|  |  |
| --- | --- |
| Ctrl+F6 switch bashes , Ctrl+F1 terug naar vorige  **BASIC NAVIGATION**  rm -r [dir] #remove directory dir recursively  mkdir -p [/a/b] #folder creatie met parents  ln -s [file] [link] #create symbolic link to file  head/tail [file] #output the first/last 10 lines of file (opt -f): live  alias [name 'command'] #create an alias for a command  df #show disk usage  locate [file] #Find file  whereis[command] #Find binary / source / manual  file [path] #Find out what type of item a file or directory is.  sort [file] #Sort lines in a given way. -n nummeric sort / -r revrese sort  wc [fil] #How many words / -l lines / -w words / -c characters  uniq [file] #distinct / -c count / -i ignore case  diff [filename1] [filename2] #compares files, and shows where they differ  quota -v #shows what your disk quota is  set -o noclobber # blocks > file / turn off +o / bypass >|  who #all logged in users  file [fle/dir] #typefile bekijken  wget [location] #download file  diff [file1 file2] #verschill tusse twee files  nl [file] #nulmmer elke lijn  Tail –3 [file] #laatste 3regels tonen | **FIND**  find . -name "\*.conf" #find all files in cur dir ending in conf  find . -type f/d -name "\*.conf" #find all files of type FILE/DIR in cur dir ending in conf |
| **COMPRESSION**  tar cf file.tar [files] #create a tar named file.tar  tar xf file.tar #extract the files from file.tar  gzip [file] #compresses file and renames it to file.gz  gzip -d [file.gz] #decompresses file.gz back to file |
| **PROCESSEN**  ps #display your currently active processes  top #display all running processes  kill [pid] #kill process id pid |
| **NETWERK**  ping [host] #ping host and output results  whois [domain] #get information for domain  dig [domain] #get DNS information for domain  w #whois online  finger user #display info about user |

|  |  |
| --- | --- |
| **GREP COMMAND**  grep 'string' [filename1] [filename2] #Find a string in 1 or more files  grep -i 'string' [filename] #Case insens­itive search  grep –E 'regex' [filename] #Use regular expres­sions (regex)  grep -w 'word' [filename] #Look for words  grep -A/B/C n 'string' [filename] #Display n lines after/before/around matching string  grep -v 'warning' [dir] #Return all lines which don't match the pattern  grep -e 'string1' -e 'string2' [filename] #Use regex  grep -e '^al' [filename] #Return lines starting with 'al'  grep -E 'w+' [filename] #Get lines containing 1+ w / -E = extended regex  grep -l 'string' /var/log/\* #Display filename  grep '[FG]oo' \* # find Foo or Goo in all files in the current dir  grep –o ‘string’ #focuseren enkel op de gevonden | **CUT COMMAND**  cut -d: -f1 [file] #Print the first field out of a CSV file (-d = delimeter “:” )  cut -c1-25 [dir/file] #Print characterscolumn 1 - 25 in  cut -c1 [file] #print first column / c1,3 1st and 3nd charactercolumn print /  cut -f1 [file] #print column 1 / working with tabs only  cut -d” “, -f1 [file] #print 1column in CSV / -d" " = seperate by space |
| **JOIN COMMAND IDENTICAL**  join [file1 file2]  **DIFFERENT FIELDS**  join -1 3 -2 1 [file1 file2]  -1 = firstfile / 3 = column / -2 = second file / 1 = column |

