

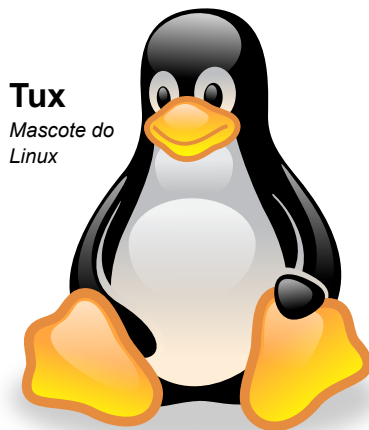


Laboratório 1.1: Introdução ao Linux, Linha de Comando e Arquivos

Introdução

Neste e nos próximos laboratórios faremos uma breve introdução ao sistema operacional Linux. Em outros laboratórios, mais para o fim da disciplina, veremos como o Linux está relacionado com o Android e o AOSP (*Android Open Source Project*).

O Linux é um sistema operacional livre e de código fonte aberto baseado no Unix. Criado em 1991 por Linus Torvalds, e divulgado pela primeira vez em um fórum [Usenet](#) chamado `comp.os.minix`:



Tux
Mascote do
Linux

Hello everybody out there using minix -

I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386(486) AT clones. This has been brewing since april, and is starting to get ready. I'd like any feedback on things people like/dislike in minix, as my OS resembles it somewhat (same physical layout of the file-system (due to practical reasons) among other things).

I've currently ported bash(1.08) and gcc(1.40), and things seem to work. This implies that I'll get something practical within a few months, and I'd like to know what features most people would want. Any suggestions are welcome, but I won't promise I'll implement them :-)

Linus (torvalds@kruuna.helsinki.fi)

Surgido inicialmente como um *hobby*, o Linux é hoje considerado um dos maiores projetos cooperativos da atualidade.

O Linux pode ser usado principalmente como:

- Sistema desktop tradicional
- Sistema para servidores
- Sistema para dispositivos embarcados

Neste curso, focaremos mais no último aspecto, em especial no uso do **Linux para o desenvolvimento e personalização do Android em sistemas embarcados**. Este é o principal motivo que utilizaremos muito o terminal (*shell*) do sistema, mas é importante deixar claro que hoje em dia o Linux pode ser utilizado como desktop sem precisar, na maioria das vezes, abrir o terminal. Além disso, ele é utilizado por muitas pessoas como o *único* sistema desktop em seus Laptops/PC, sendo completamente capaz de substituir outros sistemas operacionais.

O Linux possui programas correspondentes à maioria dos programas usados em outros sistemas:

- Navegador: *Chrome, Firefox*
- Word, Planilhas e Apresentações: *Google Docs, Sheets e Slides*
- Edição de Fotos: *Krita, Gimp*
- Edição de Imagens Vetoriais: *Inkscape*
- Edição de Vídeos: *Kdenlive, Olive Editor*
- Desenvolvimento: *Kdevelop, Eclipse, IntelliJ, Sublime, Visual Code Studio*

Sendo que todos esses programas são livres, não precisam de licença e são fáceis de instalar. Uma outra vantagem do Linux são os *drivers*. O sistema de instalação consegue reconhecer todos os dispositivos e utilizar os drivers correspondentes, sem precisar sair procurando um monte de drivers para download após a primeira instalação. Por fim, a atualização do sistema é rápida e sem precisar reiniciar o computador.

Obviamente, como todo sistema novo, se você nunca usou o Linux como seu desktop principal, haverá um certo período para se adaptar e se acostumar com a nova interface e procedimentos.

"Linux é vida." - Chat da disciplina Técnicas Avançadas de Programação / 2022

1. Distribuições Linux

Para facilitar o uso do Linux, o mesmo é utilizado através de "distribuições". Uma distribuição contém não apenas o kernel Linux, mas uma série de outros programas que permitem o uso do mesmo. As distribuições mais conhecidas hoje em dia são:

- Debian
- Ubuntu
- Fedora
- Mint



Neste curso, utilizaremos o [Linux Mint \(edição Cinnamon\)](#) versão 20.3. Você pode utilizar outras distribuições, mas recomendamos que seja uma baseada no Debian, como o Ubuntu ou alguma outra edição do Mint.

