

# Introdução ao GNU/Linux

## Comandos e Scripts

Felipe Dias de Oliveira



Escola Técnica Estadual  
Governador Eduardo Campos

*fdoprof@gmail.com*

5 de novembro de 2021

## 1 Introdução

- O que é Software Livre?
- O que é Software Livre?
- Um Breve histórico...
- O que são Distribuições?
- Algumas Distribuições...

## 2 Primeiros Passos

- Sistema de Arquivos e Diretórios
- Manipulando arquivos e diretórios
- Coringas
- Comandos Úteis

## Objetivos

## Objetivos

- Descobrir o que é GNU/Linux;

## Objetivos

- Descobrir o que é GNU/Linux;
- Entender a filosofia do Software Livre;

## Objetivos

- Descobrir o que é GNU/Linux;
- Entender a filosofia do Software Livre;
- Conhecer um pouco da história;

## Objetivos

- Descobrir o que é GNU/Linux;
- Entender a filosofia do Software Livre;
- Conhecer um pouco da história;
- Entender o que são Distribuições

# O que é Software Livre?



# O que é Software Livre?

- Software: funcionalidade traduzida para linguagem de máquina;

# O que é Software Livre?

- Software: funcionalidade traduzida para linguagem de máquina;
  - Alguém sistematiza essa funcionalidade e a codifica;

# O que é Software Livre?

- Software: funcionalidade traduzida para linguagem de máquina;
  - Alguém sistematiza essa funcionalidade e a codifica;
  - O código pode ser traduzido para a linguagem de máquina (compilação)

# O que é Software Livre?

- Software: funcionalidade traduzida para linguagem de máquina;
  - Alguém sistematiza essa funcionalidade e a codifica;
  - O código pode ser traduzido para a linguagem de máquina (compilação)
- No modelo convencional, apenas o código binário é empacotado e distribuído;

# O que é Software Livre?

- Software: funcionalidade traduzida para linguagem de máquina;
  - Alguém sistematiza essa funcionalidade e a codifica;
  - O código pode ser traduzido para a linguagem de máquina (compilação)
- No modelo convencional, apenas o código binário é empacotado e distribuído;
  - A empresa detém todo o conhecimento necessário para o desenvolvimento do software;

# O que é Software Livre?

- Software: funcionalidade traduzida para linguagem de máquina;
  - Alguém sistematiza essa funcionalidade e a codifica;
  - O código pode ser traduzido para a linguagem de máquina (compilação)
- No modelo convencional, apenas o código binário é empacotado e distribuído;
  - A empresa detém todo o conhecimento necessário para o desenvolvimento do software;
- Em 1980, Richard Stallman já trocava código com outros hackers;

# O que é Software Livre?

- Software: funcionalidade traduzida para linguagem de máquina;
  - Alguém sistematiza essa funcionalidade e a codifica;
  - O código pode ser traduzido para a linguagem de máquina (compilação)
- No modelo convencional, apenas o código binário é empacotado e distribuído;
  - A empresa detém todo o conhecimento necessário para o desenvolvimento do software;
- Em 1980, Richard Stallman já trocava código com outros hackers;
- Ao precisar do código fonte do programa que controlava uma impressora;

# O que é Software Livre?

- Software: funcionalidade traduzida para linguagem de máquina;
  - Alguém sistematiza essa funcionalidade e a codifica;
  - O código pode ser traduzido para a linguagem de máquina (compilação)
- No modelo convencional, apenas o código binário é empacotado e distribuído;
  - A empresa detém todo o conhecimento necessário para o desenvolvimento do software;
- Em 1980, Richard Stallman já trocava código com outros hackers;
- Ao precisar do código fonte do programa que controlava uma impressora;
  - Percebeu a necessidade de um movimento que desse garantias de liberdade do compartilhamento de software;



# O que é Software Livre?

- Software: funcionalidade traduzida para linguagem de máquina;
  - Alguém sistematiza essa funcionalidade e a codifica;
  - O código pode ser traduzido para a linguagem de máquina (compilação)
- No modelo convencional, apenas o código binário é empacotado e distribuído;
  - A empresa detém todo o conhecimento necessário para o desenvolvimento do software;
- Em 1980, Richard Stallman já trocava código com outros hackers;
- Ao precisar do código fonte do programa que controlava uma impressora;
  - Percebeu a necessidade de um movimento que desse garantias de liberdade do compartilhamento de software;

# O que é Software Livre?

# O que é Software Livre?

- Em 1984, cria o projeto GNU (GNU is not Unix);

# O que é Software Livre?

- Em 1984, cria o projeto GNU (GNU is not Unix);
  - Sistema operacional livre;

# O que é Software Livre?

- Em 1984, cria o projeto GNU (GNU is not Unix);
  - Sistema operacional livre;
  - Para isso, estabeleceu um manifesto de 4 liberdades

# O que é Software Livre?

- Em 1984, cria o projeto GNU (GNU is not Unix);
  - Sistema operacional livre;
  - Para isso, estabeleceu um manifesto de 4 liberdades
- Em 1991, o projeto possuía um sistema operacional quase completo;

# O que é Software Livre?

- Em 1984, cria o projeto GNU (GNU is not Unix);
  - Sistema operacional livre;
  - Para isso, estabeleceu um manifesto de 4 liberdades
- Em 1991, o projeto possuía um sistema operacional quase completo;
  - Não possuía um núcleo;

# O que é Software Livre?

- Em 1984, cria o projeto GNU (GNU is not Unix);
  - Sistema operacional livre;
  - Para isso, estabeleceu um manifesto de 4 liberdades
- Em 1991, o projeto possuía um sistema operacional quase completo;
  - Não possuía um núcleo;
  - Linus T. libera uma versão funcional do kernel;



# O que é Software Livre?

- Em 1984, cria o projeto GNU (GNU is not Unix);
  - Sistema operacional livre;
  - Para isso, estabeleceu um manifesto de 4 liberdades
- Em 1991, o projeto possuía um sistema operacional quase completo;
  - Não possuía um núcleo;
  - Linus T. libera uma versão funcional do kernel;
  - Os primeiros usuários uniram os dois projetos e criaram o GNU/Linux;

# O que é Software Livre?

- Em 1984, cria o projeto GNU (GNU is not Unix);
  - Sistema operacional livre;
  - Para isso, estabeleceu um manifesto de 4 liberdades
- Em 1991, o projeto possuía um sistema operacional quase completo;
  - Não possuía um núcleo;
  - Linus T. libera uma versão funcional do kernel;
  - Os primeiros usuários uniram os dois projetos e criaram o GNU/Linux;

# O que é Software Livre?

Para que esse mundo continue livre, **Richard Stallman** fundou a **FSF - Free Software Foundation**, que mantém a licença chamada **GNU GPL - GNU General Public License**.

# O que é Software Livre?

Para que esse mundo continue livre, **Richard Stallman** fundou a **FSF - Free Software Foundation**, que mantém a licença chamada **GNU GPL - GNU General Public License**.

- **liberdade 0** - liberdade para rodar o programa para quaisquer propósitos;

# O que é Software Livre?

Para que esse mundo continue livre, **Richard Stallman** fundou a **FSF - Free Software Foundation**, que mantém a licença chamada **GNU GPL - GNU General Public License**.

- **liberdade 0** - liberdade para rodar o programa para quaisquer propósitos;
- **liberdade 1** - liberdade para estudar como o programa trabalha e adaptá-lo às suas necessidades. Ter acesso ao código fonte é essencial para isso.

# O que é Software Livre?

Para que esse mundo continue livre, **Richard Stallman** fundou a **FSF - Free Software Foundation**, que mantém a licença chamada **GNU GPL - GNU General Public License**.

- **liberdade 0** - liberdade para rodar o programa para quaisquer propósitos;
- **liberdade 1** - liberdade para estudar como o programa trabalha e adaptá-lo às suas necessidades. Ter acesso ao código fonte é essencial para isso.
- **liberdade 2** - liberdade de redistribuir cópias de forma que você possa ajudar outras pessoas.

# O que é Software Livre?

Para que esse mundo continue livre, **Richard Stallman** fundou a **FSF - Free Software Foundation**, que mantém a licença chamada **GNU GPL - GNU General Public License**.

- **liberdade 0** - liberdade para rodar o programa para quaisquer propósitos;
- **liberdade 1** - liberdade para estudar como o programa trabalha e adaptá-lo às suas necessidades. Ter acesso ao código fonte é essencial para isso.
- **liberdade 2** - liberdade de redistribuir cópias de forma que você possa ajudar outras pessoas.
- **liberdade 3** - liberdade para melhorar o programa e disponibilizar as melhorias para o público, de forma que toda a comunidade possa se beneficiar disso. Ter acesso ao código fonte é essencial para isso.

# O que é Software Livre?

Atualmente a GPL está disponível em três versões, **GPLv1**, **GPLv2** e **GPLv3**. Veja suas diferenças em:  
<http://www.gnu.org/licenses/gpl.html>



# O que é Software Livre?

Atualmente a GPL está disponível em três versões, **GPLv1**, **GPLv2** e **GPLv3**. Veja suas diferenças em:

<http://www.gnu.org/licenses/gpl.html>

Existem outras licenças e variações:

[https://pt.wikipedia.org/wiki/Licença\\_de\\_software\\_livre](https://pt.wikipedia.org/wiki/Licença_de_software_livre)

Como todos tem acesso ao código fonte, o conhecimento é compartilhado!

# Um Breve histórico...

# Um Breve histórico...

- 1991 - Linus Torvalds cria um *kernel* baseado no Minix

# Um Breve histórico...

- 1991 - Linus Torvalds cria um *kernel* baseado no Minix
  - Deixou o código fonte aberto;

# Um Breve histórico...

- 1991 - Linus Torvalds cria um *kernel* baseado no Minix
  - Deixou o código fonte aberto;
  - O kernel é uma forma de controlar o hardware por software;

# Um Breve histórico...

- 1991 - Linus Torvalds cria um *kernel* baseado no Minix
  - Deixou o código fonte aberto;
  - O kernel é uma forma de controlar o hardware por software;
  - Criar um sistema operacional onde fosse possível fazer alterações conforme as necessidades;

# Um Breve histórico...

- 1991 - Linus Torvalds cria um *kernel* baseado no Minix
  - Deixou o código fonte aberto;
  - O kernel é uma forma de controlar o hardware por software;
  - Criar um sistema operacional onde fosse possível fazer alterações conforme as necessidades;
- 1992 - Surgem as primeiras distribuições

# Um Breve histórico...

- 1991 - Linus Torvalds cria um *kernel* baseado no Minix
  - Deixou o código fonte aberto;
  - O kernel é uma forma de controlar o hardware por software;
  - Criar um sistema operacional onde fosse possível fazer alterações conforme as necessidades;
- 1992 - Surgem as primeiras distribuições
  - Formou-se comunidades de cooperação;



# Um Breve histórico...

- 1991 - Linus Torvalds cria um *kernel* baseado no Minix
  - Deixou o código fonte aberto;
  - O kernel é uma forma de controlar o hardware por software;
  - Criar um sistema operacional onde fosse possível fazer alterações conforme as necessidades;
- 1992 - Surgem as primeiras distribuições
  - Formou-se comunidades de cooperação;
  - O desenvolvimento é totalmente distribuído pelo mundo;

# Um Breve histórico...

- 1991 - Linus Torvalds cria um *kernel* baseado no Minix
  - Deixou o código fonte aberto;
  - O kernel é uma forma de controlar o hardware por software;
  - Criar um sistema operacional onde fosse possível fazer alterações conforme as necessidades;
- 1992 - Surgem as primeiras distribuições
  - Formou-se comunidades de cooperação;
  - O desenvolvimento é totalmente distribuído pelo mundo;
    - O desenvolvimento é **24/7**;

# Um Breve histórico...

- 1991 - Linus Torvalds cria um *kernel* baseado no Minix
  - Deixou o código fonte aberto;
  - O kernel é uma forma de controlar o hardware por software;
  - Criar um sistema operacional onde fosse possível fazer alterações conforme as necessidades;
- 1992 - Surgem as primeiras distribuições
  - Formou-se comunidades de cooperação;
  - O desenvolvimento é totalmente distribuído pelo mundo;
    - O desenvolvimento é **24/7**;
- 1994 - O Linux atinge 900 mil usuários

# Um Breve histórico...

- 1991 - Linus Torvalds cria um *kernel* baseado no Minix
  - Deixou o código fonte aberto;
  - O kernel é uma forma de controlar o hardware por software;
  - Criar um sistema operacional onde fosse possível fazer alterações conforme as necessidades;
- 1992 - Surgem as primeiras distribuições
  - Formou-se comunidades de cooperação;
  - O desenvolvimento é totalmente distribuído pelo mundo;
    - O desenvolvimento é **24/7**;
- 1994 - O Linux atinge 900 mil usuários
  - Kernel 1.0 é oficialmente lançado

# Um Breve histórico...

- 1991 - Linus Torvalds cria um *kernel* baseado no Minix
  - Deixou o código fonte aberto;
  - O kernel é uma forma de controlar o hardware por software;
  - Criar um sistema operacional onde fosse possível fazer alterações conforme as necessidades;
- 1992 - Surgem as primeiras distribuições
  - Formou-se comunidades de cooperação;
  - O desenvolvimento é totalmente distribuído pelo mundo;
    - O desenvolvimento é **24/7**;
- 1994 - O Linux atinge 900 mil usuários
  - Kernel 1.0 é oficialmente lançado
  - A NASA lança a 1ª versão do **cluster beowulf**

# Um Breve histórico...

- 1991 - Linus Torvalds cria um *kernel* baseado no Minix
  - Deixou o código fonte aberto;
  - O kernel é uma forma de controlar o hardware por software;
  - Criar um sistema operacional onde fosse possível fazer alterações conforme as necessidades;
- 1992 - Surgem as primeiras distribuições
  - Formou-se comunidades de cooperação;
  - O desenvolvimento é totalmente distribuído pelo mundo;
    - O desenvolvimento é **24/7**;
- 1994 - O Linux atinge 900 mil usuários
  - Kernel 1.0 é oficialmente lançado
  - A NASA lança a 1ª versão do **cluster beowulf**

# Um Breve histórico...

- 1995 - O Brasil entra na comunidade Linux

# Um Breve histórico...

- 1995 - O Brasil entra na comunidade Linux
  - A distribuição Conectiva é criada



# Um Breve histórico...

- 1995 - O Brasil entra na comunidade Linux
  - A distribuição Conectiva é criada
- 1996/2000 - Diferentes interfaces gráficas foram criadas

# Um Breve histórico...

- 1995 - O Brasil entra na comunidade Linux
  - A distribuição Conectiva é criada
- 1996/2000 - Diferentes interfaces gráficas foram criadas
  - KDE;

# Um Breve histórico...

- 1995 - O Brasil entra na comunidade Linux
  - A distribuição Conectiva é criada
- 1996/2000 - Diferentes interfaces gráficas foram criadas
  - KDE;
  - GNOME;

# Um Breve histórico...

- 1995 - O Brasil entra na comunidade Linux
  - A distribuição Conectiva é criada
- 1996/2000 - Diferentes interfaces gráficas foram criadas
  - KDE;
  - GNOME;
- 2001 - A IBM anuncia que irá investir 1 bilhão em Linux

# Um Breve histórico...

- 1995 - O Brasil entra na comunidade Linux
  - A distribuição Conectiva é criada
- 1996/2000 - Diferentes interfaces gráficas foram criadas
  - KDE;
  - GNOME;
- 2001 - A IBM anuncia que irá investir 1 bilhão em Linux
  - Marcelo Tosatti é indicado por Linus T. para ser mantenedor mundial do kernel;

# Um Breve histórico...

- 1995 - O Brasil entra na comunidade Linux
  - A distribuição Conectiva é criada
- 1996/2000 - Diferentes interfaces gráficas foram criadas
  - KDE;
  - GNOME;
- 2001 - A IBM anuncia que irá investir 1 bilhão em Linux
  - Marcelo Tosatti é indicado por Linus T. para ser mantenedor mundial do kernel;
  - O kernel 2.4 é lançado;

# Um Breve histórico...

- 1995 - O Brasil entra na comunidade Linux
  - A distribuição Conectiva é criada
- 1996/2000 - Diferentes interfaces gráficas foram criadas
  - KDE;
  - GNOME;
- 2001 - A IBM anuncia que irá investir 1 bilhão em Linux
  - Marcelo Tosatti é indicado por Linus T. para ser mantenedor mundial do kernel;
  - O kernel 2.4 é lançado;
  - A NASA anuncia um assistente pessoal de satélite.

