



Scripting no Processamento de Linguagem Natural

Luís Filipe da Costa Cunha
lfilipecc1@gmail.com

José João Almeida
jj@di.uminho.pt





Github

Enviar email para: lfilipecc1@gmail.com

Assunto: SPLN-2122-Github:axxxx

Corpo:

Nome Completo

Número

Username Github

Lista módulos python interessantes para NLP



Sistemas Unix

- Write programs that do one thing and do it well.
- Write programs to work together.
- Write programs to handle text streams, because that is a universal interface.

Peter H. Salus. A Quarter-Century of Unix. Addison-Wesley. 1994



Filtros Unix

- Programas que recebem um input e produzem um output.
- Podem ser usados em operações de stream.
- Podem ser combinados para se criarem operações complexas.

Exemplo: tail

- Input do stdin ou ficheiros (argumentos)

```
$ find | tail
```

```
$ tail file.txt
```
- Output para o stdout ou pode ser redirecionado

```
$ tail file.text | nr
```

```
$ tail file.text > last_lines.txt
```
- O comportamento varia conforme os argumentos

```
$ tail -n 20          # show 20 lines instead of 10
```



Filtros UNIX

- **head**: Imprime linhas no início de um ou mais ficheiros
- **tail**: Imprime linhas no final de um ou mais ficheiros
- **cat**: Concatena ficheiros e imprime para o stdo
- **tac**: Concatena ficheiros e imprime para o stdo de forma inversa
- **cut**: remove secções de cada linha de ficheiros
- **nl**: numera linhas
- **wc**: conta linhas, palavras e caracteres
- **grep**: filtra por padrão



Filtros UNIX

- **tr**: translate or delete caracteres
- **sed**: search, replace, delete
- **awk**: pattern scanning and processing language
- **sort**: sort lines
- **uniq**: report or omit repeated lines
- **tee**: read from standard input and write to standard output and files
- **paste**: merge lines of files



Filtros UNIX

Command:

```
$ cat -n file.txt
```

```
1 200 - Alexandra
2  Pedro
3  Maria
4  Manuel
5  João
6  Manuel
7  1 - Filipe
8  2 - João
9  pedro
10 maria
11 manuel
```

Command:

```
$ cat file.txt | sort
```

```
1 - Filipe
200 - Alexandra
2 - João
João
manuel
Manuel
Manuel
maria
Maria
pedro
Pedro
```

Command:

```
$ sort -n file.txt
```

```
João
manuel
Manuel
Manuel
maria
Maria
pedro
Pedro
1 - Filipe
2 - João
200 - Alexandra
```


Filtros UNIX



Command:

```
$ head ocorr.txt
```

output:

```
755 Maria de Jesus
483 Maria
462 Manuel
256 João
226 Antónia de Jesus
224 José
169 António
102 Carolina de Jesus
93 Antónia
93 Francisco
```

Command:

```
$ sort -n ocorr.txt | tail -n 10
```

output:

```
93 Antónia
93 Francisco
102 Carolina de Jesus
169 António
224 José
226 Antónia de Jesus
256 João
462 Manuel
483 Maria
755 Maria de Jesus
```

Command:

```
$ cat -n nomes.txt
```

output:

```
1  Manuel
2  Constantina
3  Luísa
4  Albina
```

```
$ tac cat -n nomes.txt |tac
```

output:

```
4  Albina
3  Luísa
2  Constantina
1  Manuel
```

Filtros UNIX



```
$ cat linux.txt
```

linux é um bom os. unix é opensource.
aprender filtros unix
linux linux qual escolher.
linux é fácil de aprender.

```
$ echo "Adoro Utilizar filtros Unix"  
| sed 's/\(\b[A-Z]\)/\(\1\)/g'
```

(A)doro (U)tilizar filtros (U)nix

```
$ sed '3d' linux.txt
```

```
$ sed '$,3 s/linux/unix/g' linux.txt
```

linux é um bom os. unix é opensource.
aprender filtros unix
unix unix qual escolher.
unix é fácil de aprender.

```
$ sed '/unix/d' linux.txt
```

linux linux qual escolher.
linux é fácil de aprender.

```
$ sed '3,$d' linux.txt
```

