universidade de aveiro

Ferramentas de Descoberta do Espaço Ocupado em Disco

Unidade Curricular: Sistemas Operativos;

Preparado por: Luís Fonseca nº 89066;

Pedro Escaleira nº 88821;

Índice

1. Índice de Imagens

2. Introdução

Inserido na unidade curricular de Sistemas Operativos, o presente documento visa a detalhar de forma sucinta de que maneira foi realizado o primeiro trabalho prático requerido.

Para tal, foi dado especial ênfase no que diz respeito à forma como a informação é apresentada, tal que esta seja facilmente lida e percebida.

Assim, este documento é de enorme importância para a perceção de como o trabalho foi abordado e que métodos foram usados para tal.

De uma maneira geral, o *script* criado em **bash**, permite procurar ficheiros/diretorias com determinadas propriedades e da forma estabelecida pelo Utilizador, indicando finalmente quais os ficheiros/diretorias dentro destes parâmetros.

3. Abordagem usada para Resolver o Problema

O script apresentado começa por fazer uma leitura dos parâmetros inseridos pelo utilizador, verificando a correta inserção dos mesmos, para que, caso tal não se verifique, o programa seja imediatamente encerrado com a devida mensagem de erro. Pelo contrário, caso se verifique a sua correção, de acordo com a sintaxe exigida pelos requisitos, estes são registados de forma a serem posteriormente tratados.

Finalizado este processo, chega a altura da utilização destes, de modo a que se obtenha o *output* pretendido. Este papel é assegurado pela função ***seeDirs()***, caracterizada por fazer uma pesquisa recursiva em todas as subdiretorias pertencentes à(s) diretoria(s) passada(s).

Nos seguintes pontos iremos, de forma breve, explicar a implementação que usamos para chegar ao resultado final esperado.

3.1 Tratamento de argumentos

Os argumentos inseridos pelo Utilizador são lidos de forma sequencial e tratados de forma individual, tal que, isolados dos outros, se garanta que a ordem pela qual são inseridos não seja relevante.

O primeiro passo a tomar é saber que tipo de parâmetro o Utilizador estará a especificar para que, em seguida, se possa tratar e registar a respetiva escolha.

Seguidamente, é necessário verificar se tal parâmetro tem um "valor" atribuído. Esta verificação é feita pela função ***verify()***, que não faz mais do que retornar um *boolean,* consoante a sua existência, tornando o código mais robusto. Desta forma, já "fora da função", o programa é abortado no caso do retorno ser negativo.

O tratamento feito para cada opção é:

- **-n**, **-d** e **-e** (apenas no caso do script nespace.sh) é analisado se contém um argumento associado
- **-l** e **-L** é analisado se contém um argumento associado, se esse argumento é um número e se as duas aparecem em conjunto
- **-r** e **-a** não é necessária qualquer verificação, a não ser a se foram ou não evocada.

Após todo este processo, é chamada a função ***seeDirs()*** com os respetivos parâmetros.

3.2 Tratamento da opção -n

```
22  array_files=()
23  for i in ${files[@]}; do   #tratamento da opção -n
24      if [[ $( echo $i | rev | cut -d "/" -f1 | rev ) =~ $2 ]]; then
25          array_files=("${array_files[@]}" "$i")
26      fi
27  done
```

A opção **-n** tem como objetivo fazer a seleção dos ficheiros a serem considerados, através da passagem duma expressão regular, a qual vai ser verificada no nome de cada ficheiro. Assim, a forma que encontramos para resolver este problema foi ler todos os nomes dos ficheiros de cada subdiretoria (antes do tratamento de cada opção, é armazenado num

array de nome ***files*** o caminho relativo para cada ficheiro da subdiretoria que está a ser tratada), de forma a selecionar os adequados, fazendo, para isso, uso do operador binário **=~**, com o objetivo de verificar se a expressão regular referida anteriormente faz ou não parte do nome de cada um. Por fim, cada um destes ficheiros é armazenado num novo array, de seu nome ***array_files***.