O Linux, propriamente dito, é apenas o kernel (núcleo) do sistema. A interface gráfica que vemos é implementada por um *gerenciador de janelas* conhecido como [X.Org](#). Já o desktop é conhecido como *desktop environment*. O Linux possui vários *desktop environments*. Os principais são:

- [GNOME](#)
- [KDE](#)
- [Cinnamon](#)
- [Xfce](#)



2. Instalação do Linux

Não entraremos nos detalhes da instalação de uma distribuição, pois isso varia de uma para outra. Entretanto, os passos principais e comuns entre eles são:

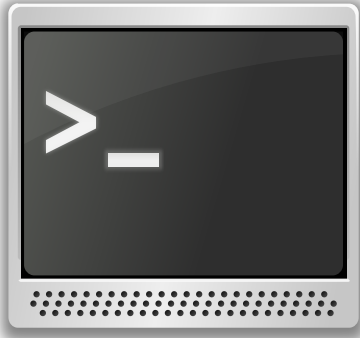
1. Entrar no site da distribuição
2. Baixar o arquivo ISO da versão mais recente
3. Criar um pendrive USB de boot
4. Reiniciar o Laptop/PC a partir do pendrive
5. Seguir os passos de instalação que, geralmente, envolvem:
 - Detecção e configuração de hardware
 - Selecionar o esquema de particionamento do disco
 - Cópia dos arquivos
 - Reboot

Para os detalhes de instalação (criação do pendrive, etc), entre no site da distribuição e siga os passos descritos lá. Para este curso, estamos considerando que você já está usando um Linux pré-instalado.

3. Sobre o Terminal

O terminal é uma interface texto para o sistema, em contraste com a interface gráfica. É também conhecido como *shell*, *console*, linha de comando, ou mesmo *prompt*. Em geral, as pessoas mais experientes com o Linux costumam usar muito o terminal, apesar de hoje em dia não ser mais tão necessário.

Neste curso, quase **tudo será feito no terminal**. O motivo para isso é que, por ser em modo texto, o terminal é o ambiente ideal para seguir instruções em tutoriais e



laboratórios disponíveis na Internet por permitir copiar e colar comandos. Além disso, muitos dos comandos usados pelo Android/AOSP não possuem correspondentes gráficos.

Se você não possui experiência com o terminal do Linux, não se preocupe! A ideia é você ir aprendendo e se familiarizando à medida que você for fazendo esse e os próximos laboratórios.

Abrindo um Terminal

Vamos começar abrindo um terminal. Clique no botão de "Iniciar" da sua interface gráfica (provavelmente no canto inferior/esquerdo) e procure por um programa chamado `terminal` ou `console`. Você pode começar a digitar o nome logo depois de clicar no botão iniciar (ele deve fazer uma pesquisa nos programas instalados).

O terminal, ao iniciar, executa um interpretador de comandos. Existem vários interpretadores de comandos no Linux. O mais utilizado e o padrão na maioria das distribuições Linux é o `bash` (*Bourne Again Shell*).


Executando Comandos

O terminal é utilizado através da execução de comandos. Tais comandos são, na maioria das vezes, programas existentes e instalados no sistema. A única diferença é que estes programas foram criados para serem executados em modo texto. Por exemplo, utilizando o terminal aberto, execute o comando abaixo para listar os arquivos do seu diretório Home.

```
$ ls
```

Copiando e Colando no Terminal

Neste e nos próximos laboratórios, será pedido para você executar comandos no terminal e copiar/colar a saída do comando de volta aqui no laboratório. Além de servir como um indicativo de que você está seguindo o laboratório, isso servirá como uma forma de você completar o laboratório transformando-o em um tutorial completo para consultas futuras.

