Estratégia Concursos

Aula 11

BNB (Analista Bancário) Informática - 2023 (Pré-Edital)

Autor:

Diego Carvalho, Renato da Costa, Equipe Informática e TI

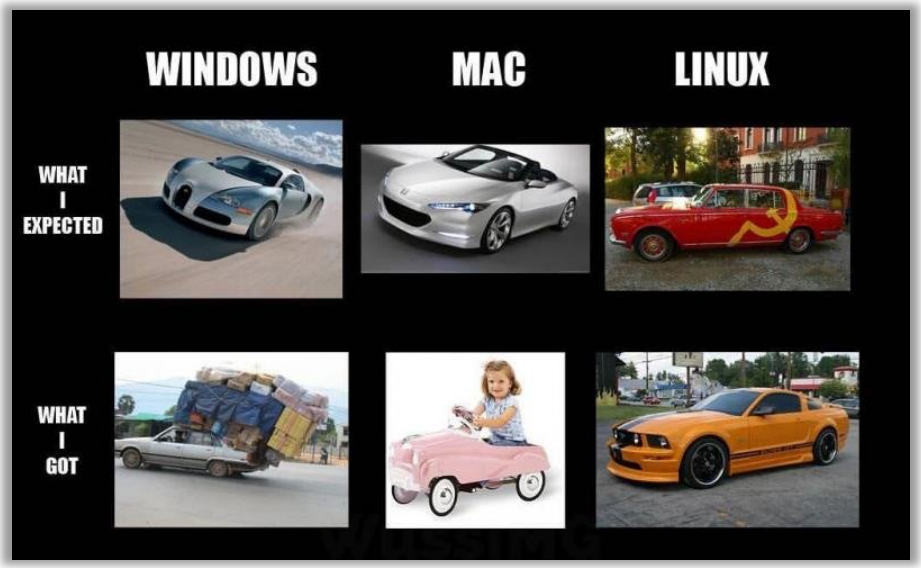
31 de Maio de 2023

Índice

Apresentação da Aula

Pessoal, o tema da nossa aula é Linux. Sim, aquele sistema operacional que a maioria de vocês só ouviu falar, mas nunca utilizou. **Galera, esse é um assunto um pouco assustador para quem nunca utilizou o Linux, mas não fiquem com medo dele.** Ele é bastante parecido com o Windows, exceto uma ou outra particularidade. Vocês verão que o foco maior é em comandos, mas fiquem tranquilos porque eu destrinchei um por um para que vocês não tenham dificuldades. Vamos lá...

PROFESSOR DIEGO CARVALHO - WWW.INSTAGRAM.COM/PROFESSORDIEGOCARVALHO



Galera, todos os tópicos da aula possuem Faixas de Incidência, que indicam se o assunto cai muito ou pouco em prova. *Diego, se cai pouco para que colocar em aula?* Cair pouco não significa que não cairá justamente na sua prova! A ideia aqui é: se você está com pouco tempo e precisa ver somente aquilo que cai mais, você pode filtrar pelas incidências média, alta e altíssima; se você tem tempo sobrando e quer ver tudo, vejam também as incidências baixas e baixíssimas. *Fechado?*

INCIDÊNCIA EM PROVA: BAIXÍSSIMA

INCIDÊNCIA EM PROVA: BAIXA

INCIDÊNCIA EM PROVA: MÉDIA

INCIDÊNCIA EM PROVA: ALTA

INCIDÊNCIA EM PROVA: ALTÍSSIMA

Além disso, essas faixas não são por banca – é baseado tanto na quantidade de vezes que caiu em prova independentemente da banca e também em minhas avaliações sobre cada assunto...

#ATENÇÃO

Avisos Importantes

O curso abrange todos os níveis de conhecimento...

Esse curso foi desenvolvido para ser acessível a **alunos com diversos níveis de conhecimento diferentes.** Temos alunos mais avançados que têm conhecimento prévio ou têm facilidade com o assunto. Por outro lado, temos alunos iniciantes, que nunca tiveram contato com a matéria ou até mesmo que têm trauma dessa disciplina. A ideia aqui é tentar atingir ambos os públicos - iniciantes e avançados - da melhor maneira possível..

Por que estou enfatizando isso?

O **material completo** é composto de muitas histórias, exemplos, metáforas, piadas, memes, questões, desafios, esquemas, diagramas, imagens, entre outros. Já o **material simplificado** possui exatamente o mesmo núcleo do material completo, mas ele é menor e bem mais objetivo. *Professor, eu devo estudar por qual material?* Se você quiser se aprofundar nos assuntos ou tem dificuldade com a matéria, necessitando de um material mais passo-a-passo, utilize o material completo. Se você não quer se aprofundar nos assuntos ou tem facilidade com a matéria, necessitando de um material mais direto ao ponto, utilize o material simplificado.

Por fim...

O curso contém diversas questões espalhadas em meio à teoria. Essas questões possuem um comentário mais simplificado porque **têm o único objetivo de apresentar ao aluno como bancas de concurso cobram o assunto previamente administrado**. A imensa maioria das questões para que o aluno avalie seus conhecimentos sobre a matéria estão dispostas ao final da aula na lista de exercícios e **possuem comentários bem mais completos, abrangentes e direcionados.**

LINUX

Contexto Histórico

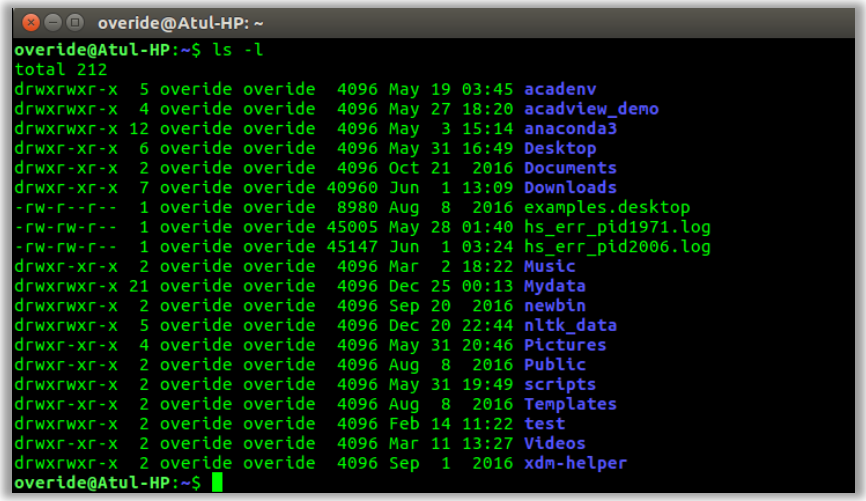
INCIDÊNCIA EM PROVA: BAIXA

Um Sistema Operacional é basicamente um software especial executado em seu computador para possibilitar a inicialização e a execução de programas. Atualmente, existem diversos sistemas operacionais no mercado! **Um deles é maduro, eficiente, poderoso, moderno e atual – além de acompanhar todas as mudanças tecnológicas de hardware e software dos últimos quarenta anos**! Não, eu não estou falando do Windows! Eu estou falando sobre o Unix...

Professor, você escreveu errado: é Linux! Não, galera... estou falando sobre o Unix! **O Unix é um sistema operacional multitarefa e multiusuário, disponível para diversas plataformas de hardware**. Ele foi criado no final da década de 1960, quando os computadores eram grandes, caros e de difícil acesso a pessoas comuns. Logo, era imprescindível desenvolver um sistema operacional multiusuário, multitarefa e que funcionasse em diferentes computadores.

O Unix multiusuário, isto é, permitia que vários usuários utilizassem o mesmo computador ao mesmo tempo por meio de terminais remotos. Ademais, **ele era multitarefa**, isto é, permitia que vários programas fossem executados simultaneamente. Não é só isso: ele apresenta uma vasta gama de possibilidades relacionadas à rede, com o sistema de cota de disco, FTP, e-mail, WWW, DNS, possibilidade de diferentes níveis de acesso, de executar programas em background etc.

No início, o Unix era distribuído gratuitamente pela AT&T Corporation para as universidades. Mais tarde, porém, percebendo o sucesso do Unix no meio comercial, a empresa passou a disponibilizá-lo por um preço muito alto. Já os departamentos de ciência da computação de diversas universidades no mundo inteiro começaram a desenvolver programas comerciais para o Unix, gerando um grande número de usuários e desenvolvedores de utilitários e programas.



Até meados da década de 1980, o Unix ainda não possuía uma interface gráfica própria. *Como assim, professor?* Galera, vocês – assim como eu – são jovens, então vocês não sabem o que é um sistema operacional sem interface gráfica, somente com linha de comando! O Professor Renato da Costa – que tem o dobro da minha idade – conhece muito bem, porque ele já trabalhou muito com isso. **Então, para quem nunca viu, eu**

mostro abaixo um sistema operacional somente com linha de comando – não tem ícone, não tem janela, não tem nada disso...

Claro que, com o passar do tempo, ele implementou uma interface gráfica bonitinha e tal. **Ele se tornou referência na comunidade tecnológica, em grande parte devido ao seu design elegante, a sua simplicidade e a sua portabilidade**. E foi o sistema operacional mais popular e relevante do mundo até o início da década de noventa, quando o Microsoft Windows começou a se popularizar. No entanto, foi com as distribuições de Linux que o Unix teve mais influência.

Professor, não gosto desse tal de Unix! Parece antigo, feio e coisa desses nerds de informática! Pequeno gafanhoto, você sabe qual sistema operacional que roda nesses dois computadores acima? Mac OS – um sistema operacional baseado em Unix! **Então, diga não ao preconceito contra o Unix – ele é um excepcional sistema operacional e merece uma chance**. Uma outra versão do Unix é o tema da nossa aula de hoje: Linux! Vamos ver um pouquinho sobre ele...

Como afirma o Professor Rubem E. Ferreira: "*Linux é um clone de Unix criado como uma alternativa barata e funcional para quem não está disposto a pagar o alto preço de um sistema Unix comercial ou não tem um computador suficientemente rápido*". Vamos contar melhor essa história: havia um pesquisador chamado Richard Stallman que fundou a *Free Software Foundation*, **uma organização para difundir softwares cujo código-fonte é livre para qualquer um acessar, estudar, copiar e modificar**.

Um dos projetos da Free Software Foundation era o Projeto GNU, que tinha como finalidade criar uma cópia melhorada e livre do sistema operacional Unix, mas que não utilizasse seu código-fonte. Como nós já vimos, o Unix era bastante caro na época. Pessoal, era um projeto extremamente ambicioso e trabalhoso: havia a necessidade de desenvolver o Kernel, Utilitários de Programação, Administração de Sistema, Administração de Rede, Comandos, entre outros.

Uma pequena pausa: **kernel é o núcleo de um Sistema Operacional – ele é o responsável por se comunicar e controlar o hardware**! Para quem gosta de carros, eu gosto de dizer que o Kernel é como o chassi de um veículo, isto é, a estrutura que suporta outros componentes. Da mesma forma que o chassi – que pode ser utilizado em vários carros –, o kernel pode ser utilizado em vários sistemas operacionais. Na imagem abaixo, vemos dois carros com o mesmo chassi!

Voltando à história: no final da década de 1980, o projeto havia fracassado – apenas alguns utilitários de programação e alguns comandos estavam prontos; o kernel, nada. Nessa mesma época, havia outras iniciativas paralelas: **Dr. Andrew Tanenbaum desenvolveu o Minix – um clone do Unix – como instrumento de ensino para seus alunos sobre os princípios estruturais dos sistemas operacionais**.

No entanto, o Minix era limitado para um processador bastante específico e tinha dificuldades em lidar com mais de uma tarefa. *Professor, desenrola!* Calma, estamos chegando lá! No final da década de oitenta, **um garoto de 22 anos chamado Linus Benedict Torvalds – aluno da Universidade de**

Helsinque – percebeu que havia um processador que seria perfeito para executar o clone do Unix: Intel 80386.

Galera, pensem numa aposta bem-feita: esse processador revolucionou o mundo naquela época. Perguntem sobre ele para alguém da sua família com um pouco mais de idade que tenha adquirido um computador no início da década de noventa! Ele provavelmente te falará com saudade: "*Claro que eu lembro. Meu primeiro computador foi um 386!*"! Certeza que depois disso cairá uma lágrima dos olhos dele...