3.3 Tratamento da opção -d

```
29  array_date=();
30  for i in ${array_files[@]}; do   #tratamento da opção -d
31      actual="$(stat -c %x $i)"
32      file_date=$(date -d "$actual" +%s)
33      compare_date=$(date -d "$4" +%s)
34      if [[ $file_date -le $compare_date ]]; then
35          array_date+=($i)
36      fi
37  done;
```

Por sua vez **-d** sofre um tratamento bastante simples: a partir do array de nome ***array_files***, faz-se uma iteração pelo mesmo de forma a obter a data de acesso mais recente ao mesmo. Assim, e tendo também registada a data atual, faz-se uma conversão destas para um valor numérico (correspondente aos segundos passados desde 1970-01-01 00:00:00) o que se torna muito útil, uma vez que permite uma fácil comparação entre as datas. Posto isto, ainda durante o *loop*, verifica-se quais os ficheiros cuja data de acesso é inferior à atual, para que em caso afirmativo estes sejam adicionados a um array temporário ***array_date***. Finalmente o resultado é atualizado em ***array_files***.

3.4 Tratamento da opção -L

```
39  array_files=(${array_date[@]});
40  if [[ $5 != "" && ${#array_files[@]} -ne 0 ]]; then   #tratamento da opção -L
41      for ((num=0; num<$5; num++)); do
42          if [[ ${array_files[num]} == "" ]]; then
43              break
44          fi
45          bigger=(${bigger[@]} "${array_files[num]}")
46      done
47      bigger=($(ls -S $(printf "%s\n" "${bigger[@]}")))
48
49      new_bigger=()
50      for ((num=0; num<$5; num++)); do
51          new_bigger=(${new_bigger[@]} ${bigger[num]})
52      done
53      bigger=(${new_bigger[@]})
```

Esta opção tem como objetivo indicar o número *n* de ficheiros, de entre os maiores de todas as diretorias, a serem considerados. Desta forma, a abordagem utilizada para reproduzir esse resultado foi armazenar os caminhos relativos dos *n* maiores ficheiros da diretoria que está a ser lida num array de nome ***bigger*** e, de seguida, selecionar os *n* maiores ficheiros desse array, para de seguida os respetivos caminhos serem armazenados nesse mesmo array.

3.5 Tratamento da opção -l

```
55  elif [[ $5 == "" ]]; then  #tratamento da opção -l
56      size=()
57
58      if [[ ${#array_files[@]} -gt 0 ]]; then
59          stat --printf="%s\n" "${array_files[@]}" &> /dev/null || NA=0
60          if [[ NA -ne 0 ]]; then
61              size=($( stat --printf="%s\n" "${array_files[@]}"))
62          fi
63      fi
64
65      if [[ $l == "" || $l -gt ${#size[@]} ]]; then
66          l=${#size[@]}
67      fi
```

Na opção **-l**, o utilizador pode indicar quantos ficheiros *n*, de entre os maiores, devem ser considerados em cada diretoria. Sendo assim, em cada subdiretoria lida, são armazenados num array chamado ***size*** os tamanhos dos ficheiros dos ficheiros da mesma por ordem decrescente, sendo que deste array são depois retirados os *n* maiores tamanhos para serem incluídos na soma do espaço ocupado por essa subdiretoria.

3.6 Tratamento das opções -r e -a

```
197 #tratamento de -r e -a
198 if [[ $a -eq 0 ]]; then
199     if [[ $r -eq 1 ]]; then
200         printf "%s\n" "${result[@]}" | sort -k2
201     else
202         printf "%s\n" "${result[@]}" | sort -k2 -r
203     fi
204 elif [[ $r -eq 0 ]]; then
205     printf "%s\n" "${result[@]}" | sort -n
206 else
207     printf "%s\n" "${result[@]}" | sort -n -r
208 fi
```

A opção **-r** tem como foco alterar a forma como o *output* final é mostrado ao Utilizador, pelo que apenas é "utilizada" após o termo da função ***seeDirs()***. O que esta opção faz é, quando selecionada, inverter a ordem natural do *output* original.

Tal como a opção **-r,** a opção **-a** também tem como objetivo modificar o *output* final, tornando-se útil, mais uma vez, finalizada ***seeDirs()***. Esta opção, quando escolhida, ordena o *output* por ordem alfanumérica (do caminho relativo das diretorias).

