|  |  |
| --- | --- |
|  |  |
| Monitorização de interfaces de rede em bash |  |
|  |  |
|  | 2021/2022L. Engenharia Informática |
|  | Rafael Remígio 102435João Correia 104360 |
|  | Professor Nuno LauProfessor Guilherme Campos |

|  |  |  |
| --- | --- | --- |
| Índice |  | |
| [***Introdução \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***](#_bookmark0)***3***  [***Main \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4***](#_bookmark0)  [***Looping \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***](#_bookmark0)***9***  [***Functions \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***](#_bookmark0)***10***  [***Testes Realizados\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***](#_bookmark0) ***13***  ***Bibliografia\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_19*** | |  |
|  | |  |

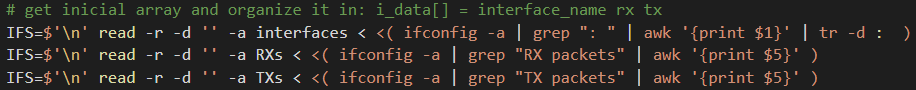
##### Introdução

#### Neste documento, explicaremos a resolução do 1º Trabalho Prático da disciplina de Sistemas Operativos que tem como objetivo o desenvolvimento e teste de um Script Bash capaz da monitorização de interfaces de rede apresentando esta informação para o utilizador no terminal. Deve permitir a visualização de todas as interfaces de rede disponíveis, o tamanho de pacotes transmitidos e recebidos durante um período de tempo imposto pelo Utilizador, e respetivas taxas de transferências, tal como ferramentas de Sorting e Seleção.

#### Explicaremos os métodos usados para cada função, tratamento de dados e ferramentas necessárias para o funcionamento do programa seguindo a ordem de construção, tal como também mostraremos como foi o código foi testado e analisado de forma a eliminar erros.

##### Main

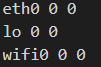
A partir do comando bash “ifconfig -a” recolhemos a informação necessária e organizamos estes dados. Criamos 3 arrays intermédios onde iremos salvar os valores de RX, TX, e o nome das interfaces, usando awk grep e tr para a selecionar a informação necessária.



Organizamos em seguida um array “i\_data” com três colunas divididas por um espaço (como uma matriz n por 3) capaz de ser percorrida a partir de coluna e linha através do awk e guardada toda a informação num só array, isto ajudará na organização, Sorting, e no corte necessário de alguma interface.

Exemplo de como ficará organizada a informação:





Usamos de seguida o comando “sleep” com o último argumento imposto pelo Utilizador.

Mais uma vez a partir do comando “ifconfig -a” recolhemos a informação e organizando-a da mesma forma anterior neste caso guardando a no array “data”. Depois de termos os dois arrays fazemos a subtração dos valores dos RX e TX e salvamos na variável (“data”), tendo deste modo toda a informação necessária para escrever e fazer quaisquer cálculos ou sortings intermédios.

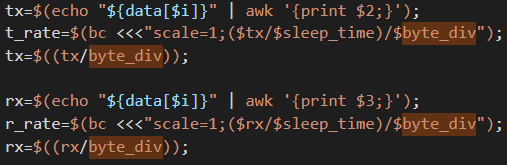
### Argumentos passados pelo Utilizador

Iteramos os argumentos passados pelo Utilizador e utilizando a instrução swicth (case) identificamos os argumentos passados e dar erro se for introduzido um valor não especificado, tal como verifica se o último argumento passado pelo utilizador é um número inteiro.

-c: muda o valor da variável booleana de modo que na próxima iteração leia o Regex introduzido pelo Utilizador

-p: muda o valor da variável booleana de modo que na próxima iteração leia o número de interfaces a mostrar. Na iteração seguinte e verificado se o argumento introduzido é inteiro é dado a variável “max”.

