**Curso de Shell Script - Módulo 1 (Comandos básicos)**

**Geofisicando (Rodolfo Dirack)**

Vídeos:

<https://www.youtube.com/watch?v=HRRfgufskaw&list=PLLCFxfe9wkl-k0w-c_1i4sdZPUYt0Yc2P>

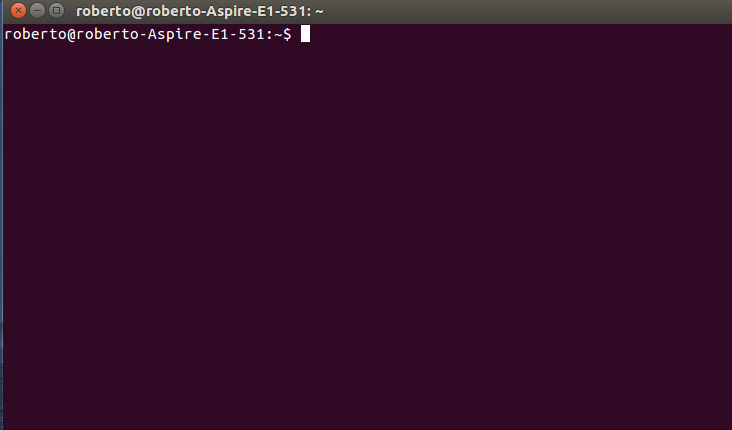
Resumo feito por Roberto Pinheiro

<https://github.com/betopinheiro1005>

# Aula 01 - Terminal

## Abrindo o terminal no Linux Ubuntu

- No Linux Ubuntu, para abrir um terminal pressione <Ctrl><Alt><T>



## Fechando o terminal

Para fechar a tela do terminal, use o comando:

exit

# Aula 02 - Comandos clear, reset e Ctrl + L

Estes comandos limpam a tela do terminal.

## clear

O comando clear ou pressionar <Ctrl><L> limpa a tela.

- Na realidade, o comando só rola a tela para que a informação existente não apareça.

## reset

O comando reset, de fato limpa a toda a tela.

# Aula 03 - O que é shell e o que é shell script

## Exibindo o nome do shell

echo $SHELL



bash é o nome do shell.

/bin/bash é o endereço do shell

- Existem vários tipos de shell

## O que é o Shell

- Shell é o interpretador de comandos no terminal, além disso, o Shell verifica se não há erros de síntaxe na digitação dos comandos. Caso não aja erros o Shell executa o comando.

- O shell é uma interface que se comunica com a parte central do sistema: o kernell

## O que é Shell Script

- O Shell entende uma linguagem específica chamada Shell Script que é uma linguagem com os comandos que iremos estudar durante o curso.

- Podemos utilizar essa linguagem como uma linguagem de programação.

- Shell Script é uma sequência de comandos que executa alguma tarefa.

# Aula 04 - Comando pwd

- O comando pwd obtém o endereço do diretório atual no terminal.

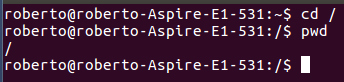
pwd



~ representa a pasta do usuário (/home/betopinheiro):

cd /

pwd



/ é o diretório raiz, ou seja, contém todos os outros diretórios.

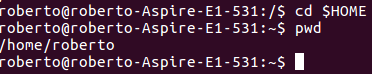
# Aula 05 - Comando cd

- O comando cd (change directory) altera o diretório corrente no terminal.

cd ~ ou cd $HOME

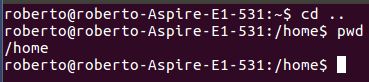
pwd

Vai para a pasta do usuário.

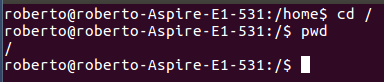


cd .. ou cd -

Volta uma pasta (sobe um nível).



cd /

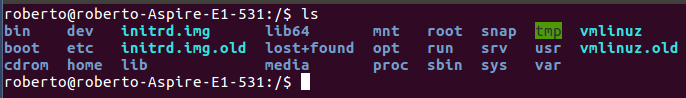


Vai para o diretório raiz.

# Aula 06 - Comando ls

- O comando ls exibe o conteúdo de um diretório no terminal.

ls

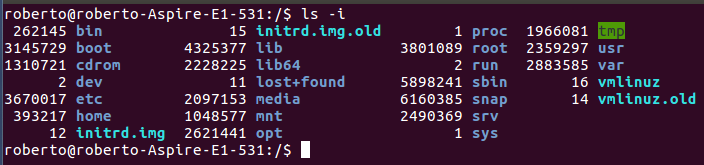


- Cada tipo de arquivo é exibido com uma cor diferente.

* em vermelho: arquivos compactados ou pacotes deb (debian)
* em roxo: arquivos de som
* em azul: diretórios
* em preto: arquivos comuns
* em azul claro: links
* em verde: arquivos executáveis (comandos de script do shell)

## Exibindo o inode dos arquivos

ls -i



inode é um número de identificação de cada arquivo.

## Listando arquivos com o caractere coringa (\*)

cd /sbin

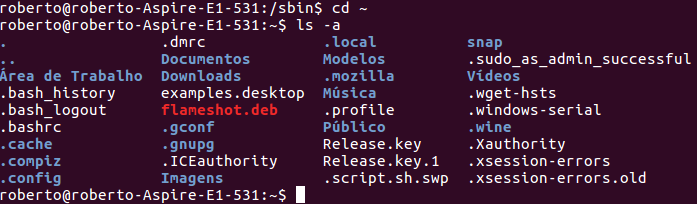
ls n\*



## Listando também os arquivos ocultos

cd ~

ls -a



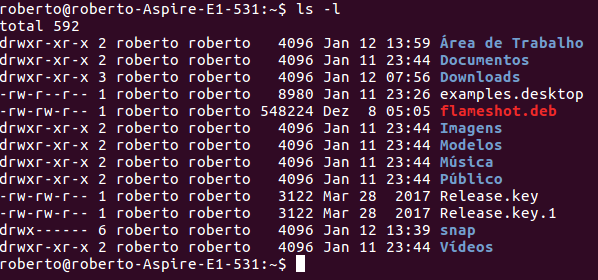
- No Linux arquivos ocultos começam com ponto.

- Em geral, são representados dessa maneira para que o usuário não altere esses arquivos (exemplo: arquivos de configuração ou de sistema).

## Lista longa

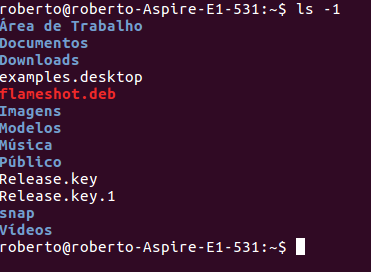
- Exibe as permissões de cada arquivo.

ls -l



## Listando os arquivos em uma única coluna

ls -1



# Aula 07 - Comando echo

- O comando echo exibe mensagens na tela do terminal.

echo "Olá mundo shell! :)"



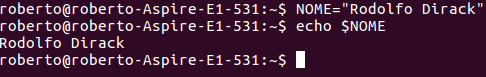
- Também pode ser utilizado para exibir variáveis.

echo $HOME



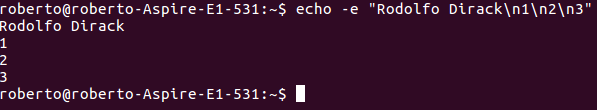
NOME="Rodolfo Dirack"

echo $NOME



## Interpretando caracteres especiais

echo -e "Rodolfo Dirack\n1\n2\n3"

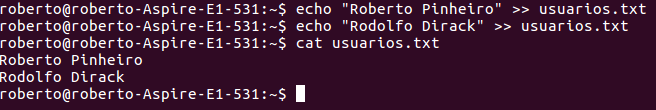


## Direcionando um texto para um arquivo

echo "Roberto Pinheiro" >> usuarios.txt

echo "Rodolfo Dirack" >> usuarios.txt

cat usuarios.txt



>>

Se o arquivo não existe ele será criado, se existe o texto será adicionado no final do arquivo.

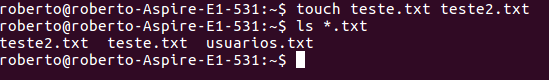
# Aula 08 - Comandos chmod e ls -l

- O comando chmod permite alterar permissões de arquivos no Linux.

- Na pasta do usuário, crie 2 arquivos de texto vazios:

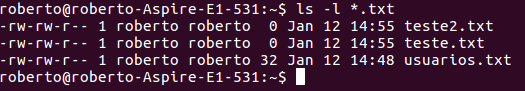
touch teste.txt teste2.txt

ls \*.txt



- O comando ls -l exibe a listagem longa (com as permissões) dos arquivos da pasta

ls -l \*.txt



## Permissões de arquivo

1. r: permissão de leitura
2. w: permissão de escrita
3. x: permissão de execução

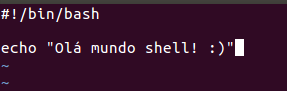
- Crie um shell script com o seguinte conteúdo:

#!/bin/bash

echo "Olá mundo shell! :)"

vi script.sh

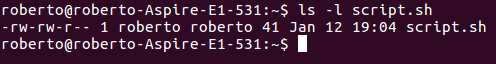
<Insert>



<Esc> :wq

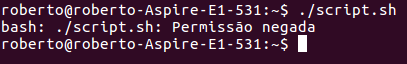
## Permissão de execução para o arquivo

ls -l script.sh



1. Primeiro grupo de permissões: para o usuário (usuário proprietário do arquivo).
2. Segundo grupo de permissões: para um grupo específico de usuários.
3. Terceiro grupo de permissões: para todos os demais usuários

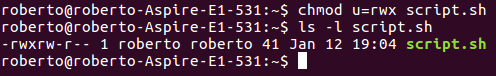
./script.sh



### Alterando as permissões do usuário proprietário do arquivo (u)

chmod u=rwx script.sh

ls -l script.sh



- Agora o usuário (proprietário do arquivo → betopinheiro) tem as três permissões (rwx) e portanto pode executar o shell script.

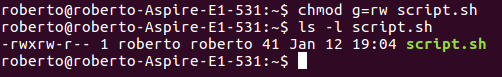
./script.sh



### Alterando as permissões do grupo (g)

chmod g=rw script.sh

ls -l script.sh

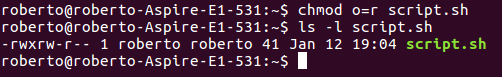


- Agora o grupo tem permissão apenas para leitura e escrita.

### Alterando as permissões para os demais usuários (o)

chmod o=r script.sh

ls -l script.sh



- Agora os demais usuários tem permissão apenas para leitura.

### Resumindo

1. O usuário proprietário do arquivo pode ler, escrever ou executar o script.
2. O grupo pode ler ou escrever o arquivo, mas não pode executá-lo.
3. Todos os demais só podem ler o arquivo.

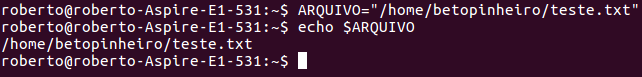
# Aula 09 - Comandos basename e dirname

- Esses comandos são muito úteis dentro de um script.

## Criando uma variável

ARQUIVO="/home/betopinheiro/teste.txt"

echo $ARQUIVO



## comando dirname

- O comando dirname exibe apenas o diretório do arquivo (ou seja: o caminho do arquivo).

dirname $ARQUIVO



## comando basename

- O comando basename exibe apenas o nome do arquivo.

basename $ARQUIVO



## Exibindo apenas o nome do arquivo, sem a extensão

basename $ARQUIVO .txt

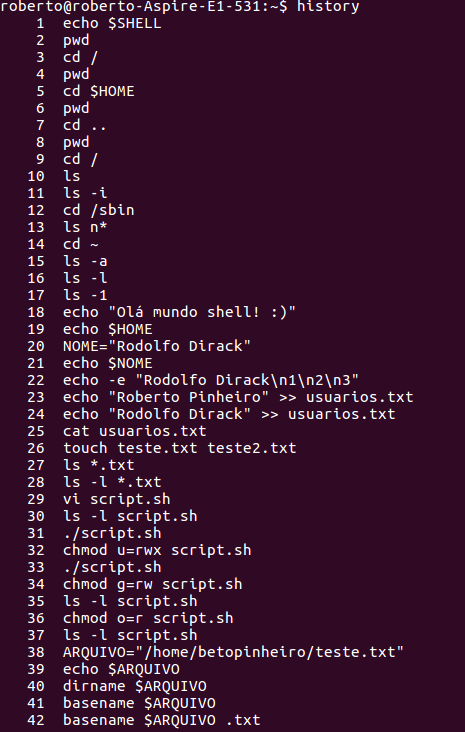


# Aula 10 - Comando history

- O comando history exibe o histórico de comandos.

## Exibindo o histórico de comandos

history



- O comando history armazena até 2000 comandos.

## Executando um comando existente no history

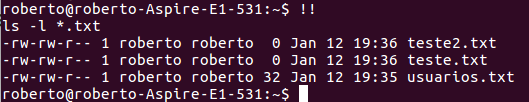
### Executando um comando específico do histórico

!28



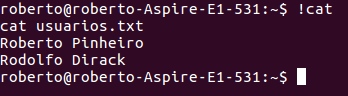
### Executando o último comando do histórico

!!



### Executando um último comando específico

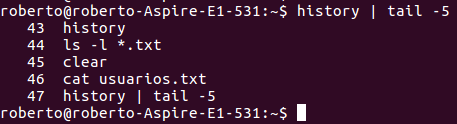
!cat



- Executa o último comando cat encontrado no histórico de comandos.