Filtros UNIX



Command:

```
$ echo "{unix é fácil de aprender}" | tr '{} ' '()'
```

output: (unix é fácil de aprender)

```
$ echo "adoro spln" | tr 'a-z' 'A-Z'
```

output: ADORO SPLN

```
$ echo "adoro spln" | tr 'a-z' 'g-zabcdef'
```

output: gjuxu yvtr

```
$ echo "tr@é#umaMferramenta~poderosa" | tr -c 'a-zé' ' ' '
```

output: tr é uma ferramenta poderosa

```
$ cat filtros.txt
```

filtros unix poupam tempo

filtros unix poupam tempo

spln é uma uc de MEI

filtros unix poupam tempo

```
$ uniq -c filtros.txt
```

2 filtros unix poupam tempo

1 spln é uma uc de MEI

1 filtros unix poupam tempo

-i ignore case

-d imprime apenas linhas repetidas

Composição de Filtros



Command: Calcula quais os comandos mais usados

```
$ history \  
| awk '{a[$2]++}END {for( i in a){print(a[i] "\t" i)}}'\ \  
| sort -rn \  
| head -n 20 \  
| nl
```

Command: Calcula o número de ocorrências

```
$ sort | uniq -c | sort nr
```

Command: Gera N caracteres aleatórios

```
$ cat /dev/urandom | tr -dc "a-z0-9A-Z" | head -c 20
```

Exercício



Qual dos seguintes comandos devolve os 3 ficheiros com menor número de linhas?

```
$ wc -l * > sort -n > head -n 3
```

```
$ wc -l * | sort -n | head -n 1-3
```

```
$ wc -l * | head -n 3 | sort -n
```

```
$ wc -l * | sort -n | head -n 3
```

Qual dos comandos deve ser usado para devolver o número total de cada tipo de animal?

```
2012-11-05,deer
```

```
2012-11-05,rabbit
```

```
2012-11-05,raccoon
```

```
2012-11-06,rabbit
```

```
...
```

```
$ sort animals.txt | uniq -c
```

```
$ sort -t, -k2,2 animals.txt | uniq -c
```

```
$ cut -d, -f 2 animals.txt | uniq -c
```

```
$ cut -d, -f 2 animals.txt | sort | uniq -c
```

```
$ cut -d, -f 2 animals.txt | sort | uniq -c | wc -l
```



System Calls

```
import subprocess
```

```
exit_code = subprocess.call([ "ls", "-l" ])  
print(exit_code)
```

```
# returns output as byte string  
returned_output = subprocess.check_output( 'date' )
```

```
# using decode() function to convert byte string to string  
print('Current date is:', returned_output.decode( "utf-8" ))
```

```
import os
```

```
os.system()
```



System Calls


```
import subprocess

output = subprocess.run([ "ls", "-l"], capture_output=True)

print('Return code:', output.returncode)

print('Output:', output.stdout.decode( "utf-8"))
```

Filtros Unix no python



```
import getopt
import sys

options, remainder = getopt.getopt(sys.argv[1:], 'abc:', \
    ['output=', 'verbose', 'version='])
dict_opts = dict(options)

# input from STDIN or file
if remainder:
    input = open(remainder[0])
else:
    input = sys.stdin

# output to STDOUT or file
out = dict_opts.get('--output', None)
if out:
    output = open(out, 'w+')
else:
    output = sys.stdout

output.write("".join(input.readlines()))
```




Filtros Unix no python (simplificado)

```
import getopt, sys, fileinput

opts, args = getopt.getopt(sys.argv[ 1:], c:)

for line in fileinput.input(args):
    line = line.strip()
    line = line.lower()
    #....
```

Exercícios



1. Escreva um filtro Unix que imprima as primeiras 5 linhas que contenham datas, as 2 linhas anteriores e posteriores a cada ocorrência.
2. Escreva uma sequência de filtros Unix que imprima as linhas 40 a 50 de um ficheiro.
3. Escreva um programa em python que faça o mesmo, utilizando system calls.
4. Reescreva o programa sem utilizar system calls.
5. Escreva um programa que calcule o número de ocorrências de palavras de um ou mais ficheiros, com opções de ordenação alfabética e numérica.



Extra

1. Escreva um programa que substitui as contrações num ficheiro.

Exemplo: “I’m” deve ser substituído por “I am”.



Scripting no Processamento de Linguagem Natural

Luís Filipe da Costa Cunha
lfilipecc1@gmail.com

José João Almeida
jj@di.uminho.pt