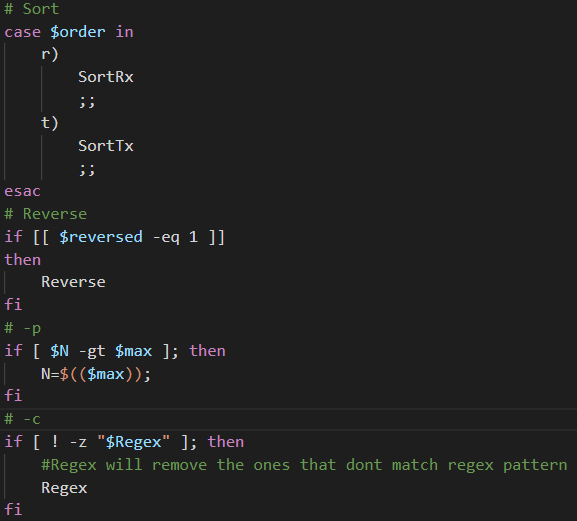
-b/k/m: são opções que mudam o “byte\_divisor”, uma variável que usamos para dividir os valores de RX e TX quando escrevemos a informação.



-r / t: Seleciona o uso da função SortRX e da função SortTX respetivamente através de um variável booleana. (Funções explicadas na secção sobre Funções)

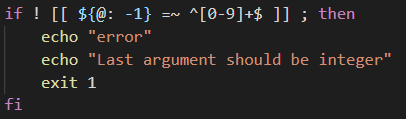
-R / T: Seleciona o uso da função SortRX e da função SortTX respetivamente através de um variável booleana. (Funções explicadas na secção sobre Funções). Neste caso são selecionadas as mesmas funções que na seleção de r ou t pois o T\_Rate e X\_Rate irão sempre ser proporcionais aos valores de RX e TX

-v: Seleciona o uso da função Reverse através de um variável booleana. (Funções explicadas na secção sobre Funções)



-l: Muda o valor da variável booleana looping (explicado abaixo com mais detalhe)

Verificamos que o último argumento é um número inteiro da seguinte forma:



## Argumentos não reconhecidos e erros

Caso sejam passados argumentos não suportados, é apresentada a mensagem:



Caso o último argumento passado não seja um número inteiro, é apresentada a mensagem:



Selecionando as opções -c e -p e o argumento seguinte será lido e introduzido numa variável $Regex e $max respetivamente. Se o valor introduzido a seguir a -p não for um inteiro é apresentada a seguinte mensagem de erro:



### Escrever a tabela com os valores desejados

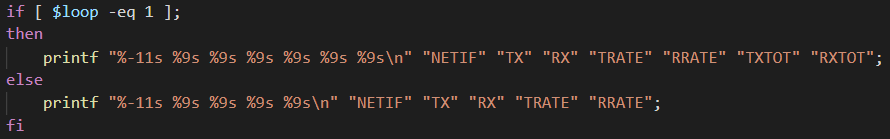
## Cabeçalho:

Dependendo do valor da variável booleana “$looping” o cabeçalho poderá ser formatado das seguintes formas:

NETIF TX RX TRATE RRATE

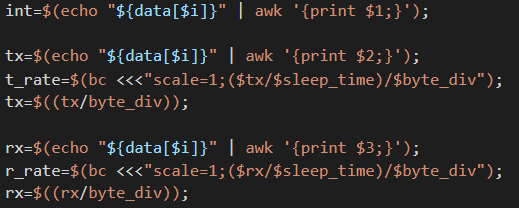
ou

NETIF TX RX TRATE RRATE TXTOT RXTOT



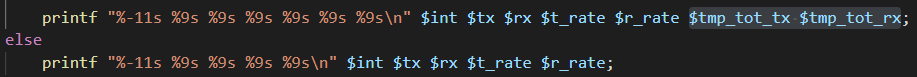
## Escrever os valores

Iteramos o array “data” recolhendo os valores de TX e RX de cada interface e inserindo-os numa nova variavel “tx” e “rx”, TX\_Rate e RX\_Rate respetivamente, dividindo os valores inseridos pela variável “byte\_division” explicada anteriormente e no caso de TX\_Rate e RX\_Rate dividimos também pelo nosso tempo de espera



Depois de termos os estas variáveis usamos o comando “printf” do bash imprimimos a informação com a formatação desejada.