### Exibindo um determinado número de linhas de comandos do histórico

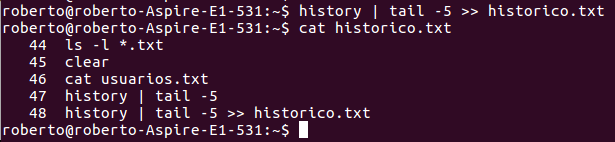
history | tail -5



## Direcionando a saída do comando para um arquivo

history | tail -5 >> historico.txt

cat historico.txt



## Limpando o histórico

history -c

# Aula 11 - Comandos mkdir e mktemp

- Estes comandos permitem criar diretórios e arquivos temporários.

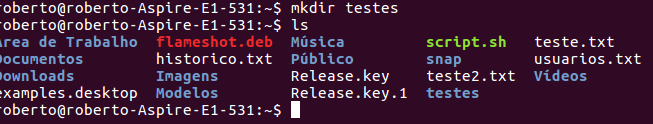
## Criando um diretório

### mkdir

O comando mkdir cria um diretório.

mkdir testes

ls



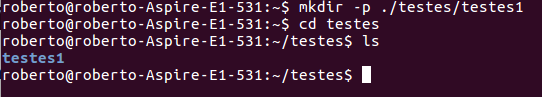
- Foi criado um diretório chamado "testes" vazio.

### Criando um diretório dentro de outro

mkdir -p ./testes/testes1

cd testes

ls



ou:

ls testes



## Criando um arquivo temporário

### mktemp

- O comando mktemp cria arquivos temporários

- Por padrão ele vai gerar os arquivos dentro de uma pasta chamada /tmp (pasta de arquivos temporários do shell)

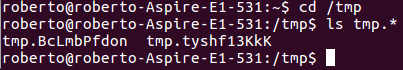
mktemp



- O comando criou um arquivo com um nome aleatório (tmp.BcLmbPfdon) dentro da pasta /tmp

cd /tmp

ls tmp.\*



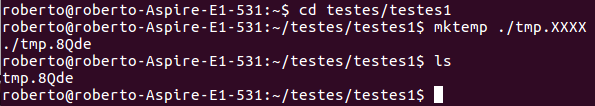
### Criando um arquivo temporário dentro da pasta atual

cd ~

cd testes/testes1

mktemp ./tmp.XXXX

ls



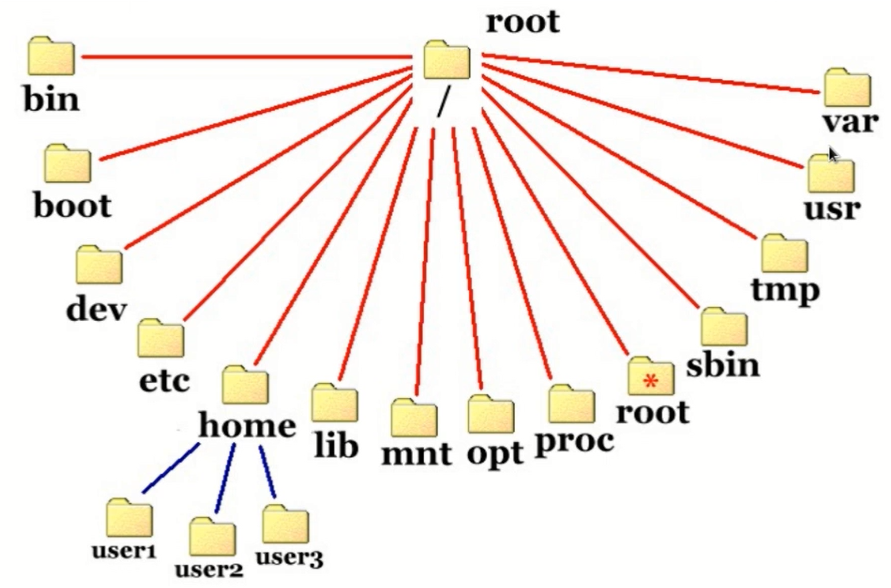
tmp.XXXX

XXXX é um placeholder, ou seja, o mktemp ao criar o arquivo temporário vai substituir XXXX por 4 caracteres aleatórios (para que os nomes dos arquivos não se repitam).

- Foi criado um arquivo temporário chamado tmp.8Qde dentro da pasta atual

# Aula 12 - Árvore de diretórios no Linux e revisão dos comandos

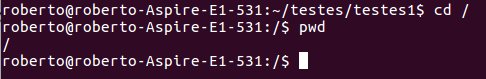
## Árvore de diretórios no Linux



- No diretório raiz se for digitado apenas o comando cd, o usuário é direcionado para a pasta home:

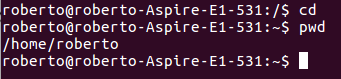
cd /

pwd



cd

pwd



# Aula 13 - Diretórios ponto e ponto ponto no Linux

ls -a

Exibe todos os arquivos do diretório, inclusive os ocultos (que iniciam com .)

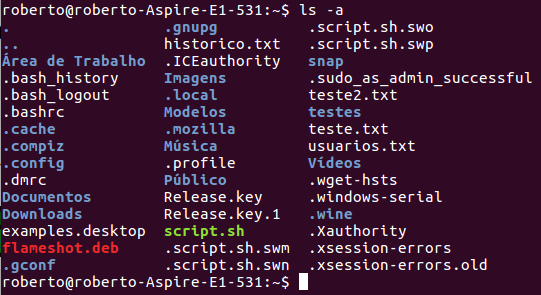


## Diretório atual

O diretório . é uma referência para o diretório atual.

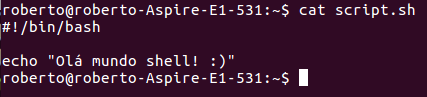
cd .

pwd



### Executando um shell script no diretório atual

cat script.sh



./script.sh



- Equivale a digitar o comando:

/home/roberto/script.sh



## Um diretório acima do atual

cd ..



.. se refere a um diretório acima

- Todas as pastas possuem os diretórios . e ..

# Aula 14 - Comando touch

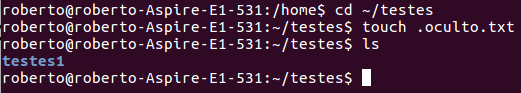
O comando touch permite criar arquivos em branco e arquivos ocultos.

## Criando um arquivo oculto

cd ~/testes

touch .oculto.txt

ls



- Com o comando ls, arquivos ocultos não são exibidos.

### Listando todos os arquivos, inclusive os ocultos

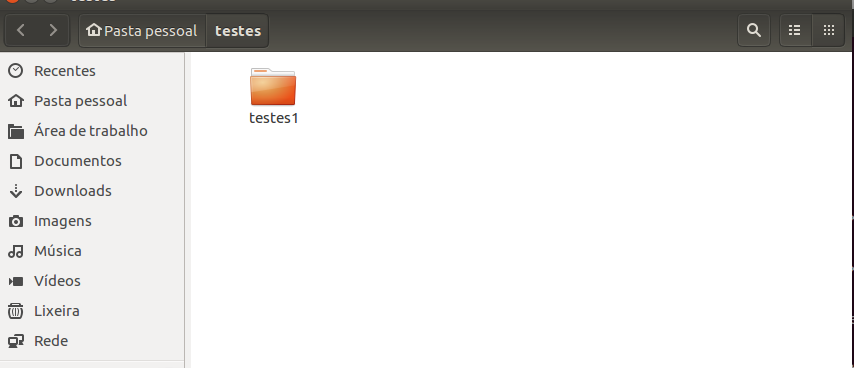
- Para exibir todos os arquivos, inclusive os ocultos, deve-se utilizar o comando:

ls -a

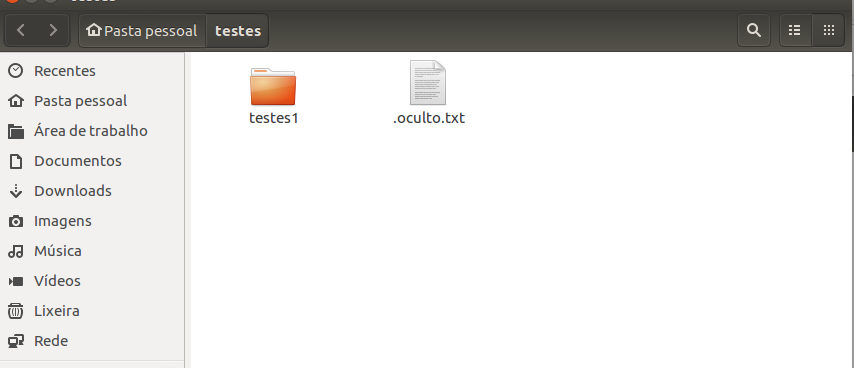


## Abrindo o navegador do Linux (Nautilus)

nautilus . &



- Pressione as teclas <Ctrl><h>



- O arquivo oculto passa a ser exibido.

- Se for pressionado novamente as teclas <Ctrl><h> o arquivo oculto não será exibido. E assim sucessivamente.

<Ctrl><h> habilita ou desabilita a exibição de arquivos ocultos.

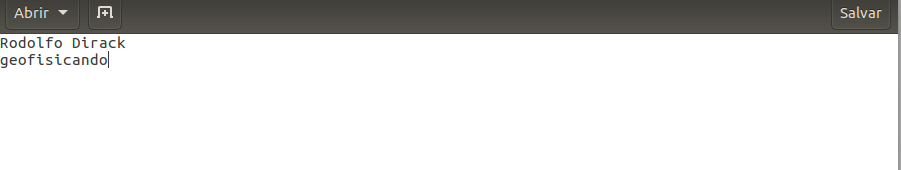
- Abra o arquivo no editor gedit:

gedit .oculto.txt &

& é usado para não vincular o terminal com o editor de texto

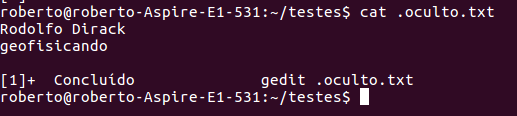
Rodolfo Dirack

geofisicando



- Salve o arquivo e saia do editor.

cat .oculto.txt



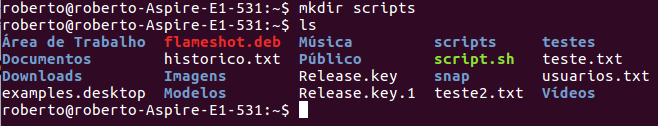
# Aula 15 - Comando mv

- O comando mv permite mover arquivos e diretórios pela linha de comando.

- Dentro da pasta de usuário, crie uma subpasta chamada scripts:

mkdir scripts

ls

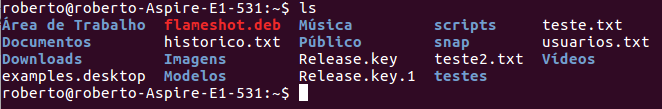


## Movendo arquivos

- Mova o arquivo script.sh para dentro dela, executando o comando:

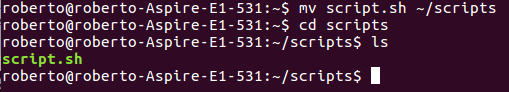
mv script.sh ~/scripts

ls



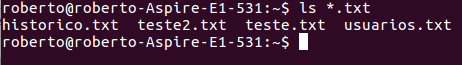
cd scripts

ls



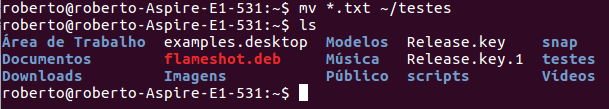
### Movendo arquivos com a extensão txt para a pasta testes

ls \*.txt



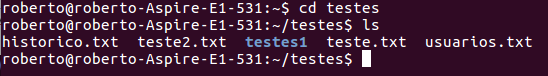
mv \*.txt ~/testes

ls



cd testes

ls

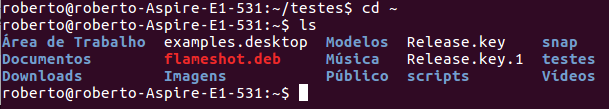


- Os arquivos historico.txt, usuarios.txt, teste.txt e teste2.txt foram movidos da pasta ~ para a pasta ~/testes

## Movendo pastas

cd ~

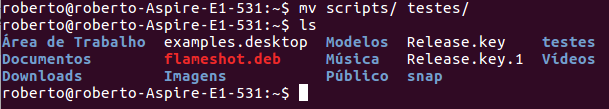
ls



- Mova a pasta scripts existente dentro da pasta de usuário para a pasta testes:

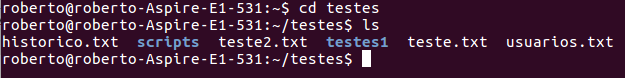
mv scripts/ testes/

ls



cd testes

ls



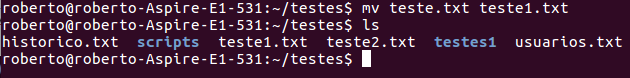
## Renomeando arquivo

- O comando mv também serve para renomear arquivos e/ou diretórios.