# Um Breve histórico...

- 1995 - O Brasil entra na comunidade Linux
  - A distribuição Conectiva é criada
- 1996/2000 - Diferentes interfaces gráficas foram criadas
  - KDE;
  - GNOME;
- 2001 - A IBM anuncia que irá investir 1 bilhão em Linux
  - Marcelo Tosatti é indicado por Linus T. para ser mantenedor mundial do kernel;
  - O kernel 2.4 é lançado;
  - A NASA anuncia um assistente pessoal de satélite.



# O que são Distribuições?

# O que são Distribuições?

Em uma definição mais abrangente, podemos dizer que é um sistema operacional de código-fonte aberto, e que é distribuído gratuitamente pela internet.

# O que são Distribuições?

Em uma definição mais abrangente, podemos dizer que é um sistema operacional de código-fonte aberto, e que é distribuído gratuitamente pela internet.

- Tecnicamente falando, o GNU/Linux é um **kernel**.

# O que são Distribuições?

Em uma definição mais abrangente, podemos dizer que é um sistema operacional de código-fonte aberto, e que é distribuído gratuitamente pela internet.

- Tecnicamente falando, o GNU/Linux é um **kernel**.
- Kernel + Sistema GNU = GNU/Linux.

# O que são Distribuições?

Em uma definição mais abrangente, podemos dizer que é um sistema operacional de código-fonte aberto, e que é distribuído gratuitamente pela internet.

- Tecnicamente falando, o GNU/Linux é um **kernel**.
- Kernel + Sistema GNU = GNU/Linux.
- Kernel + Sistema GNU + Aplicativos + Empacotamento =  
**Distribuição Linux**

# O que são Distribuições?

**Distribuições Livres** - mantidas por comunidades de colaboradores sem fins lucrativos. Exemplos são: Debian, Ubuntu, Slackware, Gentoo, Knoppix e CentOS, entre outras.

# O que são Distribuições?

**Distribuições Livres** - mantidas por comunidades de colaboradores sem fins lucrativos. Exemplos são: Debian, Ubuntu, Slackware, Gentoo, Knoppix e CentOS, entre outras.

**Distribuições Corporativas** - mantidas por empresas que vendem o suporte ao seu sistema. Exemplos são: RedHat, SuSe e Mandriva.

# O que são Distribuições?

**Distribuições Livres** - mantidas por comunidades de colaboradores sem fins lucrativos. Exemplos são: Debian, Ubuntu, Slackware, Gentoo, Knoppix e CentOS, entre outras.

**Distribuições Corporativas** - mantidas por empresas que vendem o suporte ao seu sistema. Exemplos são: RedHat, SuSe e Mandriva.



# O que são Distribuições?

**Distribuições Convencionais** - são distribuídas da forma tradicional, ou seja, uma ou mais mídias que são utilizadas para instalar o sistema no disco rígido;

# O que são Distribuições?

**Distribuições Convencionais** - são distribuídas da forma tradicional, ou seja, uma ou mais mídias que são utilizadas para instalar o sistema no disco rígido;

**Distribuições Live** - são distribuídas em mídias com o intuito de rodarem a partir delas, sem a necessidade de instalar no HD.

# O que são Distribuições?

**Distribuições Convencionais** - são distribuídas da forma tradicional, ou seja, uma ou mais mídias que são utilizadas para instalar o sistema no disco rígido;

**Distribuições Live** - são distribuídas em mídias com o intuito de rodarem a partir delas, sem a necessidade de instalar no HD.



# O que são Distribuições?

Ainda para entender um pouco mais das distribuições, é necessário lembrar de mais duas características: **From scratch** e **Provenientes (Baseadas)**

# O que são Distribuições?

Ainda para entender um pouco mais das distribuições, é necessário lembrar de mais duas características: **From scratch** e **Provenientes (Baseadas)**

**Distribuições From Scratch** - São desenvolvidas do zero, ou seja, utiliza um kernel linux, alguns programas GNU e a grande maioria das suas particularidades é desenvolvida especificamete para ela. Exemplos: Debian, RedHat, Gentoo, Slackware etc.

# O que são Distribuições?

Ainda para entender um pouco mais das distribuições, é necessário lembrar de mais duas características: **From scratch** e **Provenientes (Baseadas)**

**Distribuições From Scratch** - São desenvolvidas do zero, ou seja, utiliza um kernel linux, alguns programas GNU e a grande maioria das suas particularidades é desenvolvida especificamente para ela. Exemplos: Debian, RedHat, Gentoo, Slackware etc.

**Distribuições Provenientes (Baseadas)** - Aproveitam ferramentas e bases já desenvolvidas por outras distribuições. Distribuições baseadas usam distribuições from scratch para alcançar seus objetivos mais rápido, dando maior atenção para ao propósito da distribuição. Exemplos: Ubuntu, Fedora, Kubuntu, Mint, etc.



Tipo: corporativa;





Tipo: corporativa;

Descrição: primeira distribuição corporativa a ser criada. Muito utilizada nas empresas por oferecer suporte técnico e ter seu sistema compatível com as diversas tecnologias disponíveis;



Tipo: corporativa;

Descrição: primeira distribuição corporativa a ser criada. Muito utilizada nas empresas por oferecer suporte técnico e ter seu sistema compatível com as diversas tecnologias disponíveis;

Interface padrão: GNOME;



Tipo: corporativa;

Descrição: primeira distribuição corporativa a ser criada. Muito utilizada nas empresas por oferecer suporte técnico e ter seu sistema compatível com as diversas tecnologias disponíveis;

Interface padrão: GNOME;

Sistema de pacote: RPM - RedHat Package Manager;



Tipo: corporativa;

Descrição: primeira distribuição corporativa a ser criada. Muito utilizada nas empresas por oferecer suporte técnico e ter seu sistema compatível com as diversas tecnologias disponíveis;

Interface padrão: GNOME;

Sistema de pacote: RPM - RedHat Package Manager;

Site oficial: <http://www.redhat.com>

# Algumas Distribuições...





Tipo: corporativa;



Tipo: corporativa;

Descrição: é a principal concorrente da RedHat, atuando no meio corporativo tanto em servidores quanto em desktops. Assim como a RedHat, possui parcerias com diversas empresas, a fim de manter seu sistema compatível com produtos de terceiros;



Tipo: corporativa;

Descrição: é a principal concorrente da RedHat, atuando no meio corporativo tanto em servidores quanto em desktops. Assim como a RedHat, possui parcerias com diversas empresas, a fim de manter seu sistema compatível com produtos de terceiros;

Interface padrão: GNOME;





Tipo: corporativa;

Descrição: é a principal concorrente da RedHat, atuando no meio corporativo tanto em servidores quanto em desktops. Assim como a RedHat, possui parcerias com diversas empresas, a fim de manter seu sistema compatível com produtos de terceiros;

Interface padrão: GNOME;

Sistema de pacote: baseado em RPM, mas não segue o formato da RedHat à risca, tendo implementado algumas variações;

# Algumas Distribuições...



Tipo: corporativa;

Descrição: é a principal concorrente da RedHat, atuando no meio corporativo tanto em servidores quanto em desktops. Assim como a RedHat, possui parcerias com diversas empresas, a fim de manter seu sistema compatível com produtos de terceiros;

Interface padrão: GNOME;

Sistema de pacote: baseado em RPM, mas não segue o formato da RedHat à risca, tendo implementado algumas variações;

Site oficial: <http://www.suse.com>





Distribuição: livre;



Distribuição: livre;

Descrição: criada com o intuito de prover um sistema operacional totalmente livre e gratuito, foi uma das primeiras distribuições GNU/Linux a serem criadas. Atualmente é uma das maiores distribuições e a que mais gerou distribuições derivadas. Por ser uma referência em sistemas GNU/Linux, é a distribuição mais utilizada em órgãos públicos e governos;



Distribuição: livre;

Descrição: criada com o intuito de prover um sistema operacional totalmente livre e gratuito, foi uma das primeiras distribuições GNU/Linux a serem criadas. Atualmente é uma das maiores distribuições e a que mais gerou distribuições derivadas. Por ser uma referência em sistemas GNU/Linux, é a distribuição mais utilizada em órgãos públicos e governos;

Interface padrão: GNOME;



Distribuição: livre;

Descrição: criada com o intuito de prover um sistema operacional totalmente livre e gratuito, foi uma das primeiras distribuições GNU/Linux a serem criadas. Atualmente é uma das maiores distribuições e a que mais gerou distribuições derivadas. Por ser uma referência em sistemas GNU/Linux, é a distribuição mais utilizada em órgãos públicos e governos;

Interface padrão: GNOME;

Sistema de pacote: DEB - Debian Package;



Distribuição: livre;

Descrição: criada com o intuito de prover um sistema operacional totalmente livre e gratuito, foi uma das primeiras distribuições GNU/Linux a serem criadas. Atualmente é uma das maiores distribuições e a que mais gerou distribuições derivadas. Por ser uma referência em sistemas GNU/Linux, é a distribuição mais utilizada em órgãos públicos e governos;

Interface padrão: GNOME;

Sistema de pacote: DEB - Debian Package;

Site oficial: <http://www.debian.org>







Distribuição: livre (convencional e Live);



Distribuição: livre (convencional e Live);

Descrição: com seu slogan *Linux for Human Beings* - é voltada para o usuário final, apesar de ter versão para servidores. Patrocinada pelo milionário Mark Shuttleworth é, atualmente, a maior distribuição em número de downloads.



Distribuição: livre (convencional e Live);

Descrição: com seu slogan *Linux for Human Beings* - é voltada para o usuário final, apesar de ter versão para servidores. Patrocinada pelo milionário Mark Shuttleworth é, atualmente, a maior distribuição em número de downloads.

Interface padrão: GNOME ou KDE (para Kubuntu);



Distribuição: livre (convencional e Live);

Descrição: com seu slogan *Linux for Human Beings* - é voltada para o usuário final, apesar de ter versão para servidores. Patrocinada pelo milionário Mark Shuttleworth é, atualmente, a maior distribuição em número de downloads.

Interface padrão: GNOME ou KDE (para Kubuntu);

Sistema de pacote: DEB - Debian Package;



Distribuição: livre (convencional e Live);

Descrição: com seu slogan *Linux for Human Beings* - é voltada para o usuário final, apesar de ter versão para servidores. Patrocinada pelo milionário Mark Shuttleworth é, atualmente, a maior distribuição em número de downloads.

Interface padrão: GNOME ou KDE (para Kubuntu);

Sistema de pacote: DEB - Debian Package;

Site oficial: <http://www.ubuntu.com>





Distribuição: livre;





Distribuição: livre;

Descrição: mantida pela RedHat, serve de teste para o carro chefe da empresa, o RedHat Enterprise.



Distribuição: livre;

Descrição: mantida pela RedHat, serve de teste para o carro chefe da empresa, o RedHat Enterprise.

Interface padrão: GNOME;



Distribuição: livre;

Descrição: mantida pela RedHat, serve de teste para o carro chefe da empresa, o RedHat Enterprise.

Interface padrão: GNOME;

Sistema de pacote: RPM - RedHat Package Manager;



Distribuição: livre;

Descrição: mantida pela RedHat, serve de teste para o carro chefe da empresa, o RedHat Enterprise.

Interface padrão: GNOME;

Sistema de pacote: RPM - RedHat Package Manager;

Site oficial: <http://fedora.redhat.com>



gentoo linux™

Distribuição: livre (Live);



gentoo linux™

Distribuição: livre (Live);

Descrição: Todos os programas são compilados na própria máquina. As principais vantagens são a performance e a personalização conforme as necessidades do usuário. A principal desvantagem é o trabalho e tempo necessários a sua instalação.



Distribuição: livre (Live);

Descrição: Todos os programas são compilados na própria máquina. As principais vantagens são a performance e a personalização conforme as necessidades do usuário. A principal desvantagem é o trabalho e tempo necessários a sua instalação.

Interface padrão: A escolha do usuário;



Distribuição: livre (Live);

Descrição: Todos os programas são compilados na própria máquina. As principais vantagens são a performance e a personalização conforme as necessidades do usuário. A principal desvantagem é o trabalho e tempo necessários a sua instalação.

Interface padrão: A escolha do usuário;

Sistema de pacote: Emerge, código fonte;





Distribuição: livre (Live);

Descrição: Todos os programas são compilados na própria máquina. As principais vantagens são a performance e a personalização conforme as necessidades do usuário. A principal desvantagem é o trabalho e tempo necessários a sua instalação.