Se for escolhida neste caso a opção -l serão também adicionadas as variáveis “$tmp\_tot\_tx” e “$tmp\_tot\_rx” (o seu uso será explicado na secção Looping)

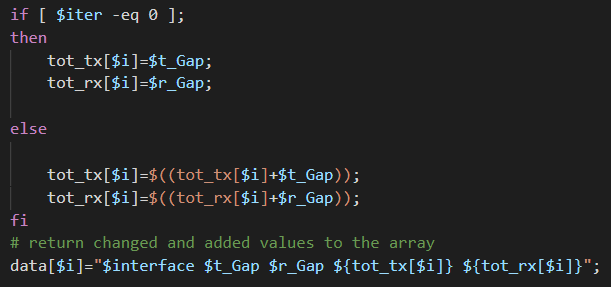


As interfaces serão sempre escritas pela ordem existente no array “data”, logo todas as funções de ordenação e remoção e adição de interfaces é feita antes da escrita no terminal, através de funções como SortRX, SortTX, Reverse e Regex, e o número de interfaces a mostrar é definido pela variável “max”, inicializada na passagem de argumentos.

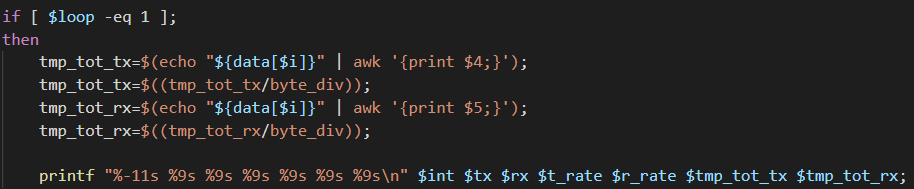
A ordem pela qual as interfaces são escritas é por predefinição a ordem alfabética, podendo ser alterada pelas opções -r, -t, -R e -T .

##### Looping

#### Caso a seja selecionada a função de loop (-l) o script será iterado sem terminar fim sendo só possível interrompe-lo forçando a paragem. Será visualizado pelo Utilizador a informação normal a visualizar tal como duas colunas com a informação do tamanho total de informação transmitida e recebida por interface. O script escrevera no ecrã em intervalos de tempo iguais ao introduzido pelo utilizador, incrementado a cada iteração o valor total das variáveis “tot\_tx” e “tot\_rx” e de seguida inserindo esta informação no array “data”.



#### A informação será apresentada através do uso de variáveis intermédias “tmp\_tot\_tx” e “tmp\_tot\_rx”

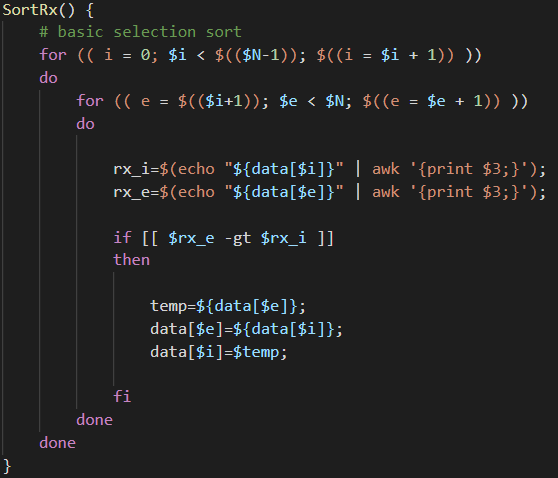


As funções Reverse, SortRX e SortTX são executadas em todas as iterações do código isto devido as mudanças no tamanho da informação recebida e transmitida por cada interface podendo ser ordenadas de formas diferentes em diferentes iterações do Loop, visto dependerem dos valores recebidos.