### Renomeando arquivo teste.txt para teste1.txt

mv teste.txt teste1.txt

ls

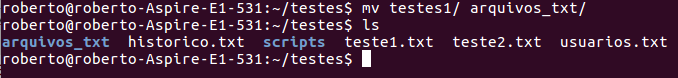


## Renomeando diretório

### Renomeando diretório testes1 para arquivos\_txt

mv testes1/ arquivos\_txt/

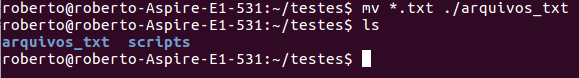
ls



- Mova os arquivos com extensão txt para a pasta arquivos\_txt:

mv \*.txt ./arquivos\_txt

ls



ls ./arquivos\_txt



- Agora os arquivos com a extensão txt estão dentro da pasta arquivos\_txt

# Aula 16 - Comando cp

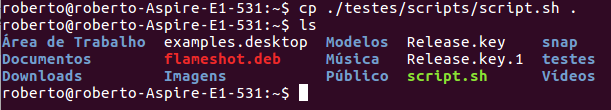
- O comando cp permite copiar arquivos e diretórios.

## Copiando arquivos

- A partir do diretório de usuário, copie o arquivo script.sh existente dentro da pasta ~/testes/scripts para a pasta atual (~):

cp ./testes/scripts/script.sh .

. representa o diretório atual



## Copiando diretórios

- Para copiar um diretório é necessário utilizar a opção -r:

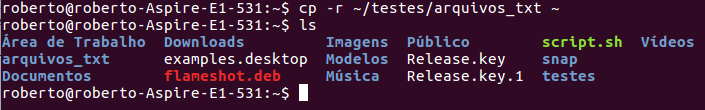
- Utilizando essa opção, é feita a cópia do diretório com todos os seus arquivos.

Exemplo:

- Copie o diretório arquivos\_txt existente dentro de ~/testes para a pasta de usuário:

cp -r ~/testes/arquivos\_txt ~

ls



# Aula 17 - Comando cat

- Dentro da pasta de testes (~/testes), crie uma pasta chamada testeCat:

mkdir testeCat; cd testeCat

O ponto-e-vírgula (;) permite executar mais de um comando simultaneamente no shell.

- Dentro dessa pasta, crie dois arquivos de texto: arq1.txt e arq2.txt:

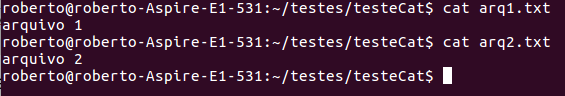
echo "arquivo 1" >> arq1.txt

echo "arquivo 2" >> arq2,txt

## Exibindo o conteúdo dos arquivos

cat arq1.txt

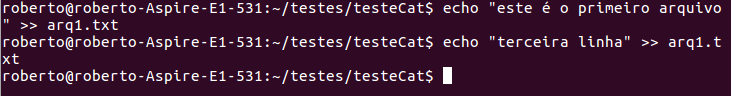
cat arq2.txt



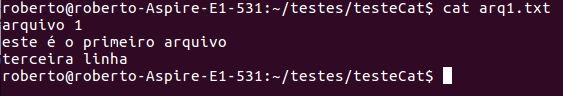
### Inserindo mais linhas nos arquivos

echo "este é o primeiro arquivo" >> arq1.txt

echo "terceira linha" >> arq1.txt

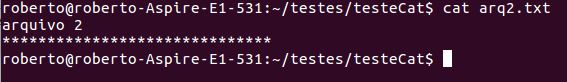


cat arq1.txt



echo "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" >> arq2.txt

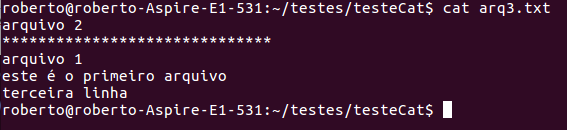
cat arq2.txt



## Concatenando arquivos em um novo arquivo

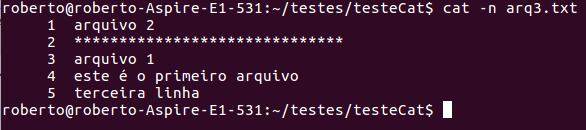
cat arq2.txt arq1.txt >> arq3.txt

cat arq3.txt



## Exibindo o número de linhas de um arquivo

cat -n arq3.txt



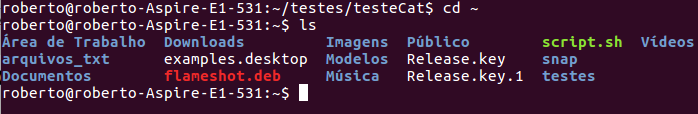
# Aula 18 - Comandos rm e rmdir

O comando rm remove arquivos.

O comando rmdir remove diretórios.

## Removendo arquivos

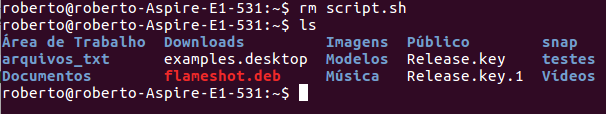
ls



- No diretório de usuário (~) remova o arquivo script.sh:

rm script.sh

ls



### Remoção de arquivos mediante confirmação

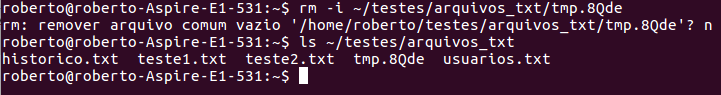
Para solicitar confirmação de remoção utiliza-se a opção -i

ls ~/testes/arquivos\_txt



rm -i ~/testes/arquivos\_txt/tmp.8Qde

ls



digitando n na confirmação, o arquivo não é removido.

rm -i ~/testes/arquivos\_txt/tmp.8Qde



ls



digitando y na confirmação, o arquivo é removido.

## Removendo diretórios

- O comando rmdir sem nenhuma opção remove um diretório apenas se ele estiver vazio.

rmdir arquivos\_txt



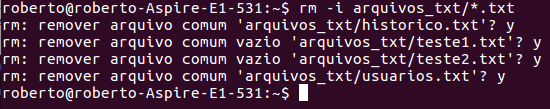
- Para remover um diretório contendo arquivos deve-se usar rmdir -r. Neste caso, será removido o diretório com todos os seus arquivos.

- Para solicitar confirmação de exclusão de cada um dos arquivos existentes dentro de um diretório utiliza-se rm -ir

### Removendo arquivos de um diretório com confirmação

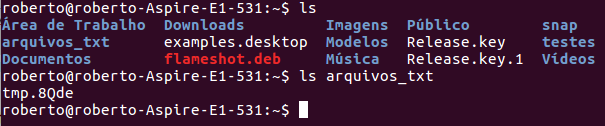
No diretório de usuário, execute o comando:

rm -i arquivos\_txt/\*.txt



ls

ls arquivos\_txt

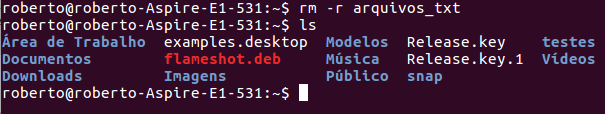


Observe que ainda existe um arquivo dentro da pasta.

### Removendo o diretório

rm -r arquivos\_txt

ls



- O diretório arquivos\_txt com o seu arquivo tmp.8Qde foram removidos.

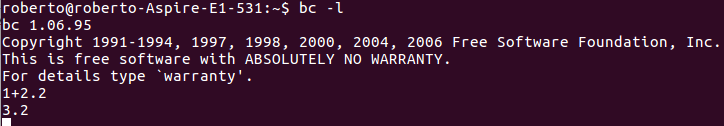
# Aula 19 - O que é um Pipe - Como redirecionar saídas dos comandos

O pipe (|) é utilizado para redirecionar a saída de um comando para outro comando.

## Utilizando a calculadora do Linux

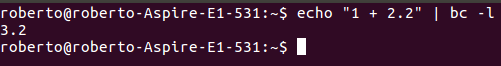
bc -l

a opção -l permite utilizar números decimais.



## Redirecionando o cálculo que se deseja fazer para a calculadora

echo "1 + 2.2" | bc -l



## Redirecionamento com >> e >

>> e > são utilizados para redirecionar a saída.

### Redirecionando com >>

- Se o arquivo não existe, ele será criado com o conteúdo informado.

- Se o arquivo já existe, será adicionada uma nova linha, com o conteúdo informado, no final do arquivo.

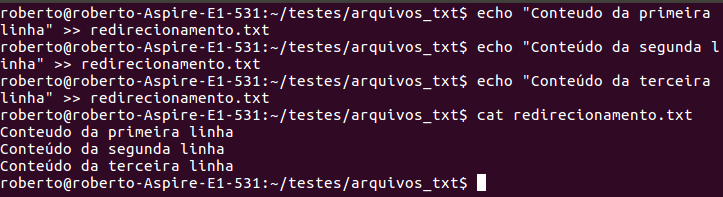
exemplo:

echo "Conteudo da primeira linha" >> redirecionamento.txt

echo "Conteúdo da segunda linha" >> redirecionamento.txt

echo "Conteúdo da terceira linha" >> redirecionamento.txt

cat redirecionamento.txt



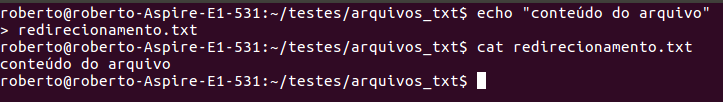
## Redirecionando com >

- Se o arquivo não existe, ele será criado com o conteúdo informado.

- Se o arquivo já existe, seu conteúdo será substituido pelo conteúdo informado.

echo "conteúdo do arquivo" > redirecionamento.txt

cat redirecionamento.txt

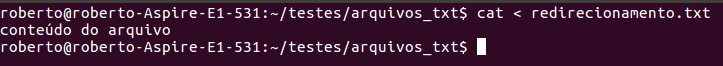


## Redirecionando com <

< é utilizado para redirecionar a entrada.

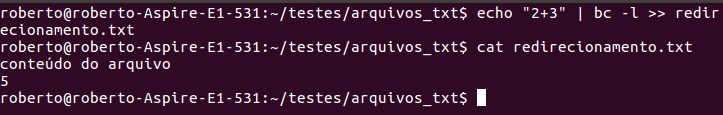
cat < redirecionamento.txt

Redireciona o conteúdo do arquivo redirecionamento.txt para o comando cat (para que seja exibido).



## Combinando redirecionamentos

echo "2+3" | bc -l >> redirecionamento.txt

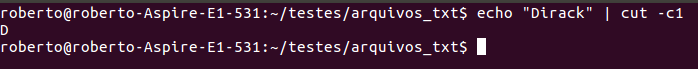


# Aula 20 - Comando cut

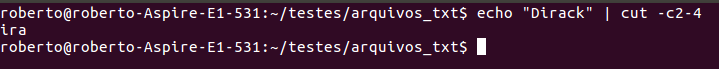
- O comando cut filtra saídas do comando cat e de arquivos csv

## Filtrando saídas do comando cat

echo "Dirack" | cut -c1



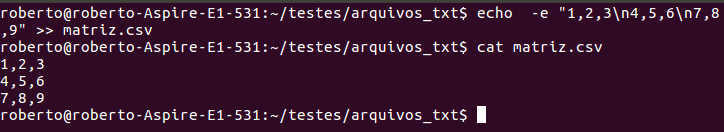
echo "Dirack" | cut -c2-4



## Selecionando colunas de um arquivo csv

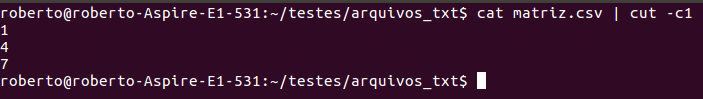
echo -e "1,2,3\n4,5,6\n7,8,9" >> matriz.csv

cat matriz.csv



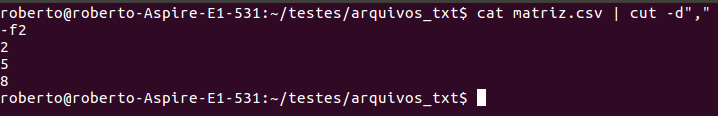
### Selecionando a primeira colunas do arquivo

cat matriz.csv | cut -c1



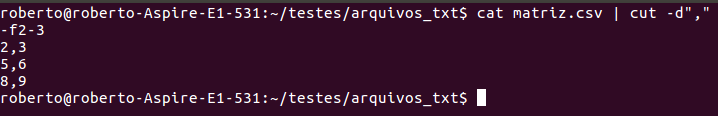
### Selecionando a segunda coluna do arquivo

cat matriz.csv | cut -d"," -f2



### Selecionando a segunda e terceira coluna do arquivo

cat matriz.csv | cut -d"," -f2-3



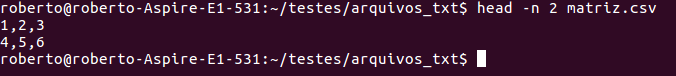
# Aula 21 - Comando head

O comando head exibe as primeiras linhas de um arquivo.

Exemplo:

- Para exibir as duas primeiras linhas do arquivo matriz.csv, execute o comando:

head -n 2 matriz.csv

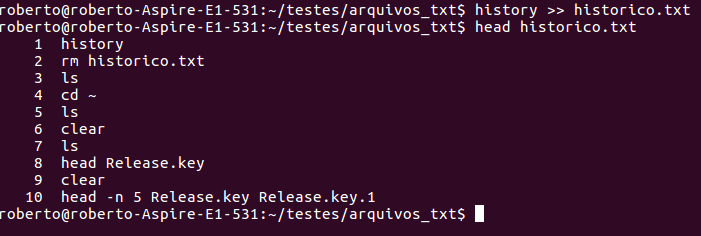


- Por padrão, o comando head sem a flag -n exibe as dez primeiras linhas de um arquivo.

