UNIX

Lecture 11 : Filtering commands

Filipe Vasconcelos[1]
[1]ESME, Lille, filipe.vasconcelo@esme.fr

ESME
PARIS•BORDEAUX•LILLE•LYON

Outline

Intended Learning Outcomes

ILO2

Recognize, describe, and use basic Bash commands, or quickly find documentation about these commands.

ILO4

Produce, test, and verify the results of Shell scripts (in Bash) to perform tasks ranging from simple to complex algorithms.

Outline

Filters

- A filter is a command that reads data from standard input, performs operations on the received lines, and writes the result to standard output.
- Filters are used to manipulate text data.
- Filters can be used with files or applied to standard outputs using pipes.

grep

Example already encountered

grep: "g/re/p"

- Usage: This command searches and displays all lines that match a pattern (string or regular expression).
- Syntax: `grep [options] <pattern> <file>`
- Result: Lines from the file containing the pattern
- Note: `grep -E` is equivalent to `egrep`

Outline

cut

Extract from the man page

```
NAME
       cut - Remove sections from each line of files.

DESCRIPTION
       cut displays selected parts of each line of each input file, or from
       standard input if none are given.
       A file name of `-' denotes standard input.

       The selected parts are typically selected from each line by specifying
       arguments to one of the
       following options:

       -b, --bytes list
       -c, --characters list
       -f, --fields list
```

cat count.txt

```
cat count.txt
```

```
un one um ichi
deux two dois ni
trois three três san
quatre four quatro yon
cinq five cinco go
six six seis roku
```

cut -c<list>

```
cut -c1-4 count.txt
```

```
un o
deux
troi
quat
cinq
six
```

cut -d<delimitor> -f<fields>

```
cut -d ' ' -f 1,3 compter.txt
```

```
un um
deux dois
trois três
quatre quatro
cinq cinco
six seis
```

Outline

head
excerpt from the manual

```
NAME
       head - Display the beginning of files

SYNOPSIS
       head [OPTION] ... [FILE] ...

DESCRIPTION
       Display  the first 10 lines of each FILE to the standard output.

       -c, --bytes=[-]K
              print the first K bytes of each file

       -n, --lines=[-]K
              print the first K lines instead of the first 10; with the leading `-',
              print all but the last K lines of each file
```

```
head compter2.txt
```

```
un one um ichi
deux two dois ni
trois three três san
quatre four quatro yon
cinq five cinco go
six six seis roku
sept seven sete nana
huit eight oito hachi
neuf nine nove kyū
dix ten dez jū
```

```
head -n 3 compter2.txt
```

```
un one um ichi
deux two dois ni
trois three três san
```

Outline

tail
excerpt from the manual

```
NAME
       tail - Display the last part of files

SYNOPSIS
       tail [OPTION] ... [FILE] ...

DESCRIPTION
       Display  the last 10 lines of each FILE to the standard output.

       -c
              display the last K bytes;

       -f
              output appended data as the file grows;

       -n, --lines=K
              display the last K lines, instead of the last 10;
       -s, --sleep-interval=N
              with -f, sleep for approximately N seconds (default 1.0)
              between iterations
```

```
tail compter2.txt
```

```
trois three três san
quatre four quatro yon
cinq five cinco go
six six seis roku
sept seven sete nana
huit eight oito hachi
neuf nine nove kyū
dix ten dez jū
onze eleven onze jûichi
douze twelve doze jûni
```

```
tail -n2 compter2.txt
```

```
onze eleven onze jûichi
douze twelve doze jûni
```

Monitoring with tail

tail -f

```
simulation > log &
tail -f log
```

```
iter : 325    value = 18040  Elapsed Time : 1101  (s)
iter : 326    value = 3281   Elapsed Time : 1103  (s)
iter : 327    value = 7102   Elapsed Time : 1113  (s)
iter : 328    value = 5786   Elapsed Time : 1116  (s)
iter : 329    value = 13874  Elapsed Time : 1119  (s)
iter : 330    value = 19195  Elapsed Time : 1121  (s)
iter : 331    value = 134    Elapsed Time : 1123  (s)
iter : 332    value = 32008  Elapsed Time : 1143  (s)
```