3.7 Funcionalidades Acrescentadas em nespace.sh

3.7.A.1 Tratamento da opção -e

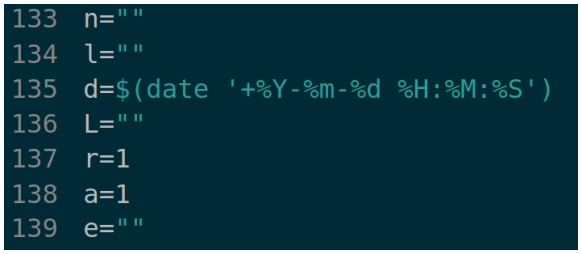
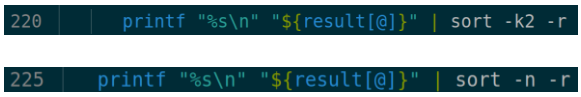
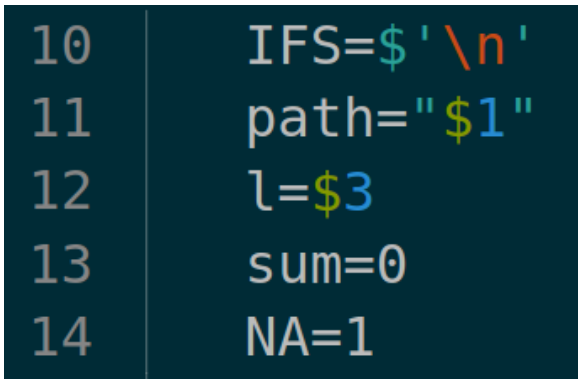
```
23 array_files=()
24 for i in ${files[@]}; do   #tratamento da opção -n e -e
25    v=1
26    for n in $e; do
27       if [[ "$i" == "$n" ]]; then
28          v=0
29          break
30       fi
31    done
32    if [[ $v -eq 1 ]] && [[ $( echo $i | rev | cut -d "/" -f1 | rev ) =~ $2 ]]; then
33       array_files=("${array_files[@]}" "$i")
34    fi
35 done
```

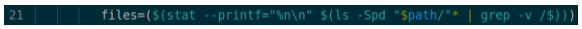
Esta opção é restrita ao script **nespace.sh**, tendo como objetivo ignorar ficheiros importantes para o utilizador, sendo para isso necessário que os caminhos para estes sejam armazenados num ficheiro à parte, cujo nome é passado como argumento desta opção. Internamente, os caminhos guardados neste ficheiro são armazenados numa variável de nome ***e***, após a leitura do ficheiro, sendo que posteriormente na função ***seeDirs()***, estes vão ser comparados com os caminhos de cada ficheiro de cada subdiretoria lida, pelo que todos os diferentes são os considerados na opção **-n**.

3.8 Tratamento de erros

Neste segmento do problema, os scripts têm de conseguir analisar se é possível ou não aceder a todas as subdiretorias e aos tamanhos dos ficheiros. Para isso, antes de cada uma destas tarefas ser realizada, é feita uma tentativa de acesso a uma subdiretoria ou de obtenção o tamanho dos ficheiros. Caso essa tentativa resulte num insucesso, ao espaço ocupado pelos ficheiros da respetiva diretoria é atribuído o valor -1, para que no resultado a ser mostrado ao utilizador este seja apresentado como NA.

4. Explicação parcial do código feito

***Prints* do código**	**Explicação**
	Neste trecho, há a inicialização das variáveis que serão utilizadas no processo de leitura e verificação de argumentos que ocorre imediatamente a seguir. Tendo em conta que se trata do interpretador de comandos **bash**, tal inicialização não seria necessária. Não obstante, foi mantida, de modo a tornar o código mais conciso e legível.
	Aqui, nesta linha constituinte de ***verify(),*** é feita a verificação de se o argumento passado é *string null* ou não. Ou seja se j á lhe foi atribuído um valor ou não.
	Parque integral do final do *script,* onde se dá *print* do resultado de tudo o que foi feito anteriormente. Uma vez que este resultado deve ser impresso de uma certa forma ordenada, o *output* de *printf* é "redirecionado", através do uso de um *pipe* para *sort,* onde através do parâmetro "k" se define em relação a que campo se quer ordenar; do parâmetro "n" se define que se quer uma ordenação numérica; e do parâmetro "r" se define que se quer em ordem inversa.
	Variáveis inicializadas sempre que a função ***seeDirs()*** é chamada: • ***IFS*** é uma variável especial da **bash**, que define como fazer a divisão de linhas em palavras. Neste caso é usada de forma a que os espaços que possam aparecer nos nomes das diretorias ou ficheiros seja tratado da forma correta durante a execução da função • ***path*** é a variável que vai conter o caminho para a pasta que está a ser analisada cada vez que a função é executada • ***l*** contém o valor da opção **-l** passada (ou não) como argumento na inicialização do programa • ***sum*** representa a soma de cada diretoria e, sendo assim, vai ter de ser inicializada