Exemplo:

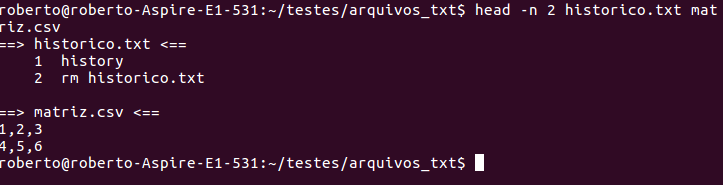
history >> historico.txt

head historico.txt



### Usando o comando head para exibir linhas de mais de um arquivo

head -n 2 historico.txt matriz.csv

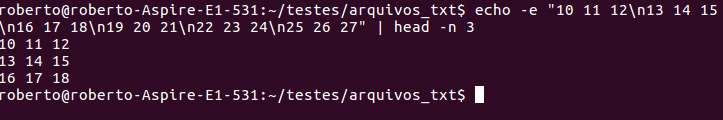


### Redirecionamento

O comando head aceita redirecionamento.

Exemplo:

echo -e "10 11 12\n13 14 15\n16 17 18\n19 20 21\n22 23 24\n25 26 27" | head -n 3



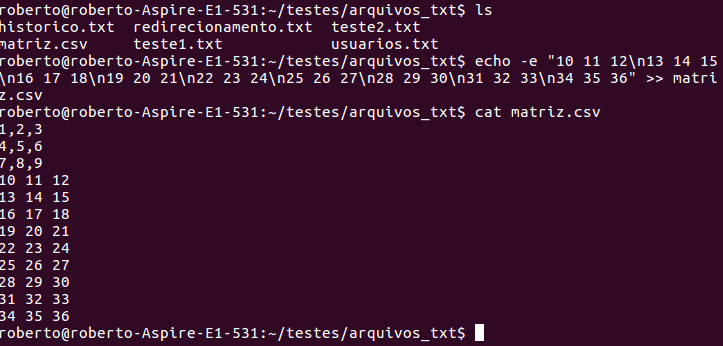
# Aula 22 - Comando tail

O comando tail exibe as últimas linhas de um arquivo.

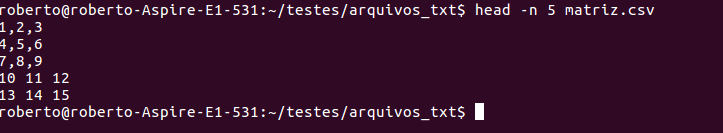
ls

echo -e "10 11 12\n13 14 15\n16 17 18\n19 20 21\n22 23 24\n25 26 27\n28 29 30\n31 32 33\n34 35 36" >> matriz.csv

cat matriz.csv

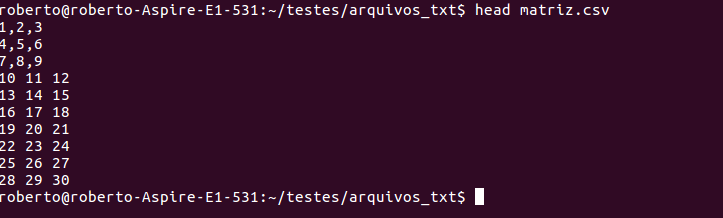


head -n 5 matriz.csv



- O comando tail sem a flag -n exibe as dez últimas linhas do arquivo.

head matriz.csv

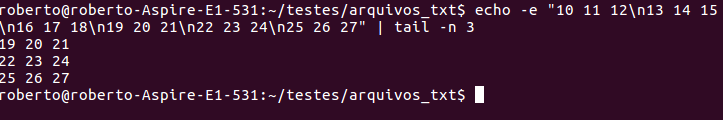


### Redirecionamento

O comando tail também aceita redirecionamento.

Exemplo:

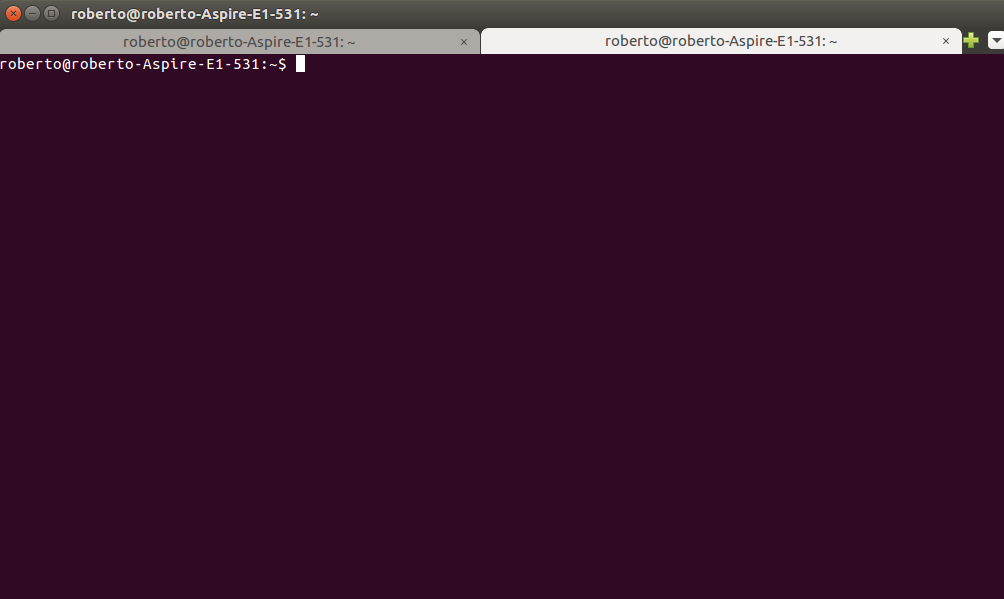
echo -e "10 11 12\n13 14 15\n16 17 18\n19 20 21\n22 23 24\n25 26 27" | tail -n 3



# Aula 23 - Comando tail -f

O comando tail -f permite monitorar arquivos.

- Abra duas abas do terminal, pressionando <Ctrl><Alt><t> para abrir a primeira tela do terminal e em seguida <Ctrl><Shift><t> para abrir uma nova aba do terminal.



- Para alternar entre as abas pressione <Alt><1> (para ir para a primeira aba) ou <Alt><2> (para ir para a segunda aba).

Na aba 1 (do lado esquerdo), vá para o diretório ~/testes/arquivos\_txt e execute o comando:

echo "Primeira linha" >> tail.txt; cat tail.txt



## Monitorando o arquivo tail.txt

- Na aba 2, vá para a pasta ~/testes/arquivos\_txt e execute o comando:

tail -f tail.txt



- Observe que será exibido o prompt piscando indicando que o arquivo tail.txt está sendo monitorado.

- Na aba 1, acrescente uma nova linha ao arquivo tail.txt, executando o comando:

echo "Segunda linha" >> tail.txt

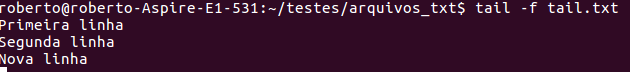
- Na aba 2, será exibido:



- Na aba 1, execute o comando:

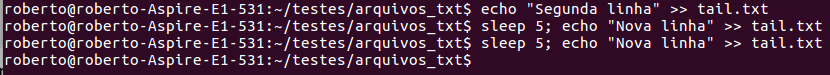
sleep 5; echo "Nova linha" >> tail.txt

- Voltando para a aba 2 é possível monitorar que 5 segundos a entrada do comando, a nova linha será acrescentada ao arquivo tail.txt.

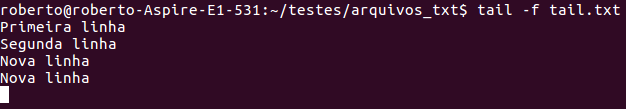


- Repita o mesmo comando na aba 1:

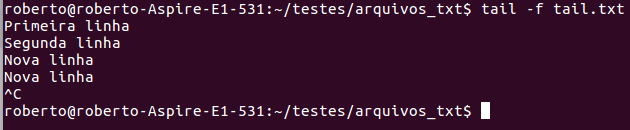
sleep 5; echo "Nova linha" >> tail.txt



- Voltando para a aba 2 é possível monitorar que 5 segundos a entrada do comando, a nova linha será acrescentada ao arquivo tail.txt.



- Na aba 2 para sair da monitoração do arquivo tail.txt pressione <Ctrl><c>



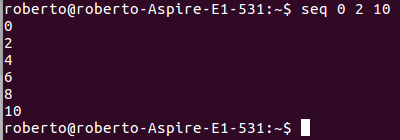
# Aula 24 - Comando seq

O comando seq exibe sequências de números e laços de repetição.

## Exibindo uma sequência de números

O comando a seguir, exibe uma sequência de números de 0 a 10, com passo 2:

seq 0 2 10



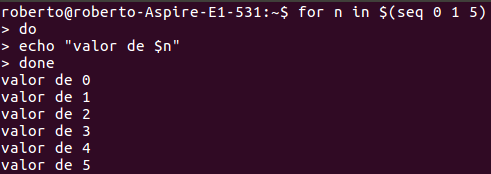
- Execute o comando:

for n in $(seq 0 1 5)

do

echo "valor de $n"

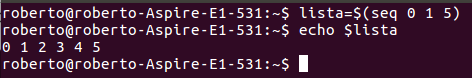
done



## Armazenando o resultado numa variável

lista=$(seq 0 1 5)

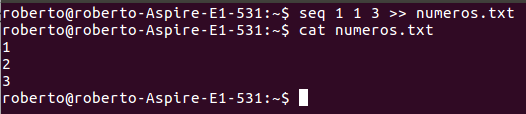
echo $lista



## Usando redirecionamento

seq 1 1 3 >> numeros.txt

cat numeros.txt



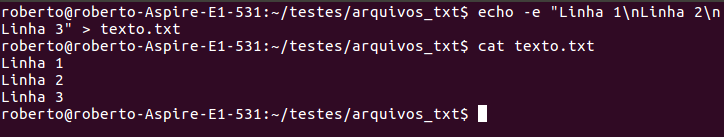
# Aula 25 - Comando wc

O comando wc exibe número de linhas, palavras e caracteres de arquivo.

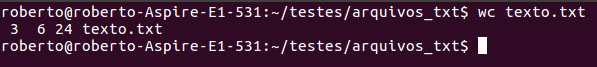
## Exibindo o número de linhas, de palavras e de caracteres de um arquivo

echo -e "Linha 1\nLinha 2\nLinha 3" > texto.txt

cat texto.txt



wc texto.txt

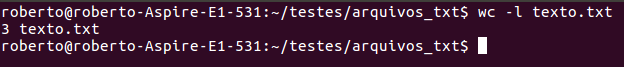


Ou seja, o arquivo texto.txt possui 3 linhas, 6 palavras e 24 caracteres.

## Exibindo as informações separadamente

### Número de linhas

wc -l texto.txt



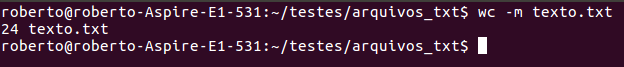
### Número de palavras

wc -w texto.txt



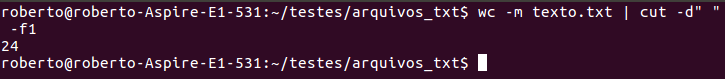
### Número de caracteres

wc -m texto.txt

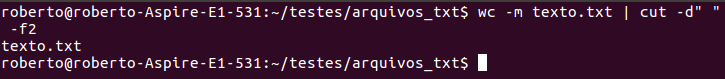


## Redirecionamento

wc -m texto.txt | cut -d" " -f1



wc -m texto.txt | cut -d" " -f2

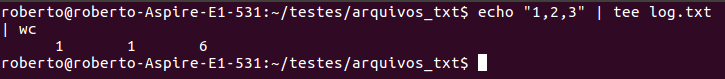


# Aula 26 - Comando tee

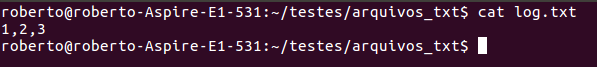
O comando tee gera arquivos de log das saídas de um pipe.

## Criando um arquivo de log

echo "1,2,3" | tee log.txt | wc

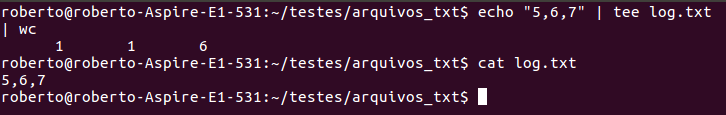


cat log.txt



echo "5,6,7" | tee log.txt | wc

cat log.txt



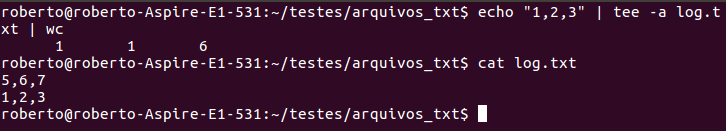
- Como o arquivo log.txt já existia ele foi sobrescrito.

## Adicionando linhas ao arquivo de log ao invés de sobrescrevê-lo

- Para isso usa-se a flag -a:

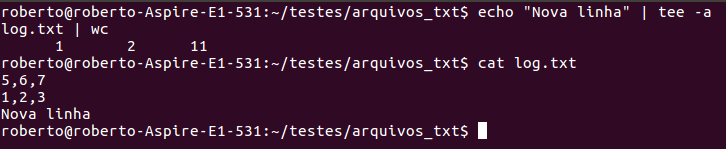
echo "1,2,3" | tee -a log.txt | wc

cat log.txt



echo "Nova linha" | tee -a log.txt | wc

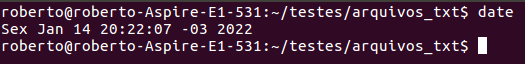
cat log.txt



# Aula 27 - Comando date (Parte 1)

- O comando date exibe data e hora local em formato específico.