Interface padrão: A escolha do usuário;

Sistema de pacote: Emerge, código fonte;

Site oficial: <http://www.gentoo.org>

# Qual distribuição utilizar?

# Qual distribuição utilizar?

Testar distribuições Linux online: <https://distrotest.net/>



- Entender a estrutura do sistema operacional;

- Entender a estrutura do sistema operacional;
- Descobrir as funcionalidades do Shell;

- Entender a estrutura do sistema operacional;
- Descobrir as funcionalidades do Shell;
- Executar os primeiros comandos no sistema.

- Entender a estrutura do sistema operacional;
- Descobrir as funcionalidades do Shell;
- Executar os primeiros comandos no sistema.
- Entendendo a estrutura do Linux



# Primeiros Passos

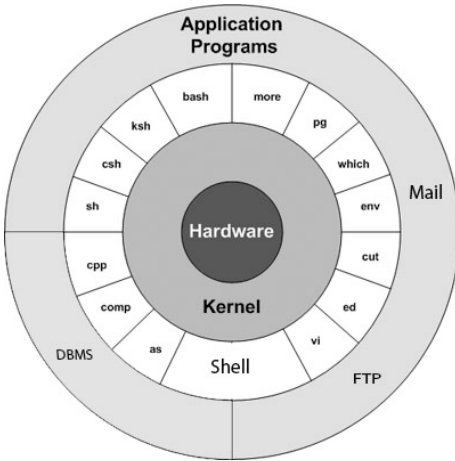


Figura: Arquitetura de uma Distribuição

- **Hardware** - tais como cd-rom, placa de rede, controlador de disco entre outros;

- **Hardware** - tais como cd-rom, placa de rede, controlador de disco entre outros;
- **Kernel** - O núcleo do sistema operacional, essa camada é quem faz todas as interações com o hardware da máquina, interpretando todas as requisições das camadas acima;

- **Hardware** - tais como cd-rom, placa de rede, controlador de disco entre outros;
- **Kernel** - O núcleo do sistema operacional, essa camada é quem faz todas as interações com o hardware da máquina, interpretando todas as requisições das camadas acima;
- **Shell** - é a interface entre o usuário e o kernel do sistema e por meio dele, podemos digitar os comandos;

- **Hardware** - tais como cd-rom, placa de rede, controlador de disco entre outros;
- **Kernel** - O núcleo do sistema operacional, essa camada é quem faz todas as interações com o hardware da máquina, interpretando todas as requisições das camadas acima;
- **Shell** - é a interface entre o usuário e o kernel do sistema e por meio dele, podemos digitar os comandos;
- **ttyN** - Terminais Virtuais aonde são executados comandos e setadas as configurações. As tty's interpretam os comandos dados por um humano e convertem os mesmos para uma linguagem que a máquina entenda;

O shell interpreta o usuário que irá efetuar uma ação de duas maneiras:

O shell interpreta o usuário que irá efetuar uma ação de duas maneiras:

- **Super usuário ou root** - O usuário root é o administrador do sistema, e seu diretório (pasta) padrão é o /root. Demais usuários que ficam dentro de /home. O shell de um usuário root é diferente de um usuário comum. Antes do cursor, ele é identificado com # (jogo-da-velha)

O shell interpreta o usuário que irá efetuar uma ação de duas maneiras:

- **Super usuário ou root** - O usuário root é o administrador do sistema, e seu diretório (pasta) padrão é o /root. Demais usuários que ficam dentro de /home. O shell de um usuário root é diferente de um usuário comum. Antes do cursor, ele é identificado com # (jogo-da-velha)
- **Usuário comum** - qualquer usuário do sistema que não seja root e não tenha poderes administrativos no sistema. Como já havíamos dito anteriormente, o diretório padrão para os usuários é o /home. Antes do cursor, o shell de um usuário comum é identificado com \$ (cifrão)



# Primeiros Passos

- As teclas seta para cima e seta para baixo, para navegar entre comandos digitados anteriormente;

- As teclas seta para cima e seta para baixo, para navegar entre comandos digitados anteriormente;
- Para rolarmos a tela para cima, seguramos o **Shift** e pressionamos o **Page Up**. Para rolarmos a tela para baixo, seguramos o **Shift** e pressionamos o *Page Down*. Isto é útil para ver textos que rolaram rapidamente para cima.

Acessando os terminais virtuais:

- Para acessar outros terminais virtuais, segure a tecla **Ctrl+ALT** e pressionando **F1** até **F6**;

Acessando os terminais virtuais:

- Para acessar outros terminais virtuais, segure a tecla **Ctrl+ALT** e pressionando **F1 até F6**;
- Cada tecla tem função correspondente a um número de terminal do 1 ao 6, isso é por default, e pode ser mudado (o sétimo, por default, é usado pelo ambiente gráfico - Xorg)

Acessando os terminais virtuais:

- Para acessar outros terminais virtuais, segure a tecla **Ctrl+ALT** e pressionando **F1 até F6**;
- Cada tecla tem função correspondente a um número de terminal do 1 ao 6, isso é por default, e pode ser mudado (o sétimo, por default, é usado pelo ambiente gráfico - Xorg)
- **Logon** é a entrada do usuário, root ou comum, onde deve ser digitado seu nome de usuário. Na sequência, digite sua senha. Caso você digite algo de forma errada, irá aparecer uma mensagem de erro e você não será logado no sistema.

Acessando os terminais virtuais:

- Para acessar outros terminais virtuais, segure a tecla **Ctrl+ALT** e pressionando **F1 até F6**;
- Cada tecla tem função correspondente a um número de terminal do 1 ao 6, isso é por default, e pode ser mudado (o sétimo, por default, é usado pelo ambiente gráfico - Xorg)
- **Logon** é a entrada do usuário, root ou comum, onde deve ser digitado seu nome de usuário. Na sequência, digite sua senha. Caso você digite algo de forma errada, irá aparecer uma mensagem de erro e você não será logado no sistema.
- O terminal do Linux permite que você guarde 500 comandos por padrão, assim não precisa redigitar o comando quando precisar dele novamente.

Acessando os terminais virtuais:

- Para acessar outros terminais virtuais, segure a tecla **Ctrl+ALT** e pressionando **F1 até F6**;
- Cada tecla tem função correspondente a um número de terminal do 1 ao 6, isso é por default, e pode ser mudado (o sétimo, por default, é usado pelo ambiente gráfico - Xorg)
- **Logon** é a entrada do usuário, root ou comum, onde deve ser digitado seu nome de usuário. Na sequência, digite sua senha. Caso você digite algo de forma errada, irá aparecer uma mensagem de erro e você não será logado no sistema.
- O terminal do Linux permite que você guarde 500 comandos por padrão, assim não precisa redigitar o comando quando precisar dele novamente.

**\$ history**



Acessando os terminais virtuais:

- **Logout** é a saída do sistema. Ela é feita pelos comandos:

Acessando os terminais virtuais:

- **Logout** é a saída do sistema. Ela é feita pelos comandos:  
**\$ logout**

Acessando os terminais virtuais:

- **Logout** é a saída do sistema. Ela é feita pelos comandos:  
\$ **logout**  
\$ **exit**

Acessando os terminais virtuais:

- **Logout** é a saída do sistema. Ela é feita pelos comandos:

**\$ logout**

**\$ exit**

**\$ Ctrl+D**

Acessando os terminais virtuais:

- **Logout** é a saída do sistema. Ela é feita pelos comandos:

**\$ logout**

**\$ exit**

**\$ Ctrl+D**

Desligando ou reiniciando o sistema.

Para desligar o computador, primeiro digite um dos comandos abaixo (como root):

```
# shutdown -h now
```

Para desligar o computador, primeiro digite um dos comandos abaixo (como root):

```
# shutdown -h now
```

```
# halt
```

Para desligar o computador, primeiro digite um dos comandos abaixo (como root):

```
# shutdown -h now
```

```
# halt
```

```
# poweroff
```



Para desligar o computador, primeiro digite um dos comandos abaixo (como root):

```
# shutdown -h now
```

```
# halt
```

```
# poweroff
```

A palavra **halt** vem do comando em assembly chamado **HTL**, que quer dizer “parada de processamento”.

Para desligar o computador, primeiro digite um dos comandos abaixo (como root):

```
# shutdown -h now
```

```
# halt
```

```
# poweroff
```

A palavra **halt** vem do comando em assembly chamado **HTL**, que quer dizer “parada de processamento”.

Os comandos **halt** e **poweroff** disparam uma série de procedimentos, como encerramento de serviços e desligamento de sistemas de arquivos, que são executados antes da máquina ser desligada.

Para desligar o computador, primeiro digite um dos comandos abaixo (como root):

```
# shutdown -h now
```

```
# halt
```

```
# poweroff
```

A palavra **halt** vem do comando em assembly chamado **HTL**, que quer dizer “parada de processamento”.

Os comandos **halt** e **poweroff** disparam uma série de procedimentos, como encerramento de serviços e desligamento de sistemas de arquivos, que são executados antes da máquina ser desligada.

O comando **shutdown** tem a seguinte sintaxe:

O comando **shutdown** tem a seguinte sintaxe:  
**# shutdown ação tempo**

O comando **shutdown** tem a seguinte sintaxe:

**# shutdown ação tempo**

Onde:

O comando **shutdown** tem a seguinte sintaxe:

**# shutdown ação tempo**

Onde: ação - o que você quer fazer, As opções são:

O comando **shutdown** tem a seguinte sintaxe:

**# shutdown ação tempo**

Onde: ação - o que você quer fazer, As opções são:

**-h** para desligar



O comando **shutdown** tem a seguinte sintaxe:

**# shutdown ação tempo**

Onde: ação - o que você quer fazer, As opções são:

**-h** para desligar

**-r** para reiniciar.

O comando **shutdown** tem a seguinte sintaxe:

**# shutdown ação tempo**

Onde: ação - o que você quer fazer, As opções são:

**-h** para desligar

**-r** para reiniciar.

**tempo** - tempo em **minutos** que você deseja para começar a executar a ação.

Exemplo:

# Primeiros Passos

Exemplo:

Desligar agora:

Exemplo:

Desligar agora:

```
# shutdown -h now
```

# Primeiros Passos

Exemplo:

Desligar agora:

```
# shutdown -h now
```

Desligar daqui a 12 minutos

Exemplo:

Desligar agora:

```
# shutdown -h now
```

Desligar daqui a 12 minutos

```
# shutdown -h 12
```

Exemplo:

Desligar agora:

```
# shutdown -h now
```

Desligar daqui a 12 minutos

```
# shutdown -h 12
```

Reiniciar agora:



Exemplo:

Desligar agora:

```
# shutdown -h now
```

Desligar daqui a 12 minutos

```
# shutdown -h 12
```

Reiniciar agora:

```
# shutdown -r now
```

Exemplo:

Desligar agora:

```
# shutdown -h now
```

Desligar daqui a 12 minutos

```
# shutdown -h 12
```

Reiniciar agora:

```
# shutdown -r now
```

Reiniciar daqui a 5 minutos:

Exemplo:

Desligar agora:

```
# shutdown -h now
```

Desligar daqui a 12 minutos

```
# shutdown -h 12
```

Reiniciar agora:

```
# shutdown -r now
```

Reiniciar daqui a 5 minutos:

```
# shutdown -r 5
```

Exemplo:

Desligar agora:

```
# shutdown -h now
```

Desligar daqui a 12 minutos

```
# shutdown -h 12
```

Reiniciar agora:

```
# shutdown -r now
```

Reiniciar daqui a 5 minutos:

```
# shutdown -r 5
```

Pressione a tecla:

Pressione a tecla:

**Back Space** para apagar um caractere à esquerda do cursor;

Pressione a tecla:

**Back Space** para apagar um caractere à esquerda do cursor;

**Delete** para apagar o caractere acima do cursor;

Pressione a tecla:

**Back Space** para apagar um caractere à esquerda do cursor;

**Delete** para apagar o caractere acima do cursor;

**Home** para ir ao começo da linha de comando;



Pressione a tecla:

**Back Space** para apagar um caractere à esquerda do cursor;

**Delete** para apagar o caractere acima do cursor;

**Home** para ir ao começo da linha de comando;

**End** para ir ao final da linha de comando;

Pressione a tecla:

**Back Space** para apagar um caractere à esquerda do cursor;

**Delete** para apagar o caractere acima do cursor;

**Home** para ir ao começo da linha de comando;

**End** para ir ao final da linha de comando;

**Ctrl + A** para mover o cursor para o início da linha de comandos;

Pressione a tecla:

**Back Space** para apagar um caractere à esquerda do cursor;

**Delete** para apagar o caractere acima do cursor;

**Home** para ir ao começo da linha de comando;

**End** para ir ao final da linha de comando;

**Ctrl + A** para mover o cursor para o início da linha de comandos;

**Ctrl + E** para mover o cursor para o fim da linha de comandos;

# Primeiros Passos

Pressione a tecla(s):

Pressione a tecla(s):

**Ctrl + U** para apagar o que estiver à esquerda do cursor. O conteúdo apagado é copiado para uso com **Ctrl + y**;

Pressione a tecla(s):

**Ctrl + U** para apagar o que estiver à esquerda do cursor. O conteúdo apagado é copiado para uso com **Ctrl + y**;

**Ctrl + K** para apagar o que estiver à direita do cursor. O conteúdo apagado é copiado para uso com **Ctrl + y**;

# Primeiros Passos

Pressione a tecla(s):

**Ctrl + U** para apagar o que estiver à esquerda do cursor. O conteúdo apagado é copiado para uso com **Ctrl + y**;

**Ctrl + K** para apagar o que estiver à direita do cursor. O conteúdo apagado é copiado para uso com **Ctrl + y**;

**Ctrl + L** para limpar a tela e manter a linha de comando na primeira linha. Se você der um **Shift + Page Up** você ainda consegue enxergar o conteúdo. O **Ctrl + L** funciona igual ao comando **clear**, que tem a mesma função;

# Primeiros Passos

Pressione a tecla(s):

**Ctrl + U** para apagar o que estiver à esquerda do cursor. O conteúdo apagado é copiado para uso com **Ctrl + y**;

**Ctrl + K** para apagar o que estiver à direita do cursor. O conteúdo apagado é copiado para uso com **Ctrl + y**;

**Ctrl + L** para limpar a tela e manter a linha de comando na primeira linha. Se você der um **Shift + Page Up** você ainda consegue enxergar o conteúdo. O **Ctrl + L** funciona igual ao comando **clear**, que tem a mesma função;

**Ctrl + C** para abrir uma nova linha de comando, na posição atual do cursor;



# Primeiros Passos

Pressione a tecla(s):

**Ctrl + U** para apagar o que estiver à esquerda do cursor. O conteúdo apagado é copiado para uso com **Ctrl + y**;

**Ctrl + K** para apagar o que estiver à direita do cursor. O conteúdo apagado é copiado para uso com **Ctrl + y**;

**Ctrl + L** para limpar a tela e manter a linha de comando na primeira linha. Se você der um **Shift + Page Up** você ainda consegue enxergar o conteúdo. O **Ctrl + L** funciona igual ao comando **clear**, que tem a mesma função;

**Ctrl + C** para abrir uma nova linha de comando, na posição atual do cursor;

**Ctrl + D** para sair do shell. Este é equivalente ao comando **exit**;

# Primeiros Passos

Pressione a tecla(s):

**Ctrl + U** para apagar o que estiver à esquerda do cursor. O conteúdo apagado é copiado para uso com **Ctrl + y**;

**Ctrl + K** para apagar o que estiver à direita do cursor. O conteúdo apagado é copiado para uso com **Ctrl + y**;

**Ctrl + L** para limpar a tela e manter a linha de comando na primeira linha. Se você der um **Shift + Page Up** você ainda consegue enxergar o conteúdo. O **Ctrl + L** funciona igual ao comando **clear**, que tem a mesma função;

**Ctrl + C** para abrir uma nova linha de comando, na posição atual do cursor;

**Ctrl + D** para sair do shell. Este é equivalente ao comando **exit**;

**Ctrl + R** para procurar x letra relacionada ao último comando digitado que tinha x letra como conteúdo do comando;



## Objetivos

## Objetivos

- Entender o que é FHS;

## Objetivos

- Entender o que é FHS;
- Conhecer a estrutura de diretórios do sistema;

## Objetivos

- Entender o que é FHS;
- Conhecer a estrutura de diretórios do sistema;
- Descobrir alguns diretórios e suas determinadas finalidades.