Linus – esse senhor educado da foto ao lado – estava disposto a construir um kernel clone do Unix que possuísse memória virtual, multitarefa preemptiva e capacidade de multiusuários. **Era um trabalho gigantesco e, na prática, impossível para apenas uma pessoa concluí-lo, ainda que estivesse familiarizada com as complexidades dos sistemas operacionais**. Na primavera de 1991, ele iniciou seu projeto particular, inspirado no seu interesse pelo Minix. Depois de algum tempo de trabalho em seu projeto solitário, conseguiu criar um kernel capaz de executar os utilitários de programação e os comandos do Unix

Reconhecendo que não conseguiria continuar a desenvolver sozinho o Linux, ele enviou pela internet a seguinte mensagem de desafio para uma lista de discussão:

==========

De: torvalds@klaava.Helsinki.FI (Linus Benedict Torvalds)
Lista de Discussão: comp.os.minix
Assunto: O que você gostaria que o Minix tivesse?
Resumo: Pequena votação sobre o meu Sistema operacional.
Message-ID: <1991Aug25.205708.9541@klaava.Helsinki.FI>
Data: 25 Aug 1991 20:57:08 GMT
Organização: University of Helsinki

Olá pessoal por aí usando minix –

Estou fazendo um sistema operacional (gratuito) (apenas um hobby, não será grande e profissional como o gnu) para 386 (486) clones da AT. Isso vem crescendo desde abril, e está começando a ficar pronto. Eu gostaria de receber qualquer feedback sobre coisas que as pessoas gostam / não gostam no Minix, já que meu sistema operacional se parece um pouco (mesmo layout físico do sistema de arquivos (devido a razões práticas) entre outras coisas). No momento, tenho portado o bash (1.08) e o gcc (1.40), e as coisas parecem funcionar. Isso significa que vou conseguir algo prático em alguns meses e gostaria de saber quais recursos a maioria das pessoas desejaria. Todas as sugestões são bem-vindas, mas eu não prometo que vou implementá-las :-)
Linus (torvalds@kruuna.helsinki.fi)

Obs: Sim, está livre de qualquer código minix e tem um fs multi-threaded. Não é portável (usa a alternância de tarefas do 386, etc), e provavelmente nunca irá suportar nada além de discos rígidos AT, já que é tudo que eu tenho :-(.
==========

Em 5 de outubro de 1991, Linus Torvalds lançou a primeira versão de seu sistema operacional: Linux 0.02. A partir dessa data, muitos programadores no mundo inteiro têm colaborado e ajudado

a fazer do Linux o sistema operacional que é atualmente. E foi assim que um cara lá nos cafundós gelados da Finlândia criou um Sistema Operacional e hoje – trinta anos depois – você tem que estudá-lo para passar em um concurso público :-)

Conceitos Básicos

INCIDÊNCIA EM PROVA: MÉDIA

Quando se liga um computador, o sistema operacional é acionado, possibilitando inicializar o hardware e gerenciá-lo, tornando possível sua utilização pelo usuário – ele é descarregado da memória quando o computador é desligado. O Linux – sistema operacional bastante difundido atualmente e adotado por grandes empresas – é multitarefa, multiusuário e multiprocessamento, além de possuir memória virtual por paginação, bibliotecas compartilhadas, etc.

Ao contrário de um software proprietário, ele é um software livre, cujo código-fonte está aberto e disponível sob a Licença GPL (*General Public License*) para que o usuário possa ter acesso ao código-fonte com o intuito de utilizá-lo executá-lo, estudá-lo, modificá-lo e distribuí-lo livremente de acordo com os termos da licença. **Ele é desenvolvido, em geral, por uma comunidade de programadores voluntários espalhados pelo mundo que contribuem para melhorá-lo**.

Ele é um exemplo de sistema operacional livre amplamente difundido que pode ser utilizado tanto em servidores de grandes empresas – onde ele é mais frequente – quanto em computadores pessoais, passando por diversas arquiteturas ou plataformas de hardware diferentes. **Existem muitas questões que querem enganar o aluno dizendo que ele só funciona em servidores**. MEN-TI-RA! Ele funciona tanto em servidores quanto em computadores pessoais.

Há muitas coisas que as pessoas acham que não podem ser feitas no Linux, tais como: permitir a conexão de pendrive, acessar a internet por meio do Protocolo TCP/IP, realizar backup, instalar aplicativos de armazenamento em nuvem (Ex: Google Drive, Dropbox, etc)... tem gente que acha que Linux não tem nem interface gráfica com papel de parede, ícones, menus, etc. **Galera, Linux tem tudo isso, tem internet, tem navegadores, arquivos, pastas, etc**. *Bacana?*

Professor, é verdade que o Linux não é seguro? Não, esse é outro mito! **O Linux não é mais ou menos seguro que outros sistemas operacionais – como MS-Windows**. Lembrem-se: todos os sistemas operacionais são vulneráveis a eventuais softwares maliciosos. *Beleza?* Dito isso, vamos finalizar essa parte de conceitos básicos do nosso sistema operacional visualizando uma tabela com várias características do Linux. Vejam só:

CARACTERÍSTICAS DO LINUX
É multitarefa, isto é, o sistema pode executar mais de uma aplicação ao mesmo tempo.
É multiusuário, isto é, um mesmo computador pode ter várias contas de usuário.
É preemptivo, isto é, permite a interrupção de processos.

Suporta nomes extensos de arquivos e pastas (255 caracteres).
Conectividade com outros tipos de plataformas como: Apple, Sun, Macintosh, Sparc, Unix, Windows, DOS, etc.
Utiliza permissões de acesso a arquivos, pastas e programas em execução na memória RAM.
Proteção entre processos executados na Memória RAM.
Modularização: ele só carrega para a memória o que é utilizado durante o processamento, liberando totalmente a memória, assim que o programa/dispositivo é finalizado.
Não há a necessidade de se reiniciar o sistema após modificar a configuração de qualquer periférico de computador ou parâmetros da rede – exceto em falhas de hardware.
Em geral, não necessita de um processador potente para funcionar.
Suporta diversos dispositivos e periféricos disponíveis no mercado, tanto os novos como os obsoletos.
Possui controles de permissão de acesso (Login e Logout).

Interface Gráfica

INCIDÊNCIA EM PROVA: BAIXA

Vamos falar um pouco sobre interface de usuário! *O que é isso, professor?* Galera, é o meio de interação entre humanos e máquina. Todas as pessoas quando utilizam um computador estão se comunicando por meio de uma interface de usuário. Em geral, essas interfaces podem ser de dois tipos: **CLI** (***Command Line Interface***) e **GUI** (***Graphic User Interface***). Basicamente, a primeira é uma Interface de Linha de Comando e a segunda é a uma Interface Gráfica.

Quando a informática ainda engatinhava, os computadores funcionavam em uma interface de linha de comando em que o usuário digitava uma série de instruções, o computador as processava e retornava com as informações processadas para o usuário. Os comandos eram simples, diretos e precisos, mas apresentavam um inconveniente: era necessário que o usuário soubesse todos esses comandos ou procurassem em um manual (geralmente em inglês).

Foi aí que a Apple e a Microsoft desenvolveram uma interface gráfica que – aliada ao mouse – eliminou as barreiras e a necessidade de usar comandos e preparou o mundo da informática para o que é hoje: ícones, janelas, menus, botões, etc. **O computador se tornava mais amigável e convidativo, e eliminava-se a necessidade de aprender todos os comandos do computador – além de abrir portas para novos tipos de aplicativos**.

Com o surgimento da interface gráfica, o terminal de comandos foi ficando cada vez mais escondido dos usuários, agindo de vez em quando nos bastidores ou pulando por alguns segundos numa instalação mais complexa ou outra. Apesar disso, o terminal ainda é uma ferramenta muito

forte e presente em sistemas operacionais UNIX (Ex: Mac OS e Linux). **Isso causa um certo ar de desconfiança e desdém por quem utiliza Windows – especialmente a geração mais nova**.

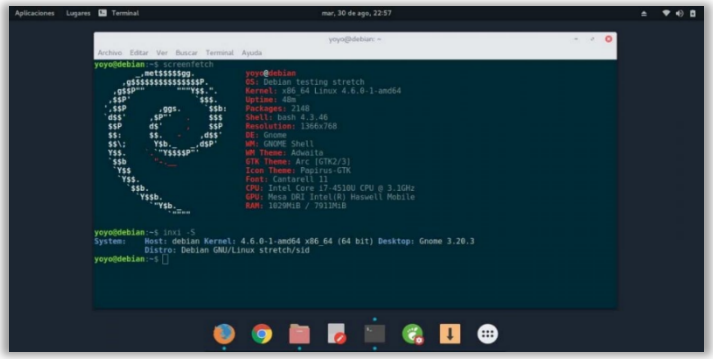
De todo modo, a Microsoft simplesmente esqueceu dele e apostou todas as suas cartas na Interface Gráfica. **Hoje em dia, ambas as interfaces coexistem na imensa maioria dos sistemas operacionais**. Bem... poucas pessoas sabem, mas as interfaces gráficas geralmente possuem um nome. *Ah é, professor?* Sim, a interface gráfica do Windows 7 se chamava Aero; do Windows 8 se chamava Metro; e atualmente a interface gráfica do Windows 10 se chama Fluent.

Só que – quando nós chegamos no Universo Linux – devemos lembrar que o código é livre para qualquer pessoa desenvolver recursos e funcionalidades. Diferentemente do Windows em que você não pode escolher qual interface utilizar, o Linux permite que você escolha entre diversas interfaces gráficas diferentes. Existe uma variedade imensa de interfaces gráficas – algumas mais leves que outras, outras são mais elegantes, outras são mais funcionais e assim por diante.

Dentre as opções de interface gráfica (também chamadas de Ambiente Gráfico ou Ambiente X), podemos destacar: **Gnome, KDE, XFCE, Unity, LXDE, Mate, Cinnamon, OpenBox, BlackBox, Window Maker, etc, etc, etc**. Calma, galera... nós não vamos ver em detalhes todas elas em nossa aula! As únicas interfaces gráficas relevantes para provas de concurso são as quatro primeiras. Então, vamos fazer alguns exercícios e depois vamos detalhá-las...

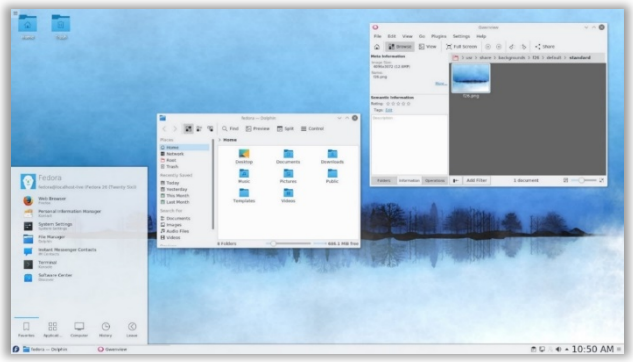
GNOME

GNOME é um projeto de software livre abrangendo o ambiente gráfico para os usuários e os desenvolvedores. O projeto dá ênfase especial a usabilidade, acessibilidade e internacionalização. É um dos mais populares do mundo e não se limita apenas à interface, mas a diversas aplicações que compõem toda ela.



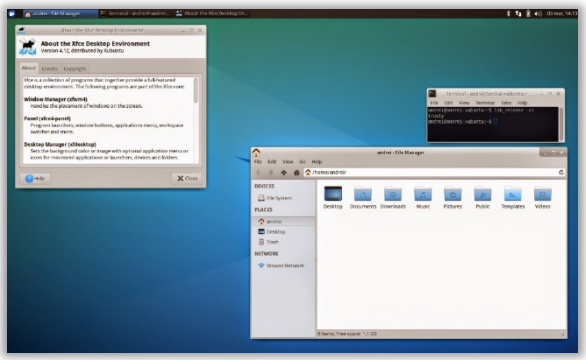
KDE

KDE é um ambiente gráfico multiplataforma mais similar ao Windows, **portanto é o mais comum quando se migra do Windows para o Linux**. Ele é altamente configurável e flexível.