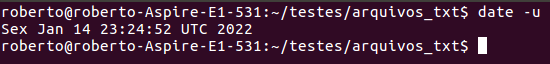
date



## Exibindo a zona de tempo UTC

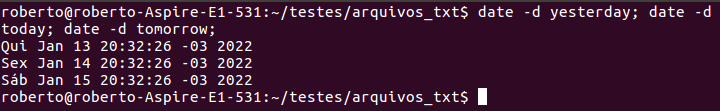
UTC é a zona de tempo padrão, sem correção.

date -u



## Exibindo a data de ontem, de hoje e de amanhã

date -d yesterday; date -d today; date -d tomorrow;



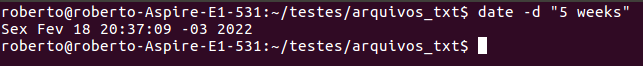
## Exibindo datas específicas

### Dois dias atrás

date -d "2 days ago"

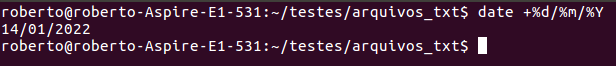
### Cinco semanas adiante

date -d "5 weeks"

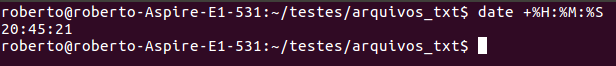


## Usando date com descritores de formato

date +%d/%m/%Y



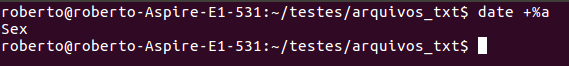
date +%H:%M:%S



## Exibindo o nome do dia

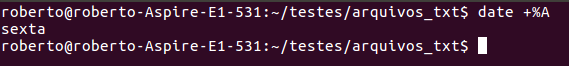
### Abreviado

date +%a



### Completo

date +%A



# Aula 28 - Comando date (Parte 2)

## Descritores de formato

### Representando o caractere literal %

"%" é um metacaractere. É um caractere especial para o comando date que indica que será passado um descritor de formato.

Para utilizar "%" na forma literal, usar "%%":

date +%d**%%**%m**%%**%Y



### Utilizando tabulação

Utiliza-se %t

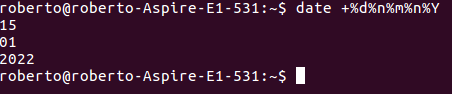
date +%d**%t**%m**%t**%Y



### Utilizando quebra de linha

Utiliza-se %n

date +%d**%n**%m**%n**%Y



## Exibindo o nome do mês

### Abreviado

date +%b



### Completo

date +%B



## Exibindo o número do dia do ano

date +%j



## Exibindo o número de seguindos desde 01 de janeiro de 1970

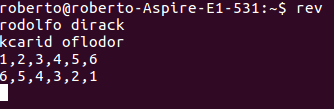
date +%s



# Aula 29 - Comando rev

- O comando rev reverte sequências de caracteres

rev



- Para sair: <Ctrl><c>

## Obtendo o conteúdo do último campo

Os campos são: nome, idade, email, senha

echo "rodolfo,27,rodolfo@gmail.com,123456" > users.txt

cat users.txt



### Obtendo o valor do campo senha

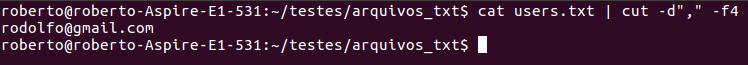
cat users.txt | cut -d"," -f4



- O campo senha é sempre o último. Mas se for inserido um novo campo neste arquivo, o comando anterior não mais exibirá a senha e sim o email.

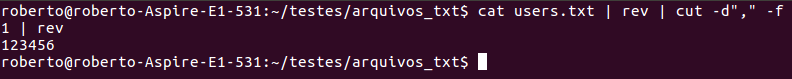
echo "rodolfo,dirack,27,rodolfo@gmail.com,123456" > users.txt

cat users.txt | cut -d"," -f4



- Para resolver esse problema podemos utilizar o comando "rev"

cat users.txt | rev | cut -d"," -f1 | rev



- Dessa forma mesmo se colunas forem acrescentadas ou removidas, a senha será exibida.

# Aula 30 - Comando paste

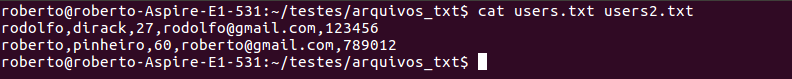
- O comando paste concatena arquivos por colunas.

## Concatenando arquivos por linhas

- O comando cat concatena arquivos por linhas.

echo "roberto,pinheiro,60,roberto@gmail.com,789012" > users2.txt

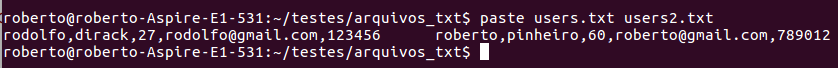
cat users.txt users2.txt



## Concatenando arquivos por colunas

- O comando paste concatena arquivos por colunas.

paste users.txt users2.txt



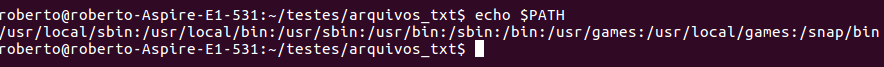
### Alterando o caractere de separação de colunas

paste users.txt users2.txt -d:



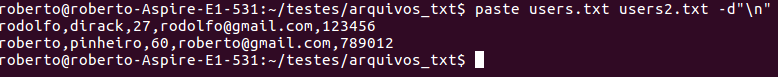
- Na variável de ambiente $PATH, a separação é feita dessa forma:

echo $PATH



- O delimitador pode ser "\n". Nesse caso o comando paste se comportará da mesma forma que o comando cat, ou seja, exibirá a concatenação por linhas.

paste users.txt users2.txt -d"\n"



## Armazenando a concatenação em um novo arquivo

paste users.txt users2.txt -d"\n" > cadastro.txt

ls

cat cadastro.txt

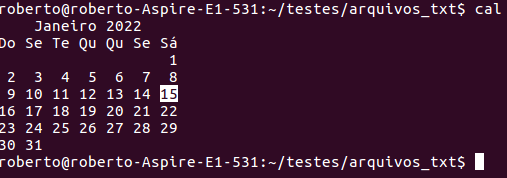


# Aula 31 - Comando cal

- O comando cal exibe o calendário na tela do terminal.

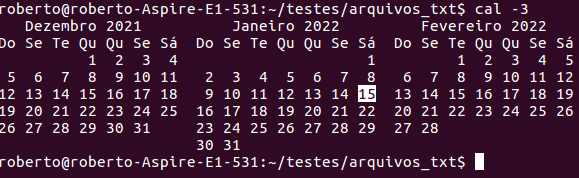
## Exibindo o calendário do mês atual

cal



## Exibindo o calendário do mês atual, o mês anterior e o mês próximo

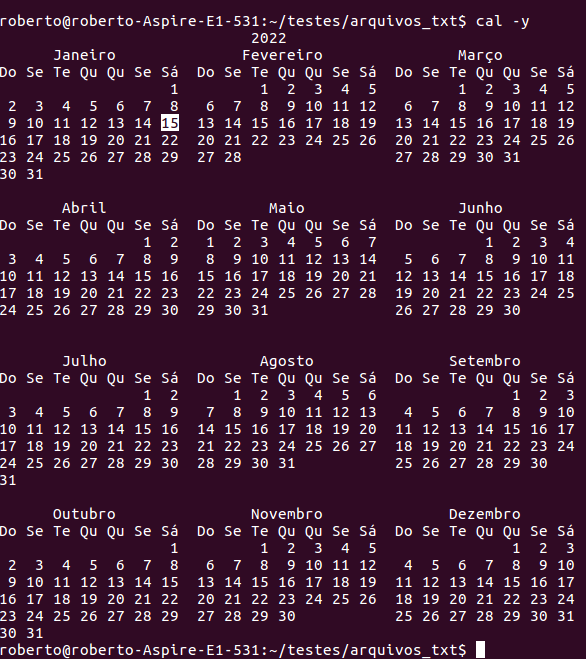
cal -3



- Exibe o mês atual, um mês antes e um mês depois.

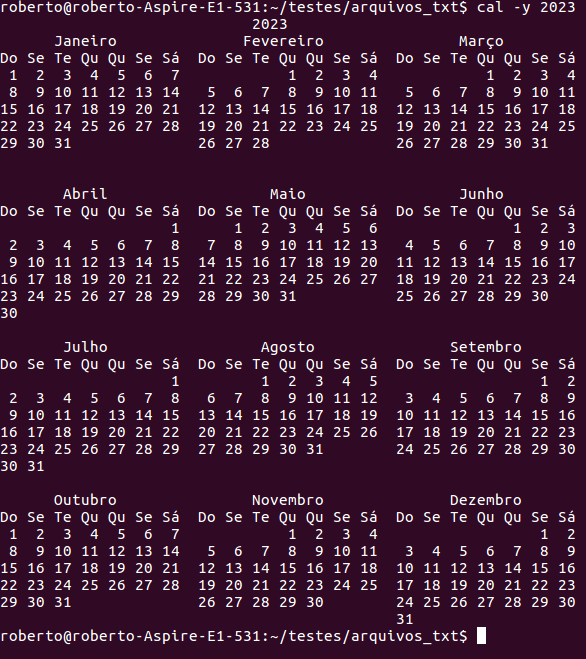
## Exibindo o calendário completo do ano atual

cal -y



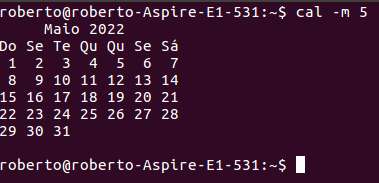
## Exibindo o calendário de um ano específico

cal -y 2023



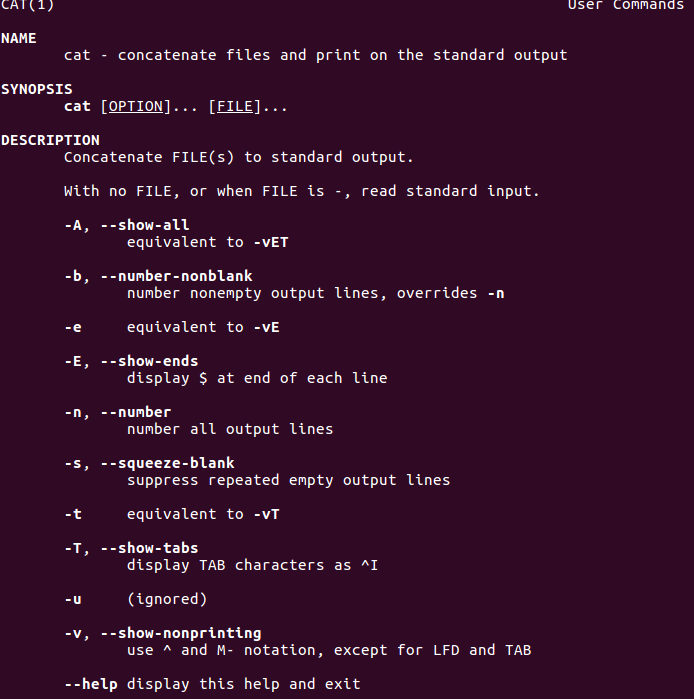
## Exibindo um mês do ano atual

cal -m 5



## Consultando as opções do comando cal

man cal





- Pressione <q> para sair do manual;

# Aula 32 - Comando sort (Parte 1) - Organizar lista em ordem alfabética e numérica

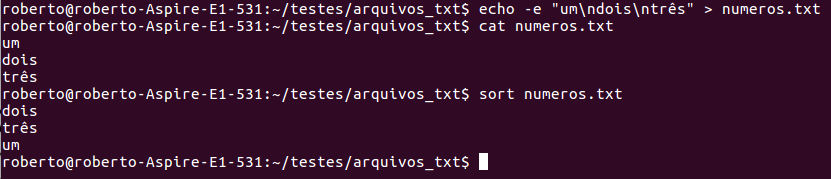
- O comando sort organiza lista em ordem alfabética e numérica.

## Organizando em ordem alfabética

echo -e "um\ndois\ntrês" > numeros.txt

cat numeros.txt

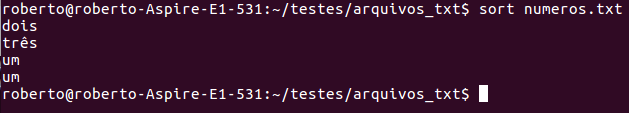
sort numeros.txt



## Exibindo ocorrências únicas

echo "um" >> numeros.txt

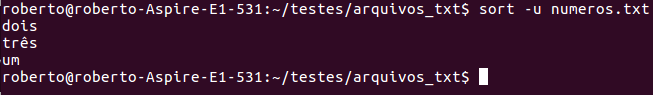
sort numeros.txt



"um" é exibido duas vezes.