Como alguns comandos serão grandes, o ideal é copiá-los daqui do laboratório e colá-los no terminal. Para isso, você pode usar o botão , que aparece no canto superior direito ao colocar o mouse em cima da caixa. Este botão automaticamente copia todo o código para a área de transferência. No terminal, pressione `Ctrl+Shift+V` para colar. O comando provavelmente gerará uma ou mais linhas de saída. Para copiar estas linhas,

selecione-as e pressione Ctrl+Shift+C. Em seguida, volte aqui para o laboratório e pressione Ctrl+V para colar o resultado no campo pedido.

Por exemplo, o comando executado anteriormente (`ls`) mostrou os arquivos do seu diretório *Home*. Para saber o nome e caminho completo do seu diretório *Home*, execute o comando abaixo:

```
$ pwd
```

```
/home/marcos
```

Como você deve ter notado, o campo ficará **verde caso sua resposta esteja correta e vermelho caso contrário**. As caixas de códigos também ficam verdes indicando que você já executou os comandos. Por fim, tudo feito no laboratório é **automaticamente salvo no servidor**. Você pode fechar o seu navegador a qualquer momento. Quando você voltar, mesmo usando um navegador diferente, os dados do servidor serão carregados de volta.

Como um segundo teste, na primeira seção falamos das distribuições Linux. Veja a descrição da sua distribuição atual executando o comando abaixo:

```
$ cat /etc/lsb-release
```

```
DISTRIB_ID=Ubuntu  
DISTRIB_RELEASE=20.04  
DISTRIB_CODENAME=focal  
DISTRIB_DESCRIPTION="Ubuntu 20.04.3 LTS"
```

Nas seções seguintes, serão mostrados os comandos mais comumente utilizados no terminal e, em especial, os comandos importantes para se desenvolver no Android/AOSP. **Não se preocupe em decorá-los**. Você irá memorizando os comandos à medida que for utilizando-os. Neste laboratório, se preocupe apenas em ter uma visão geral desses comandos.

4. Listando Arquivos

Até aqui, já executamos três comandos Linux diferentes, o `ls`, o `pwd` e o `cat`. Como dissemos anteriormente, eles nada mais são do que programas instalados no sistema. Podemos ver os detalhes dos seus executáveis usando o próprio comando `ls`:

```
$ ls -l /bin/ls
```

```
-rwxr-xr-x 1 root root 142144 Sep  5 2019 /bin/ls
```

Comando ls

O `ls` lista o conteúdo de um diretório. Executado sozinho, ele mostra todo o conteúdo do diretório atual sem considerar os arquivos ocultos. Podemos colocar um filtro após o comando, em que ele só listará os arquivos e diretórios que correspondem ao filtro. Por exemplo, para listar todos os arquivos do diretório `/etc` que começam com a palavra `host`, execute:

```
$ ls /etc/host*
```

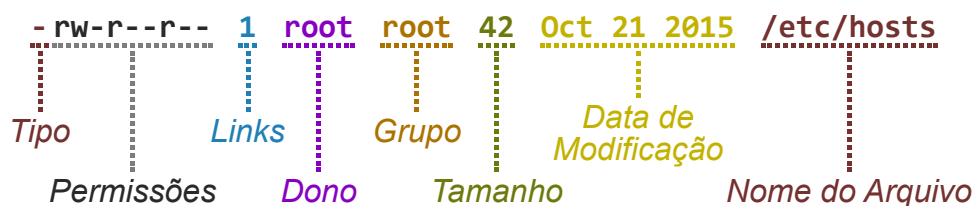
```
/etc/host.conf /etc/hostname /etc/hosts /etc/hosts.allow
/etc/hosts.deny
```

O `ls` também aceita opções, que modificam a sua saída. No Linux, as opções dos comandos são identificadas por começarem com o caractere `-` (traço) ou `--` (dois traços). Por exemplo, uma opção muito útil do comando `ls` é a opção `-l`, que mostra um arquivo por linha com seus detalhes:

```
$ ls -l /etc/host*
```

```
-rw-r--r-- 1 root root 92 Dec 5 2019 /etc/host.conf
-rw-r--r-- 1 root root 7 May 4 18:12 /etc/hostname
-rw-r--r-- 1 root root 517 May 4 18:12 /etc/hosts
-rw-r--r-- 1 root root 411 Aug 19 2021 /etc/hosts.allow
-rw-r--r-- 1 root root 711 Aug 19 2021 /etc/hosts.deny
```