	a 0 no início de cada leitura de cada diretoria • ***NA*** é a variável de controlo de erros ao ler o tamanho dos ficheiros ou abrir diretorias, sendo que é inicializada com o valor 1 (false) e passada a 0 (true) sempre que ocorra um dos referidos erros
`16  ls $path &> /dev/null || NA=0`	***&>*** é um comando usado para fazer redireccionamento do conteúdo de stdout e stderr (prints "normais" e de erros no terminal) para o ficheiro que aparece à frente desse comando. Desta forma, sempre que haja um erro (neste caso a abrir a pasta atual), ou não, o que é escrito no terminal é redirecionado para o ficheiro **/dev/null**, ficheiro este que destrói toda a informação nele escrita. Para além disso, sempre que haja um dos referidos erros, a variável ***NA*** passa a ter o valor 0 (true)
	Opções passadas para ***ls***: • **-S** é usado para que os ficheiros e diretorias sejam listados por ordem de tamanho • ***-p*** é usado para que as diretorias listadas contenham o caracter '/' no final do seu nome • ***-d*** é usado para que as diretorias e ficheiros listados contenham o caminho relativo
	***rev*** é um comando que troca a ordem dos caracteres da string passada
`39  actual="$(stat -c %x $i)"`	***stat -c %x*** com o parâmetro "c" é especificado a *stat* o formato no qual o queremos, sendo neste caso com a indicação apenas da última data de acesso, em formado *human readable*.
`40  file_date=$(date -d "$actual" +%s)`	***date -d <data> +%s*** faz a conversão da data passada para segundos, para que se possa fazer comparação entre datas em qualquer formato

5. Testes Realizados para validar a solução

```
./totalspace.sh -L 3 SO
13480 SO/aula 9/incrementer/incrementerSafe
13280 SO/aula 9/incrementer/incrementer
12992 SO/aula 7/myls
```

Figura 1 - ./totalspace.sh -L 3 SO

```
./totalspace.sh -a -L 3 SO
12992 SO/aula 7/myls
13280 SO/aula 9/incrementer/incrementer
13480 SO/aula 9/incrementer/incrementerSafe
```

Figura 2 - ./totalspace.sh -a -L 3 SO

```
./totalspace.sh -a SO
411241 SO
669 SO/aula 2
4290 SO/aula 3
12425 SO/aula 4
85727 SO/aula 5
74484 SO/aula 6
64886 SO/aula 7
91376 SO/aula 8
77328 SO/aula 9
14539 SO/aula 9/dinner
48273 SO/aula 9/incrementer
14516 SO/aula 9/prodcon
```

Figura 3 - ./totalspace.sh -a SO

```
./totalspace.sh -r -a -L 3 SO
13480 SO/aula 9/incrementer/incrementerSafe
13280 SO/aula 9/incrementer/incrementer
12992 SO/aula 7/myls
```

Figura 4 - ./totalspace.sh -r -a -L 3 SO

```
./totalspace.sh -r -a SO
14516 SO/aula 9/prodcon
48273 SO/aula 9/incrementer
14539 SO/aula 9/dinner
77328 SO/aula 9
91376 SO/aula 8
64886 SO/aula 7
74484 SO/aula 6
85727 SO/aula 5
12425 SO/aula 4
4290 SO/aula 3
669 SO/aula 2
411241 SO
```

Figura 5 - ./totalspace.sh -r -a SO

```
./totalspace.sh -r SO
669 SO/aula 2
4290 SO/aula 3
12425 SO/aula 4
14516 SO/aula 9/prodcon
14539 SO/aula 9/dinner
48273 SO/aula 9/incrementer
64886 SO/aula 7
74484 SO/aula 6
77328 SO/aula 9
85727 SO/aula 5
91376 SO/aula 8
411241 SO
```

Figura 6 - ./totalspace.sh -r SO

```
./totalspace.sh -d "nov 25 13:30" -r -a -L 3 SO
13480 SO/aula 9/incrementer/incrementerSafe
13280 SO/aula 9/incrementer/incrementer
12992 SO/aula 7/myls
```

Figura 7 - ./totalspace.sh -d "nov 25 13:30" -r -a -L 3 SO

```
./totalspace.sh -d "nov 25 13:30" -r -a SO
14516 SO/aula 9/prodcon
48273 SO/aula 9/incrementer
14539 SO/aula 9/dinner
77328 SO/aula 9
91376 SO/aula 8
64886 SO/aula 7
74484 SO/aula 6
85727 SO/aula 5
12425 SO/aula 4
4290 SO/aula 3
669 SO/aula 2
411241 SO
```