##### Funções

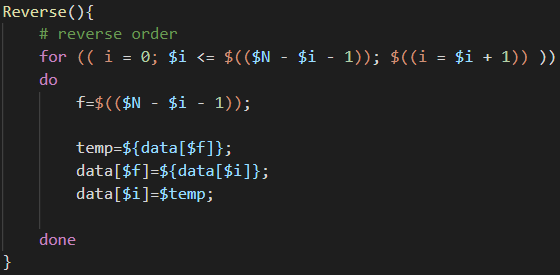
## SortTX() e SortRX()

As funções SortTX e SortRX ordenam o array “data” através de algoritmo de seleção das variáveis da tx ou rx respetivamente. O algoritmo de seleção ordena comparando cada elemento do array com os elementos seguintes posicionando o valor mais alto no índex acima, correndo até terminar o array.

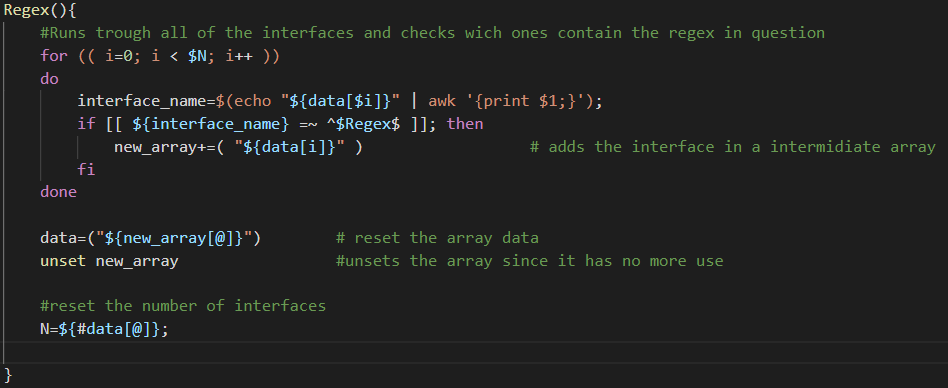


## Reverse()

A função Reverse inverte a ordem das interfaces no array “data” trocando os indexes das entradas no array com a mesma distância ao centro.



## Regex()

 A função Regex itera o array “data” recolhendo o nome da interface e comparando-a à expressão regular introduzida pela Utilizador nos argumentos da função. Se o nome da interface corresponder à expressão regular então a interface será adiciona a um array intermédio que depois será passado como array “data” original. Precisamos, depois de retirar as interfaces que não correspondem ao “regex”, mudar o número de iterações a correr quando escrevemos no ecrã.

##### Conclusão

A função **netifstat.sh** foi implementada com sucesso, mostrando todos os resultados pretendidos e implementadas todas as opções e funcionalidades pedidas pelo enunciado. Este trabalho levou-nos a aprender e implementar algumas ferramentas do **bash,** tal como o **awk, grep, sleep** e **printf.** Maior parte do conhecimento necessário para a completação do script foi adquirido nas aulas práticas através de guiões anteriores ou duvidas retiradas ao professor presente, tal como através do uso de fórums ou de documentação online e claro o manual do **bash.**

Estamos satisfeitos com a nossa implementação do programa apesar de nos apercebermos que ainda existe muitas formas de melhorar o uso de memoria tal existem algumas redundâncias e implementações pouco eficientes.

Tendo realizado todos testes e tendo estes dados os resultados esperados e bem-sucedidos damos o trabalho realizado com êxito.

Ambos tivemos o mesmo nível de envolvimento e aprendizagem no trabalho. Deste modo a nota deverá ser repartida de igual forma.

##### Teste realizados

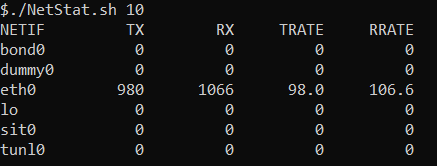


Figura 01.

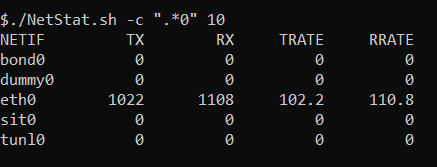


Figura 02.