A figura a seguir identifica os detalhes de um dos arquivos acima:



Já a opção `-h` do `ls` faz com que os tamanhos dos arquivos sejam legíveis a humanos (com K, M ou G), ao invés de serem em bytes. Note também no exemplo abaixo, que podemos combinar opções usando só um traço:

```
$ ls -lh /bin/ls
```

```
-rwxr-xr-x 1 root root 139K Sep 5 2019 /bin/ls
```

Arquivos Ocultos

Por fim, a opção `-a` (*all*) lista também os arquivos ocultos. No Linux, arquivos ocultos são arquivos cujos nomes começam com o caractere `.` (ponto). A opção `-a` lista também os diretórios `.` e `..` que correspondem aos diretórios atual e anterior, respectivamente.

```
$ ls -al
```

```
drwxr-xr-x 4 marcos marcos 4096 May  4 19:29 .  
drwxr-xr-x 3 root    root   4096 Feb 26 17:39 ..  
-rw----- 1 marcos marcos  602 May  4 19:08 .bash_history  
-rw-r--r-- 1 marcos marcos  220 Feb 26 17:39 .bash_logout  
-rw-r--r-- 1 marcos marcos 3771 Feb 26 17:39 .bashrc  
-rw-r--r-- 1 marcos marcos 4096 Feb 26 17:39 .bandwidth
```

5. Documentação dos Comandos

Comando man

O Linux possui um manual para a maioria dos seus comandos. Tais manuais são acessados através do comando `man`. Por exemplo, para uma documentação completa do comando `ls`, digite:

```
$ man ls
```

Pressione a tecla 'Q' para sair

```
ls - list directory contents
```

No comando acima, note que usamos um "comentário" (tudo após o `#`) para indicar como sair do manual. O comentário é ignorado pelo terminal. Como você pode observar, o comando `man` mostra uma documentação bem detalhada do comando, com todas as opções e, em alguns casos, até alguns exemplos de execução.

Conforme mencionado, a maioria dos comandos Linux possuem um `man`. Você pode executá-lo a qualquer momento para acessar a documentação de qualquer comando.

Comando tldr

O problema do `man` é que ele é muito detalhado e às vezes demoramos a encontrar algo que queremos fazer. Além disso, muitos manuais não possuem exemplo de execução, como é o caso do manual do `ls`. Para ver uma ajuda mais direta, mostrando logo os exemplos que queremos, execute o comando `tldr` (*too long; didn't read*), a seguir:

```
$ tldr ls
```

Note que a saída dele é bem mais direta, mostrando exemplos dos casos mais comuns de uso do programa. Se você não tiver o programa `tldr` instalado, execute:

```
$ sudo apt install tldr
```

Veremos os detalhes do comando acima mais adiante, mas ele executa o comando `apt` como administrador (`sudo`) para instalar o programa `tlldr`. Só será possível executar o `sudo` se o seu usuário atual pertencer ao grupo de administradores. Chame o monitor/professor caso seu usuário não esteja no grupo. Você precisará dessa permissão para outros comandos futuros.

Na grande maioria das vezes, instalar um programa no Linux é tão simples quanto executar um comando `apt`. Ou seja, é bem mais simples que em outros sistemas operacionais, em que precisamos entrar no site do programa que queremos para encontrar o link de download (após ver várias propagandas e ter que preencher formulários com seu e-mail), depois executar o instalador, que irá pedir para clicar em "Next" várias vezes e que, no final, ainda irá fazer o grande favor de instalar uma barra nova de propagandas no seu navegador preferido.