|  |  |
| --- | --- |
| **SED COMMAND** stream editor  cat [file] | sed 's/oldString/newString/' #replace 1st word in line  -n option is numbering of line = cat -n [file] | sed 's/o...  sed 's/oldString/newString/g' #replace all matches in line  sed 's/^/newString/g' #replace all starting lines(^) with newvalue / $-endofline  sed G #line space after line / 'G;G' 2line spaces  sed -n 3,5p #printin 3d en 5th line  sed '3d' #delete 3rd line / delete last line '$d'  sed -e '1d' -e 'G' #apply multiple actions with -e  sed '/25/ s/oldString/newString/g' #replace all matches where 25 like keyword is found  sed '1,3p' #dublicate 1 en 3rd line  \U uppercase /ig' ignore case | **SORT COMMAND**  sort [file] #alphbet sort / -n nummerically sort  sort -r [file] #reverse sort  sort -k 2 [file] #sort on number of column  sort -t "|" -k 2 [file] #seperate by delimiter en sort by 2nd column  sort -c [file] #check sorted or not |
| **TR trancation**  cat [file] | tr 'e' 'E' #change all letters from e to E  cat [file] | tr '\n' ' ' #change all new lines to spaces  tr -s 'd' #squeez to one cchar dddd -> d |
| **AWK**  awk -F”:” ‘ { print $1 } ‘ [file/dir] #-F”:” - delimeter / $1 column to print  awk -F”:” ‘ NR==1,NR==10{ print $1 ,$3} ‘ [file/dir] #print rows 1-10 ,1col en 3 col  awk -F”:” ‘ { print length($1) } ‘ [file/dir] #print rows 1-10 en lengte van 1ste col  Ook: ‘ { printf “%s %d\n”, $1,$3 }’  Awk –F: ‘$3 >= 1000 {print $1}' passwd.txt | sort > users.txt  awk -F: '$7 == "/bin/bash" {printf "%s:%s:%s\n", $1, $3, $6}' passwd.txt | sort #users printen met bash | **Paste**  Paste[file1 file2] #almost like join but..  Paste –d: [file1 flie2] #join by delimiter |

|  |  |
| --- | --- |
| **GEBRUIKERS GROEPS**  Su [user] #inliggen onder user  id -u/g [name] #id opvragen van user of group  less /etc/passwd #alle gebruikers / less /etc/group #groepen  sudo -i inloggen als sudo  useradd [name] / usermod -aG [grname] [name]# gebruiker toeveogen aan groep  groupadd [grname] #grouptoeveogen  usermod -G [grname] [name] #reset to group enkel de grname  usermod -g [grname] [name] #primaire groep instellen  userdel [name] #delete user / -r #delete homedir  useradd -g sporten -G judo bob | **PERMISSIES**  rwx 421 uid4 gid2 stbi1 - octal - chmod 775 [file]  chmod u=rwx,g=rw,o=r [file]  chown [name] [grname] [/dirvanfile] #veranderen eigenaar van dir of file  umask #permissies aanpassen  umask 022  **f**ind / -type f -perm -04000 > suid #alle bestanden met suid aan  chmod 3775 [dir] #stickyb+gid voor dir  chgrp [group] [file] #group van de file veranderen  Chmod 6771 [dir/file] #niew file van groepseogenaar  chmod g+s [dir] #groeps eigenaar automatisch eig van new best in dir  chmod +t [dir] enkel eig mag verwijderen  chmod +t [dir] enkel eig mag verwijderen 1755 or -rwxr-xr-t // 2764 or -rwxrwSr-- // 4754 or drwsr-xr-- |

|  |  |
| --- | --- |
| **Scripts**  ${BASH\_VERSION}  ${SHLVL} #leverl van shell  Bash #bash binnen bash openen  Export [var] #maakt declare –p [var] -> declare –x ipv -- | **E**nv #omgevingsvariabelen  Set #alle functies en variabelen  Chmod +x [file] #maak exectutable // ./[file] --- execute  echo $(cat /etc/passwd | cut -d: -f1 | sort) #uitvoer van command in scr  Shift #schuif een paramater naar rechts  Set $(date) #set de gegeven variabele als optie |