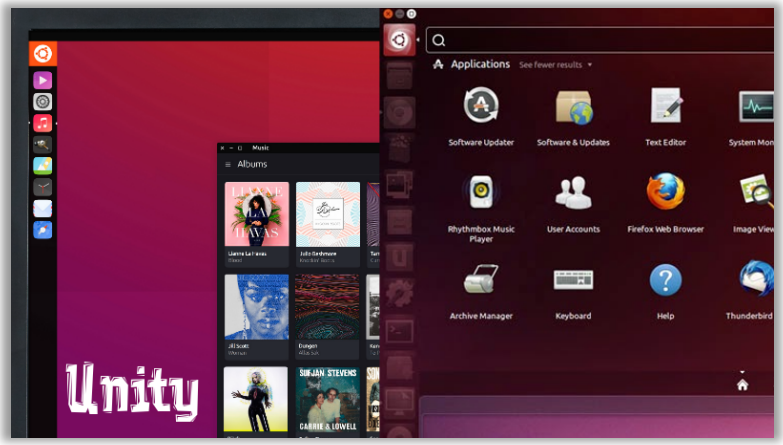
XFCE

Trata-se de um ambiente gráfico que pretende ser rápido e leve, enquanto ainda é visualmente atraente e fácil de usar – **além de incorporar a filosofia UNIX de modularidade e reutilização**.



Unity

Unity é uma interface para o ambiente desktop. **Ela foi desenhada inicialmente para fazer um uso mais eficiente do espaço das telas limitadas dos netbooks**, porém devido ao sucesso tornou-se a interface padrão do Ubuntu até 2017.



Rotinas de Inicialização

INCIDÊNCIA EM PROVA: BAIXÍSSIMA

Algumas pessoas ficam até interessadas em experimentar o Linux, mas pensam que – para isso – elas devem se desfazer do Windows. Galera, isso não é necessário! *Como assim, professor?* **Você pode ter diversos sistemas operacionais em seu computador**. Claro que você não troca de sistema operacional com um simples atalho de teclado – lembrem-se que o sistema operacional é um software grande e pesado que ocupa grande parte da memória do seu computador.

Então, como é feito? Pessoal, vocês já sabem que um disco rígido (*Hard Disk* – HD) é aquele disco que armazena dados em seu computador mesmo após você desligá-lo da tomada. **Esse disco é dividido em milhões de setores, que são pequenas áreas para armazenamento de dados**. Todo disco rígido possui um setor específico chamado setor de inicialização, também conhecido como *Master Boot Record* (MBR) – Registro Mestre de Inicialização.

Esse setor armazena um código executável que funciona como um carregador do sistema operacional instalado. *Como é, professor?* Voltemos para o exemplo dos carros: *como você faz para ligar o motor bem potente e pesado de um carro em segundos?* Você insere a chave e inicia o sistema de ignição do carro. Esse sistema é responsável por fazer com que aquele motor pesado, potente e frio saia da inércia e, em segundos, esteja funcionando a todo vapor.

A MBR armazena um software (código executável) semelhante ao sistema de ignição! **Dessa forma, toda vez que você liga um computador, a primeira coisa carregada na memória é esse código executável capaz de inicializar o sistema operacional desejado**. Agooooora vem o pulo do gato: se você possui mais de um sistema operacional instalado em seu computador, um software exibirá em tela uma lista de sistemas operacionais para que você possa escolher qual você deseja inicializar. *Computação é lindo demais, gente...*

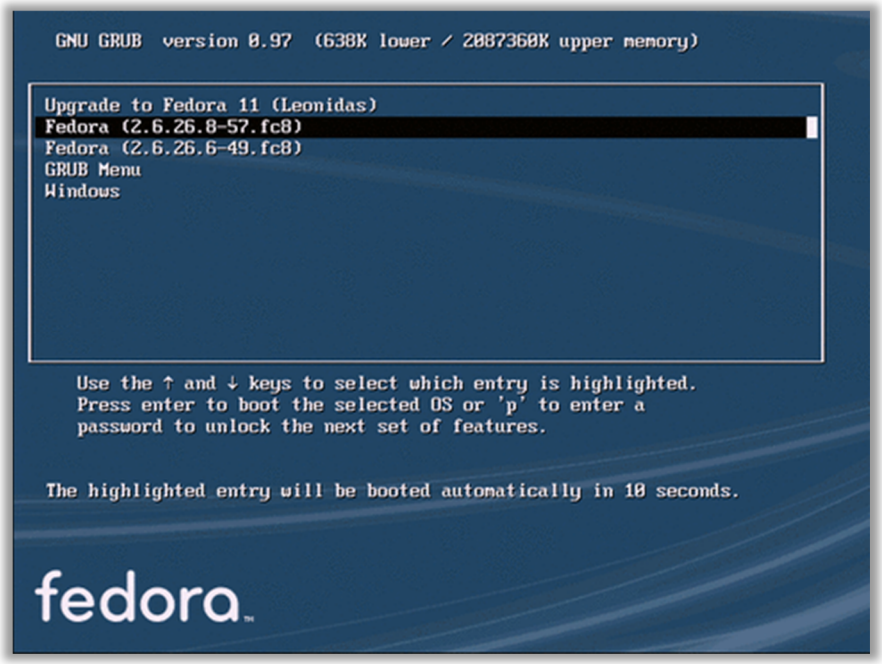
Galera, esse procedimento é chamado de Dual Boot, porque na maioria dos casos possui duas opções de sistemas operacionais. Bem, já vou adiantando para vocês que esses softwares gerenciadores de inicialização ou partida – conhecidos como *Boot Loader* – não são bonitinhos: **eles são acessados por meio de uma interface de linha de comando**. Nós veremos a seguir um pouquinho sobre os principais softwares de inicialização do Universo Linux: LILO e GRUB!

> **Gerenciadores de Inicialização (Boot Loaders):** são programas que carregam um sistema operacional e/ou permitem escolher qual será iniciado. Normalmente estes programas são gravados no setor de boot (inicialização) da partição ativa ou no Master Boot Record (MBR) do disco rígido.

LILO

O **LILO** (***LI****nux* ***LO****ader*) é o gerenciador de inicialização mais antigo – vejam que ele tem uma interface mais simples e rústica. Era o carregador de boot mais popular para Linux até 2001, quando seu concorrente – GRUB – começou a substituí-lo. Ele permite selecionar qual sistema operacional será iniciado (caso você possua mais de um) e funciona em diversos tipos de discos rígidos. Existem muito mais coisas a se falar sobre ele, mas não cai em concurso público:-)

GRUB



O **GRUB** (***GR****and* ***U****nified* ***B****ootloader*) é um gerenciador de inicialização mais recente – notem como ele é mais bonito e moderno que o anterior. Ele é mais poderoso que o LILO e suporta um número ilimitado de entradas de sistemas operacionais, além de permitir sistemas de arquivo maiores. Trata-se de um gerenciador de inicialização flexível, funcional e poderoso, podendo inicializar diversos sistemas operacionais diferentes, com diversos sistemas de arquivos diferentes.

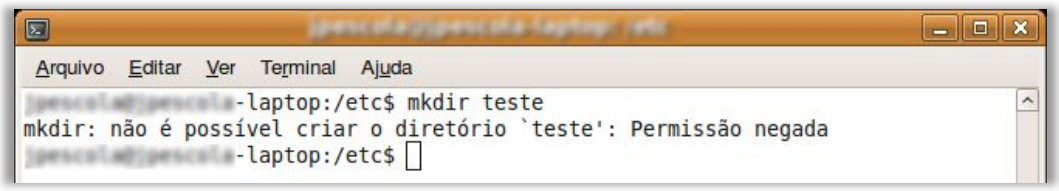
Tipo de Usuário

INCIDÊNCIA EM PROVA: BAIXA

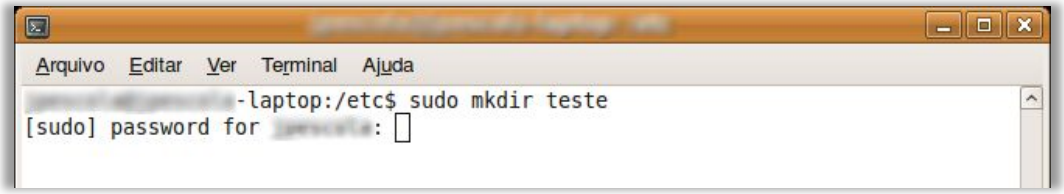
Um usuário é alguém que possui uma identificação no sistema. Essas informações permitem ao Linux controlar como o acesso é garantido aos usuários e o que eles podem fazer depois de obter a permissão de acesso. Vamos conhecer os três tipos de usuário do Linux:

TIPO DE USUARIO	DESCRIÇÃO
USUÁRIO COMUM	São aqueles que possuem contas para utilização do sistema operacional. Basicamente, esses usuários possuem um diretório base (/home/username, exemplo) e podem criar e manipular arquivos em seu diretório, além de executar tarefas simples como criar e editar documentos, navegar na internet, ouvir música etc. Ao contrário do usuário administrador, o usuário comum é inviabilizado para realização de algumas tarefas a nível de sistema. Em geral, vem com um símbolo de cifrão ($) na linha de comando.
USUÁRIO ADMINISTRADOR (ROOT)	Também chamado de Root, é responsável por controlar todo o sistema e não possui quaisquer tipos de restrições. Sempre que executado algum software ou atividade que precise de acesso administrativo, é necessário o root, que é chamado por meio do comando sudo. Por exemplo: sempre que for instalar um programa ou realizar um upgrade de todo o sistema operacional, é utilizado o comando sudo para se ter as permissões de root e conseguir efetuar essas tarefas. Em geral, vem com um símbolo de cerquilha (#) na linha de comando.
USUÁRIO DE SISTEMA	Usuários que não necessitam estar logados no sistema para controlar alguns serviços. Estes comumente não possuem senhas e, diferentemente dos usuários comuns, não se conectam. São contas usadas para propósitos específicos do sistema e não são de propriedade de uma pessoa em particular. Um exemplo desse tipo de usuário é o www-data, que pode ser utilizado para controlar servidores web como Apache e Nginx.

Todos os usuários conseguem listar os conteúdos dos diretórios, mas somente o Usuário Root pode criar arquivos e/ou pastas em um diretório diferente de seu diretório pessoal. Isso quer dizer que, se eu quiser logar no sistema como um usuário comum, somente poderei criar arquivos e/ou pastas em meu diretório pessoal, ou seja, o diretório **/home/<usuário>.** Dito isso, vamos ver alguns exemplos: **eu – como usuário comum – tentando criar uma pasta em outro diretório**.



Vejam que a permissão foi negada! Agora notem o que ocorre quando tentamos criar a pasta como administrador (root) usando o comando **sudo**:



O terminal de linha de comando vai solicitar a sua senha pessoal e, depois que você digitá-la, ele vai executar o comando solicitado. Nesse momento, você pode pensar: *Ué, professor! Se qualquer usuário que digite o comando sudo poderá executar os comandos avançados, do que adianta?* Bom saber que vocês estão atentos à aula! Belíssima pergunta...

Na realidade, o que ocorre é que o usuário que instalou o sistema operacional na máquina tem direitos de executar comandos como administrador, porque – em tese – ele é responsável pela máquina, ele que instalou tudo, etc. Por essa razão, ele pode executar comandos como o sudo. **Se você posteriormente pensar em criar um novo usuário no computador, você notará que ele não conseguirá executar comandos com o sudo**. *Entendido, pessoal?* Bom demais...

Distribuições Linux

INCIDÊNCIA EM PROVA: BAIXA

Galera, o Linux não é como o Windows! *Como assim, professor?* Cara, a Microsoft combina internamente todos os bits do Windows para produzir cada nova versão do Windows e o distribui como um único pacote. Se você quiser o Windows, precisará escolher uma das versões que a Microsoft oferece. **O Linux funciona de maneira diferente: ele não é produzido por uma única organização. Diferentes organizações e pessoas trabalham em diferentes partes**.

Nós temos o Kernel (núcleo do Sistema Operacional), os Utilitários de Shell (interpretador de comandos), o Servidor X (que produz um desktop gráfico), o Ambiente de Desktop (que roda no Servidor X para fornecer uma área de trabalho gráfica), etc. Galera, serviços do sistema, programas gráficos, comandos do terminal - **muitos deles são desenvolvidos de forma independente um do outro e são todos softwares de código aberto**.