6. Navegando em Diretórios

Comandos `cd` e `pwd`

O comando `cd` (*change dir*), muda o diretório atual. Na linha de comando, sempre estamos dentro de algum diretório. Este diretório é o diretório atual de trabalho. Para saber o diretório atual, executamos o comando `pwd` (*print working directory*). Quando abrimos um terminal novo, somos normalmente colocados no diretório *Home*, que é o diretório do usuário atual, um dos poucos que o mesmo terá permissão de escrita/modificação. Nos comandos seguintes, mudaremos o diretório de trabalho para ser o `/etc` e executaremos o comando `pwd` para verificar se realmente estamos dentro do diretório:

```
$ cd /var/log          # Muda o diretório de trabalho atual
$ pwd                  # Imprime o diretório de trabalho
atual
```

```
/var/log
```

Note como no exemplo anterior, temos dois comandos a serem executados. Você pode executar um por vez (copiando e colando linha por linha) ou pode copiar/colar todos os comandos de uma só vez.

Como o diretório *Home* é um dos mais utilizados, ele possui um atalho que é o `~` (til):

```
$ cd ~                  # Vai para o diretório home
$ pwd
```

```
/home/marcos
```


Para facilitar ainda mais ir para o diretório *Home*, o ~ (til) é, na verdade, opcional:

```
$ cd /var/log          # Muda o diretório de trabalho atual
$ cd                  # Volta para o Home
$ pwd
```

```
/home/marcos
```

Na seção anterior, mencionamos os diretórios especiais "." e "..", vistos quando executamos o comando `ls -al`. O diretório "." é uma referência para o diretório atual e o ".." é uma referência para o diretório anterior. Nós podemos "entrar" nestes diretórios:

```
$ cd .                # "Entra" no diretório atual (i.e.,
continua aonde está)
$ pwd
```

```
/home/marcos
```

```
$ cd ..              # Vai para o diretório anterior
$ pwd
```

```
/home
```

Por fim, se você acabou de entrar em um diretório e quer voltar para o diretório que você estava anteriormente, execute:

```
$ cd -               # Volta para o diretório que você
estava antes de entrar no atual
$ pwd
```

```
/home/marcos
```

7. Lendo Arquivos

Comando cat

O comando mais comum para ver o conteúdo de um arquivo é o `cat` (*concatenate*). O objetivo principal dele é, na verdade, concatenar arquivos e mostrar o conteúdo resultante na saída padrão (geralmente o terminal). Entretanto, se você só colocar o nome de um único arquivo, ele irá mostrar o conteúdo deste arquivo.

```
$ cat /etc/hostname
```

```
Marcos
```

Comando more

Se o arquivo for grande (muitas linhas), ele será todo impresso no terminal de uma só vez. Você terá que usar a barra de rolagem para olhar o conteúdo desde o início. Outra opção é usar o comando `more`, que mostra o conteúdo do arquivo começando pelo início, mas até o limite de linhas do seu terminal atual. Para continuar vendo o restante do arquivo, pressione *enter* para mostrar mais uma linha ou *espaço* para pular uma página inteira (quantidade de linhas do seu terminal). Para sair do `more`, pressione *espaço* várias vezes para chegar no final do arquivo ou pressione a tecla `Q` para sair sem imprimir o restante do arquivo:

```
$ more /etc/login.defs          # Pressione Q para sair
```

Comandos head e tail

Você pode também querer ver apenas as primeiras ou as últimas linhas de um arquivo. Para isso, existem os comandos `head` e `tail`, respectivamente. Por exemplo, para ver as primeiras 5 linhas do arquivo `/etc/passwd`, execute:

```
$ head -5 /etc/passwd
```

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
```

Já para ver as 5 últimas linhas, execute:

```
$ tail -5 /etc/passwd
```

```
tcpdump:x:108:113::/nonexistent:/usr/sbin/nologin
sshd:x:109:65534::/run/sshd:/usr/sbin/nologin
landscape:x:110:115::/var/lib/landscape:/usr/sbin/nologin
pollinate:x:111:1::/var/cache/pollinate:/bin/false
marcos:x:1000:1000:,,,:/home/marcos:/bin/bash
```

Ainda sobre o comando `tail`, uma opção interessante dele é a `-f`. Esta opção mostra as últimas linhas do arquivo, mas continua monitorando o arquivo aberto por novas

linhas. Assim que o conteúdo do arquivo muda, o `tail -f` exibe as novas linhas. Isso é bom quando queremos acompanhar as novas linhas em um arquivo de log:

```
$ tail -f /var/log/syslog      # Pressione Ctrl+C para sair
```

O comando acima ficará monitorando o arquivo de log do sistema por novos eventos (novas linhas). Espere alguns segundos ou force uma saída abrindo uma nova aba do terminal (Ctrl+Shift+T) e executando o comando `logger "Mensagem de Teste"`. Esse comando irá escrever uma mensagem de sistema. Voltando para o terminal anterior, você deverá ver a nova linha. Pressione Ctrl+C para sair.

8. Editando Arquivos

O Linux possui vários editores de texto em linha de comando. Os mais conhecidos são o `vi` e o `nano`. Falaremos rapidamente deles aqui. Entretanto, o editor que usaremos mesmo durante o curso será um editor em modo gráfico, que falaremos mais a seguir.

Comando vi/vim

O `vi` (ou `vim -- vi improved`) é um editor em modo texto desenvolvido principalmente para funcionar em qualquer terminal local ou remoto, incluindo os mais antigos. Ele possui dois modos: o modo de navegação (padrão) e o modo de edição. Ao editar um arquivo usando o `vi`, ele inicia no modo de navegação, em que você pode usar as setas do teclado para navegar no arquivo. Para poder alterar o arquivo, você precisa entrar no modo edição. Para isso, pressione o caractere `i` (*insert*). Agora, você já pode alterar o arquivo. Para sair do modo de edição, pressione a tecla `esc`. Por fim, para sair do editor, pressione o caractere `:` (dois pontos), digite o comando `wq` (de *write quit*) e pressione `enter`.

Vamos tentar usar o `vi`! Execute o comando abaixo para criar e editar um arquivo novo chamado `ArquivoTeste.txt`:

```
$ vi ArquivoTeste.txt
```

Siga os seguintes passos:

- Digite `i` para entrar no modo de edição;
- Digite a frase "Este é um arquivo de teste!";
- Pressione `esc` para sair do modo de edição e voltar para o modo de navegação;
- Digite `:wq` seguido de `enter` para salvar o arquivo e sair.



Nota: caso você fique preso dentro do vi/vim, não se preocupe! Isso é comum e já até virou meme. Chame o professor/monitor para te ajudar :).

Veja se o arquivo foi criado com sucesso executando o comando abaixo:

```
$ cat ArquivoTeste.txt
```

```
Este é um arquivo de teste!
```

Comando nano

Já o nano, é um editor em modo texto mais amigável. Ao abrir um arquivo com o nano, ele já está pronto para ser editado. A ajuda do nano com os comandos para salvar e sair são mostrados na parte de baixo da tela. Pressione Ctrl+O para salvar as mudanças a qualquer momento e Ctrl+X para sair.

```
$ nano ArquivoTeste.txt
```

Editores em Modo Gráfico

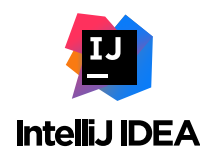
O Linux possui vários editores em modo gráfico. Alguns mais simples incluem: xed, gedit e kate. Existem ainda editores mais próprios para trabalhar com códigos fonte:

- Atom
- Sublime
- Visual Studio Code



Por fim, o Linux possui ainda IDEs (*Integrated Development Environments*) completos como:

- Eclipse
- KDevelop
- IntelliJ



Neste curso, usaremos o editor nano, mas você pode utilizar qualquer outro de sua preferência:

```
$ nano ArquivoTeste.txt
```

9. Copiando, Movendo e Renomeando Arquivos

Comando cp

Para copiar arquivos (e diretórios), usamos o comando `cp`:

```
$ cp ArquivoTeste.txt ArquivoCopia.txt           # Faz uma cópia
do arquivo no mesmo diretório
$ ls -lh Arquivo[C\|T]*.txt
```

