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
```

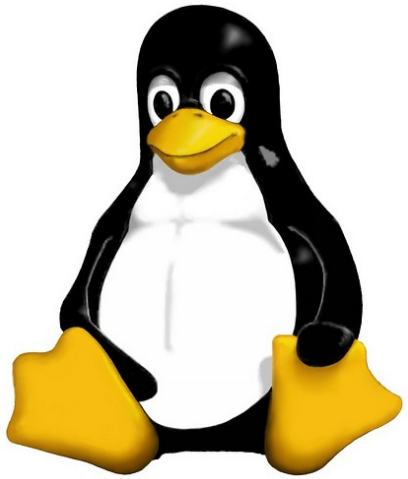
Definire funcții

```
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.

freeworld.posterous.com

Linux Bash Shell Cheat Sheet



(works with about every distribution, except for apt-get which is Ubuntu/Debian exclusive)

Legend:

Everything in “<>” is to be replaced, ex: <fileName> --> iLovePeanuts.txt
Don't include the '=' in your commands
'..' means that more than one file can be affected with only one command ex: rm file.txt file2.txt movie.mov .. ..

Linux Bash Shell Cheat Sheet

Basic Commands

Basic Terminal Shortcuts

CTRL L = Clear the terminal
CTRL D = Logout
SHIFT Page Up/Down = Go up/down the terminal
CTRL A = Cursor to start of line
CTRL E = Cursor the end of line
CTRL U = Delete left of the cursor
CTRL K = Delete right of the cursor
CTRL W = Delete word on the left
CTRL Y = Paste (after CTRL U,K or W)
TAB = auto completion of file or command
CTRL R = reverse search history
!! = repeat last command
CTRL Z = stops the current command (resume with fg in foreground or bg in background)

Basic Terminal Navigation

ls -a = list all files and folders
ls <folderName> = list files in folder
ls -lh = Detailed list, Human readable
ls -l *.jpg = list jpeg files only
ls -lh <fileName> = Result for file only

cd <folderName> = change directory
if folder name has spaces use “ “
cd / = go to root
cd .. = go up one folder, tip: ../../../

du -h: Disk usage of folders, human readable
du -ah: “ “ “ files & folders, Human readable
du -sh: only show disc usage of folders

pwd = print working directory

man <command> = shows manual (RTFM)

Basic file manipulation

cat <fileName> = show content of file
(less, more)
head = from the top
-n <#oflines> <fileName>

tail = from the bottom
-n <#oflines> <fileName>

mkdir = create new folder
mkdir myStuff ..
mkdir myStuff/pictures/ ..

cp image.jpg newimage.jpg = copy and rename a file
cp image.jpg <folderName>/ = copy to folder
cp image.jpg folder/sameImageNewName.jpg
cp -R stuff otherStuff = copy and rename a folder
cp *.txt stuff/ = copy all of *<file type> to folder

mv file.txt Documents/ = move file to a folder
mv <folderName> <folderName2> = move folder in folder
mv filename.txt filename2.txt = rename file
mv <fileName> stuff/newfileName
mv <folderName>/ .. = move folder up in hierarchy

rm <fileName> .. = delete file (s)
rm -i <fileName> .. = ask for confirmation each file
rm -f <fileName> = force deletion of a file
rm -r <foldername>/ = delete folder

touch <fileName> = create or update a file

ln file1 file2 = physical link
ln -s file1 file2 = symbolic link

Linux Bash Shell Cheat Sheet

Basic Commands

Researching Files

The slow method (sometimes very slow):

```
locate <text> = search the content of all the files
locate <fileName> = search for a file
sudo updatedb = update database of files

find = the best file search tool (fast)
find -name “<fileName>”
find -name “text” = search for files who start with the word text
find -name “*text” = “      “    “    “   end   “    “   “    “
```

Advanced Search:

```
Search from file Size (in ~)
      find ~ -size +10M = search files bigger than.. (M,K,G)

Search from last access
      find -name “<filetype>” -atime -5
            ('-' = less than, '+' = more than and nothing = exactly)

Search only files or directory’s
      find -type d --> ex: find /var/log -name "syslog" -type d
      find -type f = files

More info: man find, man locate
```