**CONDITIONS**

|  |  |  |
| --- | --- | --- |
| n1 -eq n2 #Check to see if n1 equals n2.  n1 -ne n2 #Check to see if n1 is not equal to n2.  n1 -lt n2 #Check to see if n1 < n2.  n1 -le n2 #Check to see if n1 <= n2.  n1 -gt n2 #Check to see if n1 > n2.  n1 -ge n2 #Check to see if n1 >= n2. | $0 #Name of this shell script itself.  $1 #Value of first command line parameter (similarly $2, $3, etc)  $# #In a shell script, the number of command line parameters.  $\* #een per een opties/:alle opties seperated  $- #Options given to the shell.  $? #Return the exit status of the last command.  $$ #Process id of script (really id of the shell running the script) | -r file #Check if file is readable.  -w file #Check if file is writable.  -x file #Check if we have execute access to file.  -f file #Check if file is an ordinary file (as opposed to a directory, a device special file, etc.)  -s file #Check if file has size greater than 0.  -d file #Check if file is a directory.  -e file #Check if file exists. Is true even if file is a directory.  -z file #empty String |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **LOOPS**  for i in /etc/rc.\*;  do  echo $i  done | for ((i = 0 ; i < 100 ; i++)); do  echo $i  done | **FUNCTIONS**  myfunc() {  echo "hello $1"  }  Myfunc ilya | myfunc() {  local myresult='some value'  echo $myresult  } | **Apache en netwerk**  Dnf install –y httpd php  Service httpf status #status bekijke  Netstat –tl #overzicht connecties  Ps –ef overzicht seervices  Service httpd start #service starten  Ifconfig #ipadressen bekijken  Route –n #gateway bekijken  Nslookup [website] #ipadr kijken  /etc/resolf.conf #beschikb dns serv  /etc/sysconfig/network-scripts  Vi ifcfg-eth0 #netkarten bekijken |
| for i in {1..5}; do  echo "Welcome $i"  done | < file.txt | while read line; do  echo $line  done | myfunc() {  return 1  } | if myfunc; then  echo "success"  else  echo "failure" fi |

|  |  |  |  |
| --- | --- | --- | --- |
| **CASE**  CASE  case "$1" in  a) cmd1 ;;  b) cmd2 ;;  c) cmd3 ;;  \*) cmd4 ;;  esac | **IFELSE**  if [[ -z "$string" ]]; then  echo "String is empty"  elif [[ -n "$string" ]]; then  echo "String is not empty"  fi  If (()) #ease  test 56 -gt 55 && echo true || echo false | **ARRAYS DICT**  Fruits=('Apple' 'Banana' 'Orange')  Fruits[0]="Apple" | /var/log #log bestanden  Var/log/journal of dnf.log #syslogs  Journalctl –f #live logs bekijken  Journalctl –u httpd #log van httpd bekijken  Systemctl restart network  **Func**  my\_function () { func\_result="some result"}my\_functione  cho $func\_result |
| sounds[dog]="bark"  echo ${sounds[dog]}  for val in "${sounds[@]}"; do  echo $val  done |

|  |  |  |  |
| --- | --- | --- | --- |
| **WHILE**  i=100; while [ spatie $i -ge 0 spatie] ; do echo Counting down, from 100 to 0, now at $i; let i--; done | **UNTIL**  let i=100; until [ $i -le 0 ] ; do echo Counting down, from 100 to 1, now at $i; let i--; done | ls \*.txt > /dev/null 2>&1 if [ $? -ne 0 ] then echo "There are 0 files ending in .txt" else let i=0 for file in \*.txt do let i++ done echo "There are $i files ending in .txt" fi | if [ "$#" == "0" ] then echo You have to give at least one parameter. exit 1 fi while (( $# )) do echo You gave me $1 shift done |

|  |  |  |
| --- | --- | --- |
| option="${1}"  case "${option}" in  -h|--help|-?) usage exit 0 ;; -  v|--verbose) verbose=y shift ;;  \*) printf 'Unrecognized option: %s\n' "${option}" usage exit 1 ;;  esac  var\_a=afoo() {  var\_b=b echo "${var\_a} ${var\_b}"} | while getopts ":afz" option;  do  case $option in  a) echo received -a ;;  f) echo received -f ;;  z) echo received -z ;;  \*) echo "invalid option -$OPTARG" ;;  esac  done | **The (( )) allows for evaluation of numerical expressions:**  (( 42 > 33 )) && echo true || echo false  **let** built-in shell function instructs the shell to perform of **arithmetic:**  let x="10 + 100/10" ; echo $x  **between 1 and 100**  if [ $n1 -lt 1 -o $n1 -gt 100 ] then echo Wrong number... exit 1 fi |