Outline

tr
excerpt from the manual

```
NAME
       tr - Translate or delete characters

SYNOPSIS
       tr [OPTION]... SET1 [SET2]

DESCRIPTION
       Translate, squeeze, and/or delete characters from standard input,
       writing to standard output.

       -c, -C, --complement
              use the complement of SET1

       -d, --delete
              delete characters in SET1, do not translate

       -s, --squeeze-repeats
              replace each sequence of repeated characters in the input
              that is in SET1 with a single occurrence of that character
```

tr Example

```
cat example_course_tr.txt
```

```
9143 777 258
```

tr Example (GitHub)

```
cat example_course_tr.txt | tr 12345789 'achsm>db'
```

```
bash >> cmd
```

Outline

uniq
extrait du man

```
NAME
       uniq - report or omit repeated lines

SYNOPSIS
       uniq [OPTION]... [INPUT [OUTPUT]]

DESCRIPTION
       Filter  adjacent  matching lines from INPUT (or standard input), writing to
       OUTPUT (or standard output).

       With no options, matching lines are merged to the first occurrence.

       Mandatory arguments to long options are mandatory for short options too.

       -c, --count
              prefix lines by the number of occurrences

       -d, --repeated
              only print duplicate lines, one for each group

       -D     print all duplicate lines
```

Outline

sed Command

The sed command is a stream editor that operates on a stream of text using internal sed commands.

Consider the following file (GitHub)

```
cat distrib.txt
```

```
Linux
Solaris
Ubuntu
Fedora
Debian
RedHat
Mandriva
<blank line>
```

By default, sed does nothing

```
cat distrib.txt | sed
```

The 'd' command (delete) erases the first line

```
cat distrib.txt | sed '1d'
```

```
Solaris
Ubuntu
Fedora
Debian
RedHat
Mandriva
<blank line>
```

The 'd' command (delete) erases multiple lines

```
cat distrib.txt | sed '1,4d'
```

```
Debian
RedHat
Mandriva
<blank line>
```

Substitution

`sed s/a/u/ distrib.txt` (first occurence)

```
Linux
Soluris
Ubuntu
Fedoru
Debiun
RedHut
Mundriva
```

Substitution

`sed s/a/u/g distrib.txt` (global)

```
Linux
Soluris
Ubuntu
Fedoru
Debiun
RedHut
```

Examples of Using sed (GitHub)

Delete all // comment lines and empty lines

```
sed s/^$/d lignes_vides.txt

sed -e '/^$/d' -e '/^*/d' code.c
```

```
#include <stdio.h>
int main(void)
{
    int n = 10;
    for (int i =0;i<n;i++)
    {
        printf("i = %d",i)
    }
}
```

Outline

Some Miscellaneous Commands

- `sort`: sorts the lines of a file
- `uniq`: displays lines by removing duplicates usage: `sort <file> | uniq`
- `comm`: compares lines between sorted files
- `diff`: displays differences between two files
- `basename`: removes suffixes
- `pr`: formats output for printing
- `awk`: a language for processing files line by line (syntax inspired by C)
- ...

Application Exercises (GitHub)

Exercise 1

The file `nat2019.csv` lists all the first names given in France since 1900. How many distinct first names have been given since 1900? Commands that can be used: `cat`, `sort`, `uniq`, `cut`, and `wc`

Exercise 2

From the file of emails (`liste_inge1`) of students in the 2020-2021 Inge1 class in Lille, create a two-column file with "First Name Last Name" (with capital letters at the beginning of each string). Commands that can be used: `cat`, `sort`, `cut`, and `sed`