- Para que não sejam exibidas opções duplicatas, utiliza-se a flag -u:

sort -u numeros.txt



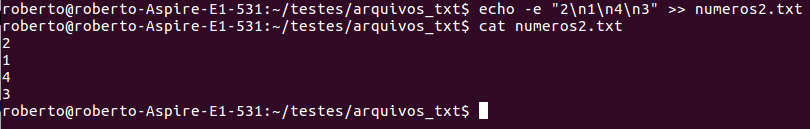
# Aula 33 - Comando sort (Parte 2) - Organizar lista por coluna e na ordem inversa

## Redirecionando a saída de um comando sort para um novo arquivo

O redirecionamento pode ser feito com a flag -o:

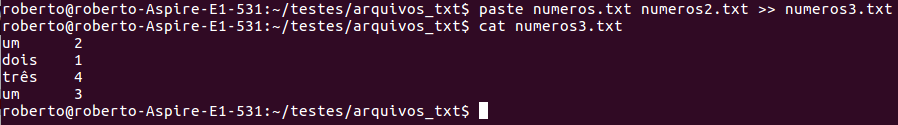
rm numeros2.txt

echo -e "2\n1\n4\n3" >> numeros2.txt



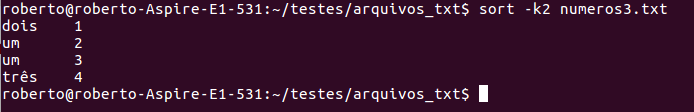
paste numeros.txt numeros2.txt >> numeros3.txt

cat numeros3.txt



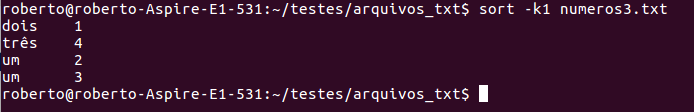
## Ordenando pela coluna 2

sort -k2 numeros3.txt



## Ordenando pela coluna 1

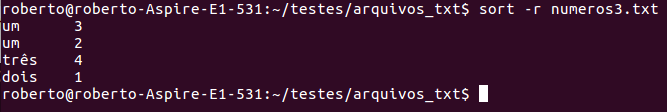
sort -k1 numeros3.txt



## Ordenando em ordem decrescente

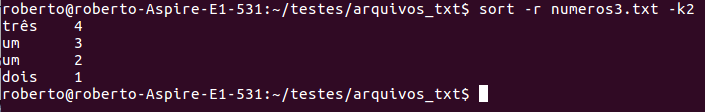
Por padrão, a ordenação é feita pela primeira coluna:

sort -r numeros3.txt



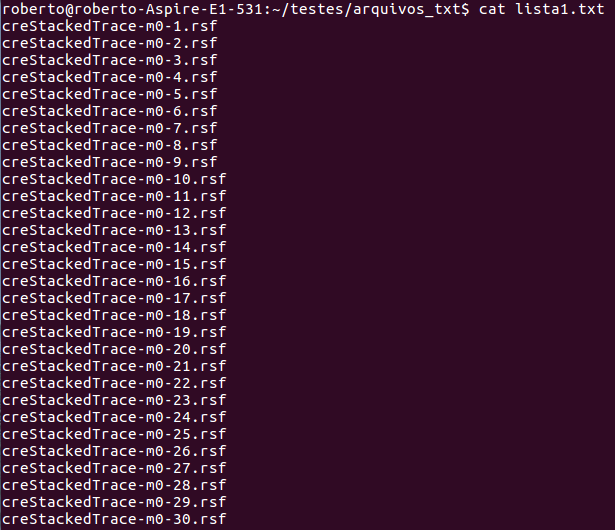
## Ordenando em ordem decrescente pela segunda coluna

sort -r numeros3.txt -k2

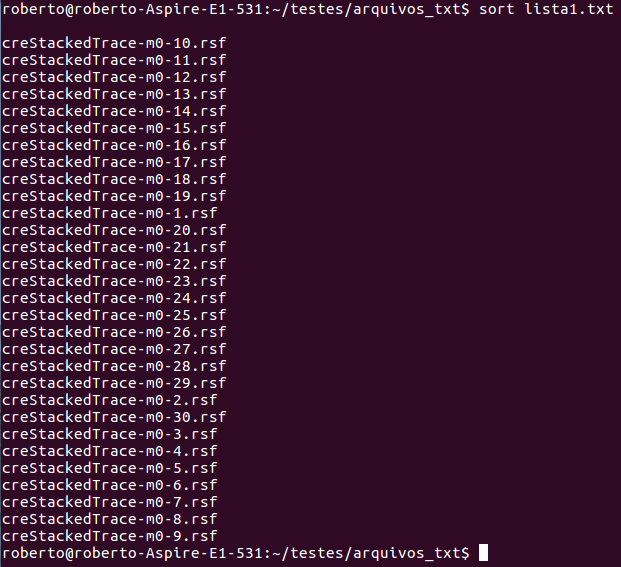


# Aula 34 - Comando sort (Parte 3) - Organizar lista grande de arquivos por coluna

cat lista1.txt

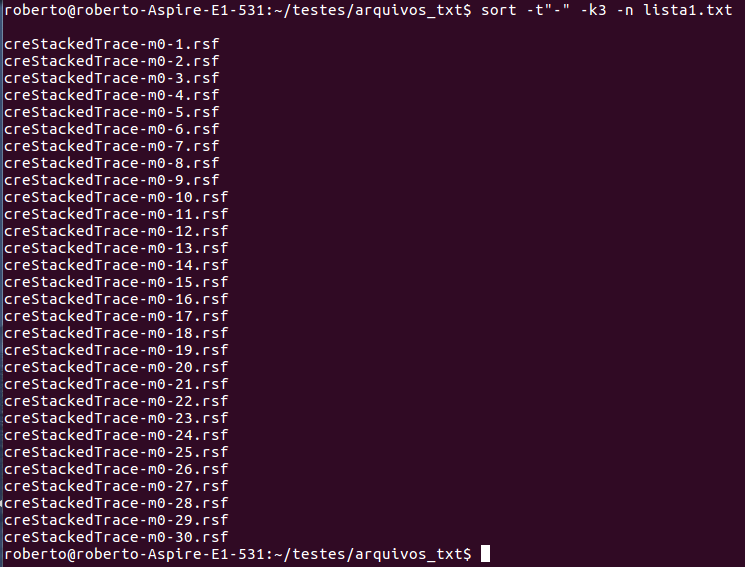


sort lista1.txt



- A ordenação não é feita como o desejado. Para isso execute o comando:

sort -t"-" -k3 -n lista1.txt



- A flag -t indica qual será o separador

- A flag -k indica qual é a coluna

- A flag -n indica ordenação numérica

# Aula 35 - Comando xargs

- O comando xargs executa comandos, baseado em uma lista de argumentos, em loop sem usar laços.

## Separando os argumentos

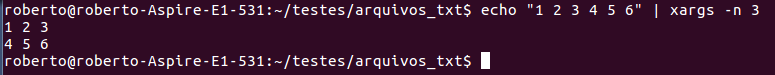
### De dois em dois

echo "1 2 3 4 5 6" | xargs -n 2



### De três em três

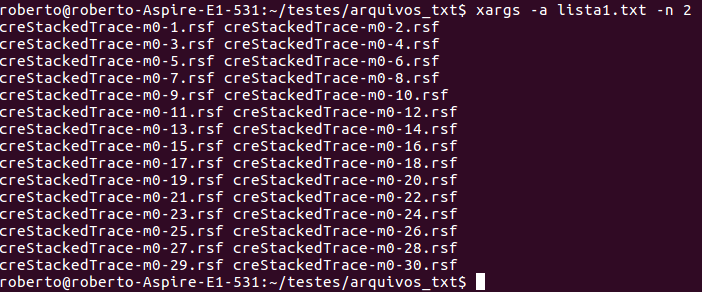
echo "1 2 3 4 5 6" | xargs -n 3



## Lendo a partir de um arquivo

- Isso pode ser feito utilizando a flag -a

xargs -a lista1.txt -n 2

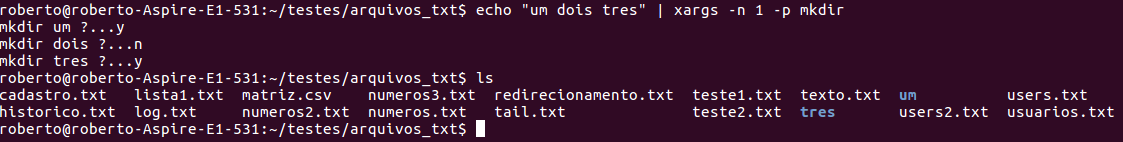


## Executando uma sequência de comandos em loop

### Executando uma sequência de comandos com confirmação

echo "um dois tres" | xargs -n 1 -p mkdir

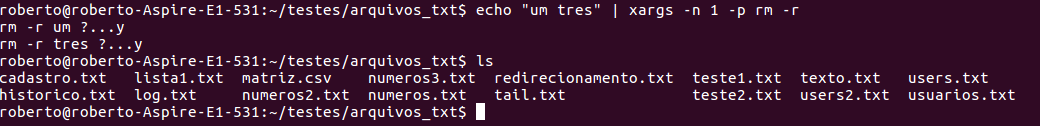
ls



- Foram criados os diretórios "um" e "tres"

### Removendo os diretórios criados

echo "um tres" | xargs -n 1 -p rm -r

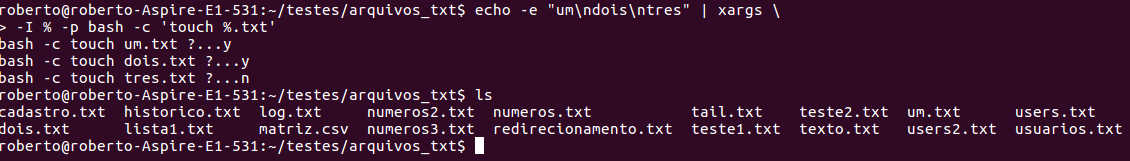


### Forma mais complexa de criar arquivos de texto em loop

echo -e "um\ndois\ntres" | xargs \

> -I % -p bash -c 'touch %.txt'

ls



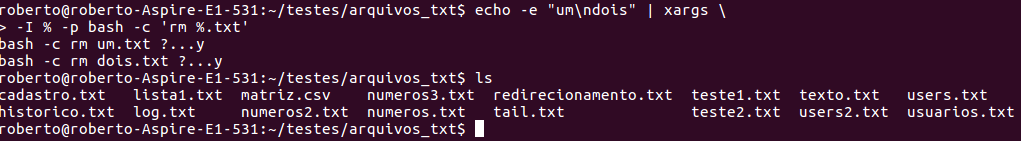
- Foram criados os arquivos: um.txt e dois.txt.

### Forma mais complexa de remover arquivos de texto em loop

echo -e "um\ndois" | xargs \

> -I % -p bash -c 'rm %.txt'

ls



# Aula 36 - Comando sed (Parte 1) - Editor de fluxo de texto do terminal

- O comando sed é um editor de texto do terminal. Ele edita os textos a partir de expressões regulares (regex).

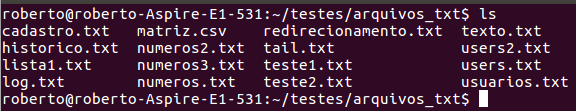
## Exemplo de expressão regular

ls \*[0-9].txt

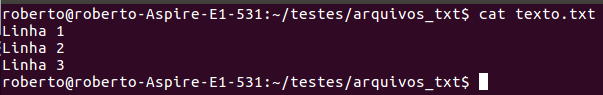


## Editando um arquivo de texto

ls



cat texto.txt



- Edite o arquivo texto.txt com o editor gedit:

gedit texto.txt

- Apague o texto existente e digite o seguinte texto:

Testo qualque

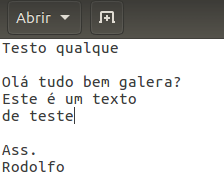
Olá tudo bem galera?

Este é um texto

de teste

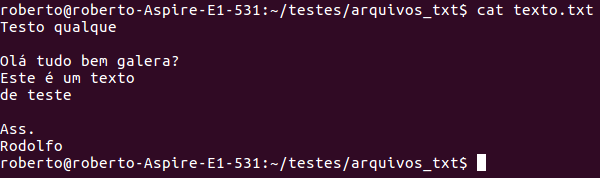
Ass.