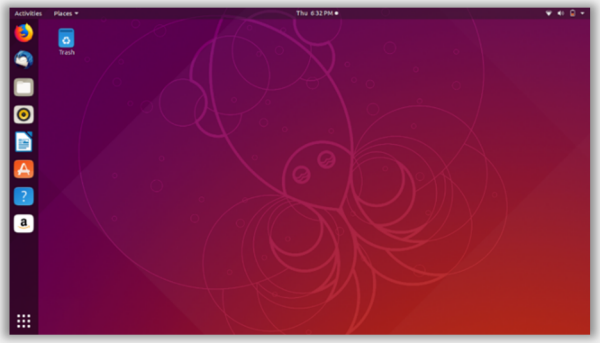
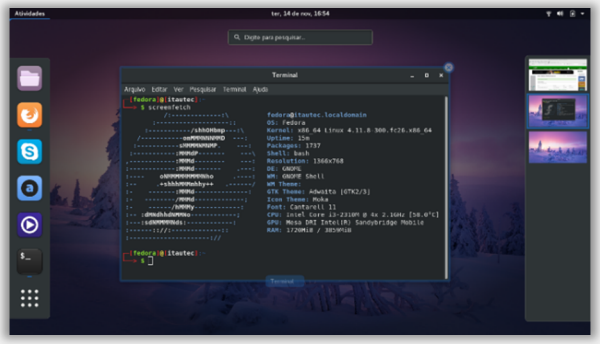
Se você quisesse, poderia pegar o código-fonte do kernel do Linux, os Utilitários de Shell, o Servidor X e todos os outros programas em um Sistema Operacional Linux – montando tudo sozinho. **No entanto, compilar o software levaria muito tempo - sem mencionar o trabalho envolvido em fazer com que todos os diferentes programas funcionassem corretamente juntos**.

Uma Distribuição Linux faz o trabalho pesado para você! Ela pega todo o código dos projetos de código aberto, compilando-o e combinando-o em um único sistema operacional que você pode instalar e inicializar. Eles também fazem escolhas para você, como escolher o ambiente de desktop padrão, o navegador, etc. A maioria das distribuições adiciona seus próprios toques finais para gerar uma identidade de cada distribuição.

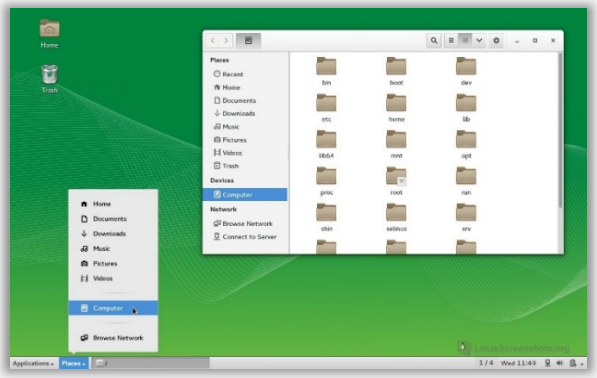
Portanto, o que é uma distribuição? **Trata-se de um sistema operacional criado a partir de uma coleção de software construído sobre o Kernel do Linux**. Cada distribuição possui recursos que a tornam única. Algumas distribuições são projetadas para uso geral, enquanto outras são projetadas para um caso de uso muito específico, como um firewall ou um servidor da Web. A escolha da distribuição que funciona melhor para você pode levar algum tempo.

As distribuições podem ser comerciais, isto é, o usuário paga pelo sistema e recebe um suporte técnico; ou podem ser livres, isto é, não há cobrança pelo uso, qualquer um pode fazer o download, estudar, executar, modificar, distribuir, entre outros. **Claro, você não terá suporte técnico gratuito, no entanto existe uma comunidade gigantesca no mundo para ajudá-lo, caso você necessite**. *Bacana, não é?*

As principais distribuições atualmente são: Debian, Ubuntu, RedHat, Fedora, Suse, Mint, CentOS, Mandrake, Slackware, Kurumin, Conectiva, Kalango e Mandriva (as quatro últimas são brasileiras).

DISTRIBUIÇÃO	DESCRIÇÃO
	**Distribuição Ubuntu:** Sem dúvida nenhuma, é a distribuição mais popular no mundo devido ao fato de ser uma das mais amigáveis e fáceis de instalar, utilizar e de se obter ajuda para resolver problemas. Ela se preocupa muito com o usuário final de computadores desktop. Originalmente baseada no Debian, diferencia-se além do foco no desktop, em sua forma de publicação de novas versões, que são lançadas semestralmente.
	**Distribuição Debian:** Trata-se de uma distribuição baseada Projeto GNU, conhecida por ser a distribuição mais estável e segura do Linux. No Debian, cada pacote para por inúmeros testes até ser considerado estável. Algumas pessoas que estão começando a usar podem considerar Debian uma distribuição complexa demais para iniciantes. Ela serviu de base para a criação de diversas outras distribuições populares, tais como Ubuntu e Kurumin.
	**Distribuição Fedora:** Trata-se de uma das mais populares e estáveis distribuições que existem atualmente. Ele era, no começo, uma bifurcação para a comunidade, liberado e mantido pela gigante RedHat que, na época, estava fechando seu sistema e concentrando-se no mercado corporativo. Isso significa que, desde o princípio, o Fedora já contava com o que há de mais moderno em tecnologia de software, assim

como também contava com uma das mais competentes equipes em seu desenvolvimento.



Distribuição Suse:

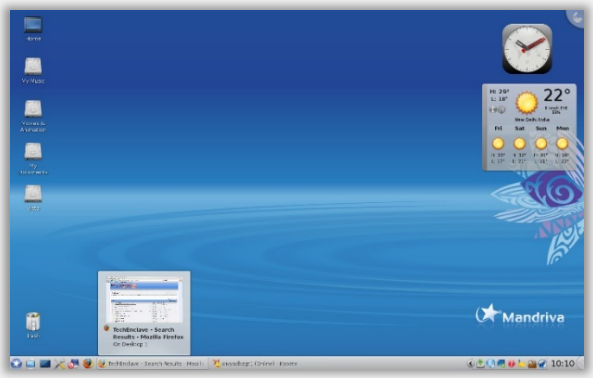
Trata-se de uma das distribuições Linux mais antigas e, assim como o RedHat, sua história está mesclada com os primeiros passos no Linux no mundo empresarial. O openSUSE, a versão comunitária do SUSE Enterprise Linux, numa relação semelhante ao que o Red Hat tem com o CentOS, porém ainda mais unificado entre os projetos, que são basicamente o mesmo sistema. O openSUSE pode ser utilizado tanto como Desktop, como em servidores.

Distribuição Mint:

É uma das distribuições Linux preferidas dos usuários iniciantes no Linux e também é considerada uma das distribuições mais fáceis de usar. Ela é uma distribuição baseada no Ubuntu com o qual é totalmente compatível e partilha os mesmos repositórios. Diferencia-se do Ubuntu por incluir drivers e codificadores proprietários por padrão e por alguns recursos que permitem fazer em modo gráfico configurações que no Ubuntu são feitas de modo texto.

Distribuição CentOS:

É uma distribuição de classe Enterprise derivada de códigos fonte gratuitamente distribuídos pela RedHat Enterprise Linux e mantida pelo CentOS Project. CentOS proporciona um grande acesso aos softwares padrão da indústria, incluindo total compatibilidade com os pacotes de softwares preparados especificamente para os sistemas da RHEL. Isso lhe dá o mesmo nível de segurança e suporte, através de updates, que outras soluções enterprise, porém sem custo.

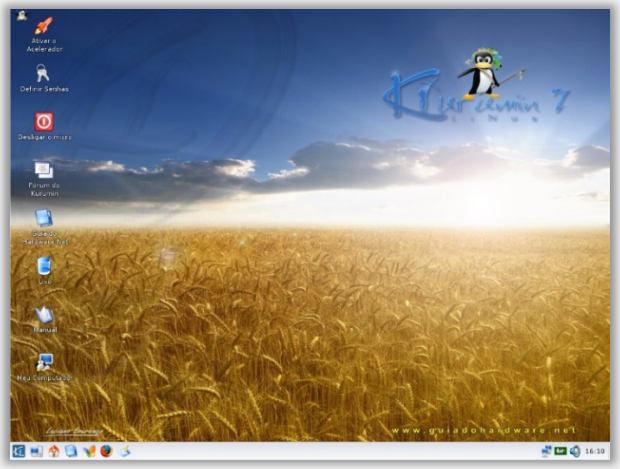


Distribuição Mandriva:

É uma das maiores distribuições da atualidade, nasceu da fusão entre o antigo Mandrake e a brasileira Conectiva. A Empresa Francesa Mandriva se dedica à distribuição e suporte do sistema operacional Mandriva, tem sua sede administrativa em Paris e um centro de desenvolvimento em Curitiba, no Brasil. O Mandriva conta também com um grande número de contribuidores pelo mundo, o público-alvo do Mandriva Linux engloba usuários iniciantes no mundo Linux assim como usuários com mais experiência.

Distribuição Slackware:

Junto com Debian e RedHat, é uma das distribuições que deram origem a todas as outras. Embora seja considerada por muitos uma distribuição difícil de se usar, voltada para usuário expert ou hacker, possui um sistema de gerenciamento de pacotes simples, assim como sua interface de instalação, que é uma das poucas que continua em modo-texto, mas nem por isso se faz complicada.



Distribuição Kurumin:

Trata-se de uma distribuição brasileira desenvolvida pela equipe do Guia do Hardware e colaboradores, com o objetivo de ser um sistema fácil de usar, voltado especialmente para iniciantes e ex-usuários do Windows. Todos os componentes e scripts usados são abertos, o que possibilitou também o surgimento de versões modificadas do sistema. Apesar de ter feito um grande sucesso e ter sido durante algum tempo uma das distribuições mais usadas no país, o projeto acabou falhando em atrair um grupo de desenvolvedores interessados em participar de forma ativa do desenvolvimento, sendo descontinuado em 2008.

Gerenciamento de Sistemas de Arquivos

Sistemas de Arquivos

INCIDÊNCIA EM PROVA: BAIXA

Todo sistema operacional necessita de uma estrutura que possa dar suporte a ele para acessar e ler informações contidas no disco rígido. O recurso que constrói uma base lógica estrutural para o sistema operacional é o sistema de arquivos. *Como assim, professor?* **Um sistema de arquivos é uma espécie de gerenciador e organizador que permitirá ao sistema operacional ler os arquivos que estão no disco rígido – esta é a finalidade básica de um sistema de arquivos**.

Imagine a seguinte situação: **digamos que você necessite de um determinado documento que está armazenado em um depósito com milhões de outros documentos, mas você não pode pegá-lo diretamente – você deve fazê-lo por meio de um atendente**. Dessa forma, você recorre ao atendente e este lhe mostra exatamente onde está o documento que você deseja dentro dessa estrutura lógica de organização dos arquivos. O sistema de arquivos é como esse atendente! :)

O Sistema de Arquivos permite gravar, ler, localizar, remover e realizar funções em um dispositivo de armazenamento – em geral, um disco rígido. Galera, a maioria dos usuários Unix/Linux já foram

ou ainda são usuários Windows. Nesse sistema operacional, existem basicamente três sistemas de arquivos: FAT16, FAT32 e NTFS. **No Linux, existem muito mais opções, tais como: EXT2, EXT3, EXT4, RaiserFS, etc**. Vamos vê-los a seguir...

EXT2

Um dos primeiros sistemas de arquivos utilizado nas primeiras versões do Linux foi o EXT2 (*Second Extended FileSystem*) – embora ele tenha sido uma espécie de padrão não era muito eficiente.

EXT3

Trata-se de uma versão do EXT2, porém com suporte a *journaling*[1]. Essa característica foi uma evolução e tornou o EXT3 um sistema de arquivos muito estável e robusto.

EXT4

Este é uma espécie de versão do EXT3 que surgiu com a prerrogativa de melhorar o desempenho de compatibilidade, formatos e limites de armazenamentos.

ReiserFS

Criado recentemente e suportado por quase todas as distribuições, apresenta ótima performance, principalmente para um número muito grande de arquivos pequenos.

Por fim, é importante mencionar o gerenciador de arquivos e pastas ao utilizar a interface Gnome é Nautilus (assim como o Windows Explorer é o gerenciador de arquivos e pastas do Windows).