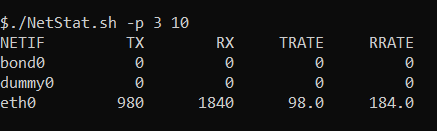


Figura 03

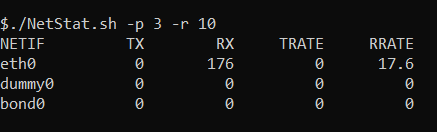


Figura 04

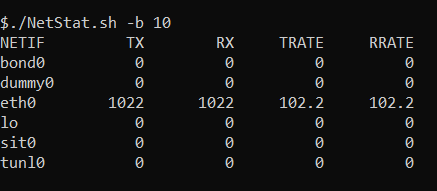


Figura 05

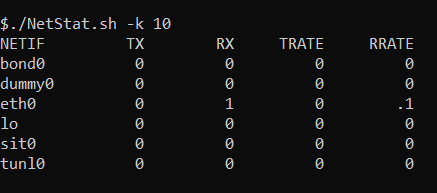


Figura 06

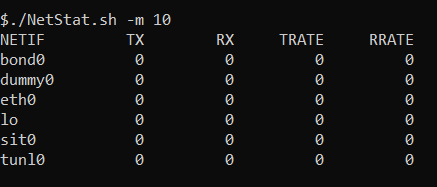


Figura 07

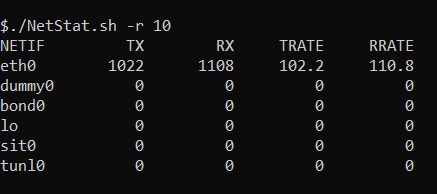


Figura 08

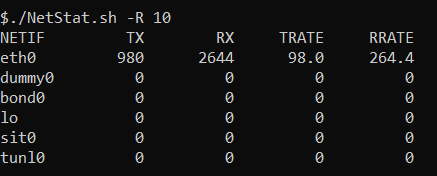


Figura 09

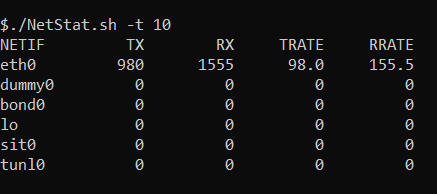


Figura 10

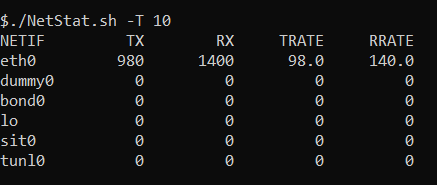


Figura 11

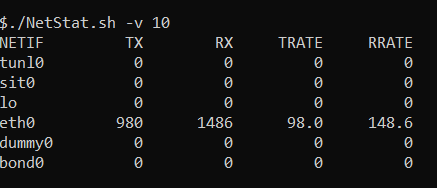


Figura 12

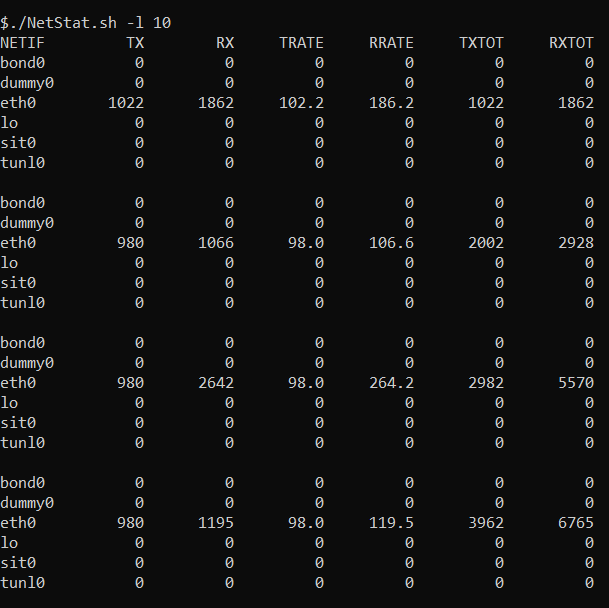


Figura 13