## Objetivos

- Entender o que é FHS;
- Conhecer a estrutura de diretórios do sistema;
- Descobrir alguns diretórios e suas determinadas finalidades.



- **FHS - Filesystem Hierarchy Standard, ou Hierarquia Padrão do Sistema de Arquivos** - define que tipo de arquivo deve ser guardado em cada diretório;

- **FHS - Filesystem Hierarchy Standard, ou Hierarquia Padrão do Sistema de Arquivos** - define que tipo de arquivo deve ser guardado em cada diretório;
- Importante para manter a compatibilidade entre as diferentes distribuições e aplicações desenvolvidas no padrão FHS;

- **FHS - Filesystem Hierarchy Standard, ou Hierarquia Padrão do Sistema de Arquivos** - define que tipo de arquivo deve ser guardado em cada diretório;
- Importante para manter a compatibilidade entre as diferentes distribuições e aplicações desenvolvidas no padrão FHS;
- Para conhecer o documento detalhado:  
<http://www.pathname.com/fhs>

- **FHS - Filesystem Hierarchy Standard, ou Hierarquia Padrão do Sistema de Arquivos** - define que tipo de arquivo deve ser guardado em cada diretório;
- Importante para manter a compatibilidade entre as diferentes distribuições e aplicações desenvolvidas no padrão FHS;
- Para conhecer o documento detalhado:  
<http://www.pathname.com/fhs>

A árvore de diretórios do GNU/Linux tem a seguinte estrutura:

A árvore de diretórios do GNU/Linux tem a seguinte estrutura:

/

A árvore de diretórios do GNU/Linux tem a seguinte estrutura:

```
/
bin cdrom etc lib mnt proc root var sys
```

A árvore de diretórios do GNU/Linux tem a seguinte estrutura:

```
/
bin cdrom etc lib mnt proc root var sys
boot dev home media opt sbin srv tmp usr
```



A árvore de diretórios do GNU/Linux tem a seguinte estrutura:

```
/
bin cdrom etc lib mnt proc root var sys
boot dev home media opt sbin srv tmp usr
```

# Sistema de Arquivos e Diretórios

O FHS determina que um sistema GNU/Linux deve conter obrigatoriamente 14 diretórios:

# Sistema de Arquivos e Diretórios

O FHS determina que um sistema GNU/Linux deve conter obrigatoriamente 14 diretórios:

- / (**raiz**) - É no diretório raiz que ficam todos os demais diretórios do sistema;

# Sistema de Arquivos e Diretórios

O FHS determina que um sistema GNU/Linux deve conter obrigatoriamente 14 diretórios:

- **/ (raiz)** - É no diretório raiz que ficam todos os demais diretórios do sistema;
- **/bin** - Guarda os comandos essenciais para o funcionamento do sistema;

# Sistema de Arquivos e Diretórios

O FHS determina que um sistema GNU/Linux deve conter obrigatoriamente 14 diretórios:

- **/ (raiz)** - É no diretório raiz que ficam todos os demais diretórios do sistema;
- **/bin** - Guarda os comandos essenciais para o funcionamento do sistema;
- **/boot** - Estão os arquivos estáticos necessários à inicialização do sistema, e o gerenciador de boot;

# Sistema de Arquivos e Diretórios

O FHS determina que um sistema GNU/Linux deve conter obrigatoriamente 14 diretórios:

- **/ (raiz)** - É no diretório raiz que ficam todos os demais diretórios do sistema;
- **/bin** - Guarda os comandos essenciais para o funcionamento do sistema;
- **/boot** - Estão os arquivos estáticos necessários à inicialização do sistema, e o gerenciador de boot;
- **/dev** - Ficam todos os arquivos de dispositivos. O Linux faz a comunicação com os periféricos por meio de links especiais;

# Sistema de Arquivos e Diretórios

O FHS determina que um sistema GNU/Linux deve conter obrigatoriamente 14 diretórios:

- **/ (raiz)** - É no diretório raiz que ficam todos os demais diretórios do sistema;
- **/bin** - Guarda os comandos essenciais para o funcionamento do sistema;
- **/boot** - Estão os arquivos estáticos necessários à inicialização do sistema, e o gerenciador de boot;
- **/dev** - Ficam todos os arquivos de dispositivos. O Linux faz a comunicação com os periféricos por meio de links especiais;

# Sistema de Arquivos e Diretórios

- `/etc` - Arquivos de configuração, tais como: scripts de inicialização do sistema, tabela do sistema de arquivos, etc.;



# Sistema de Arquivos e Diretórios

- **/etc** - Arquivos de configuração, tais como: scripts de inicialização do sistema, tabela do sistema de arquivos, etc.;
- **/lib** - Bibliotecas compartilhadas e módulos do kernel;

# Sistema de Arquivos e Diretórios

- **/etc** - Arquivos de configuração, tais como: scripts de inicialização do sistema, tabela do sistema de arquivos, etc.;
- **/lib** - Bibliotecas compartilhadas e módulos do kernel;
- **/media** - Ponto de montagem para dispositivos removíveis, tais como: cd, dvd, pendrive, etc.;

# Sistema de Arquivos e Diretórios

- **/etc** - Arquivos de configuração, tais como: scripts de inicialização do sistema, tabela do sistema de arquivos, etc.;
- **/lib** - Bibliotecas compartilhadas e módulos do kernel;
- **/media** - Ponto de montagem para dispositivos removíveis, tais como: cd, dvd, pendrive, etc.;
- **/mnt** - é utilizado para montagem temporária de sistemas de arquivos, tais como compartilhamentos de arquivos entre Windows e Linux, Linux e Linux, etc.;

# Sistema de Arquivos e Diretórios

- **/etc** - Arquivos de configuração, tais como: scripts de inicialização do sistema, tabela do sistema de arquivos, etc.;
- **/lib** - Bibliotecas compartilhadas e módulos do kernel;
- **/media** - Ponto de montagem para dispositivos removíveis, tais como: cd, dvd, pendrive, etc.;
- **/mnt** - é utilizado para montagem temporária de sistemas de arquivos, tais como compartilhamentos de arquivos entre Windows e Linux, Linux e Linux, etc.;
- **/opt** - é utilizado por programas proprietários ou que não fazem parte oficialmente da distribuição;

# Sistema de Arquivos e Diretórios

- **/etc** - Arquivos de configuração, tais como: scripts de inicialização do sistema, tabela do sistema de arquivos, etc.;
- **/lib** - Bibliotecas compartilhadas e módulos do kernel;
- **/media** - Ponto de montagem para dispositivos removíveis, tais como: cd, dvd, pendrive, etc.;
- **/mnt** - é utilizado para montagem temporária de sistemas de arquivos, tais como compartilhamentos de arquivos entre Windows e Linux, Linux e Linux, etc.;
- **/opt** - é utilizado por programas proprietários ou que não fazem parte oficialmente da distribuição;
- **/sbin** - guarda os comandos utilizados para inicializar, reparar, restaurar e/ou recuperar o sistema;

# Sistema de Arquivos e Diretórios

- **/etc** - Arquivos de configuração, tais como: scripts de inicialização do sistema, tabela do sistema de arquivos, etc.;
- **/lib** - Bibliotecas compartilhadas e módulos do kernel;
- **/media** - Ponto de montagem para dispositivos removíveis, tais como: cd, dvd, pendrive, etc.;
- **/mnt** - é utilizado para montagem temporária de sistemas de arquivos, tais como compartilhamentos de arquivos entre Windows e Linux, Linux e Linux, etc.;
- **/opt** - é utilizado por programas proprietários ou que não fazem parte oficialmente da distribuição;
- **/sbin** - guarda os comandos utilizados para inicializar, reparar, restaurar e/ou recuperar o sistema;

# Sistema de Arquivos e Diretórios

- **/sys** - Podemos encontrar o quase o mesmo conteúdo do proc, mas de uma forma bem mais organizada para nós administradores;

# Sistema de Arquivos e Diretórios

- **/sys** - Podemos encontrar o quase o mesmo conteúdo do `proc`, mas de uma forma bem mais organizada para nós administradores;
- **/home** - Contém os diretórios pessoais dos usuários cadastrados no sistema.;



# Sistema de Arquivos e Diretórios

- **/sys** - Podemos encontrar o quase o mesmo conteúdo do proc, mas de uma forma bem mais organizada para nós administradores;
- **/home** - Contém os diretórios pessoais dos usuários cadastrados no sistema.;
- **/root** - Diretório pessoal do superusuário root;

# Sistema de Arquivos e Diretórios

- **/sys** - Podemos encontrar o quase o mesmo conteúdo do proc, mas de uma forma bem mais organizada para nós administradores;
- **/home** - Contém os diretórios pessoais dos usuários cadastrados no sistema.;
- **/root** - Diretório pessoal do superusuário root;
- **/var** - O diretório /var contém arquivos de dados variáveis;

# Sistema de Arquivos e Diretórios

- **/sys** - Podemos encontrar o quase o mesmo conteúdo do proc, mas de uma forma bem mais organizada para nós administradores;
- **/home** - Contém os diretórios pessoais dos usuários cadastrados no sistema.;
- **/root** - Diretório pessoal do superusuário root;
- **/var** - O diretório /var contém arquivos de dados variáveis;
- **/opt** - é utilizado por programas proprietários ou que não fazem parte oficialmente da distribuição;

# Sistema de Arquivos e Diretórios

- **/sys** - Podemos encontrar o quase o mesmo conteúdo do proc, mas de uma forma bem mais organizada para nós administradores;
- **/home** - Contém os diretórios pessoais dos usuários cadastrados no sistema.;
- **/root** - Diretório pessoal do superusuário root;
- **/var** - O diretório /var contém arquivos de dados variáveis;
- **/opt** - é utilizado por programas proprietários ou que não fazem parte oficialmente da distribuição;
- **/proc** - é um diretório virtual, mantido pelo kernel, onde encontramos a configuração atual do sistema, dados estatísticos, dispositivos já montados, interrupções, endereços e estados das portas físicas, dados sobre as redes, etc.;

- **/srv** - Diretório para dados de serviços fornecidos pelo sistema cuja aplicação é de alcance geral, ou seja, os dados não são específicos de um usuário;

- **/srv** - Diretório para dados de serviços fornecidos pelo sistema cuja aplicação é de alcance geral, ou seja, os dados não são específicos de um usuário;
- **/tmp** - Diretório para armazenamento de arquivos temporários;

- **/srv** - Diretório para dados de serviços fornecidos pelo sistema cuja aplicação é de alcance geral, ou seja, os dados não são específicos de um usuário;
- **/tmp** - Diretório para armazenamento de arquivos temporários;
- **/usr** - O diretório `/usr` contém programas que não são essenciais ao sistema e que seguem o padrão GNU/Linux, como, por exemplo, navegadores, gerenciadores de janelas, etc.;

# Navegando na Árvore de Diretórios

- O comando **pwd** exibe o diretório corrente. Ele é muito útil quando estamos navegando pelo sistema e não lembramos qual o diretório atual;



# Navegando na Árvore de Diretórios

- O comando **pwd** exibe o diretório corrente. Ele é muito útil quando estamos navegando pelo sistema e não lembramos qual o diretório atual;

Exemplo:

# Navegando na Árvore de Diretórios

- O comando **pwd** exibe o diretório corrente. Ele é muito útil quando estamos navegando pelo sistema e não lembramos qual o diretório atual;

Exemplo:

```
$ pwd
```

# Navegando na Árvore de Diretórios

- O comando **pwd** exibe o diretório corrente. Ele é muito útil quando estamos navegando pelo sistema e não lembramos qual o diretório atual;

Exemplo:

```
$ pwd
```

- O comando **cd** é utilizado para mudar o diretório atual de onde o usuário está.

# Navegando na Árvore de Diretórios

- O comando **pwd** exibe o diretório corrente. Ele é muito útil quando estamos navegando pelo sistema e não lembramos qual o diretório atual;

Exemplo:

```
$ pwd
```

- O comando **cd** é utilizado para mudar o diretório atual de onde o usuário está.

Exemplo:

# Navegando na Árvore de Diretórios

- O comando **pwd** exibe o diretório corrente. Ele é muito útil quando estamos navegando pelo sistema e não lembramos qual o diretório atual;

Exemplo:

```
$ pwd
```

- O comando **cd** é utilizado para mudar o diretório atual de onde o usuário está.

Exemplo:

Ir para o diretório home do usuário logado:

# Navegando na Árvore de Diretórios

- O comando **pwd** exibe o diretório corrente. Ele é muito útil quando estamos navegando pelo sistema e não lembramos qual o diretório atual;

Exemplo:

```
$ pwd
```

- O comando **cd** é utilizado para mudar o diretório atual de onde o usuário está.

Exemplo:

Ir para o diretório home do usuário logado:

```
$ cd
```

# Navegando na Árvore de Diretórios

- O comando **pwd** exibe o diretório corrente. Ele é muito útil quando estamos navegando pelo sistema e não lembramos qual o diretório atual;

Exemplo:

```
$ pwd
```

- O comando **cd** é utilizado para mudar o diretório atual de onde o usuário está.