PRINCIPAIS CARACTERÍSTICAS
No Linux, dispositivos de armazenamento são representados por diretórios cuja posição na hierarquia de diretórios é definida no momento de montagem, por exemplo: /mnt/floppy, /mnt/cd-rom.
No Linux, podemos utilizar nomes de arquivos com até 255 caracteres e mais de um ponto, por exemplo: Programa1.src.tar.gz.
O Linux, como qualquer sistema operacional Unix, diferencia letras maiúsculas de minúsculas, portanto, relatório, RELATÓRIO e Relatório são arquivos diferentes.
É possível inserir espaços no meio do nome dos arquivos.
Não há extensões compulsórias como .COM e .EXE para programas, .BAT para arquivos de lote, .BAK para backup ou outras.

1 Característica que dá permissão ao sistema operacional de manter um log (*journal*) de todas as mudanças no sistema de arquivos antes de escrever os dados no disco e assim permitindo recuperar o disco após um desastre em uma velocidade maior.

Estrutura de Diretórios

INCIDÊNCIA EM PROVA: MÉDIA

O Linux organiza seus diretórios em uma estrutura hierárquica conhecida como árvore de diretórios que segue o **Padrão FHS (*Filesystem Hierarchy Standard*)**[2]. *Como assim, professor?* É mais ou menos assim: arquivos relacionados a dispositivos de hardware ficam situados no diretório **dev**; arquivos relacionados a bibliotecas essenciais ficam situados no diretório **lib**; arquivos relacionados a arquivos de configuração se situam no diretório **etc**.

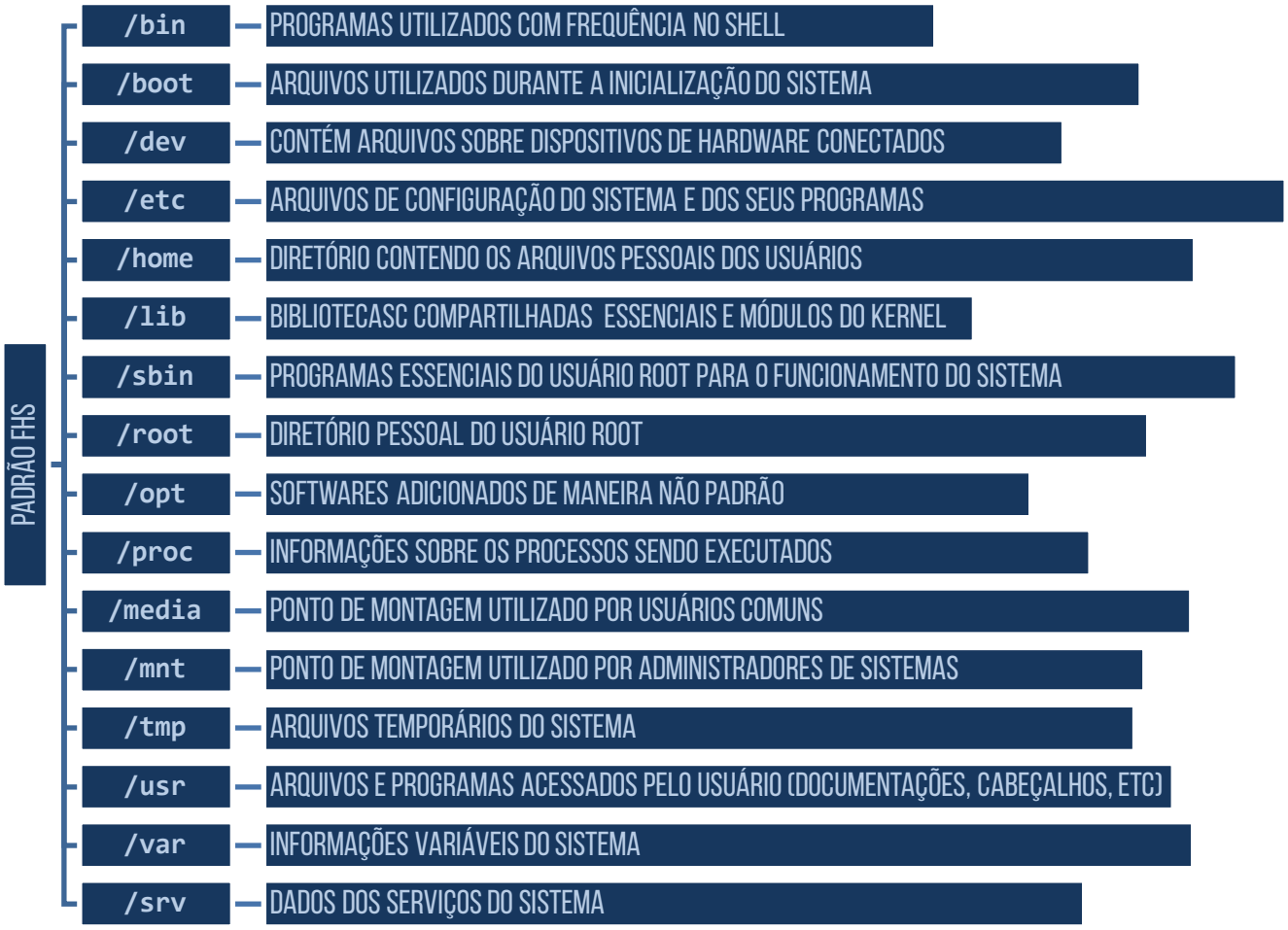
Galera, é claro que cada diretório possui outros subdiretórios, mas vocês não precisam conhecê-los para prova. Basicamente vocês precisam conhecer apenas os subdiretórios do diretório raiz "/" que estão apresentados abaixo! *Como assim diretório raiz, professor?* Galera, eu falei que é uma estrutura de árvore e, em uma árvore, tudo tem origem na raiz. **Aqui é a mesma coisa: todos os subdiretórios têm origem no diretório raiz**. Então vejam só...

DIRETÓRIO	DESCRIÇÃO
/	Trata-se do diretório raiz do Linux. É aqui que encontrará todos os diretórios e todos os dados que se encontram em seu sistema. Até mesmo um CD/DVD, disco externo ou qualquer outro periférico se encontram dentro da raiz do sistema. Na linha de comandos, você navega para o root digitando cd /.
/bin	Trata-se do diretório onde ficam guardados arquivos binários que têm de estar acessíveis a todos os utilizadores do sistema. Estes arquivos binários são programas que o próprio sistema inicia de forma autónoma.
/boot	Trata-se do diretório que contém arquivos necessários para a inicialização do sistema.
/dev	Trata-se do diretório onde ficam arquivos especiais associados aos dispositivos do sistema. Estes ficheiros são especiais porque representam os dispositivos do sistema. Por exemplo: um disco rígido do sistema aparecerá como /dev/sda.
/etc	Trata-se do diretório onde se encontram todos os arquivos globais de configuração do sistema. Na sua grande maioria, estes arquivos podem ser editados com o uso de um simples editor de texto. Repare que neste diretório encontram-se arquivos de configuração do sistema e, não, de um usuário específico. Os arquivos de configuração de um usuário específico encontram-se no diretório home de cada utilizador.
/home	Trata-se do diretório onde encontramos os arquivos de cada usuário existente no sistema. Sempre que adicionamos um novo usuário, por exemplo, com o nome profdiego2 no diretório /home, é criado um arquivo para este usuário como /home/profdiego2/. Dentro desse diretório, ficam todos os arquivos de configurações específicas para aquele usuário, bem como todos os seus arquivos de dados.
/lib	Trata-se do diretório onde estão armazenadas as bibliotecas compartilhadas no sistema. Estas bibliotecas podem variar de acordo com a distribuição utilizada e podem ser, por exemplo, bibliotecas

[2] É importante mencionar que é Case Sensitive, logo diferencia letras maiúsculas de letras minúsculas (Ex: **/teste** é diferente de **/Teste**). É importante mencionar também que – ao contrário do MS-DOS –, ele utiliza barra (/) e, não, contrabarra (\). Por fim, é importante saber que não pode existir dois arquivos com o mesmo nome em um diretório, ou um subdiretório com um mesmo nome de um arquivo em um mesmo diretório.

	de linguagens como Perl, Python, C, etc. É também neste diretório que estão os módulos do Kernel do Sistema Operacional.
/mnt	Trata-se do diretório em que os administradores de sistema montam sistemas de arquivos temporários enquanto os utilizam. Por exemplo: se você estiver montando uma partição do Windows para executar algumas operações de recuperação de arquivos, você pode montá-lo em /mnt/windows. No entanto, você pode montar outros sistemas de arquivos em qualquer lugar no sistema.
/proc	Trata-se do diretório onde se encontram arquivos especiais associados aos processos do sistema. Estes arquivos são especiais porque representam os processos em funcionamento no sistema. Por exemplo: haverá um arquivo que fornece informação sobre o funcionamento do processador ou sobre outras operações que ocorram no sistema.
/root	Trata-se do diretório do superusuário do sistema. Este diretório não é a mesma coisa que o diretório raiz do sistema – de onde descendem todos os restantes diretórios. Trata-se, na verdade, de um diretório dedicado ao superusuário root.
/sbin	Trata-se do diretório destinado aos arquivos binários que são utilizados pelo superusuário root e para administração do sistema. Pode-se dizer que este diretório é semelhante ao /bin, mas com a peculiaridade de serem programas que normalmente não serão utilizados por usuários com permissões limitadas. Este diretório pode não existir em um sistema e pode também substituir o diretório /bin.
/tmp	Trata-se do diretório onde encontramos os arquivos temporários do sistema. Estes arquivos são normalmente gerados pelo sistema e, como o nome indica, permanecem no sistema durante um período limitado de tempo. Por exemplo: sempre que instalamos um programa, este utiliza o diretório /tmp/ para colocar arquivos que serão necessários durante a instalação, mas que não voltarão a ser necessários.
/usr	Trata-se do diretório onde estão arquivos e programas utilizados pelos usuários existentes no sistema. No caso dos programas, no diretório /usr/bin ficam todas as aplicações que não são essenciais ao sistema e, por conseguinte, não se encontram no diretório /sbin ou /bin . No caso dos programas que ficam no diretório /usr/bin, as bibliotecas associadas a estes sistemas ficam localizadas no diretório /usr/lib.
/var	Trata-se do diretório onde ficam diversos arquivos de dados vindos das contas de usuários. É neste diretório que são colocadas bases de dados locais pertencentes a programas instalados pelos utilizadores.
/boot	Trata-se do diretório onde se encontram variados arquivos necessários para a inicialização do sistema operacional. É neste diretório, por exemplo, que podemos encontrar os arquivos BootLoader – responsáveis por gerir a inicialização do sistema.
/opt	Trata-se do diretório que contém subdiretórios para pacotes de software opcionais. É comumente usada por softwares proprietários, que não obedecem à hierarquia do sistema de arquivos-padrão. Por exemplo: um programa proprietário pode colocar seus arquivos em /opt/aplicativo quando você instalá-lo.
/media	Trata-se do diretório que contém subdiretórios em que os dispositivos de mídia removível inseridos no computador são montados. Por exemplo: quando você insere um CD em seu sistema Linux, um

	diretório será criado automaticamente dentro do diretório /media. Você pode acessar o conteúdo do CD dentro desse diretório.
/srv	Trata-se do diretório que contém dados para serviços prestados pelo sistema. Se você usa o servidor Apache em um site, por exemplo, provavelmente armazena os arquivos do seu site em um diretório dentro do /srv.



Gerenciamento de Privilégios

INCIDÊNCIA EM PROVA: MÉDIA

Galera, uma das razões que torna o Linux um sistema operacional bastante seguro é a sua exigência de que cada recurso tenha dono com permissões de uso específicas. Dessa forma, para que seja possível restringir ou permitir o acesso e o uso de determinados recursos a uma ou mais pessoas, é necessário que cada uma dessas pessoas tenha sido devidamente cadastrada como um usuário no sistema operacional.

Criar uma conta para cada usuário no sistema operacional não é útil apenas para restringir ou permitir o acesso aos recursos oferecidos, mas também para respeitar o espaço que cada pessoa tem. **Por meio de uma conta, um usuário poderá ter os seus próprios diretórios, personalizar o**

seu desktop, criar atalhos customizados, fazer configurações específicas para os seus programas preferidos, entre outros.

Além disso, mesmo que o computador seja utilizado apenas por uma pessoa, é recomendável criar um usuário próprio para ela. *Por que se já existe o usuário root?* Porque o usuário root é o que "manda" no sistema – ele tem poderes de administrador e tem acesso a todos os recursos. Logo, usá-lo no dia-a-dia não é recomendável, visto que – se o computador for tomado por outra pessoa ou se o próprio usuário fizer algo errado – , **o sistema operacional poderá ser comprometido**.