Rodolfo



Clique no botão Salvar e saia do editor pressionando <Ctrl><q>.

cat texto.txt



## Selecionando linhas do arquivo

### Primeira linha do arquivo de texto

sed -n '1p' texto.txt



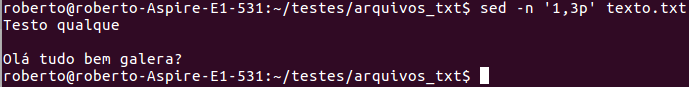
### Buscando uma linha através de um palavra

sed -n '/galera/p' texto.txt



### Imprimindo um intervalo de linhas

sed -n '1,3p' texto.txt



## Caracteres especiais do comando sed

^ representa o início da linha

$ representa o final da linha

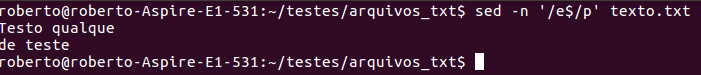
### Exibindo linha que termina com ?

sed -n '/?$/p' texto.txt



### Exibindo linhas que terminam com a letra e

sed -n '/e$/p' texto.txt



### Exibindo todas as linhas que começam com a letra O

sed -n '/^O/p' texto.txt



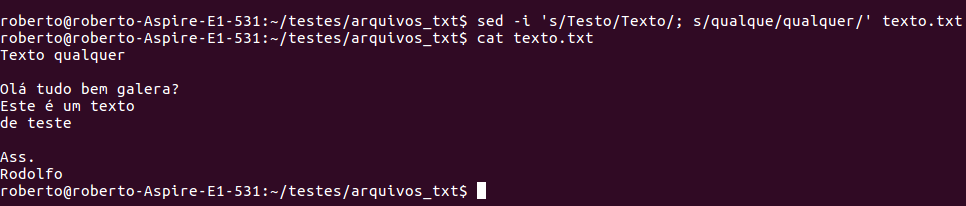
# Aula 37 - Comando sed (Parte 2) - Substituir uma string por outra

- Por padrão o comando sed não registra uma substituição no arquivo (apenas exibe). Para se fazer uma substituição que fique registrada no arquivo, utiliza-se a flag -i

## Corrigindo a primeira linha de texto

sed -i 's/Testo/Texto/; s/qualque/qualquer/' texto.txt

cat texto.txt

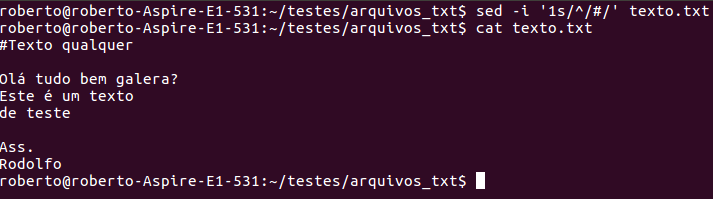


## Comentando uma parte do código

No Linux, # é o caractere que indica comentário

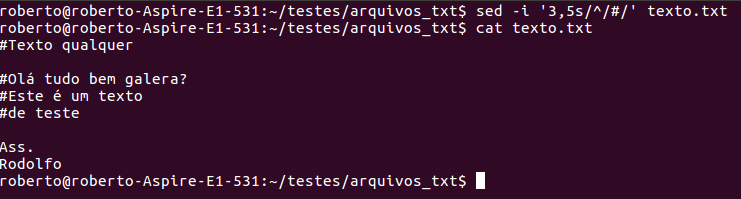
- Para comentar a primeira linha do arquivo:

sed -i '1s/^/#/' texto.txt



- Para comentar, também, as linhas 3 a 5 do arquivo:

sed -i '3,5s/^/#/' texto.txt



# Aula 38 - Comando sed (Parte 3) - Deletar linhas e strings de arquivo de texto

Faça as seguintes alterações no arquivo texto.txt:

#Texto qualquerLIXO

Olá tudo bem galera?LIXO

Esse é

BUGapenas BUGum BUGteste

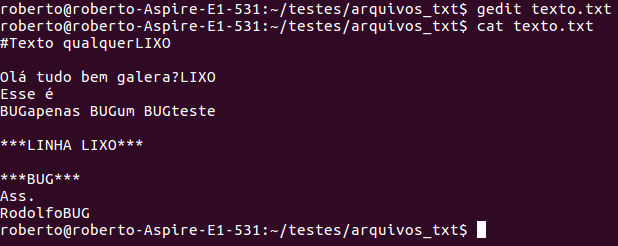
\*\*\*LINHA LIXO\*\*\*

\*\*\*BUG\*\*\*

Ass.

RodolfoBUG

cat texto.txt

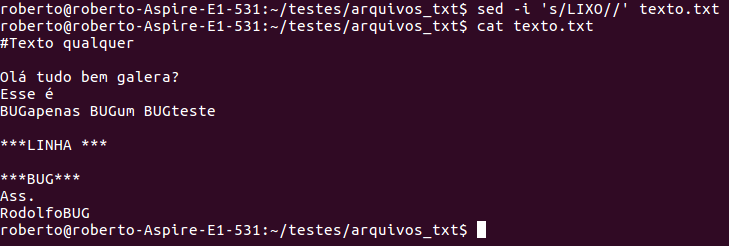


## Corrigindo o texto

### Apagando as ocorrências da palavra LIXO

sed -i 's/LIXO//' texto.txt

cat texto.txt

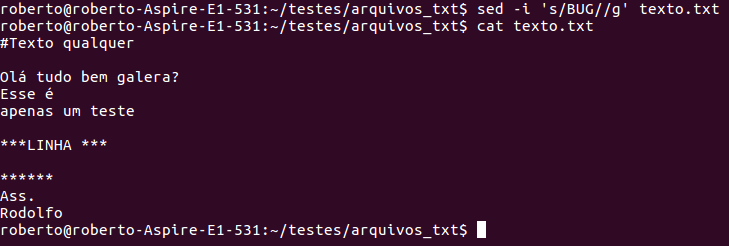


### Apagando as ocorrências da palavra BUG

- Por padrão o comando sed substitui apenas a primeira ocorrência, depois ele passa para a próxima linha. Para substituir todas as ocorrências da palavra em todas as linhas, utiliza-se o modificador g.

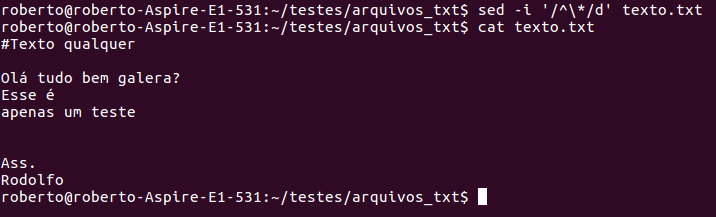
sed -i 's/BUG//g' texto.txt

cat texto.txt



### Deletando as linhas que iniciam com asterisco

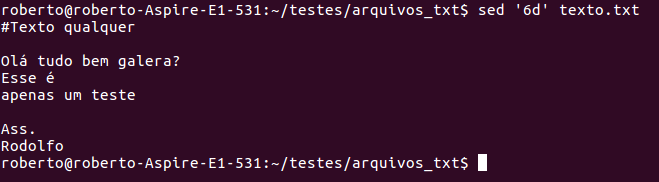
sed -i '/^\\*/d' texto.txt



### Deletando a linha 6 (linha em branco)

- Visualizando antes de deletar:

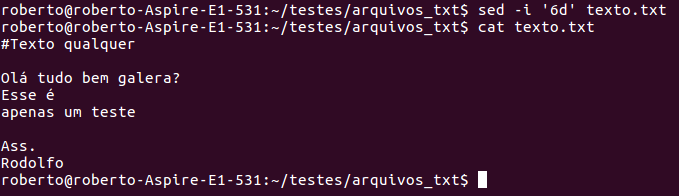
sed '6d' texto.txt



- Deletando:

sed -i '6d' texto.txt

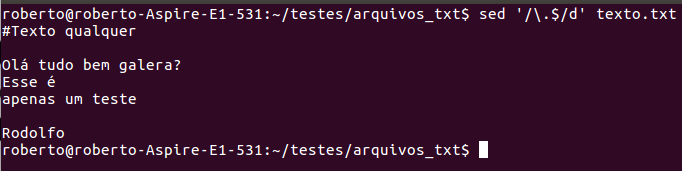
cat texto.txt



### Deletando a linha que termina com ponto

- Verificando:

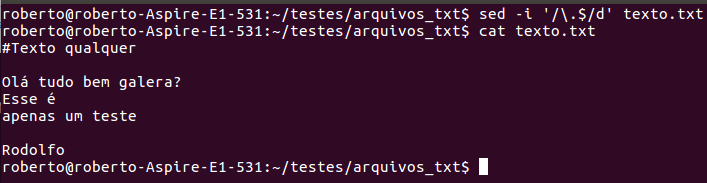
sed '/\.$/d' texto.txt



- Deletando:

sed -i '/\.$/d' texto.txt

cat texto.txt



# Aula 39 - Comando grep (Parte 1) - Pesquisar strings em arquivo de texto

grep - Globally search a regular expression and print

- Altere o arquivo texto.txt para:

#Texto qualquer

Olá tudo bem galera?

Esse é

BUGapenas BUGum BUGteste

Rodolfo

cat texto.txt



## Fazendo a busca da palavra a ser deletada

grep -n BUG texto.txt

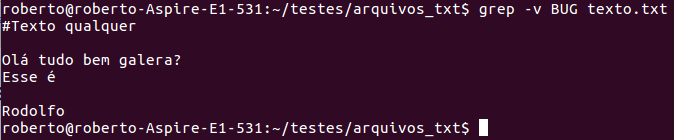
A flag -n exibe o número da linha



### Exibindo todas as linhas que não contém a palavra BUG

- Com a flag -v é exibido as linhas que não contém a palavra.

grep -v BUG texto.txt



### Exibindo o número de linhas que contém a palavra BUG

- Com a flag -c é exibido o número de linhas que contém a palavra.

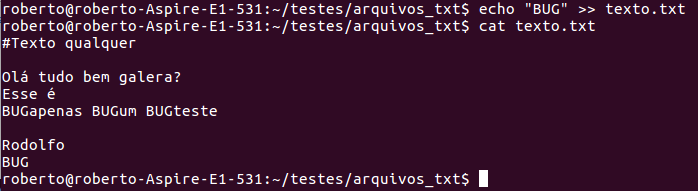
grep -c BUG texto.txt



### Acrescentando uma nova linha com a palavra BUG no final do arquivo

echo "BUG" >> texto.txt

cat texto.txt

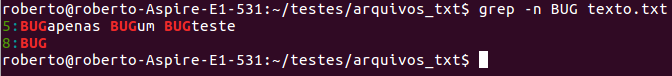


grep -v BUG texto.txt



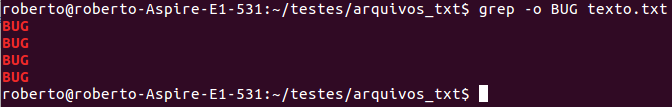
- Agora são duas linhas que contém a palavra "BUG"

grep -n BUG texto.txt



### Exibindo todas as ocorrências de uma palavra

grep -o BUG texto.txt



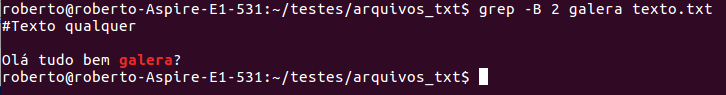
# Aula 40 - Comando grep (Parte 2) - Exibir o contexto de pesquisa

## Exibindo o contexto de uma pesquisa

### Exibindo a linha com a ocorrência da busca e n linhas antes

- A flag -B (before) exibe o número de linhas que também deve ser exibido antes da linha com a ocorrência.

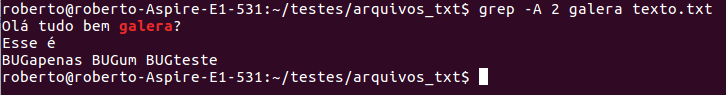
grep -B 2 galera texto.txt



### Exibindo a linha com a ocorrência da busca e n linhas depois

- A flag -A (after) exibe o número de linhas que também deve ser exibido antes da linha com a ocorrência.

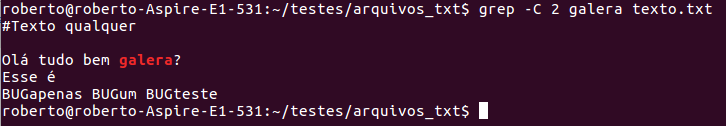
grep -A 2 galera texto.txt



### Exibindo a linha com a ocorrência da busca e n linhas antes e depois

- A flag -C (context) exibe o número de linhas que também deve ser exibido antes e depois da linha com a ocorrência.

grep -C 2 galera texto.txt

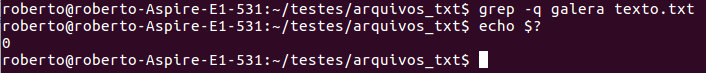


## Retornando o resultado de uma busca

- A flag -q retorna 0 se a busca encontrar alguma ocorrência, ou retorna 1 se não encontrar nenhuma ocorrência. O resultado fica armazenado na variável $?

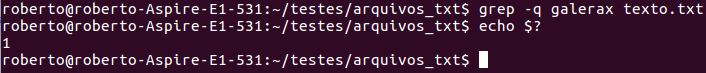
grep -q galera texto.txt

echo $?



grep -q galerax texto.txt

echo $?

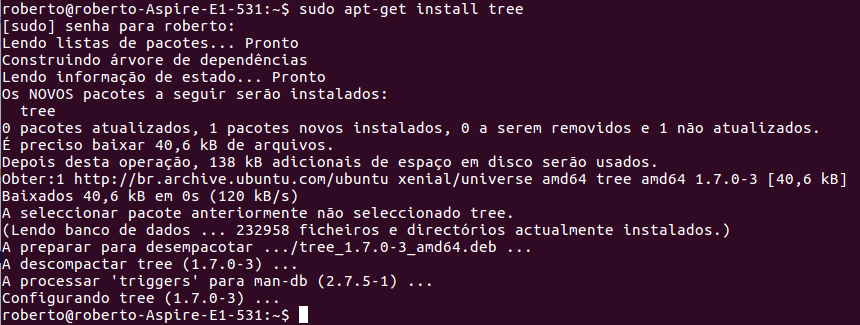


# Aula 41 - Comando tree

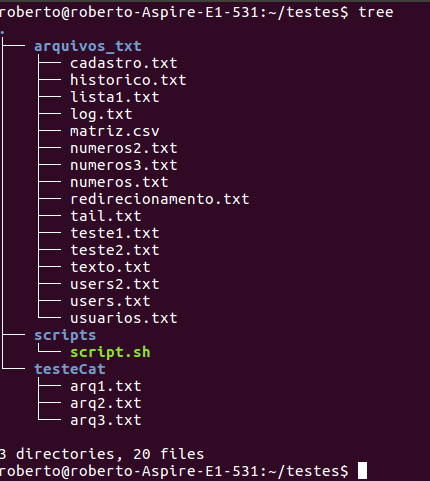
- O comando tree exibe uma lista em formato de árvore de diretórios a partir do diretório atual.