Exemplo:

Ir para o diretório home do usuário logado:

```
$ cd
```

```
$ cd ~
```

# Navegando na Árvore de Diretórios

Ir para o início da árvore de diretórios, ou seja, o diretório / :



# Navegando na Árvore de Diretórios

Ir para o início da árvore de diretórios, ou seja, o diretório / :

```
$ cd /
```

# Navegando na Árvore de Diretórios

Ir para o início da árvore de diretórios, ou seja, o diretório / :

**\$ cd /**

Ir para um diretório específico:

# Navegando na Árvore de Diretórios

Ir para o início da árvore de diretórios, ou seja, o diretório / :

```
$ cd /
```

Ir para um diretório específico:

```
# cd /etc
```

# Navegando na Árvore de Diretórios

Ir para o início da árvore de diretórios, ou seja, o diretório / :

**\$ cd /**

Ir para um diretório específico:

**# cd /etc**

Sobe um nível na árvore de diretórios:

# Navegando na Árvore de Diretórios

Ir para o início da árvore de diretórios, ou seja, o diretório / :

```
$ cd /
```

Ir para um diretório específico:

```
# cd /etc
```

Sobe um nível na árvore de diretórios:

```
# cd ..
```

# Navegando na Árvore de Diretórios

Ir para o início da árvore de diretórios, ou seja, o diretório / :

```
$ cd /
```

Ir para um diretório específico:

```
# cd /etc
```

Sobe um nível na árvore de diretórios:

```
# cd ..
```

Retorna ao diretório anterior:

# Navegando na Árvore de Diretórios

Ir para o início da árvore de diretórios, ou seja, o diretório / :

```
$ cd /
```

Ir para um diretório específico:

```
# cd /etc
```

Sobe um nível na árvore de diretórios:

```
# cd ..
```

Retorna ao diretório anterior:

```
# cd -
```

# Navegando na Árvore de Diretórios

Ir para o início da árvore de diretórios, ou seja, o diretório / :

```
$ cd /
```

Ir para um diretório específico:

```
# cd /etc
```

Sobe um nível na árvore de diretórios:

```
# cd ..
```

Retorna ao diretório anterior:

```
# cd -
```

```
# ls
```



# Navegando na Árvore de Diretórios

Ir para o início da árvore de diretórios, ou seja, o diretório / :

```
$ cd /
```

Ir para um diretório específico:

```
# cd /etc
```

Sobe um nível na árvore de diretórios:

```
# cd ..
```

Retorna ao diretório anterior:

```
# cd -
```

```
# ls
```

# Navegando na Árvore de Diretórios

Entra em um diretório específico:

# Navegando na Árvore de Diretórios

Entra em um diretório específico:

```
# cd /usr/include/X11
```

# Navegando na Árvore de Diretórios

Entra em um diretório específico:

```
# cd /usr/include/X11
```

Sobe 2 níveis da árvore de diretórios:

# Navegando na Árvore de Diretórios

Entra em um diretório específico:

```
# cd /usr/include/X11
```

Sobe 2 níveis da árvore de diretórios:

```
# cd ../../
```

# Navegando na Árvore de Diretórios

Entra em um diretório específico:

```
# cd /usr/include/X11
```

Sobe 2 níveis da árvore de diretórios:

```
# cd ../../
```

**Atenção!** Note a diferença entre caminhos absolutos e relativos:

# Navegando na Árvore de Diretórios

Entra em um diretório específico:

```
# cd /usr/include/X11
```

Sobe 2 níveis da árvore de diretórios:

```
# cd ../../
```

**Atenção!** Note a diferença entre caminhos absolutos e relativos:

**Absolutos:** /etc/ppp; /usr/share/doc; /lib/modules;

# Navegando na Árvore de Diretórios

Entra em um diretório específico:

```
# cd /usr/include/X11
```

Sobe 2 níveis da árvore de diretórios:

```
# cd ../../
```

**Atenção!** Note a diferença entre caminhos absolutos e relativos:

**Absolutos:** /etc/ppp; /usr/share/doc; /lib/modules;

**Relativos:** etc/ppp; ../doc; ../../usr;



# Navegando na Árvore de Diretórios

Entra em um diretório específico:

```
# cd /usr/include/X11
```

Sobe 2 níveis da árvore de diretórios:

```
# cd ../../
```

**Atenção!** Note a diferença entre caminhos absolutos e relativos:

**Absolutos:** /etc/ppp; /usr/share/doc; /lib/modules;

**Relativos:** etc/ppp; ../doc; ../../usr;



# Manipulando arquivos e diretórios

Objetivos:

# Manipulando arquivos e diretórios

Objetivos:

**Listar diretórios;**

# Manipulando arquivos e diretórios

Objetivos:

**Listar diretórios;**

**Criar e remover arquivos;**

# Manipulando arquivos e diretórios

Objetivos:

**Listar diretórios;**

**Criar e remover arquivos;**

**Criar e remover diretórios.**

# Manipulando arquivos e diretórios

Lista o conteúdo do diretório atual:

# Manipulando arquivos e diretórios

Lista o conteúdo do diretório atual:

**# ls**



# Manipulando arquivos e diretórios

Lista o conteúdo do diretório atual:

# **ls**

O comando **ls** possui muitos parâmetros, veremos aqui as opções mais utilizadas.

# Manipulando arquivos e diretórios

Lista o conteúdo do diretório atual:

# **ls**

O comando **ls** possui muitos parâmetros, veremos aqui as opções mais utilizadas.

A primeira dela é o **-l** que lista os arquivos ou diretórios de uma forma bem detalhada (quem criou, data de criação, tamanho, dono e grupo a qual eles pertencem).

# Manipulando arquivos e diretórios

Lista o conteúdo do diretório atual:

```
# ls
```

O comando **ls** possui muitos parâmetros, veremos aqui as opções mais utilizadas.

A primeira dela é o **-l** que lista os arquivos ou diretórios de uma forma bem detalhada (quem criou, data de criação, tamanho, dono e grupo a qual eles pertencem).

```
# ls -l /
```

# Manipulando arquivos e diretórios

Lista o conteúdo do diretório atual:

```
# ls
```

O comando **ls** possui muitos parâmetros, veremos aqui as opções mais utilizadas.

A primeira dela é o **-l** que lista os arquivos ou diretórios de uma forma bem detalhada (quem criou, data de criação, tamanho, dono e grupo a qual eles pertencem).

```
# ls -l /
```

```
drwxr-xr-x4 root root 1024 2019-10-15 23:17 boot
```

# Manipulando arquivos e diretórios

Lista o conteúdo do diretório atual:

```
# ls
```

O comando **ls** possui muitos parâmetros, veremos aqui as opções mais utilizadas.

A primeira dela é o **-l** que lista os arquivos ou diretórios de uma forma bem detalhada (quem criou, data de criação, tamanho, dono e grupo a qual eles pertencem).

```
# ls -l /
```

```
drwxr-xr-x4 root root 1024 2019-10-15 23:17 boot
```

**d** – indica que se trata de um diretório

# Manipulando arquivos e diretórios

Lista o conteúdo do diretório atual:

```
# ls
```

O comando **ls** possui muitos parâmetros, veremos aqui as opções mais utilizadas.

A primeira dela é o **-l** que lista os arquivos ou diretórios de uma forma bem detalhada (quem criou, data de criação, tamanho, dono e grupo a qual eles pertencem).

```
# ls -l /
```

```
drwxr-xr-x4 root root 1024 2019-10-15 23:17 boot
```

**d** – indica que se trata de um diretório

**l** – indica que se trata de um link (como se fosse um atalho)

# Manipulando arquivos e diretórios

Lista o conteúdo do diretório atual:

```
# ls
```

O comando **ls** possui muitos parâmetros, veremos aqui as opções mais utilizadas.

A primeira dela é o **-l** que lista os arquivos ou diretórios de uma forma bem detalhada (quem criou, data de criação, tamanho, dono e grupo a qual eles pertencem).

```
# ls -l /
```

```
drwxr-xr-x4 root root 1024 2019-10-15 23:17 boot
```

**d** – indica que se trata de um diretório

**l** – indica que se trata de um link (como se fosse um atalho)

**-** – hífen, indica que se trata de um arquivo

# Manipulando arquivos e diretórios

Lista o conteúdo do diretório atual:

```
# ls
```

O comando **ls** possui muitos parâmetros, veremos aqui as opções mais utilizadas.

A primeira dela é o **-l** que lista os arquivos ou diretórios de uma forma bem detalhada (quem criou, data de criação, tamanho, dono e grupo a qual eles pertencem).

```
# ls -l /
```

```
drwxr-xr-x4 root root 1024 2019-10-15 23:17 boot
```

**d** – indica que se trata de um diretório

**l** – indica que se trata de um link (como se fosse um atalho)

**-** – hífen, indica que se trata de um arquivo

**c** – indica dispositivo de caractere



# Manipulando arquivos e diretórios

Lista o conteúdo do diretório atual:

```
# ls
```

O comando **ls** possui muitos parâmetros, veremos aqui as opções mais utilizadas.

A primeira dela é o **-l** que lista os arquivos ou diretórios de uma forma bem detalhada (quem criou, data de criação, tamanho, dono e grupo a qual eles pertencem).

```
# ls -l /
```

```
drwxr-xr-x4 root root 1024 2019-10-15 23:17 boot
```

**d** – indica que se trata de um diretório

**l** – indica que se trata de um link (como se fosse um atalho)

**-** – hífen, indica que se trata de um arquivo

**c** – indica dispositivo de caractere

**b** – indica dispositivo de bloco

# Manipulando arquivos e diretórios

Lista o conteúdo do diretório atual:

```
# ls
```

O comando **ls** possui muitos parâmetros, veremos aqui as opções mais utilizadas.

A primeira dela é o **-l** que lista os arquivos ou diretórios de uma forma bem detalhada (quem criou, data de criação, tamanho, dono e grupo a qual eles pertencem).

```
# ls -l /
```

```
drwxr-xr-x4 root root 1024 2019-10-15 23:17 boot
```

**d** – indica que se trata de um diretório

**l** – indica que se trata de um link (como se fosse um atalho)

**-** – hífen, indica que se trata de um arquivo

**c** – indica dispositivo de caractere

**b** – indica dispositivo de bloco

```
drwxr-xr-x4 root root 1024 2019-10-15 23:17 boot
```

# Manipulando arquivos e diretórios

```
drwxr-xr-x 4 root root 1024 2019-10-15 23:17 boot
```

O campo `rw-r-xr-x` lista as **permissões**, enquanto os campos `root` indicam quem é o **usuário** e **grupo** dono desse diretório que, no nosso caso, é o administrador do sistema, o root.

```
drwxr-xr-x4 root root 1024 2019-10-15 23:17 boot
```

O campo **rwxr-xr-x** lista as permissões, enquanto os campos **root** indicam quem é o usuário e grupo dono desse diretório que, no nosso caso, é o administrador do sistema, o root.

O número antes do dono indica o número de **hard links**.

# Manipulando arquivos e diretórios

```
drwxr-xr-x4 root root 1024 2019-10-15 23:17 boot
```

O campo **rwxr-xr-x** lista as permissões, enquanto os campos **root** indicam quem é o usuário e grupo dono desse diretório que, no nosso caso, é o administrador do sistema, o root.

O número antes do dono indica o número de hard links.

O campo **1024** indica o **tamanho do arquivo**, e o campo **2019-10-15 23:17** informa **a data e hora em que o diretório foi criado**.

```
drwxr-xr-x4 root root 1024 2019-10-15 23:17 boot
```

O campo **rwxr-xr-x** lista as permissões, enquanto os campos **root** indicam quem é o usuário e grupo dono desse diretório que, no nosso caso, é o administrador do sistema, o root.

O número antes do dono indica o número de hard links.

O campo **1024** indica o tamanho do arquivo, e o campo **2019-10-15 23:17** informa a data e hora em que o diretório foi criado.

Finalmente, no último campo temos o **nome do arquivo** ou diretório listado, que, no nosso exemplo, é o **boot**.

# Manipulando arquivos e diretórios

A opção **a** lista todos arquivos, inclusive os ocultos:



# Manipulando arquivos e diretórios

A opção `a` lista todos arquivos, inclusive os ocultos:

```
# ls -a /root
```

# Manipulando arquivos e diretórios

A opção `a` lista todos arquivos, inclusive os ocultos:

```
# ls -a /root
```

```
..aptitude.bashrc.profile .rnd.ssh.vmware
```

```
.. .bash_history .kde .qt root_161206 .viminfo .Xauthority
```

No Linux, arquivos e diretórios ocultos são iniciados por um “.” (ponto).

# Manipulando arquivos e diretórios

A opção `a` lista todos arquivos, inclusive os ocultos:

```
# ls -a /root
```

```
..aptitude.bashrc.profile .rnd.ssh.vmware
```

```
.. .bash_history .kde .qt root_161206 .viminfo .Xauthority
```

No Linux, arquivos e diretórios ocultos são iniciados por um “.” (ponto).

Lista arquivos de forma recursiva, ou seja, lista também os subdiretórios que estão dentro do diretório `/`:

# Manipulando arquivos e diretórios

A opção `a` lista todos arquivos, inclusive os ocultos:

```
# ls -a /root
```

```
.. aptitude.bashrc.profile .rnd.ssh.vmware
```

```
.. .bash_history .kde .qt root_161206 .viminfo .Xauthority
```

No Linux, arquivos e diretórios ocultos são iniciados por um “.” (ponto).

Lista arquivos de forma recursiva, ou seja, lista também os subdiretórios que estão dentro do diretório `/`:

```
# ls -R /
```

# Manipulando arquivos e diretórios

A opção `a` lista todos arquivos, inclusive os ocultos:

```
# ls -a /root
```

```
..aptitude.bashrc.profile .rnd.ssh.vmware
```

```
.. .bash_history .kde .qt root_161206 .viminfo .Xauthority
```

No Linux, arquivos e diretórios ocultos são iniciados por um “.” (ponto).