Isso serve de barreira para programas mal intencionados. **Ele impede – por exemplo – que um malware apague um arquivo que não deve, envie arquivos especiais para outra pessoa ou forneça acesso da rede para que outros usuários invadam o sistema.** O Linux também impede que usuários mal intencionados instalem programas enviados por terceiros sem saber para que eles realmente servem e causem danos irreversíveis em seus arquivos, seu micro ou sua empresa.

O Gerenciamento de Privilégios permite ao administrador do sistema definir políticas para acesso aos arquivos, diretórios e programas executáveis do sistema. Como vimos, os arquivos são organizados em diretórios e, portanto, o Linux oferece facilidades de proteção dos arquivos e diretórios. Essas proteções são organizadas em três classes de privilégios: privilégios do dono, privilégios de um grupo e privilégio dos outros usuários. Vamos ver isso melhor...

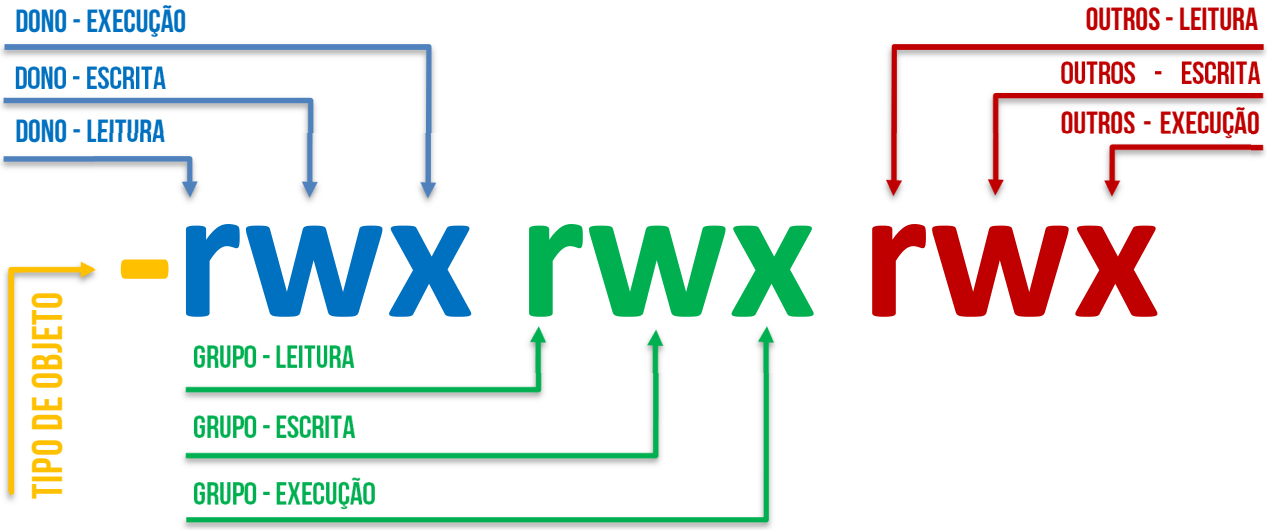
PRIVILÉGIO	DESCRIÇÃO
DONO	É a pessoa que criou o arquivo ou o diretório. O nome do dono do arquivo/diretório é o mesmo do usuário usado para entrar no sistema GNU/Linux. Somente o dono pode modificar as permissões de acesso do arquivo. As permissões de acesso do dono de um arquivo somente se aplicam ao dono do arquivo/diretório. A identificação do dono também é chamada de User ID (UID).
GRUPO	Permite que vários usuários diferentes tenham acesso a um mesmo arquivo (já que somente o dono poderia ter acesso ao arquivo). Cada usuário pode fazer parte de um ou mais grupos e então acessar arquivos que pertençam ao mesmo grupo que o seu (mesmo que estes arquivos tenham outro dono). Um usuário pode pertencer a um ou mais grupos.
OUTROS	Trata-se da categoria de usuários que não são donos ou não pertencem ao grupo do arquivo.

Em suma: dono do arquivo é normalmente aquele que criou o arquivo; o grupo é um conjunto de usuários que tem acesso a um determinado arquivo; e outros usuários são todos aqueles que não donos de arquivos ou que pertençam a um grupo. **Além disso, cada classe de privilégio é composta por três níveis básicos de permissões: permissão de leitura, permissão de escrita e permissão de execução**. Vamos ver em detalhes:

PERMISSÃO	DESCRIÇÃO
LEITURA [R]	Permissão de leitura de arquivos e listagem de conteúdo em diretórios.

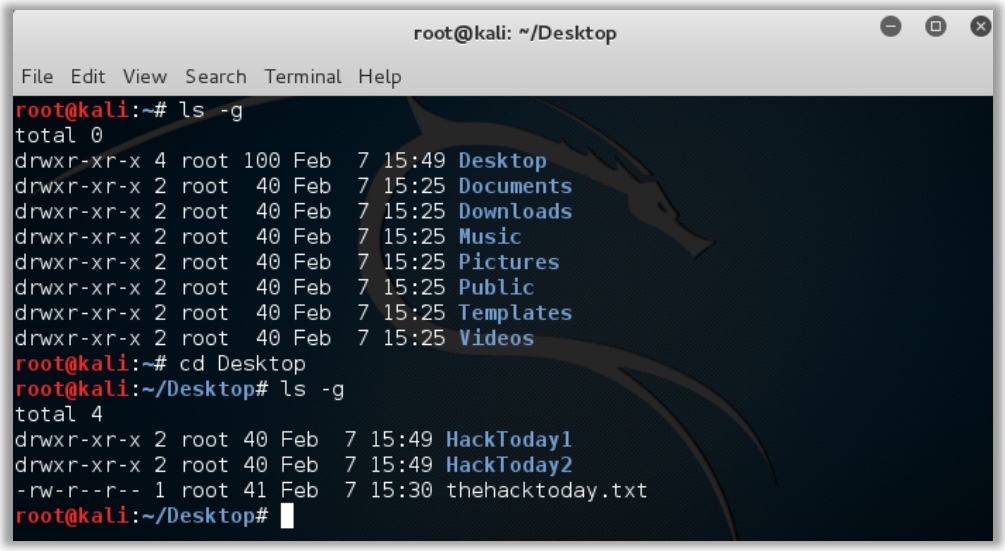
ESCRITA [W]	Permissão de escrita em arquivos ou diretórios (inclusive deleção).
EXECUÇÃO [X]	Permissão de execução de arquivos ou de acesso a diretórios.

Beleza! Nós vimos que existem três classes de privilégios e esses privilégios possuem três níveis de permissões. *Agora como eu sei que determinado arquivo, por exemplo, pode ser acessado ou modificado ou executado pelo dono, grupo ou outros usuários?* **Para isso, necessário identificar alguns dados sobre esse arquivo no terminal de linha de comando**. Todo arquivo terá um conjunto de 10 caracteres que trarão todas essas informações.



TIPOS DE OBJETO					
d	DIRETÓRIO	b	ARQUIVO DE BLOCO	c	ARQUIVO ESPECIAL DE CARACTERE
p	CANAL	s	SOCKET	-	ARQUIVO NORMAL

Professor, não entendi muito bem! Galera, vejam só: existe um comando capaz de listar arquivos detalhados de um diretório ou arquivo pelo terminal de linha de comando. **Entre esses detalhes, há os dez caracteres que eu mencionei anteriormente que indica as classes de privilégio e níveis de permissão**. *Vamos ver um exemplo?* Na imagem abaixo, é possível ver que o diretório Vídeos está listado como **drwxr-xr-x**. *O que isso significa?*



Podemos ver que é um diretório (d); podemos ver que o dono tem permissão de leitura (r), escrita (w) e execução (x); podemos ver que o grupo tem permissão para leitura (r) e execução (x), mas não de escrita (-); podemos ver que outros usuários também têm permissão para leitura (r) e execução (x), mas não de escrita (-). **Existe um comando chamado chmod responsável pela atribuição de permissões de arquivos no terminal**. Resumindo...

(1) A primeira letra diz qual é o tipo de objeto. Se for (d) é um diretório, se for (l) é um link, se for (-) é um arquivo comum, e assim por diante conforme a tabela da página anterior;

(2) Da segunda à quarta letra (rwx), exibe qual é a permissão de acesso ao dono do arquivo. Neste caso, o root tem a permissão de leitura (r), escrita (w) e execução (x).

(3) Da quinta à sétima letra (r-x), exibe qual é a permissão de acesso ao grupo do arquivo. Neste caso, todos que pertencem ao grupo têm permissão apenas de leitura (r) e execução (x).

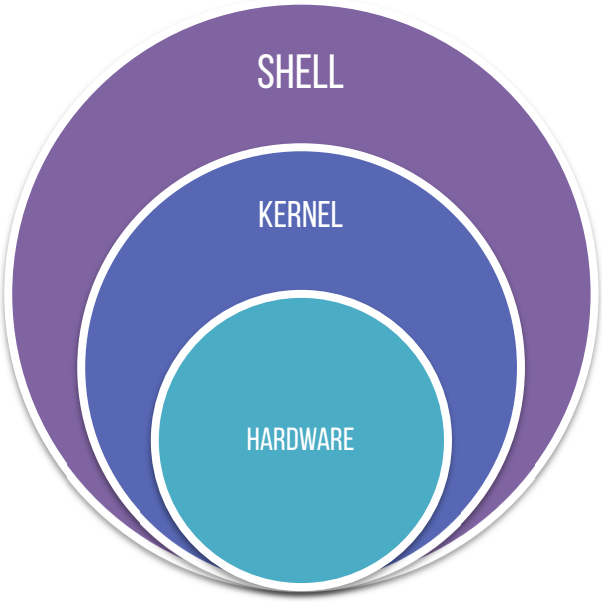
(4) Da oitava à décima letra (r-x), exibe qual é a permissão de acesso de outros usuários ao arquivo. Neste caso, todos os outros usuários têm permissão apenas de leitura (r) e execução (x).

Principais Comandos

Galera, para entender os principais comandos, nós precisamos entender o conceito de Shell! **Esse é o nome dado a uma classe de programas que funcionam como interpretador de comandos e linguagem de programação interpretada no Unix**. Não vamos tratar aqui de linguagens de programação, o nosso foco é no interpretador de comandos. Pessoal, vocês sabem que computadores só fazem o que nós mandamos fazer.

Computadores não pensam por si só! Tudo que eles fazem, são os usuários que mandam ou o que os desenvolvedores programam. Quando você quer ver o que tem dentro do seu pendrive,

você utiliza o mouse para procurar a pasta dele, dá um duplo clique e visualiza o que há na pasta. Se você quiser excluir, copiar, colar, recortar, entre diversas outras possibilidades, você pode fazer tudo isso facilmente por meio de janelas, ícones, menus, entre outros.



No entanto, há outra maneira de realizar todas essas atividades. *Como, professor?* Utilizando o Shell! **Como eu disse, o shell é uma interface de linha de comandos para acessar os serviços de um sistema operacional**[3]. Agora vejam que bacana: *shell* é o mesmo que casca, em inglês. *Vocês sabem o porquê?* Porque ele é a camada mais externa em torno do núcleo do sistema operacional. A imagem acima deixa isso mais claro...

Nós veremos à frente vários comandos diferentes que podem ser utilizados no interpretador de comandos do Linux, mas antes vamos ver que eles se dividem em **comandos internos e externos**.

Comandos Internos

São comandos que estão dentro de um shell interpretador de comandos. Quando o shell é carregado na memória, seus comandos ficam residentes nela. A maior vantagem é a velocidade, pois não precisam ser procurados no disco rígido ou criar processos. Exemplos: cd, alias e logout.

Comandos Externos

São comandos que estão localizados em diretórios específicos no disco rígido, como /bin e /sbin. O Linux precisa consultar o disco rígido sempre que um desses comandos é solicitado. A maioria dos comandos do Linux é externa. Exemplos: ls, cp, rm, mv, mkdir e rmdir.

---

[3] Se a questão não especificar nada, shell é o interpretador de comandos. No entanto, ambientes gráficos de janelas também podem – grosso modo – ser considerados um shell gráfico.

Professor, é verdade que Linux e Windows possuem comandos diferentes para executar uma mesma tarefa? Sim, um exemplo clássico são os comandos **ls** e **dir** (o segundo existe no Linux também).