```
-rw-r--r-- 1 marcos marcos 29 May  4 19:10 ArquivoCopia.txt
-rw-r--r-- 1 marcos marcos 29 May  4 19:09 ArquivoTeste.txt
```

É possível copiar um arquivo para outro diretório também:

```
$ cp ArquivoTeste.txt /tmp                       # Copia o arquivo
para o diretório /tmp
$ ls -lh ArquivoTeste.txt /tmp/ArquivoTeste.txt
```

```
-rw-r--r-- 1 marcos marcos 29 May  4 19:11 /tmp/ArquivoTeste.txt
-rw-r--r-- 1 marcos marcos 29 May  4 19:09 ArquivoTeste.txt
```

Comando `mv`

O comando `mv` move arquivos (e diretórios) entre diretórios diferentes ou renomeia um arquivo/diretório (i.e., move o arquivo para o mesmo diretório mas com nome diferente).

```
$ mv ArquivoCopia.txt /tmp                       # Move o arquivo
para o diretório /tmp
$ ls -lh /tmp/ArquivoCopia.txt
```

```
-rw-r--r-- 1 marcos marcos 29 May  4 19:10 /tmp/ArquivoCopia.txt
```

Para renomear um arquivo, é só colocar o novo nome do arquivo no segundo argumento do comando:

```
$ mv ArquivoTeste.txt ArquivoRenomeado.txt      # Renomeia o
arquivo
$ ls -lh ArquivoRenomeado.txt
```

```
-rw-r--r-- 1 marcos marcos 29 May  4 19:09 ArquivoRenomeado.txt
```

Por fim, é possível mover um arquivo para outro diretório mudando o seu nome:

```
$ mv ArquivoRenomeado.txt /tmp/ArquivoMovido.txt # Move e renomeia
o arquivo
$ ls -lh /tmp/ArquivoMovido.txt
```

```
-rw-r--r-- 1 marcos marcos 29 May  4 19:09 /tmp/ArquivoMovido.txt
```

10. Criando e Manipulando Diretórios

Comando mkdir e rmdir

Para **criar** um diretório novo, usa-se o comando mkdir:

```
$ mkdir TesteDir                # Cria o
diretório
$ cd TesteDir                    # Entra no
diretório
$ ls -al                         # Lista todos os
arquivos
```

```
total 8
drwxr-xr-x 2 marcos marcos 4096 May  4 19:14 .
drwxr-xr-x 4 marcos marcos 4096 May  4 19:14 ..
```

Para **remover** um diretório vazio, usa-se o comando rmdir:

```
$ cd ..                          # Volta para o
diretório anterior
$ rmdir TesteDir                 # Remove o
diretório vazio
```

Não é possível remover, usando o rmdir, um diretório que tenha arquivos dentro:

```
$ mkdir TesteDir2
$ cd TesteDir2
$ echo "Jebediah Kerman" > ArquivoTeste.txt
$ cd ..
$ rmdir TesteDir2
```

```
rmdir: failed to remove 'TesteDir2': Directory not empty
```

Neste caso, usa-se o comando rm, com a opção -rf (*recursive, force*):

```
$ rm -rf TesteDir2
```

Atenção! O comando `rm -rf` é um dos comandos mais perigosos do mundo Linux/Unix. Ele apaga recursivamente, de forma irreversível e sem pedir nenhuma confirmação

(opção `-f`), todos os arquivos e diretórios a partir do caminho especificado. Muito cuidado ao executá-lo! Mais cuidado ainda deve-se ter se você estiver usando o usuário *root* (administrador). Neste caso, você pode (quase) literalmente apagar todos os arquivos do sistema.

Se você precisar **renomear** um diretório (com ou sem arquivos dentro), deve-se utilizar o comando `mv` conforme feito com os arquivos normais.

Por fim, se você precisar **copiar** um diretório e todo o seu conteúdo para outro local, deve-se utilizar o comando `cp`, explicado anteriormente, mas com a opção `-r` (*recursive*).



Fall seven times, stand up eight. — Japanese Proverb

π