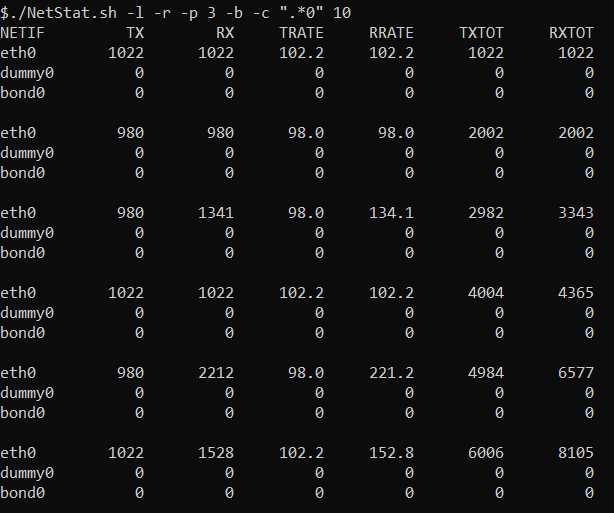
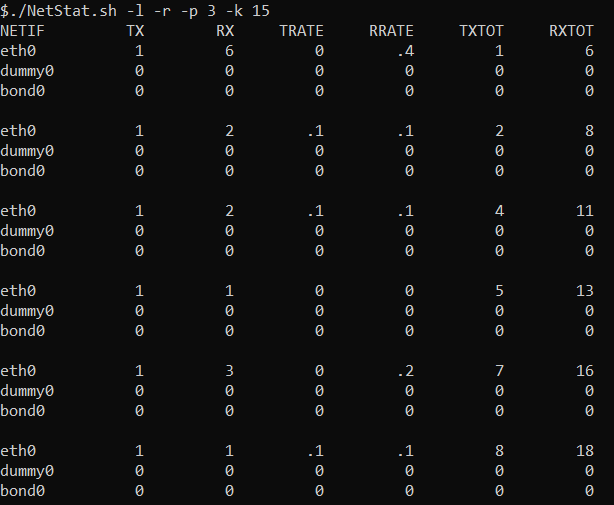


Figura 14

Figura 15

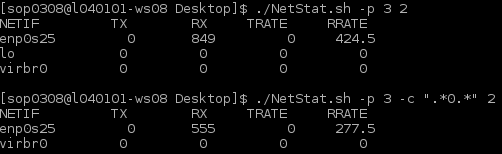


Figura 16

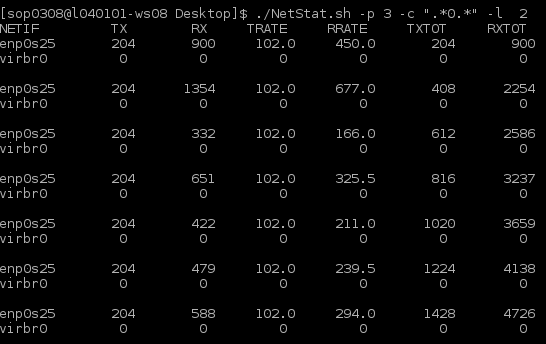


Figura 17

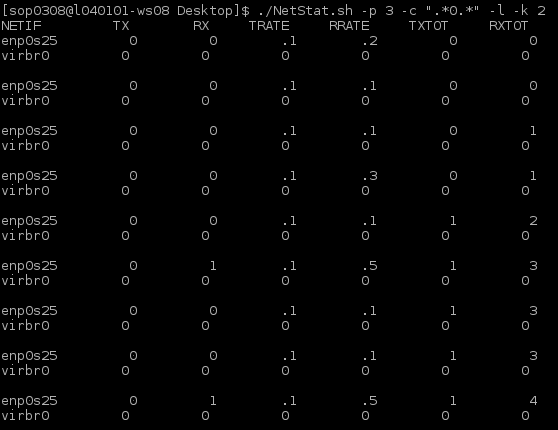


Figura 18

### Mensagens de erro



Figura 19



Figura 20



Figura 21



Figura 22

### Bibliografia

https://stackoverflow.com/

https://www.gnu.org/software/bash/manual/bash.html

https://www.gnu.org/software/gawk/manual/gawk.html

https://linuxconfig.org/

### Trabalho Realizado por:

## Rafael Remígio 102435

## João Correia 104360