Lista arquivos de forma recursiva, ou seja, lista também os subdiretórios que estão dentro do diretório `/`:

```
# ls -R /
```



Eles podem substituir uma palavra completa ou somente uma letra, seja para listar, copiar, apagar, etc. São usados três tipos de coringas no GNU/Linux:

Eles podem substituir uma palavra completa ou somente uma letra, seja para listar, copiar, apagar, etc. São usados três tipos de coringas no GNU/Linux:

- \* - Utilizado para um nome completo ou restante de um arquivo/diretório;



Eles podem substituir uma palavra completa ou somente uma letra, seja para listar, copiar, apagar, etc. São usados três tipos de coringas no GNU/Linux:

- \* - Utilizado para um nome completo ou restante de um arquivo/diretório;
- ? - Esse coringa pode substituir uma ou mais letras em determinada posição;

Eles podem substituir uma palavra completa ou somente uma letra, seja para listar, copiar, apagar, etc. São usados três tipos de coringas no GNU/Linux:

- \* - Utilizado para um nome completo ou restante de um arquivo/diretório;
- ? - Esse coringa pode substituir uma ou mais letras em determinada posição;
- [padrão]** - É utilizado para referência a uma faixa de caracteres de um arquivo/diretório;

Eles podem substituir uma palavra completa ou somente uma letra, seja para listar, copiar, apagar, etc. São usados três tipos de coringas no GNU/Linux:

- \* - Utilizado para um nome completo ou restante de um arquivo/diretório;
- ? - Esse coringa pode substituir uma ou mais letras em determinada posição;
- [padrão]** - É utilizado para referência a uma faixa de caracteres de um arquivo/diretório;

**[a-z][0-9]** - Usado para trabalhar com caracteres de a até z seguidos de um caractere de 0 até 9;

**[a-z][0-9]** - Usado para trabalhar com caracteres de a até z seguidos de um caractere de 0 até 9;

**[a,z][1,0]** - Usado para trabalhar com os caracteres a e z seguidos de um caractere 1 ou 0 naquela posição;

**[a-z][0-9]** - Usado para trabalhar com caracteres de a até z seguidos de um caractere de 0 até 9;

**[a,z][1,0]** - Usado para trabalhar com os caracteres a e z seguidos de um caractere 1 ou 0 naquela posição;

**[a-z,1,0]** - Faz referência do intervalo de caracteres de a até z ou 1 ou 0 naquela posição;

**[a-z][0-9]** - Usado para trabalhar com caracteres de a até z seguidos de um caractere de 0 até 9;

**[a,z][1,0]** - Usado para trabalhar com os caracteres a e z seguidos de um caractere 1 ou 0 naquela posição;

**[a-z,1,0]** - Faz referência do intervalo de caracteres de a até z ou 1 ou 0 naquela posição;

Lembrando que os 3 tipos de coringas mais utilizados (“\*”, “[ ]”, “?”) podem ser usados juntos.



Lembrando que os 3 tipos de coringas mais utilizados (“\*”, “[ ]”, “?”) podem ser usados juntos.

Vejamos alguns exemplos:

Lembrando que os 3 tipos de coringas mais utilizados (“\*”,“?”, “[ ]”) podem ser usados juntos.

Vejamos alguns exemplos:

Supondo que existam 5 arquivos no diretório /home/usuário. Podemos listá-los:

Lembrando que os 3 tipos de coringas mais utilizados (“\*”, “[ ]”, “[ ]””) podem ser usados juntos.

Vejamos alguns exemplos:

Supondo que existam 5 arquivos no diretório /home/usuário. Podemos listá-los:

```
# ls
```

Lembrando que os 3 tipos de coringas mais utilizados (“\*”, “[ ]”, “[ ]””) podem ser usados juntos.

Vejamos alguns exemplos:

Supondo que existam 5 arquivos no diretório /home/usuário. Podemos listá-los:

```
# ls  
arq1.txt arq2.txt arq3.txt arq4.new arq5.new
```

Lembrando que os 3 tipos de coringas mais utilizados (“\*”, “[ ]”) podem ser usados juntos.

Vejamos alguns exemplos:

Supondo que existam 5 arquivos no diretório /home/usuário. Podemos listá-los:

```
# ls  
arq1.txt arq2.txt arq3.txt arq4.new arq5.new
```

Vamos listar todos os arquivos do diretório /home/usuário. Podemos usar o coringa “\*” para visualizar todos os arquivos do diretório:

Lembrando que os 3 tipos de coringas mais utilizados (“\*”,“?”, “[ ]”) podem ser usados juntos.

Vejamos alguns exemplos:

Supondo que existam 5 arquivos no diretório /home/usuário. Podemos listá-los:

```
# ls  
arq1.txt arq2.txt arq3.txt arq4.new arq5.new
```

Vamos listar todos os arquivos do diretório /home/usuário. Podemos usar o coringa “\*” para visualizar todos os arquivos do diretório:

```
# cd /home/usuário
```

Lembrando que os 3 tipos de coringas mais utilizados (“\*”, “[ ]”) podem ser usados juntos.

Vejamos alguns exemplos:

Supondo que existam 5 arquivos no diretório /home/usuário. Podemos listá-los:

```
# ls  
arq1.txt arq2.txt arq3.txt arq4.new arq5.new
```

Vamos listar todos os arquivos do diretório /home/usuário. Podemos usar o coringa “\*” para visualizar todos os arquivos do diretório:

```
# cd /home/usuário  
# ls *
```

Lembrando que os 3 tipos de coringas mais utilizados (“\*”, “[ ]”) podem ser usados juntos.

Vejamos alguns exemplos:

Supondo que existam 5 arquivos no diretório /home/usuário. Podemos listá-los:

```
# ls  
arq1.txt arq2.txt arq3.txt arq4.new arq5.new
```

Vamos listar todos os arquivos do diretório /home/usuário. Podemos usar o coringa “\*” para visualizar todos os arquivos do diretório:

```
# cd /home/usuário  
# ls *  
arq1.txt arq2.txt arq3.txt arq4.new arq5.new
```



Lembrando que os 3 tipos de coringas mais utilizados (“\*”, “[ ]”) podem ser usados juntos.

Vejamos alguns exemplos:

Supondo que existam 5 arquivos no diretório /home/usuário. Podemos listá-los:

```
# ls  
arq1.txt arq2.txt arq3.txt arq4.new arq5.new
```

Vamos listar todos os arquivos do diretório /home/usuário. Podemos usar o coringa “\*” para visualizar todos os arquivos do diretório:

```
# cd /home/usuário  
# ls *  
arq1.txt arq2.txt arq3.txt arq4.new arq5.new
```

# Manipulando arquivos e diretórios

Para listarmos todos os arquivos do diretório `/home/usuário` que tenham "new" no nome:

# Manipulando arquivos e diretórios

Para listarmos todos os arquivos do diretório /home/usuário que tenham "new" no nome:

```
# ls *new*
```

# Manipulando arquivos e diretórios

Para listarmos todos os arquivos do diretório /home/usuário que tenham "new" no nome:

```
# ls *new*
```

```
arq4.new arq5.new
```

# Manipulando arquivos e diretórios

Para listarmos todos os arquivos do diretório /home/usuário que tenham "new" no nome:

```
# ls *new*  
arq4.new arq5.new
```

Listar todos os arquivos que começam com qualquer nome e terminam com .txt:

# Manipulando arquivos e diretórios

Para listarmos todos os arquivos do diretório `/home/usuário` que tenham "new" no nome:

```
# ls *new*  
arq4.new arq5.new
```

Listar todos os arquivos que começam com qualquer nome e terminam com .txt:

```
# ls *.txt
```

# Manipulando arquivos e diretórios

Para listarmos todos os arquivos do diretório `/home/usuário` que tenham "new" no nome:

```
# ls *new*  
arq4.new arq5.new
```

Listar todos os arquivos que começam com qualquer nome e terminam com .txt:

```
# ls *.txt
```

# Manipulando arquivos e diretórios

Para listarmos todos os arquivos do diretório `/home/usuário` que tenham "new" no nome:



# Manipulando arquivos e diretórios

Para listarmos todos os arquivos do diretório /home/usuário que tenham "new" no nome:

```
# ls *new*
```

# Manipulando arquivos e diretórios

Para listarmos todos os arquivos do diretório /home/usuário que tenham "new" no nome:

```
# ls *new*
```

```
arq4.new arq5.new
```

# Manipulando arquivos e diretórios

Para listarmos todos os arquivos do diretório /home/usuário que tenham "new" no nome:

```
# ls *new*
```

```
arq4.new arq5.new
```

Listar todos os arquivos que começam com qualquer nome e terminam com .txt:

# Manipulando arquivos e diretórios

Para listarmos todos os arquivos do diretório /home/usuário que tenham "new" no nome:

```
# ls *new*  
arq4.new arq5.new
```

Listar todos os arquivos que começam com qualquer nome e terminam com .txt:

```
# ls *.txt
```

# Manipulando arquivos e diretórios

Para listarmos todos os arquivos do diretório /home/usuário que tenham "new" no nome:

```
# ls *new*
```

```
arq4.new arq5.new
```

Listar todos os arquivos que começam com qualquer nome e terminam com .txt:

```
# ls *.txt
```

Listar todos os arquivos que começam com o nome arq, tenham qualquer caractere no lugar do coringa, e terminem com .txt:

# Manipulando arquivos e diretórios

Para listarmos todos os arquivos do diretório /home/usuário que tenham "new" no nome:

```
# ls *new*  
arq4.new arq5.new
```

Listar todos os arquivos que começam com qualquer nome e terminam com .txt:

```
# ls *.txt
```

Listar todos os arquivos que começam com o nome arq, tenham qualquer caractere no lugar do coringa, e terminem com .txt:

```
# ls arq?.txt
```

# Manipulando arquivos e diretórios

Para listarmos todos os arquivos do diretório /home/usuário que tenham "new" no nome:

```
# ls *new*  
arq4.new arq5.new
```

Listar todos os arquivos que começam com qualquer nome e terminam com .txt:

```
# ls *.txt
```

Listar todos os arquivos que começam com o nome arq, tenham qualquer caractere no lugar do coringa, e terminem com .txt:

```
# ls arq?.txt
```

# Manipulando arquivos e diretórios

Para listar todos os arquivos que começam com o nome `arq`, tenham qualquer caractere entre o número 1-3 no lugar da 4ª letra e terminem com `.txt`.



# Manipulando arquivos e diretórios

Para listar todos os arquivos que começam com o nome `arq`, tenham qualquer caractere entre o número 1-3 no lugar da 4ª letra e terminem com `.txt`.

Neste caso, se obtém uma filtragem mais exata, pois o coringa especifica qualquer caractere naquela posição e `[ ]` especifica números, letras ou intervalo que serão usados.

# Manipulando arquivos e diretórios

Para listar todos os arquivos que começam com o nome `arq`, tenham qualquer caractere entre o número 1-3 no lugar da 4ª letra e terminem com `.txt`.

Neste caso, se obtém uma filtragem mais exata, pois o coringa especifica qualquer caractere naquela posição e `[ ]` especifica números, letras ou intervalo que serão usados.

```
# ls arq[1-3].txt
```

# Manipulando arquivos e diretórios

Para listar todos os arquivos que começam com o nome `arq`, tenham qualquer caractere entre o número 1-3 no lugar da 4ª letra e terminem com `.txt`.

Neste caso, se obtém uma filtragem mais exata, pois o coringa especifica qualquer caractere naquela posição e `[ ]` especifica números, letras ou intervalo que serão usados.

```
# ls arq[1-3].txt
```

# Manipulando arquivos e diretórios

# Manipulando arquivos e diretórios

Para listar somente `arq4.new` e `arq5.new` podemos usar os seguintes métodos:

# Manipulando arquivos e diretórios

Para listar somente `arq4.new` e `arq5.new` podemos usar os seguintes métodos:

```
# ls *.new
```

# Manipulando arquivos e diretórios

Para listar somente `arq4.new` e `arq5.new` podemos usar os seguintes métodos:

```
# ls *.new
```

```
# ls *new*
```

# Manipulando arquivos e diretórios

Para listar somente `arq4.new` e `arq5.new` podemos usar os seguintes métodos:

```
# ls *.new  
# ls *new*  
# ls arq?.new
```



# Manipulando arquivos e diretórios

Para listar somente `arq4.new` e `arq5.new` podemos usar os seguintes métodos:

```
# ls *.new  
# ls *new*  
# ls arq?.new  
# ls arq[4,5].*
```

# Manipulando arquivos e diretórios

Para listar somente `arq4.new` e `arq5.new` podemos usar os seguintes métodos:

```
# ls *.new  
# ls *new*  
# ls arq?.new  
# ls arq[4,5].*  
# ls arq[4,5].new
```

# Manipulando arquivos e diretórios

Para listar somente `arq4.new` e `arq5.new` podemos usar os seguintes métodos:

```
# ls *.new  
# ls *new*  
# ls arq?.new  
# ls arq[4,5].*  
# ls arq[4,5].new
```

# Manipulando arquivos e diretórios

Uma das formas mais simples para se criar um arquivo é usando o comando **touch**:

# Manipulando arquivos e diretórios

Uma das formas mais simples para se criar um arquivo é usando o comando **touch**:

**# touch arquivo**

# Manipulando arquivos e diretórios

Uma das formas mais simples para se criar um arquivo é usando o comando **touch**:

**# touch arquivo**

O comando **mkdir** é utilizado para criar um diretório no sistema. Exemplo:

# Manipulando arquivos e diretórios

Uma das formas mais simples para se criar um arquivo é usando o comando **touch**:

```
# touch arquivo
```

O comando **mkdir** é utilizado para criar um diretório no sistema. Exemplo:  
Cria o diretório yoda:

# Manipulando arquivos e diretórios

Uma das formas mais simples para se criar um arquivo é usando o comando **touch**:

```
# touch arquivo
```

O comando **mkdir** é utilizado para criar um diretório no sistema. Exemplo:  
Cria o diretório yoda:

```
# mkdir yoda
```



# Manipulando arquivos e diretórios

Uma das formas mais simples para se criar um arquivo é usando o comando **touch**:

```
# touch arquivo
```

O comando **mkdir** é utilizado para criar um diretório no sistema. Exemplo:  
Cria o diretório yoda:

```
# mkdir yoda
```

Cria o diretório Jhonny e o subdiretório alunos:

Uma das formas mais simples para se criar um arquivo é usando o comando **touch**:

```
# touch arquivo
```

O comando **mkdir** é utilizado para criar um diretório no sistema. Exemplo:  
Cria o diretório yoda:

```
# mkdir yoda
```

Cria o diretório Jhonny e o subdiretório alunos:

```
# mkdir -p Jhonny/alunos
```

# Manipulando arquivos e diretórios

Uma das formas mais simples para se criar um arquivo é usando o comando **touch**:

```
# touch arquivo
```

O comando **mkdir** é utilizado para criar um diretório no sistema. Exemplo:  
Cria o diretório yoda:

```
# mkdir yoda
```

Cria o diretório Jhonny e o subdiretório alunos:

```
# mkdir -p Jhonny/alunos
```

A opção **-p** irá criar o diretório Jhonny e o subdiretório alunos, caso não existam.