Figura 8 - ./totalspace.sh -d "nov 25 13:30" -r -a SO

```
./totalspace.sh -d "nov 25 13:30" -r SO
669 SO/aula 2
4290 SO/aula 3
12425 SO/aula 4
14516 SO/aula 9/prodcon
14539 SO/aula 9/dinner
48273 SO/aula 9/incrementer
64886 SO/aula 7
74484 SO/aula 6
77328 SO/aula 9
85727 SO/aula 5
91376 SO/aula 8
411241 SO
```

Figura 9 - ./totalspace.sh -d "nov 25 13:30" -r SO

```
./totalspace.sh -d "nov 25 13:30" SO
411241 SO
91376 SO/aula 8
85727 SO/aula 5
77328 SO/aula 9
74484 SO/aula 6
64886 SO/aula 7
48273 SO/aula 9/incrementer
14539 SO/aula 9/dinner
14516 SO/aula 9/prodcon
12425 SO/aula 4
4290 SO/aula 3
669 SO/aula 2
```

Figura 10 - ./totalspace.sh -d "nov 25 13:30" SO

```
./totalspace.sh -l 3 -d "nov 25 13:30" -r -a -L 3 SO
ERRO!
Tentativa de uso das opções -l e -L em simultânio!
```

Figura 11 - ./totalspace.sh -l 3 -d "nov 25 13:30" -r -a -L 3 SO

```
./totalspace.sh -l 3 -d "nov 25 13:30" -r -a SO
14266 SO/aula 9/prodcon
31704 SO/aula 9/incrementer
14132 SO/aula 9/dinner
60102 SO/aula 9
26056 SO/aula 8
38568 SO/aula 7
34424 SO/aula 6
30064 SO/aula 5
11639 SO/aula 4
2041 SO/aula 3
367 SO/aula 2
203317 SO
```

Figura 12 - ./totalspace.sh -l 3 -d "nov 25 13:30" -r -a SO

```
./totalspace.sh -l 3 -d "nov 25 13:30" -r SO
367 SO/aula 2
2041 SO/aula 3
11639 SO/aula 4
14132 SO/aula 9/dinner
14266 SO/aula 9/prodcon
26056 SO/aula 8
30064 SO/aula 5
31704 SO/aula 9/incrementer
34424 SO/aula 6
38568 SO/aula 7
60102 SO/aula 9
203317 SO
```

Figura 13 - ./totalspace.sh -l 3 -d "nov 25 13:30" -r SO

```
./totalspace.sh -l 3 -d "nov 25 13:30" SO
203317 SO
60102 SO/aula 9
38568 SO/aula 7
34424 SO/aula 6
31704 SO/aula 9/incrementer
30064 SO/aula 5
26056 SO/aula 8
14266 SO/aula 9/prodcon
14132 SO/aula 9/dinner
11639 SO/aula 4
2041 SO/aula 3
367 SO/aula 2
```

Figura 14 - ./totalspace.sh -l 3 -d "nov 25 13:30" SO

```
./totalspace.sh -l 3 SO
203317 SO
60102 SO/aula 9
38568 SO/aula 7
34424 SO/aula 6
31704 SO/aula 9/incrementer
30064 SO/aula 5
26056 SO/aula 8
14266 SO/aula 9/prodcon
14132 SO/aula 9/dinner
11639 SO/aula 4
2041 SO/aula 3
367 SO/aula 2
```

Figura 15 - ./totalspace.sh -l 3 SO

```
./totalspace.sh -n ".*c" -l 3 -d "nov 25 13:30" -r -a -L 3 SO
ERRO!
Tentativa de uso das opções -l e -L em simultânio!
```

*Figura 16 - ./totalspace.sh -n ".*c" -l 3 -d "nov 25 13:30" -r -a -L 3 SO*

```
./totalspace.sh -n ".*c" -l 3 -d "nov 25 13:30" -r -a SO
13359 SO/aula 9/prodcon
31704 SO/aula 9/incrementer
12850 SO/aula 9/dinner
57913 SO/aula 9
11735 SO/aula 8
6082 SO/aula 7
21900 SO/aula 6
10195 SO/aula 5
220 SO/aula 4
621 SO/aula 3
70 SO/aula 2
108736 SO
```