Extract, sort and filter data

```
grep <someText> <fileName> = search for text in file
      -i = Doesn't consider uppercase words
      -I = exclude binary files
grep -r <text> <folderName>/ = search for file names
      with occurrence of the text
```

With regular expressions:

```
grep -E ^<text> <fileName> = search start of lines
with the word text
grep -E <0-4> <fileName> =shows lines containing numbers 0-4
grep -E <a-zA-Z> <fileName> = retrieve all lines
with alphabetical letters

sort = sort the content of files
sort <fileName> = sort alphabetically
sort -o <file> <outputFile> = write result to a file
sort -r <fileName> = sort in reverse
sort -R <fileName> = sort randomly
sort -n <fileName> = sort numbers

wc = word count
wc <fileName> = nbr of line, nbr of words, byte size
      -l (lines), -w (words), -c (byte size), -m
      (number of characters)

cut = cut a part of a file
-c --> ex: cut -c 2-5 names.txt
      (cut the characters 2 to 5 of each line)
-d (delimiter)         (-d & -f good for .csv files)
-f (# of field to cut)

more info: man cut, man sort, man grep
```

Linux Bash Shell Cheat Sheet

Basic Commands

Time settings

date = view & modify time (on your computer)

View:
date “+%H” --> If it's 9 am, then it will show 09
date “+%H:%M:%Ss” = (hours, minutes, seconds)
%Y = years
Modify:
MMDDhhmmYYYY
Month | Day | Hours | Minutes | Year

sudo date 031423421997 = March 14th 1997, 23:42

Execute programs at another time

use 'at' to execute programs in the future

Step 1, write in the terminal: at <timeOfExecution> ENTER
ex --> **at 16:45** or **at 13:43 7/23/11** (to be more precise)
or after a certain delay:
at now +5 minutes (hours, days, weeks, months, years)
Step 2: <ENTER COMMAND> ENTER
repeat step 2 as many times you need
Step 3: CTRL D to close input

atq = show a list of jobs waiting to be executed

atrm = delete a job n°<x>
ex (delete job #42) --> atrm 42

sleep = pause between commands
with ';' you can chain commands, ex: touch file; rm file
you can make a pause between commands (**m**inutes, **h**ours, **d**ays)
ex --> touch file; sleep 10; rm file <-- 10 seconds

(continued)

crontab = execute a command regularly
-e = modify the crontab
-l = view current crontab
-r = delete you crontab
In crontab the syntax is
<Minutes> <Hours> <Day of month> <Day of week (0-6, 0 = Sunday)> <COMMAND>

ex, create the file movies.txt every day at 15:47:
47 15 * * * touch /home/bob/movies.txt
* * * * * --> every minute
at 5:30 in the morning, from the 1st to 15th each month:
30 5 1-15 * *
at midnight on Mondays, Wednesdays and Thursdays:
0 0 * * 1,3,4
every two hours:
0 */2 * * *
every 10 minutes Monday to Friday:
*/10 * * * 1-5

Execute programs in the background

Add a '&' at the end of a command
ex --> cp bigMovieFile.mp4 &

nohup: ignores the HUP signal when closing the console (process will still run if the terminal is closed)
ex --> nohup cp bigMovieFile.mp4

jobs = know what is running in the background

fg = put a background process to foreground
ex: fg (process 1), f%2 (process 2) f%3, ...