# Manipulando arquivos e diretórios

Uma das formas mais simples para se criar um arquivo é usando o comando **touch**:

```
# touch arquivo
```

O comando **mkdir** é utilizado para criar um diretório no sistema. Exemplo:  
Cria o diretório yoda:

```
# mkdir yoda
```

Cria o diretório Jhonny e o subdiretório alunos:

```
# mkdir -p Jhonny/alunos
```

A opção **-p** irá criar o diretório Jhonny e o subdiretório alunos, caso não existam.

# Manipulando arquivos e diretórios

O comando **rm** é utilizado para apagar arquivos, diretórios e subdiretórios que estejam vazios ou que contenham arquivos. Exemplos:

# Manipulando arquivos e diretórios

O comando **rm** é utilizado para apagar arquivos, diretórios e subdiretórios que estejam vazios ou que contenham arquivos. Exemplos:

Remove o arquivo teste.txt:

```
# rm teste.txt
```

# Manipulando arquivos e diretórios

O comando **rm** é utilizado para apagar arquivos, diretórios e subdiretórios que estejam vazios ou que contenham arquivos. Exemplos:

Remove o arquivo teste.txt:

```
# rm teste.txt
```

Remove o arquivo yago.txt pedindo confirmação:

```
# rm -i yago.txt
```

# Manipulando arquivos e diretórios

O comando **rm** é utilizado para apagar arquivos, diretórios e subdiretórios que estejam vazios ou que contenham arquivos. Exemplos:

Remove o arquivo teste.txt:

```
# rm teste.txt
```

Remove o arquivo yago.txt pedindo confirmação:

```
# rm -i yago.txtrm: remove arquivo comum 'yago.txt'? y
```

A opção **-i** solicita a confirmação para remover o arquivo yago.txt.



# Manipulando arquivos e diretórios

O comando **rm** é utilizado para apagar arquivos, diretórios e subdiretórios que estejam vazios ou que contenham arquivos. Exemplos:

Remove o arquivo teste.txt:

```
# rm teste.txt
```

Remove o arquivo yago.txt pedindo confirmação:

```
# rm -i yago.txtrm: remove arquivo comum 'yago.txt'? y
```

A opção **-i** solicita a confirmação para remover o arquivo yago.txt.

Remove o diretório Jhonny:

```
# rm -r Jhonny
```

A opção **-r** é recursivo, ou seja, irá remover o diretório Jhonny e o seu conteúdo.

# Manipulando arquivos e diretórios

Exemplos:

# Manipulando arquivos e diretórios

Exemplos:

Remove o diretório yago:

```
# rmdir yoda
```

# Manipulando arquivos e diretórios

Exemplos:

Remove o diretório yago:

```
# rmdir yoda
```

Remove o diretório Jhonny e o subdiretório alunos:

```
# rmdir -p Jhonny/alunos
```

# Manipulando arquivos e diretórios

Exemplos:

Remove o diretório yago:

```
# rmdir yoda
```

Remove o diretório Jhonny e o subdiretório alunos:

```
# rmdir -p Jhonny/alunos
```

O comando **cp** serve para fazer cópias de arquivos e diretórios:

# Manipulando arquivos e diretórios

Exemplos:

Remove o diretório yago:

```
# rmdir yoda
```

Remove o diretório Jhonny e o subdiretório alunos:

```
# rmdir -p Jhonny/alunos
```

O comando **cp** serve para fazer cópias de arquivos e diretórios:

```
# cp arquivo-origem arquivo-destino
```

# Manipulando arquivos e diretórios

Exemplos:

Remove o diretório yago:

```
# rmdir yoda
```

Remove o diretório Jhonny e o subdiretório alunos:

```
# rmdir -p Jhonny/alunos
```

O comando **cp** serve para fazer cópias de arquivos e diretórios:

```
# cp arquivo-origem arquivo-destino
```

```
# cp arquivo-origem caminho/diretório-destino/
```

# Manipulando arquivos e diretórios

Exemplos:

Remove o diretório yago:

```
# rmdir yoda
```

Remove o diretório Jhonny e o subdiretório alunos:

```
# rmdir -p Jhonny/alunos
```

O comando **cp** serve para fazer cópias de arquivos e diretórios:

```
# cp arquivo-origem arquivo-destino
```

```
# cp arquivo-origem caminho/diretório-destino/
```

```
# cp -R diretório-origem nome-destino
```



# Manipulando arquivos e diretórios

Exemplos:

Remove o diretório yago:

```
# rmdir yoda
```

Remove o diretório Jhonny e o subdiretório alunos:

```
# rmdir -p Jhonny/alunos
```

O comando **cp** serve para fazer cópias de arquivos e diretórios:

```
# cp arquivo-origem arquivo-destino
```

```
# cp arquivo-origem caminho/diretório-destino/
```

```
# cp -R diretório-origem nome-destino
```

```
# cp -R diretório-origem caminho/diretório-destino/
```

# Manipulando arquivos e diretórios

Exemplos:

Remove o diretório yago:

```
# rmdir yoda
```

Remove o diretório Jhonny e o subdiretório alunos:

```
# rmdir -p Jhonny/alunos
```

O comando **cp** serve para fazer cópias de arquivos e diretórios:

```
# cp arquivo-origem arquivo-destino
```

```
# cp arquivo-origem caminho/diretório-destino/
```

```
# cp -R diretório-origem nome-destino
```

```
# cp -R diretório-origem caminho/diretório-destino/
```

O comando **mv** serve tanto para renomear um arquivo quanto para movê-lo:

O comando **mv** serve tanto para renomear um arquivo quanto para movê-lo:

```
# mv arquivo caminho/diretório-destino/
```

O comando **mv** serve tanto para renomear um arquivo quanto para movê-lo:

```
# mv arquivo caminho/diretório-destino/  
# mv arquivo novo-nome
```

# Manipulando arquivos e diretórios

O comando **mv** serve tanto para renomear um arquivo quanto para movê-lo:

```
# mv arquivo caminho/diretório-destino/  
# mv arquivo novo-nome  
# mv diretório novo-nome
```

# Manipulando arquivos e diretórios

O comando **mv** serve tanto para renomear um arquivo quanto para movê-lo:

```
# mv arquivo caminho/diretório-destino/  
# mv arquivo novo-nome  
# mv diretório novo-nome  
# mv diretório caminho/diretório-destino/
```

# Manipulando arquivos e diretórios

O comando **mv** serve tanto para renomear um arquivo quanto para movê-lo:

```
# mv arquivo caminho/diretório-destino/  
# mv arquivo novo-nome  
# mv diretório novo-nome  
# mv diretório caminho/diretório-destino/
```



1) Listar o conteúdo do diretório /:  
# ls /

1) Listar o conteúdo do diretório /:

```
# ls /
```

2) Listar o conteúdo do diretório /root em formato longo:

```
# ls -l /root/
```

1) Listar o conteúdo do diretório /:

```
# ls /
```

2) Listar o conteúdo do diretório /root em formato longo:

```
# ls -l /root/
```

3) Listar somente o diretório /boot em formato longo:

```
# ls -ld /boot/
```

1) Listar o conteúdo do diretório /:

```
# ls /
```

2) Listar o conteúdo do diretório /root em formato longo:

```
# ls -l /root/
```

3) Listar somente o diretório /boot em formato longo:

```
# ls -ld /boot/
```

4) Listar todos os arquivos do diretório /root, inclusive os ocultos:

```
# ls -a /root
```

1) Listar o conteúdo do diretório /:

```
# ls /
```

2) Listar o conteúdo do diretório /root em formato longo:

```
# ls -l /root/
```

3) Listar somente o diretório /boot em formato longo:

```
# ls -ld /boot/
```

4) Listar todos os arquivos do diretório /root, inclusive os ocultos:

```
# ls -a /root
```

5) Listar o conteúdo do diretório /boot de forma recursiva:

```
# ls -R /boot/
```

5) Listar o conteúdo do diretório `/boot` de forma recursiva:

```
# ls -R /boot/
```

6) Criar o diretório `estudo` dentro do diretório `/tmp`:

```
# mkdir /tmp/estudo
```

5) Listar o conteúdo do diretório `/boot` de forma recursiva:

```
# ls -R /boot/
```

6) Criar o diretório `estudo` dentro do diretório `/tmp`:

```
# mkdir /tmp/estudo
```

7) Criar a seguinte estrutura de diretórios: `/backup/2019/outubro`

```
# mkdir -p /backup/2019/outubro
```



5) Listar o conteúdo do diretório /boot de forma recursiva:

```
# ls -R /boot/
```

6) Criar o diretório estudo dentro do diretório /tmp:

```
# mkdir /tmp/estudo
```

7) Criar a seguinte estrutura de diretórios: /backup/2019/outubro

```
# mkdir -p /backup/2019/outubro
```

8) Remover o diretório /tmp/estudo utilizando o comando rmdir: **# rmdir /tmp/estudo**

5) Listar o conteúdo do diretório /boot de forma recursiva:

```
# ls -R /boot/
```

6) Criar o diretório estudo dentro do diretório /tmp:

```
# mkdir /tmp/estudo
```

7) Criar a seguinte estrutura de diretórios: /backup/2019/outubro

```
# mkdir -p /backup/2019/outubro
```

8) Remover o diretório /tmp/estudo utilizando o comando rmdir: **# rmdir /tmp/estudo**

9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

**# touch /backup/2019/outubro/estudo.txt**

9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

```
# touch /backup/2019/outubro/estudo.txt
```

```
# touch /backup/2019/outubro/alunos.txt
```

9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

```
# touch /backup/2019/outubro/estudo.txt
```

```
# touch /backup/2019/outubro/alunos.txt
```

10) Entre no diretório /backup/2019/outubro e copie o arquivo estudo.txt para aula.txt:

9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

```
# touch /backup/2019/outubro/estudo.txt
```

```
# touch /backup/2019/outubro/alunos.txt
```

10) Entre no diretório /backup/2019/outubro e copie o arquivo estudo.txt para aula.txt:

```
# cd /backup/2019/outubro
```

9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

```
# touch /backup/2019/outubro/estudo.txt
```

```
# touch /backup/2019/outubro/alunos.txt
```

10) Entre no diretório /backup/2019/outubro e copie o arquivo estudo.txt para aula.txt:

```
# cd /backup/2019/outubro
```

```
# cp estudo.txt aula.txt
```

9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

```
# touch /backup/2019/outubro/estudo.txt
```

```
# touch /backup/2019/outubro/alunos.txt
```

10) Entre no diretório /backup/2019/outubro e copie o arquivo estudo.txt para aula.txt:

```
# cd /backup/2019/outubro
```

```
# cp estudo.txt aula.txt
```

11) Copie o diretório /backup/2019/outubro para /backup/2019/setembro:



9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

```
# touch /backup/2019/outubro/estudo.txt
```

```
# touch /backup/2019/outubro/alunos.txt
```

10) Entre no diretório /backup/2019/outubro e copie o arquivo estudo.txt para aula.txt:

```
# cd /backup/2019/outubro
```

```
# cp estudo.txt aula.txt
```

11) Copie o diretório /backup/2019/outubro para /backup/2019/setembro:

```
# cp -R /backup/2019/outubro /backup/2019/setembro
```

9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

```
# touch /backup/2019/outubro/estudo.txt
```

```
# touch /backup/2019/outubro/alunos.txt
```

10) Entre no diretório /backup/2019/outubro e copie o arquivo estudo.txt para aula.txt:

```
# cd /backup/2019/outubro
```

```
# cp estudo.txt aula.txt
```

11) Copie o diretório /backup/2019/outubro para /backup/2019/setembro:

```
# cp -R /backup/2019/outubro /backup/2019/setembro
```

12) Remova o arquivo estudo.txt do diretório /backup/2019/outubro:

9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

```
# touch /backup/2019/outubro/estudo.txt
```

```
# touch /backup/2019/outubro/alunos.txt
```

10) Entre no diretório /backup/2019/outubro e copie o arquivo estudo.txt para aula.txt:

```
# cd /backup/2019/outubro
```

```
# cp estudo.txt aula.txt
```

11) Copie o diretório /backup/2019/outubro para /backup/2019/setembro:

```
# cp -R /backup/2019/outubro /backup/2019/setembro
```

12) Remova o arquivo estudo.txt do diretório /backup/2019/outubro:

```
# cd /backup/2019/outubro
```

9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

```
# touch /backup/2019/outubro/estudo.txt
```

```
# touch /backup/2019/outubro/alunos.txt
```

10) Entre no diretório /backup/2019/outubro e copie o arquivo estudo.txt para aula.txt:

```
# cd /backup/2019/outubro
```

```
# cp estudo.txt aula.txt
```

11) Copie o diretório /backup/2019/outubro para /backup/2019/setembro:

```
# cp -R /backup/2019/outubro /backup/2019/setembro
```

12) Remova o arquivo estudo.txt do diretório /backup/2019/outubro:

```
# cd /backup/2019/outubro# rm estudo.txt
```

9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

```
# touch /backup/2019/outubro/estudo.txt
```

```
# touch /backup/2019/outubro/alunos.txt
```

10) Entre no diretório /backup/2019/outubro e copie o arquivo estudo.txt para aula.txt:

```
# cd /backup/2019/outubro
```

```
# cp estudo.txt aula.txt
```

11) Copie o diretório /backup/2019/outubro para /backup/2019/setembro:

```
# cp -R /backup/2019/outubro /backup/2019/setembro
```

12) Remova o arquivo estudo.txt do diretório /backup/2019/outubro:

```
# cd /backup/2019/outubro# rm estudo.txt
```

9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

**# touch /backup/2019/outubro/estudo.txt**

9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

```
# touch /backup/2019/outubro/estudo.txt
```

```
# touch /backup/2019/outubro/alunos.txt
```



9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

```
# touch /backup/2019/outubro/estudo.txt
```

```
# touch /backup/2019/outubro/alunos.txt
```

10) Entre no diretório /backup/2019/outubro e copie o arquivo estudo.txt para aula.txt:

9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

```
# touch /backup/2019/outubro/estudo.txt
```

```
# touch /backup/2019/outubro/alunos.txt
```

10) Entre no diretório /backup/2019/outubro e copie o arquivo estudo.txt para aula.txt:

```
# cd /backup/2019/outubro
```

9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

```
# touch /backup/2019/outubro/estudo.txt
```

```
# touch /backup/2019/outubro/alunos.txt
```

10) Entre no diretório /backup/2019/outubro e copie o arquivo estudo.txt para aula.txt:

```
# cd /backup/2019/outubro
```

```
# cp estudo.txt aula.txt
```

9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

```
# touch /backup/2019/outubro/estudo.txt
```

```
# touch /backup/2019/outubro/alunos.txt
```