*Figura 17 - ./totalspace.sh -n ".*c" -l 3 -d "nov 25 13:30" -r -a SO*

```
./totalspace.sh -n ".*c" -l 3 -d "nov 25 13:30" -r SO
70 SO/aula 2
220 SO/aula 4
621 SO/aula 3
6082 SO/aula 7
10195 SO/aula 5
11735 SO/aula 8
12850 SO/aula 9/dinner
13359 SO/aula 9/prodcon
21900 SO/aula 6
31704 SO/aula 9/incrementer
57913 SO/aula 9
108736 SO
```

*Figura 18 - ./totalspace.sh -n ".*c" -l 3 -d "nov 25 13:30" -r SO*

```
./totalspace.sh -n ".*c" -l 3 -d "nov 25 13:30" SO
108736 SO
57913 SO/aula 9
31704 SO/aula 9/incrementer
21900 SO/aula 6
13359 SO/aula 9/prodcon
12850 SO/aula 9/dinner
11735 SO/aula 8
10195 SO/aula 5
6082 SO/aula 7
621 SO/aula 3
220 SO/aula 4
70 SO/aula 2
```

*Figura 19 - ./totalspace.sh -n ".*c" -l 3 -d "nov 25 13:30" SO*

```
./totalspace.sh -n ".*c" -l 3 SO
108736 SO
57913 SO/aula 9
31704 SO/aula 9/incrementer
21900 SO/aula 6
13359 SO/aula 9/prodcon
12850 SO/aula 9/dinner
11735 SO/aula 8
10195 SO/aula 5
6082 SO/aula 7
621 SO/aula 3
220 SO/aula 4
70 SO/aula 2
```

*Figura 20 - ./totalspace.sh -n ".*c" -l 3 SO*

```
./totalspace.sh -n ".*c" SO
139563 SO
74106 SO/aula 9
47897 SO/aula 9/incrementer
26383 SO/aula 6
15978 SO/aula 8
13359 SO/aula 9/prodcon
13139 SO/aula 5
12850 SO/aula 9/dinner
9046 SO/aula 7
621 SO/aula 3
220 SO/aula 4
70 SO/aula 2
```

*Figura 21 - ./totalspace.sh -n ".*c" SO*

```
./totalspace.sh   /etc/cups
NA /etc/cups/ssl
NA /etc/cups
0 /etc/cups/ppd
0 /etc/cups/interfaces
```

Figura 22 - ./totalspace.sh /etc/cups

```
./totalspace.sh /etc/cups SO
329188 SO
91376 SO/aula 8
85727 SO/aula 5
74484 SO/aula 6
64886 SO/aula 7
6842 SO/aula 4
4287 SO/aula 3
1530 SO/aula 2
NA /etc/cups/ssl
NA /etc/cups
0 /etc/cups/ppd
0 /etc/cups/interfaces
```

Figura 23- ./totalspace.sh SO /etc/cups

```
./nespace.sh -e files SO
303284 SO
91376 SO/aula 8
74484 SO/aula 6
72815 SO/aula 5
51894 SO/aula 7
6842 SO/aula 4
4287 SO/aula 3
1530 SO/aula 2
```

Figura 24- ./nespace.sh -e files SO

6. Conclusão

Desta forma, considera-se que os objetivos do trabalho foram atingidos, uma vez que todos os pontos pedidos foram implementados com sucesso. É possível verificar este êxito através dos testes realizados ao script, onde foi sempre obtido o resultado pretendido.

Assim, conclui-se que este trabalho foi um sucesso, não só pelo referido anteriormente, mas também pelo facto dos conhecimentos de **bash** terem sido significativamente aprimorados.

7. Fontes e Material de Referência

https://unix.stackexchange.com/questions/184863/what-is-the-meaning-of-ifs-n-in-bash-scripting

https://www.tldp.org/LDP/abs/html/io-redirection.html

https://stackoverflow.com/questions/43703688/exception-handling-on-ls

https://serverfault.com/questions/7503/how-to-determine-if-a-bash-variable-is-empty

https://askubuntu.com/questions/12098/what-does-outputting-to-dev-null-accomplish-in-bash-scripts

https://www.cyberciti.biz/faq/linux-ls-command-sort-by-file-size/

https://www.oreilly.com/library/view/bash-cookbook/0596526784/ch11s04.html

https://stackoverflow.com/questions/6438896/sorting-data-based-on-second-column-of-a-file

https://ss64.com/bash/