Linux Bash Shell Cheat Sheet

Basic Commands

Process Management

w = who is logged on and what they are doing

tload = graphic representation of system load average
(quit with CTRL C)

ps = Static process list
-ef --> ex: ps -ef | less
-ejH --> show process hierarchy
-u --> process's from current user

top = Dynamic process list
While in top:

- q to close top
- h to show the help
- k to kill a process

CTRL C to top a current terminal process

kill = kill a process
You need the PID # of the process
ps -u <AccountName> | grep <Application>
Then
kill <PID> .. .. ..
kill -9 <PID> = violent kill

killall = kill multiple process's
ex --> killall locate

extras:
sudo halt <-- to close computer
sudo reboot <-- to reboot

Create and modify user accounts

sudo adduser bob = root creates new user
sudo passwd <AccountName> = change a user's password
sudo deluser <AccountName> = delete an account

addgroup friends = create a new user group
delgroup friends = delete a user group

usermod -g friends <Account> = add user to a group
usermod -g bob boby = change account name
usermod -aG friends bob = add groups to a user without loosing the ones he's already in

File Permissions

chown = change the owner of a file
ex --> chown bob hello.txt
chown user:bob report.txt = changes the user owning report.txt to 'user' and the group owning it to 'bob'
-R = recursively affect all the sub folders
ex --> chown -R bob:bob /home/Daniel

chmod = modify user access/permission – simple way
u = user
g = group
o = other

d = directory (if element is a directory)
l = link (if element is a file link)
r = read (read permissions)
w = write (write permissions)
x = eXecute (only useful for scripts and programs)

Linux Bash Shell Cheat Sheet

Basic Commands

File Permissions (continued)

```
'+' means add a right
'-' means delete a right
'=' means affect a right

ex --> chmod g+w someFile.txt
       (add to current group the right to modify someFile.txt)

more info: man chmod
```

Flow redirection

```
Redirect results of commands:

'>' at the end of a command to redirect the result to a file
       ex --> ps -ejH > process.txt
'>>' to redirect the result to the end of a file
```

Redirect errors:

```
'2>' at the end of the command to redirect the result to a file
       ex --> cut -d , -f 1 file.csv > file 2> errors.log
'2>&1' to redirect the errors the same way as the standard output

Read progressively from the keyboard

<Command> << <wordToTerminateInput>
       ex --> sort << END <-- This can be anything you want
              > Hello
              > Alex
              > Cinema
              > Game
              > Code
              > Ubuntu
              > END
```

Flow Redirection (continued)

```
terminal output:
Alex
Cinema
Code
Game
Ubuntu

       Another example --> wc -m << END
```

Chain commands

```
'|' at the end of a command to enter another one
       ex --> du | sort -nr | less
```

Archive and compress data

```
Archive and compress data the long way:

Step 1, put all the files you want to compress in
the same folder: ex --> mv *.txt folder/

Step 2, Create the tar file:
tar -cvf my_archive.tar folder/
       -c : creates a .tar archive
       -v : tells you what is happening (verbose)
       -f : assembles the archive into one file

       Step 3.1, create gzip file (most current):
gzip my_archive.tar
       to decompress: gunzip my_archive.tar.gz

Step 3.2, or create a bzip2 file (more powerful but slow):
bzip2 my_archive.tar
       to decompress: bunzip2 my_archive.tar.bz2
```

Linux Bash Shell Cheat Sheet

Basic Commands

Archive and compress data (continued)

step 4, to decompress the .tar file:
tar -xvf archive.tar archive.tar

Archive and compress data the fast way:

gzip: tar -zcvf my_archive.tar.gz folder/
decompress: tar -zcvf my_archive.tar.gz Documents/

bzip2: tar -jcvf my_archive.tar.gz folder/
decompress: tar -jxvf archive.tar.bz2 Documents/

Show the content of .tar, .gz or .bz2 without decompressing it:

gzip:
gzip -ztf archive.tar.gz
bzip2:
bzip2 -jtf archive.tar.bz2
tar:
tar -tf archive.tar

tar extra:
tar -rvf archive.tar file.txt = add a file to the .tar

You can also directly compress a single file and view the file without decompressing:

Step 1, use gzip or bzip2 to compress the file:
gzip numbers.txt

Step 2, view the file without decompressing it:
zcat = view the entire file in the console (same as cat)
zmore = view one screen at a time the content of the file (same as more)
zless = view one line of the file at a time (same as less)