10) Entre no diretório /backup/2019/outubro e copie o arquivo estudo.txt para aula.txt:

```
# cd /backup/2019/outubro
```

```
# cp estudo.txt aula.txt
```

11) Copie o diretório /backup/2019/outubro para /backup/2019/setembro:

9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

```
# touch /backup/2019/outubro/estudo.txt
```

```
# touch /backup/2019/outubro/alunos.txt
```

10) Entre no diretório /backup/2019/outubro e copie o arquivo estudo.txt para aula.txt:

```
# cd /backup/2019/outubro
```

```
# cp estudo.txt aula.txt
```

11) Copie o diretório /backup/2019/outubro para /backup/2019/setembro:

```
# cp -R /backup/2019/outubro /backup/2019/setembro
```

9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

```
# touch /backup/2019/outubro/estudo.txt
```

```
# touch /backup/2019/outubro/alunos.txt
```

10) Entre no diretório /backup/2019/outubro e copie o arquivo estudo.txt para aula.txt:

```
# cd /backup/2019/outubro
```

```
# cp estudo.txt aula.txt
```

11) Copie o diretório /backup/2019/outubro para /backup/2019/setembro:

```
# cp -R /backup/2019/outubro /backup/2019/setembro
```

12) Remova o arquivo estudo.txt do diretório /backup/2019/outubro:

9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

```
# touch /backup/2019/outubro/estudo.txt
```

```
# touch /backup/2019/outubro/alunos.txt
```

10) Entre no diretório /backup/2019/outubro e copie o arquivo estudo.txt para aula.txt:

```
# cd /backup/2019/outubro
```

```
# cp estudo.txt aula.txt
```

11) Copie o diretório /backup/2019/outubro para /backup/2019/setembro:

```
# cp -R /backup/2019/outubro /backup/2019/setembro
```

12) Remova o arquivo estudo.txt do diretório /backup/2019/outubro:

```
# cd /backup/2019/outubro
```

9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

```
# touch /backup/2019/outubro/estudo.txt
```

```
# touch /backup/2019/outubro/alunos.txt
```

10) Entre no diretório /backup/2019/outubro e copie o arquivo estudo.txt para aula.txt:

```
# cd /backup/2019/outubro
```

```
# cp estudo.txt aula.txt
```

11) Copie o diretório /backup/2019/outubro para /backup/2019/setembro:

```
# cp -R /backup/2019/outubro /backup/2019/setembro
```

12) Remova o arquivo estudo.txt do diretório /backup/2019/outubro:

```
# cd /backup/2019/outubro# rm estudo.txt
```



9) Crie os arquivos estudo.txt e alunos.txt dentro de /backup/2019/outubro.

```
# touch /backup/2019/outubro/estudo.txt
```

```
# touch /backup/2019/outubro/alunos.txt
```

10) Entre no diretório /backup/2019/outubro e copie o arquivo estudo.txt para aula.txt:

```
# cd /backup/2019/outubro
```

```
# cp estudo.txt aula.txt
```

11) Copie o diretório /backup/2019/outubro para /backup/2019/setembro:

```
# cp -R /backup/2019/outubro /backup/2019/setembro
```

12) Remova o arquivo estudo.txt do diretório /backup/2019/outubro:

```
# cd /backup/2019/outubro# rm estudo.txt
```

13) Renomeie o arquivo `alunos.txt` do diretório `/backup/2019/outubro`:

- 13) Renomeie o arquivo alunos.txt do diretório /backup/2019/outubro:  
`cd /backup/2019/outubro`

- 13) Renomeie o arquivo `alunos.txt` do diretório `/backup/2019/outubro`:
- ```
cd /backup/2019/outubro  
mv alunos.txt teste.txt
```

- 13) Renomeie o arquivo `alunos.txt` do diretório `/backup/2019/outubro`:
- ```
cd /backup/2019/outubro  
mv alunos.txt teste.txt
```
- 14) Mova o diretório `/backup/2019/outubro` para `/backup/2019/abril`:

13) Renomeie o arquivo alunos.txt do diretório /backup/2019/outubro:

```
cd /backup/2019/outubro  
mv alunos.txt teste.txt
```

14) Mova o diretório /backup/2019/outubro para /backup/2019/abril:

```
mv /backup/2019/outubro /backup/2019/abril
```

13) Renomeie o arquivo `alunos.txt` do diretório `/backup/2019/outubro`:

```
cd /backup/2019/outubro  
mv alunos.txt teste.txt
```

14) Mova o diretório `/backup/2019/outubro` para `/backup/2019/abril`:

```
mv /backup/2019/outubro /backup/2019/abril
```

15) Utilize o comando `stat` para descobrir algumas informações importantes:

13) Renomeie o arquivo `alunos.txt` do diretório `/backup/2019/outubro`:

```
cd /backup/2019/outubro  
mv alunos.txt teste.txt
```

14) Mova o diretório `/backup/2019/outubro` para `/backup/2019/abril`:

```
mv /backup/2019/outubro /backup/2019/abril
```

15) Utilize o comando `stat` para descobrir algumas informações importantes:

```
# stat /backup
```



13) Renomeie o arquivo `alunos.txt` do diretório `/backup/2019/outubro`:

```
cd /backup/2019/outubro  
mv alunos.txt teste.txt
```

14) Mova o diretório `/backup/2019/outubro` para `/backup/2019/abril`:

```
mv /backup/2019/outubro /backup/2019/abril
```

15) Utilize o comando `stat` para descobrir algumas informações importantes:

```
# stat /backup
```

Redirecionando a saída padrão:

Redirecionando a saída padrão:

> - Redireciona a saída em um arquivo apagando o conteúdo anterior (se existir);

Redirecionando a saída padrão:

> - Redireciona a saída em um arquivo apagando o conteúdo anterior (se existir);

>> - Redireciona a saída no final de um arquivo, preservando-o;

Redirecionando a saída padrão:

> - Redireciona a saída em um arquivo apagando o conteúdo anterior (se existir);

>> - Redireciona a saída no final de um arquivo, preservando-o;

| (pipe, pronuncia-se *paípe*): Serve para canalizar saída de dados para outro comando;

# Comandos Úteis

Redirecionando a saída padrão:

> - Redireciona a saída em um arquivo apagando o conteúdo anterior (se existir);

>> - Redireciona a saída no final de um arquivo, preservando-o;

| (pipe, pronuncia-se *paípe*): Serve para canalizar saída de dados para outro comando;

Mostrando o conteúdo e/ou concatenando:

# Comandos Úteis

Redirecionando a saída padrão:

> - Redireciona a saída em um arquivo apagando o conteúdo anterior (se existir);

>> - Redireciona a saída no final de um arquivo, preservando-o;

| (pipe, pronuncia-se *paípe*): Serve para canalizar saída de dados para outro comando;

Mostrando o conteúdo e/ou concatenando:

**# cat arquivo**

Redirecionando a saída padrão:

> - Redireciona a saída em um arquivo apagando o conteúdo anterior (se existir);

>> - Redireciona a saída no final de um arquivo, preservando-o;

| (pipe, pronuncia-se *paípe*): Serve para canalizar saída de dados para outro comando;

Mostrando o conteúdo e/ou concatenando:

**# cat arquivo**

O comando **tac** também serve para mostrar o conteúdo e concatenar arquivos. Porém, ele mostra o conteúdo de forma reversa, linha a linha.



# Comandos Úteis

Redirecionando a saída padrão:

> - Redireciona a saída em um arquivo apagando o conteúdo anterior (se existir);

>> - Redireciona a saída no final de um arquivo, preservando-o;

| (pipe, pronuncia-se *paípe*): Serve para canalizar saída de dados para outro comando;

Mostrando o conteúdo e/ou concatenando:

**# cat arquivo**

O comando **tac** também serve para mostrar o conteúdo e concatenar arquivos. Porém, ele mostra o conteúdo de forma reversa, linha a linha.

Contagem: **wc**

# Comandos Úteis

Redirecionando a saída padrão:

> - Redireciona a saída em um arquivo apagando o conteúdo anterior (se existir);

>> - Redireciona a saída no final de um arquivo, preservando-o;

| (pipe, pronuncia-se *paípe*): Serve para canalizar saída de dados para outro comando;

Mostrando o conteúdo e/ou concatenando:

**# cat arquivo**

O comando **tac** também serve para mostrar o conteúdo e concatenar arquivos. Porém, ele mostra o conteúdo de forma reversa, linha a linha.

Contagem: **wc**

Classificação: **sort**

# Comandos Úteis

Redirecionando a saída padrão:

> - Redireciona a saída em um arquivo apagando o conteúdo anterior (se existir);

>> - Redireciona a saída no final de um arquivo, preservando-o;

| (pipe, pronuncia-se *paípe*): Serve para canalizar saída de dados para outro comando;

Mostrando o conteúdo e/ou concatenando:

**# cat arquivo**

O comando **tac** também serve para mostrar o conteúdo e concatenar arquivos. Porém, ele mostra o conteúdo de forma reversa, linha a linha.

Contagem: **wc**

Classificação: **sort**

Informações do disco: **df**

# Comandos Úteis

Redirecionando a saída padrão:

> - Redireciona a saída em um arquivo apagando o conteúdo anterior (se existir);

>> - Redireciona a saída no final de um arquivo, preservando-o;

| (pipe, pronuncia-se *paípe*): Serve para canalizar saída de dados para outro comando;

Mostrando o conteúdo e/ou concatenando:

**# cat arquivo**

O comando **tac** também serve para mostrar o conteúdo e concatenar arquivos. Porém, ele mostra o conteúdo de forma reversa, linha a linha.

Contagem: **wc**

Classificação: **sort**

Informações do disco: **df**

Encontrar uma linhas: **grep**

# Comandos Úteis

Redirecionando a saída padrão:

> - Redireciona a saída em um arquivo apagando o conteúdo anterior (se existir);

>> - Redireciona a saída no final de um arquivo, preservando-o;

| (pipe, pronuncia-se *paípe*): Serve para canalizar saída de dados para outro comando;

Mostrando o conteúdo e/ou concatenando:

# **cat arquivo**

O comando **tac** também serve para mostrar o conteúdo e concatenar arquivos. Porém, ele mostra o conteúdo de forma reversa, linha a linha.

Contagem: **wc**

Classificação: **sort**

Informações do disco: **df**

Encontrar uma linhas: **grep**

Determinando o tipo de arquivo: **file**

# Comandos Úteis

Redirecionando a saída padrão:

> - Redireciona a saída em um arquivo apagando o conteúdo anterior (se existir);

>> - Redireciona a saída no final de um arquivo, preservando-o;

| (pipe, pronuncia-se *paípe*): Serve para canalizar saída de dados para outro comando;

Mostrando o conteúdo e/ou concatenando:

# **cat arquivo**

O comando **tac** também serve para mostrar o conteúdo e concatenar arquivos. Porém, ele mostra o conteúdo de forma reversa, linha a linha.

Contagem: **wc**

Classificação: **sort**

Informações do disco: **df**

Encontrar uma linhas: **grep**

Determinando o tipo de arquivo: **file**



Mostrar o uso de memória RAM: **free**



Mostrar o uso de memória RAM: **free**  
Informações de data e hora: **date**

Mostrar o uso de memória RAM: **free**

Informações de data e hora: **date**

Mostrar por quanto tempo o computador está ligado: **uptime**

# Comandos Úteis

Mostrar o uso de memória RAM: **free**

Informações de data e hora: **date**

Mostrar por quanto tempo o computador está ligado: **uptime**

Mostrar informações sobre o sistema: **uname**

# Comandos Úteis

Mostrar o uso de memória RAM: **free**

Informações de data e hora: **date**

Mostrar por quanto tempo o computador está ligado: **uptime**

Mostrar informações sobre o sistema: **uname**

Diferença entre arquivos: **diff**

# Comandos Úteis

Mostrar o uso de memória RAM: **free**

Informações de data e hora: **date**

Mostrar por quanto tempo o computador está ligado: **uptime**

Mostrar informações sobre o sistema: **uname**

Diferença entre arquivos: **diff**

Tempo de execução de um programa: **time**

Mostrar o uso de memória RAM: **free**

Informações de data e hora: **date**

Mostrar por quanto tempo o computador está ligado: **uptime**

Mostrar informações sobre o sistema: **uname**

Diferença entre arquivos: **diff**

Tempo de execução de um programa: **time**

Linguagem de procura de padrões e processamento: **awk**

Mostrar o uso de memória RAM: **free**

Informações de data e hora: **date**

Mostrar por quanto tempo o computador está ligado: **uptime**

Mostrar informações sobre o sistema: **uname**

Diferença entre arquivos: **diff**

Tempo de execução de um programa: **time**

Linguagem de procura de padrões e processamento: **awk**

Comando de interface entre a shell e o syslog: **logger**

# Comandos Úteis

Mostrar o uso de memória RAM: **free**

Informações de data e hora: **date**

Mostrar por quanto tempo o computador está ligado: **uptime**

Mostrar informações sobre o sistema: **uname**

Diferença entre arquivos: **diff**

Tempo de execução de um programa: **time**

Linguagem de procura de padrões e processamento: **awk**

Comando de interface entre a shell e o syslog: **logger**

Editor de texto linha a linha: **sed**



Mostrar o uso de memória RAM: **free**

Informações de data e hora: **date**

Mostrar por quanto tempo o computador está ligado: **uptime**

Mostrar informações sobre o sistema: **uname**

Diferença entre arquivos: **diff**

Tempo de execução de um programa: **time**

Linguagem de procura de padrões e processamento: **awk**

Comando de interface entre a shell e o syslog: **logger**

Editor de texto linha a linha: **sed**

Imprime uma sequencia de números: **seq**

Mostrar o uso de memória RAM: **free**

Informações de data e hora: **date**

Mostrar por quanto tempo o computador está ligado: **uptime**

Mostrar informações sobre o sistema: **uname**

Diferença entre arquivos: **diff**

Tempo de execução de um programa: **time**

Linguagem de procura de padrões e processamento: **awk**

Comando de interface entre a shell e o syslog: **logger**

Editor de texto linha a linha: **sed**

Imprime uma sequencia de números: **seq**

Insere uma pausa pelo número de segundos especificado: **sleep**

Mostrar o uso de memória RAM: **free**

Informações de data e hora: **date**

Mostrar por quanto tempo o computador está ligado: **uptime**

Mostrar informações sobre o sistema: **uname**

Diferença entre arquivos: **diff**

Tempo de execução de um programa: **time**

Linguagem de procura de padrões e processamento: **awk**

Comando de interface entre a shell e o syslog: **logger**

Editor de texto linha a linha: **sed**

Imprime uma sequencia de números: **seq**

Insere uma pausa pelo número de segundos especificado: **sleep**

Fim. Dúvidas?