IMPORTANTE

Comandos podem ser modificados por meio de parâmetros – como veremos nas páginas seguintes – e os parâmetros podem ser combinados de diversas formas. Por exemplo: no exemplo apresentado logo abaixo, todos os comandos retornam exatamente o mesmo resultado apesar da combinação e ordem diferentes.

```
[dieeego88@webminal.org ~]$ls -l -a
[dieeego88@webminal.org ~]$ls -a -l
[dieeego88@webminal.org ~]$ls -la
[dieeego88@webminal.org ~]$ls -al
```

Além disso, o terminal é *case-sensitive*, isto é, ele diferencia maiúsculas de minúsculas tanto para os comandos quanto para os arquivos. Dessa forma, atente-se que **ls** é diferente de **LS**, assim como o **ls -a** é diferente de **ls -A**.

Por fim, antes de prosseguir para os comandos, é importante que vocês conheçam alguns recursos, como os caracteres curingas. *O que é isso, Diego?* **São recursos globais utilizados para especificar um conjunto de arquivos e/ou diretórios de uma única vez, permitindo – assim – manipular vários arquivos/diretórios com um único comando.** Os caracteres curingas mais comuns são *, ?, [] e {}. Para contextualizar a utilização desses caracteres, vamos utilizar o comando **rm**.

Veremos adiante, mas ele é utilizado para remover um arquivo. Consideremos um diretório contendo os oito arquivos. A execução de **rm teste1.txt** removerá esse arquivo do diretório.

DIRETÓRIO

teste1.txt	teste2.txt	teste3.txt	teste4.txt
teste5.txt	teste10.txt	teste20.txt	teste30.pdf

Caractere Asterisco (*): utilizado para substituir um conjunto de caracteres (zero ou mais).

- **Exemplo 1:** rm teste*

Todos os arquivos serão excluídos, porque o asterisco substituirá qualquer quantidade de caracteres após **teste**. Como todos os arquivos começam com **teste**, todos serão excluídos.

- **Exemplo 2:** rm teste*.txt:

Serão excluídos todos os arquivos que tenham o nome **teste**, depois qualquer quantidade de caracteres e depois **.txt**. Logo, sobrará apenas **teste30.pdf** porque começa com **teste**, mas termina com **.pdf**.

- **Exemplo 3:** rm *.pdf:

Serão excluídos todos os arquivos que tenham qualquer quantidade de caracteres, depois **.pdf**. Logo, apenas o arquivo **teste30.pdf** será excluído.

Caractere Interrogação (?): utilizado para substituir uma quantidade específica de caracteres.

- **Exemplo 4:** rm teste?.txt

Serão excluídos apenas os cinco primeiros arquivos porque uma interrogação substitui apenas um caractere. Os cinco primeiros arquivos começam com **teste**, tem um caractere e depois **.txt**.

- **Exemplo 5:** rm teste??.txt

Serão excluídos os arquivos **teste10.txt** e **teste20.txt**, porque duas interrogações substituem dois caracteres e porque eles terminam com **.txt**.

Caractere Colchete [ ]: utilizado para substituir uma faixa de caracteres (letras ou números).

- **Exemplo 6:** rm teste[2-4].txt

Serão excluídos os arquivos **teste2.txt**, **teste3.txt** e **teste4.txt**, uma vez que esse comando excluirá todos os arquivos que iniciem por **teste**, depois tenha os valores **2**, **3** ou **4**, e depois **.txt**.

- **Exemplo 7:** rm teste[2,4].txt

Com a vírgula, serão excluídos os arquivos **teste2.txt** e **teste4.txt**, uma vez que esse comando excluirá todos os arquivos que iniciem por **teste**, depois tenha os valores **2** ou **4**, e depois **.txt**.

Caractere Chave { }: utilizado para substituir caracteres em um conjunto de caracteres.

- **Exemplo 8:** rm teste{2,4}.txt (na prática, é idêntico ao exemplo anterior[4])

[4] Em regra, utiliza-se colchete com o traço para indicar uma faixa de nomes de arquivos; e se utiliza chaves com vírgula para indicar nomes de arquivos específicos. No entanto, se você utilizar a vírgula com o colchete, acabará devolvendo um resultado idêntico à chave com vírgula.

Serão excluídos os arquivos **teste2.txt** e **teste4.txt**, uma vez que esse comando excluirá todos os arquivos que iniciem por **teste**, depois tenha os valores **2** ou **4**, e depois **.txt**.

Agora vamos falar um pouquinho sobre Redirecionamento (> e >>) e Pipe (|). **Galera, nós utilizamos o caractere > para redirecionar a saída padrão de um programa, comando ou script para algum dispositivo ou arquivo ao invés do dispositivo de saída padrão.** *Galera, qual é o dispositivo de saída padrão de um computador?* A tela do computador! É por meio dela que ele exibe a saída de seus processamentos.

Quando é usado com arquivos, este redirecionamento cria ou substitui o conteúdo do arquivo. Por exemplo, o comando **ls** (veremos adiante) é utilizado para listar arquivos de um diretório no dispositivo de saída padrão (tela). Ao executar **ls > listagem.txt**, o resultado da listagem (saída) não será apresentado em tela, mas enviado para um arquivo chamado **listagem.txt** (e, se este arquivo já existir, será sobrescrito).

Ao executar **ls >> listagem.txt**, o resultado da listagem (saída) não será apresentado em tela, mas enviado para um arquivo chamado listagem.txt (e se este arquivo já existir, será adicionado ao final do arquivo). Já o Pipe é utilizado para enviar a saída de um comando para a entrada do próximo comando a fim de dar continuidade ao processamento – os dados enviados são processados pelo próximo comando que mostrará o resultado do processamento.

Por exemplo: ao executar **ls > listagem.txt | sort**, sabemos que o resultado da listagem será enviado para um arquivo chamado **listagem.txt** e isso é enviado como entrada para o comando **sort** (que é responsável por ordenar um conjunto de dados em ordem alfabética). Em suma: a principal diferença entre o **|** e o **>** é que o Pipe envolve processamento entre comandos (saída de um comando vira entrada do próximo) e o **>** redireciona a saída de um comando para um arquivo ou dispositivo.

Pronto, nós vimos os principais recursos que são utilizados em conjunto com os principais comandos do terminal do Linux. Nos próximos tópicos veremos cada um desses comandos em ordem de incidência em prova. *Fechou?* Agora a última coisa (eu juro!) – para quem não pode/quer instalar o Linux, vocês podem testar os comandos seguintes por meio de um terminal online em **www.webminal.org/terminal**. Basta cadastrar e rodar...

Comando ls

INCIDÊNCIA EM PROVA: ALTÍSSIMA

Trata-se de um comando que exibe o conteúdo de diretórios (**ls** = **l**ist **s**ource). Esse comando pode ser substituído também pelo comando **dir** (presente também no Windows) e funcionará da mesma forma. Por ser o comando mais frequente em provas, vamos vê-lo com mais detalhes...

Sintaxe: ls [parâmetros] [caminho/arquivo] [caminho1/arquivo1] ...

PARÂMETROS COMUNS	DESCRIÇÃO
ls	Permite conferir uma lista com os arquivos contidos no diretório, sem maiores detalhes, sem que sejam exibidas informações como, tamanho dos arquivos, data de modificação, entre outros.

```
root@host [/home/codigofonte]# ls
./                  codigofonte.jpg      codigofonte.php
../                 codigofontejpg.jpg   codigofonte_teste.php
codigofonte_2.jpg   .codigofonte.php     codigofonte.txt
root@host [/home/codigofonte]#
```

PARÂMETROS COMUNS	DESCRIÇÃO
ls -l	Lista os arquivos utilizando o formato longo dos nomes dos arquivos, mostrando detalhes sobre permissões, tamanho, tipo, etc (**-l** = **l**ong).

```
root@host [/home/codigofonte]# ls -l
total 540
drwxr-xr-x   2 root root   4096 Feb 11 11:26 ./
drwx--x--x. 26 root root   4096 Feb 11 11:17 ../
-rw-r--r--   1 root root  27886 Feb  5 13:50 codigofonte_2.jpg
-rw-r--r--   1 root root 211156 Dec  1 10:43 codigofonte.jpg
-rw-r--r--   1 root root 216423 Dec  1 10:23 codigofontejpg.jpg
-rw-r--r--   1 root root   1809 Feb 11 11:19 .codigofonte.php
-rw-r--r--   1 root root   1809 Feb 11 11:19 codigofonte.php
-rw-r--r--   1 root root  56141 Feb 11 11:20 codigofonte_teste.php
-rw-r--r--   1 root root  19899 Feb 11 11:19 codigofonte.txt
root@host [/home/codigofonte]#
```

PARÂMETROS COMUNS	DESCRIÇÃO
ls -a	Lista todos os arquivos de um diretório, inclusive os arquivos ocultos (**-a** = **a**ll). Galera, existe também o comando ls -A, que faz quase a mesma coisa, mas não exibe o diretório atual (./) e o de nível anterior (../) entre os arquivos listados.

```
root@host [/home/codigofonte]# ls -a
./                  codigofonte.jpg      codigofonte.php
../                 codigofontejpg.jpg   codigofonte_teste.php
codigofonte_2.jpg   .codigofonte.php     codigofonte.txt
root@host [/home/codigofonte]#
```

PARÂMETROS COMUNS	DESCRIÇÃO
ls -t	Lista os arquivos por ordem de data de modificação. Arquivos que foram modificados por último, mais recentemente, são exibidos em primeiro lugar (**-t** = **t**ime).

```
root@host [/home/codigofonte]# ls -t
codigofonte.txt   codigofonte_teste.php   codigofonte_2.jpg
codigofonte.php   .codigofonte.php        codigofonte.jpg
./                ../                     codigofontejpg.jpg
root@host [/home/codigofonte]#
```

PARÂMETROS COMUNS	DESCRIÇÃO
ls -lt	Lista os arquivos por ordem de data de modificação, também exibindo os modificados mais recentemente em primeiro lugar. No entanto, este comando lista tudo com mais detalhes.

```
root@host [/home/codigofonte]# ls -lt
total 540
-rw-r--r--   1 root root  19535 Feb 11 11:34 codigofonte.txt
-rw-r--r--   1 root root   1944 Feb 11 11:34 codigofonte.php
drwxr-xr-x   2 root root   4096 Feb 11 11:26 ./
-rw-r--r--   1 root root  56141 Feb 11 11:20 codigofonte_teste.php
-rw-r--r--   1 root root   1809 Feb 11 11:19 .codigofonte.php
drwx--x--x. 26 root root   4096 Feb 11 11:17 ../
-rw-r--r--   1 root root  27886 Feb  5 13:50 codigofonte_2.jpg
-rw-r--r--   1 root root 211156 Dec  1 10:43 codigofonte.jpg
-rw-r--r--   1 root root 216423 Dec  1 10:23 codigofontejpg.jpg
root@host [/home/codigofonte]#
```

PARÂMETROS COMUNS	DESCRIÇÃO
ls -ltr	Similar ao comando acima com a diferença de listar em ordem inversa, ou seja, os modificados mais recentemente vão ficando para o final da lista (**-r** = **r**everse).

```
root@host [/home/codigofonte]# ls -ltr
total 540
-rw-r--r--   1 root root 216423 Dec  1 10:23 codigofontejpg.jpg
-rw-r--r--   1 root root 211156 Dec  1 10:43 codigofonte.jpg
-rw-r--r--   1 root root  27886 Feb  5 13:50 codigofonte_2.jpg
drwx--x--x. 26 root root   4096 Feb 11 11:17 ../
-rw-r--r--   1 root root   1809 Feb 11 11:19 .codigofonte.php
-rw-r--r--   1 root root  56141 Feb 11 11:20 codigofonte_teste.php
drwxr-xr-x   2 root root   4096 Feb 11 11:26 ./
-rw-r--r--   1 root root   1944 Feb 11 11:34 codigofonte.php
-rw-r--r--   1 root root  19535 Feb 11 11:34 codigofonte.txt
root@host [/home/codigofonte]#
```

PARÂMETROS COMUNS	DESCRIÇÃO
ls -s	O comando abaixo exibe os arquivos de uma pasta com seu tamanho em bloco, sendo (**-s** = **s**ize). Não confundam com o **-S**, que também exibe, porém de forma ordenada por tamanho.

```
root@host [/home/codigofonte]# ls -s
total 540
  4 ./                208 codigofonte.jpg       4 codigofonte.php
  4 ../               212 codigofontejpg.jpg   56 codigofonte_teste.php
 28 codigofonte_2.jpg   4 .codigofonte.php     20 codigofonte.txt
root@host [/home/codigofonte]#
```