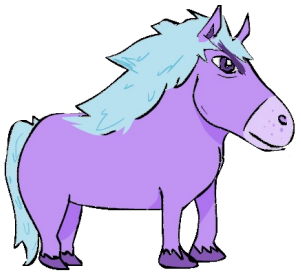
Installing software

When software is available in the repositories:
sudo apt-get install <nameOfSoftware>
ex--> sudo apt-get install aptitude

If you download it from the Internets in .gz format (or bz2) - “Compiling from source”
Step 1, create a folder to place the file:
mkdir /home/username/src <-- then cd to it

Step 2, with 'ls' verify that the file is there (if not, mv ../file.tar.gz /home/username/src/)

Step 3, decompress the file (if .zip: unzip <file>) <--
Step 4, use 'ls', you should see a new directory
Step 5, cd to the new directory
Step 6.1, use ls to verify you have an INSTALL file, then: more INSTALL
If you don't have an INSTALL file:
Step 6.2, execute **./configure** <-- creates a makefile
Step 6.2.1, run make <-- builds application binaries
Step 6.2.2 : switch to root --> su
Step 6.2.3 : make install <-- installs the software
Step 7, read the readme file



ln -s /cale denumire = legatura simbolica

cp file . = copiaza fisierul in calea curenta

cp -r folder . = copiaza folder in cale curenta

cmp file1 file2 = compara fisiere

diff -r folder1 folder = compara folderele

mv file1 file = muta fisierul

mv folder1 folder2 = muta folder recursiv

rm -rf = stergere fortat recursiva a unui director

rm Folder/* = stergere toate fisierele din director

find /cale -name nume = afisarea tuturor fisierelor cu numele nume din /cale

find /cale -size +500k = mai mari ca 500KB

whereis ls = localizeaza calea catre executabil si manual

which ls = doar calea catre executabil

zip = arhivare si compresie

tar = arhivare

gzip = compresie

tar c.. = create archive

tar v.. = verbose, shows what is happening

tar f filename = pentru numele arhivei "filename"

tar cvf tarfile.tar director_intrare/ = creeaza tar fisier

tar tf tarfile.tar = listeaza continut tarfile.tar

tar xf /cale_catre_tarfile.tar = dezarhiveaza tar-ul in directorul in care te afli acum

tar z = gzip

tar j = bzip2

tar czf tarfile.tar director_intrare/ = gzip

tar cjf tarfile.tar director_intrare/ = bzip2

stdin = 0

stdout = 1

stderr = 2

./program < fisier_intrare

./program > fisier_iesire

./program 2 > fisier_erori

./program 2 > &1 = erorile la stdouts

./ program > fisier_iesire_si_erori 2 > &1

ls > file.txt = va redirecta in file.txt

ls >> file.txt = appends

cp listare 2 > ERRORS.TXT = va redirecta erorile in ERRORS.TXT

ls > fisier.out 2 > &1 = si iesirea de eroare si iesirea standard

sudo apt install inkscape = instalarea pachetului

apt show inkscape = instalat sau nu?

sudo apt remove inkscape = sterge pachetul

sudo apt update = actualizeaza info locale despre pachete

sudo apt upgrade = actualizeaza pachetele de pe sistem

sudo apt -d remove inkscape = descarca pachetul, nu il instaleaza

apt source inkscape = descarca sursele pachetului

dpkg -l | grep '^ii' = afiseaza pachetele instalate local

dpgk -l *book* = cauta pachetele instalate local dupa expresia book

dpkg -S /usr/bin/bash = identifica pachetul care contine fisierul /usr/bin/bash

dpkg -L inkscape = afiseaza continutul pachetului inkscape, instalat local

less /etc/passwd | cut -d ':' -f1 =afiseaza toti utilizatorii din sistem

sudo usermod -d /newhome/username username = change user home

sudo passwd user = schimbi parola utilizatorului

cat /etc/passwd | grep "$(cat /bin/shells)" | cut -d ":" -f 1,7 = toti utilizatorii care au shell

tr -s " " = squishes spatiile

cat fisier | tr “[a-z]” “[A-Z]” = inlocuieste a-z cu A-Z

find /etc -maxdepth 1 -type f -ls | tr -s " " | cut -d " " -f12 = list all regular files din /etc