TP 8:

# Grep :

## Arguments:

### –n --line-number

Prefix each line of output with the 1-based line number within its

input file. (-n is specified by POSIX.)

### –c --count

Suppress normal output; instead print a count of matching lines for

each input file. With the -v, --invert-match option (see below),

count non-matching lines. (-c is specified by POSIX.)

### -i --ignore-case

Ignore case distinctions in both the PATTERN and the input files.

(-i is specified by POSIX.)

### -v --invert-match

Invert the sense of matching, to select non-matching lines. (-v is

specified by POSIX.)

### -l --files-with-matches

Suppress normal output; instead print the name of each input file

from which output would normally have been printed. The scanning

will stop on the first match. (-l is specified by POSIX.)

### -L --files-without-match

Suppress normal output; instead print the name of each input file

from which no output would normally have been printed. The scanning

will stop on the first match

## Questions :

### Comment faire apparaître le numéro de la ligne où figure le mot

### recherché ?

-n --line-number

### Comment faire pour afficher le nombre d'occurrences du mot

### recherché ? Tester.

-c –count

grep "pouet" oui.txt -c --> 2

### Comment faire pour que grep ignore la casse des caractères

### (différences entre majuscules et minuscules) dans sa recherche ?

-i --ignore-case

### Comment faire pour faire apparaître non pas les lignes où figurent le

### mot recherché, mais les noms des fichiers ? Tester

-l --files-with-matches

root@lab4ceVM:~# grep "pouet" oui\* -l

oui2.txt oui.txt

### Comment faire apparaître les lignes où ne figure pas le mot recherché ?

-v --invert-match

### Comment faire apparaître les noms des fichiers ne contenant pas le mot

### recherché ?

-L --files-without-matches

### Comment faire pour que grep ne recherche que les lignes où figure le

### mot tel quel, et non pas ses variantes ? Par exemple : on cherche le mot

travail, mais pas travailleur ou travailler.

grep "^pouet$" \*

il suffit de rajouter '^' au début du paterne er '$' a la fin de celui-ci

# CUT :

## Arguments:

### -c, --characters=LIST

select only these characters

### -d, --delimiter=DELIM

use DELIM instead of TAB for field delimiter

### -f, --fields=LIST

select only these fields; also print any line that contains no

delimiter character, unless the -s option is specified

cut -c=2-5 /etc/passwd

cut -c=-6,10 /etc/passwd

grep "home" /etc/passwd | cut -d ":" -f 1,6

## Script:

|  |
| --- |
| #!/bin/bash  if [ "$#" -ne 3 ];  then  d=$(date +%Y-%m-%d)  else  if [ ${#3} -ne 4 ];  then echo "Bad year format" ;  exit  fi  if [ ${#2} -ne 2 ];  then echo "Bad mouth format" ;  exit  fi  if [ ${#1} -ne 2 ];  then echo "Bad day format" ;  exit  fi  d="$3-$2-$1"  fi  echo $d  nbJour=$(echo $d | cut -d "-" -f 3)  nbMois=$(echo $d | cut -d "-" -f 2)  nbAnnee=$(echo $d | cut -d "-" -f 1)  echo $nbJour  echo $nbMois  declare -a dateT=( "none"  "jan."  "fev."  "mar."  "apr."  "may."  "jun."  "jul."  "aug."  "sep."  "oct."  "nov."  "dec."  )  arrayS=($(grep "session opened" auth.log.txt | grep "${dateT[$nbMois]} $2" | cut -d " " -f 11 | sort -u ))  jour=$(date -d $d +%A)  mois=$(date -d $d +%B)  echo "Les utilisateurs connectés le $jour $nbJour $mois $nbAnnee sont :"  for name in "${arrayS[@]}"  do  echo " - $name"  done |