## Instalando o tree no sistema

sudo apt-get install tree

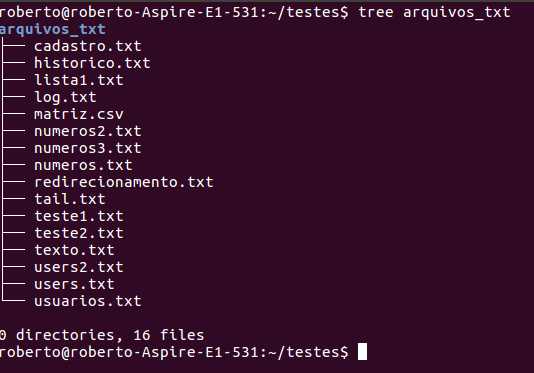


tree



## Exibindo um diretório específico

tree arquivos\_txt



## Criando pastas dentro de um diretório de testes

mkdir -p testeTree/teste1/teste2/teste3

ls

cd testeTree

tree



## Inserindo arquivos dentro dos diretórios criados

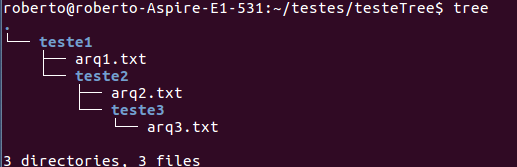
touch teste1/arq1.txt

touch teste1/teste2/arq2.txt

touch teste1/teste2/teste3/arq3.txt

## Exibindo a árvore de diretórios

tree



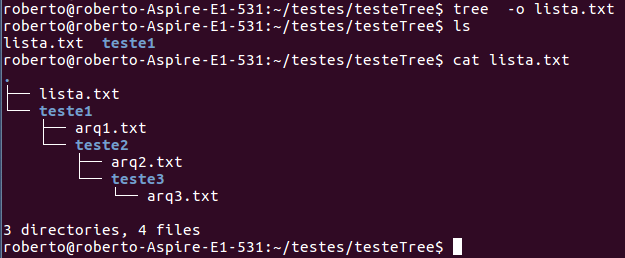
## Armazenando o resultado dentro de um arquivo de texto

Para isso utiliza-se a flag -o

tree -o lista.txt

ls

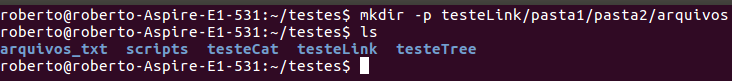
cat lista.txt



# Aula 42 - Comando ln

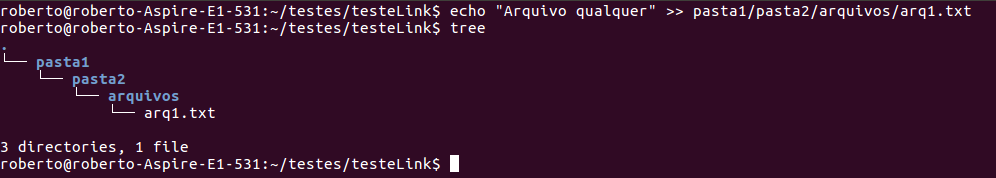
- O comando ln permite criar links simbólicos e hardlinks em shell.

mkdir -p testeLink/pasta1/pasta2/arquivos



- Dentro da pasta testeLink crie um arquivo chamado arq1.txt:

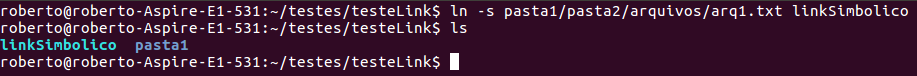
echo "Arquivo qualquer" >> pasta1/pasta2/arquivos/arq1.txt



## Criando um link simbólico para o arquivo criado

A flag -s indica que é um link simbólico (soft link)

ln -s pasta1/pasta2/arquivos/arq1.txt linkSimbolico



### Exibindo as permissões do link criado

ls -l linkSimbolico



- Agora linkSimbolico aponta para o arquivo criado arq1.txt (é um atalho para ele)

### Verificando o inode do link criado

ls -i linkSimbolico



### Verificando o inode do arquivo criado

ls -i pasta1/pasta2/arquivos/arq1.txt



- Observe que o inode dos arquivos é diferente.

- linkSimbolico aponta para arq1.txt:

cat pasta1/pasta2/arquivos/arq1.txt



cat linkSimbolico



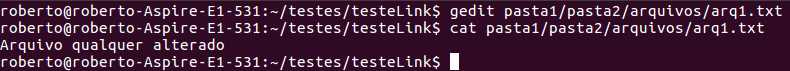
- Alterando o contéudo de arq1.txt isso irá refletir em linkSimbolico:

- Com gedit, altere o conteúdo de arq1.txt para:

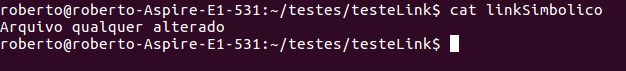
Arquivo qualquer alterado

- Salve e feche o editor.

cat pasta1/pasta2/arquivos/arq1.txt



cat linkSimbolico



## Exibindo o tipo de arquivo

O comando file exibe o tipo de arquivo.

file linkSimbolico

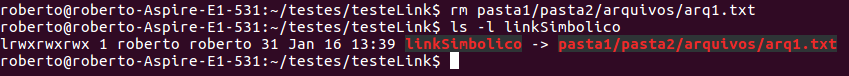


## Removendo o arquivo original

- Se o arquivo original (arq1.txt) for removido quebra-se o vínculo com o link simbólico e perde-se a informação.

rm pasta1/pasta2/arquivos/arq1.txt

ls -l linkSimbolico



file linkSimbolico



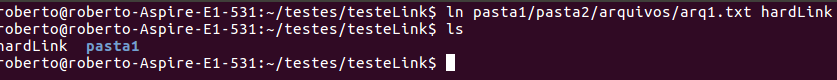
### Removendo o link quebrado

rm linkSimbolico

## Criando um hardlink

echo "Hard Link" >> pasta1/pasta2/arquivos/arq1.txt

ln pasta1/pasta2/arquivos/arq1.txt hardLink



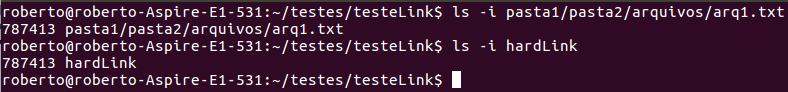
ls -l hardLink



### Verificando o inode dos arquivos

ls -i pasta1/pasta2/arquivos/arq1.txt

ls -i hardLink



- Observe que o inode dos arquivos é exatamente o mesmo.

Importante:

- Enquanto o inode do link simbólico era diferente, o inode do hardlink é igual.

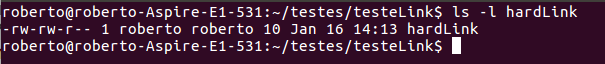
- Enquanto o soft link é um atalho para o arquivo, o hard link é um ponteiro para o inode.

## Removendo o arquivo original

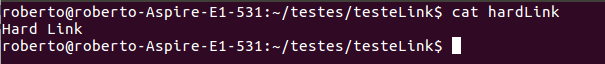
- Se o arquivo original (arq1.txt) for removido a informação não será preservada.

rm pasta1/pasta2/arquivos/arq1.txt

ls -l hardLink



cat hardLink



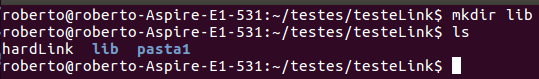
- No hard link, mesmo que o arquivo original for removido, a informação original não é perdida.

# Aula 43 - Utilizando links simbólicos como referência à uma biblioteca

- Dentro da pasta ~/testes/testeLink crie uma subpasta chamada lib:

mkdir lib

ls



- Dentro da pasta lib, com o editor gedit crie um arquivo chamado lib1.0.sh:

gedit lib1.0.sh

echo "Versão 1.0"

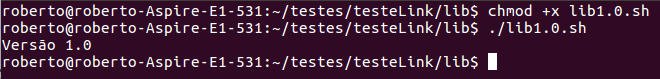


### Dando permissão de execução para o arquivo criado

chmod +x lib1.0.sh

### Executando o arquivo

./lib1.0.sh



- Dentro da pasta lib, com o editor gedit crie um arquivo chamado lib2.0.sh:

gedit lib2.0.sh

echo "Versão 2.0"

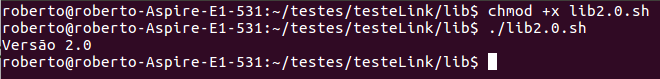


### Dando permissão de execução para o arquivo criado

chmod +x lib2.0.sh

### Executando o arquivo

./lib2.0.sh



ls



## Programa principal

Vá para a pasta ~/testes/testeLink

cd ~/testes/testeLink

gedit main.sh

./lib/lib1.0.sh



- Salve o arquivo e saia do editor.

ls

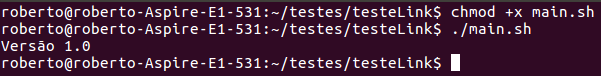


### Dando permissão de execução para o arquivo criado

chmod +x main.sh

### Executando o arquivo

./main.sh



## Utilizando link simbólico

- Altere o conteúdo do arquivo main.sh para:

gedit main.sh

./link\_lib

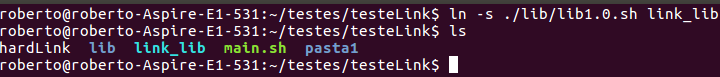


## Usando versão 1.0

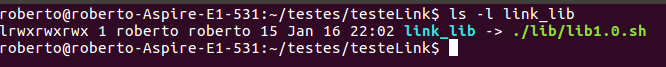
- Dentro da pasta ~/testes/TesteLink crie o link simbólico link\_db:

ln -s ./lib/lib1.0.sh link\_lib

ls

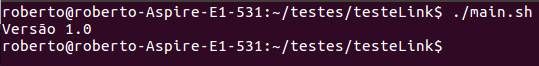


ls -l link\_lib



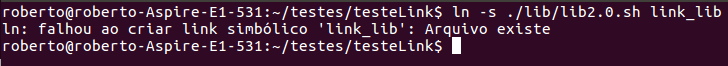
### Executando o programa principal (main.sh)

./main.sh



## Usando a versão 2.0

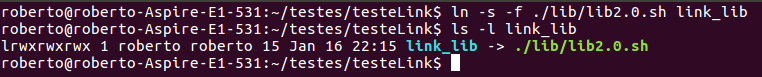
ln -s ./lib/lib2.0.sh link\_lib



- Com a flag -f (force) é forçado a alteração no link simbólico.

ln -s -f ./lib/lib2.0.sh link\_lib

ls -l link\_lib



- Perceba que agora link\_lib aponta para a versão 2.0

### Executando o programa principal (main.sh)

./main.sh

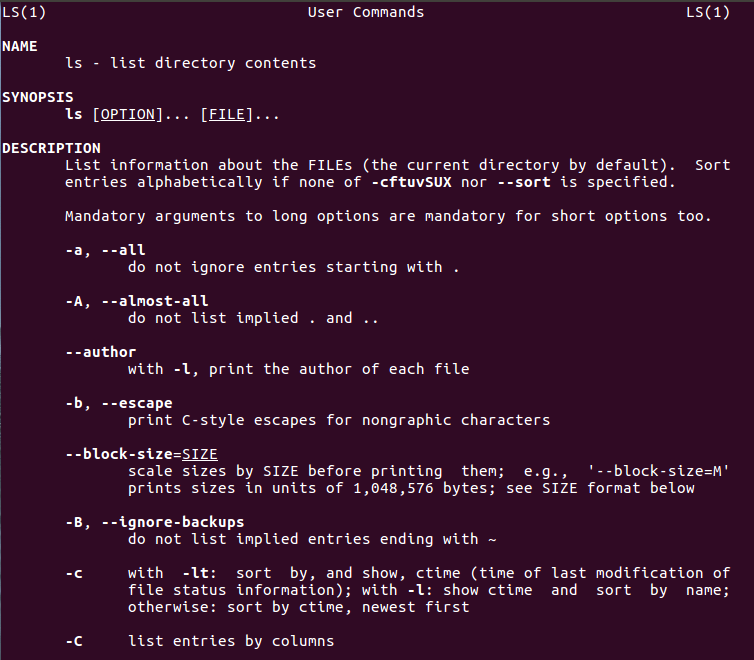


# Aula 44 - Comando main

- O comando main é utilizado para obter a documentação dos comandos do Linux.

## Exibindo a documentação do comando ls

man ls



.

.

.

- Para se mover na documentação pode-se usar as setas direcionadoras: seta para baixo ou seta para cima.

- Para sair da página pressione a tecla <q>

man man

- A documentação dos comandos está distribuída em seções.

1 Executable programs or shell commands

2 System calls (functions provided by the kernel)

3 Library calls (functions within program libraries)

4 Special files (usually found in /dev)

5 File formats and conventions eg /etc/passwd

6 Games

7 Miscellaneous (including macro packages and conventions), e.g. man(7), groff(7)

8 System administration commands (usually only for root)

9 Kernel routines [Non standard]

## Uma breve descrição dos comandos

É obtida utilizando-se a flag -f:

Exemplos:

man -f ls



man -f pwd



## Exibindo o local da documentação do comando

- Para exibir o local da documentação de um comando utiliza-se a flag -w:

man -w pwd

