if [ $a -lt 3 ]; then
    echo "Adevărat"
fi
```

Bucle

```
while condition
do
   command1
   command2
   command3
done

for i in 1 2 3 4 5 6 7 8 9 10; do ... done
for ((i = 1; i <= 10; i++)); do ... done
for i in $(seq 1 10); do ... done

for i in $(seq -f "%02g" 1 10);
do
    ...
done

for f in *; do ... done

for user in $(cut -d ':' -f 1 < /etc/passwd);
do
    ...
done

for arg in $@; do ... done
```

Definire funcții

```
function func_name()
{
    ...
}
```

Dacă trebuie să retruneze o valoare, se poate pune și un `return` ca în C. Dacă nu se folosește `return` funcția va întoarce valoarea de retur a ultimei comenzi din corp.