PARÂMETROS COMUNS	DESCRIÇÃO
ls -1	Permite fazer com que os arquivos do diretório sejam listados por linha, um em cada linha.

```
root@host [/home/codigofonte]# ls -1
./
../
codigofonte_2.jpg
codigofonte.jpg
codigofontejpg.jpg
.codigofonte.php
codigofonte.php
codigofonte_teste.php
codigofonte.txt
root@host [/home/codigofonte]#
```

PARÂMETROS COMUNS	DESCRIÇÃO
ls -lh	Para obter uma listagem de todos os arquivos que exiba seus respectivos tamanhos de uma forma mais compreensível ou *Human Readable* (humanamente legível). Dessa forma, você pode conferir os tamanhos em Kb, Mb, Gb, etc (**-h** = **h**uman).

```
root@host [/home/codigofonte]# ls -lh
total 540K
drwxr-xr-x   2 root root 4.0K Feb 11 11:26 ./
drwx--x--x. 26 root root 4.0K Feb 11 11:17 ../
-rw-r--r--   1 root root  28K Feb  5 13:50 codigofonte_2.jpg
-rw-r--r--   1 root root 207K Dec  1 10:43 codigofonte.jpg
-rw-r--r--   1 root root 212K Dec  1 10:23 codigofontejpg.jpg
-rw-r--r--   1 root root 1.8K Feb 11 11:19 .codigofonte.php
-rw-r--r--   1 root root 1.9K Feb 11 11:34 codigofonte.php
-rw-r--r--   1 root root  55K Feb 11 11:20 codigofonte_teste.php
-rw-r--r--   1 root root  20K Feb 11 11:34 codigofonte.txt
root@host [/home/codigofonte]#
```

Comando cd

INCIDÊNCIA EM PROVA: ALTÍSSIMA

Este comando permite ao usuário acessar um diretório de trabalho (**cd** = **c**hange **d**irectory). A mudança de diretório pode ser feita de forma sequencial (de diretório pai para diretório filho ou vice-versa) ou pode ser feita de forma aleatória (de um diretório qualquer para outro diretório qualquer). Lembremos que os diretórios utilizam uma estrutura de árvore, logo temos diretório raiz, diretório pai, diretório filho, diretório irmão, entre outros.

Sintaxe: cd [diretório]

PARÂMETROS COMUNS	DESCRIÇÃO
cd ou cd ~	Permite acessar o **diretório home** do usuário.
cd /	Permite acessar o **diretório raiz** do Linux.
cd diretório	Permite acessar um **diretório filho** do diretório atual.
cd .	Permite acessar o próprio **diretório atual** (na prática, não faz nada).
cd ..	Permite acessar o **diretório pai** do diretório atual.
cd ../diretório	Permite acessar um **diretório irmão** do diretório atual.
cd -	Permite acessar o **último diretório** visitado antes do diretório atual.
cd caminho-diretório	Permite acessar **qualquer diretório** quando utilizado o caminho completo.

Comando rm

INCIDÊNCIA EM PROVA: ALTÍSSIMA

Este comando apaga arquivos e também pode ser utilizado para apagar diretórios e sub-diretórios vazios ou que contenham arquivos – é uma forma curta de se referir a **remove** (remover).

Sintaxe: rm [opções] [caminho][arquivo/diretório] [caminho1][arquivo1/diretório1]

PARÂMETROS COMUNS	DESCRIÇÃO
rm -f	Apaga sem pedir confirmação (**-f** = **f**orce).
rm -i	Pede confirmação antes de apagar (**-i** = **i**nteractive).
rm -r	Apaga arquivos e seus subdiretórios (**-r** = **r**ecursive).

EXEMPLOS	DESCRIÇÃO
rm teste.txt	Apaga o arquivo teste.txt no diretório atual.
rm *.txt	Apaga todos os arquivos do diretório atual que terminam com .txt.
rm *.txt teste.novo	Apaga todos os arquivos do diretório atual que terminam com .txt e também o arquivo teste.novo.
rm -rf /tmp/teste/*	Apaga todos os arquivos e sub-diretórios do diretório /tmp/teste, mas mantém o sub-diretório /tmp/teste.
rm -rf /tmp/teste	Apaga todos os arquivos e sub-diretórios do diretório /tmp/teste, inclusive /tmp/teste.

Comando cp

INCIDÊNCIA EM PROVA: ALTA

Esse comando é utilizado para copiar arquivos (**cp** = **c**opy). **O arquivo de origem e o destino da cópia podem residir em sistemas de arquivo diferentes, ou até no mesmo diretório desde que tenham nomes diferentes**. Este comando copia também mais de um arquivo de um diretório para outro. É possível especificar mais de um arquivo no comando cp usando os curingas *, ? e [ ]. Vejamos sua sintaxe básica:

```
Sintaxe: cp [opções] [origem] [destino]
```

PARÂMETROS COMUNS	DESCRIÇÃO
cp -f	Substitui arquivos existentes sem pedir confirmação (**-f** = **f**orce).
cp -i	Pede permissão antes de substituir arquivos existentes (**-i** = **i**nteractive).
cp -r	Copia arquivos e subdiretórios (**-r** = **r**ecursivo).
cp -s	Cria um link simbólico ao invés de copiar (**-s** = **s**imbolic).

EXEMPLOS	DESCRIÇÃO
cp teste.txt teste1.txt	Copia o arquivo teste.txt para teste1.txt.
cp teste.txt /tmp	Copia o arquivo teste.txt para dentro do diretório /tmp.
cp * /tmp	Copia todos os arquivos do diretório atual para /tmp.
cp /bin/* .	Copia todos os arquivos do diretório /bin para o diretório atual.

Comando mkdir

INCIDÊNCIA EM PROVA: ALTA

Este comando é utilizado para criar um diretório no sistema (**mkdir** = **m**ake **dir**ectory). Um diretório é usado para armazenar arquivos de um determinado tipo. O diretório pode ser entendido como uma pasta onde você guarda seus papeis (arquivos). Como uma pessoa organizada, você utilizará uma pasta para guardar cada tipo de documento, da mesma forma você pode criar um diretório vendas para guardar seus arquivos relacionados com vendas naquele local.

```
Sintaxe: mkdir [opções] [caminho/diretório] [caminho1/diretório1] ...
```

PARÂMETROS COMUNS	DESCRIÇÃO
-p	Permite criar diretórios e seus subdiretórios de uma só vez.

EXEMPLOS	DESCRIÇÃO
mkdir /tmp/teste	Cria o diretório /teste em /tmp.
mkdir /teste1 /teste2	Cria o diretório /teste1 e o diretório /teste2.

Comando pwd

INCIDÊNCIA EM PROVA: ALTA

Este comando mostra o nome e o caminho do diretório atual (**pwd** = **p**resent **w**orking **d**irectory). Você pode utilizá-lo para verificar em qual diretório se encontra em determinado momento.

```
Sintaxe: pwd
```

EXEMPLOS	DESCRIÇÃO
pwd	/home/diego

Comando cat

INCIDÊNCIA EM PROVA: ALTA

O comando cat é utilizado para unir, criar e exibir arquivos. Seu nome é uma derivação da palavra inglesa ***concatenate*** ou concatenar.

```
Sintaxe: cat [opções] [diretório/arquivo] [diretório1/arquivo1] ...
```

PARÂMETROS COMUNS	DESCRIÇÃO
-n	Numera todas as linhas (**-n** = **n**umber).
-s	Não mostra mais que uma linha em branco entre um parágrafo e outro (**-s** = **s**queeze).

EXEMPLOS	DESCRIÇÃO
cat > arquivo.txt	Cria um arquivo chamado arquivo.txt.
cat arq1.txt > arq2.txt	Cria ou sobrescreve o conteúdo de arq2.txt com o conteúdo de arq1.

Comando tar

INCIDÊNCIA EM PROVA: MÉDIA

Este comando é utilizado para juntar vários arquivos em um só. O tar também é muito usado para cópias de arquivos especiais ou dispositivos do sistema. É comum encontrar arquivos com a extensão .tar, .tar.gz, .tgz, .tar.bz2, .tar.Z, .tgZ, o primeiro é um arquivo normal gerado pelo tar e

todos os outros são arquivos gerados através tar junto com outros programas de compactação (gzip (.gz), bzip2 (.bz2) e compress (.Z).

```
Sintaxe: tar [opções] [arquivo-destino] [arquivos-origem]
```

PARÂMETROS COMUNS	DESCRIÇÃO
tar -c	Cria um novo arquivo .tar (-c = create).
tar -v	Mostra o prgresso do processamento (-v = verbose)
tar -f	Indica que o resultado será do tipo arquivo (-f = file).

EXEMPLOS	DESCRIÇÃO
tar -cf a1.txt.tar /home	Criar o arquivo a1.txt combinando conteúdo dos arquivos do diretório /home.

Comando mv

INCIDÊNCIA EM PROVA: MÉDIA

Este comando é usado para mover ou renomear arquivos e diretórios (mv = move). O processo é semelhante ao do comando cp, mas o arquivo de origem é apagado após o término da cópia.

```
Sintaxe: mv [opções] [origem] [destino]
```

PARÂMETROS COMUNS	DESCRIÇÃO
-f	Move o arquivo sem pedir confirmação (-f = force).

EXEMPLOS	DESCRIÇÃO
mv teste.txt teste1.txt	Renomeia o arquivo teste.txt para teste1.txt.
mv teste.txt /tmp	Move o arquivo teste.txt para /tmp (o arquivo de origem é apagado após ser movido).
mv teste.txt teste.new	Copia o arquivo teste.txt por cima de teste.new e apaga teste.txt após terminar a cópia.

Comando find

INCIDÊNCIA EM PROVA: MÉDIA

Este comando permite procurar por arquivos/diretórios no disco. Ele pode procurar arquivos através de sua data de modificação, tamanho, etc através do uso de opções.

```
Sintaxe: find [diretório] [opções/expressão]
```

PARÂMETROS COMUNS	DESCRIÇÃO
find -name	Permite pesquisar arquivos pelo seu nome.
find -iname	Permite pesquisar arquivos pelo seu nome, ignorando maiúsculas e minúsculas.
find -size	Permite pesquisar arquivos maiores ou menores que um tamanho específico.

EXEMPLOS	DESCRIÇÃO
find / -name diego	Procura no diretório raíz e sub-diretórios um arquivo/diretório chamado diego.

Comando chmod

INCIDÊNCIA EM PROVA: MÉDIA

Trata-se do comando que permite mudar a permissão de acesso a um arquivo ou diretório. Com este comando você pode escolher se usuário ou grupo terá permissões para ler, gravar, executar um arquivo ou arquivos. Sempre que um arquivo é criado, seu dono é o usuário que o criou e seu grupo é o grupo do usuário (exceto para diretórios configurados com a permissão de grupo "s", será visto adiante) – **chmod** vem de **change mode**.

```
Sintaxe: chmod [opções] [permissões] [diretório/arquivo]
```

PARÂMETROS COMUNS	DESCRIÇÃO
u	Especifica o nível de acesso de um usuário (**-u** = **u**ser).
g	Especifica o nível de acesso de um grupo (**-g** = **g**roup).
o	Especifica o nível de acesso de outros usuários (**-o** = **o**thers).
a	Especifica o nível de acesso de todos os usuários (**-a** = **a**ll).
r	Especifica a permissão de leitura (**-r** = **r**ead).
w	Especifica a permissão de escrita (**-w** = **w**rite).
x	Especifica a permissão de execução (**-x** = **e**xecution).
+	Adiciona permissão.
-	Remove permissão.
=	Define permissão.

EXEMPLOS	DESCRIÇÃO
chmod g+r *	Permite que todos os usuários que pertençam ao grupo dos arquivos (g) tenham (+) permissões de leitura (r) em todos os arquivos do diretório atual.
chmod o-r teste.txt	Retira (-) a permissão de leitura (r) do arquivo teste.txt para os outros usuários (usuários que não são donos e não pertencem ao grupo do arquivo teste.txt).
chmod uo+x teste.txt	Inclui (+) a permissão de execução do arquivo teste.txt para o dono e outros usuários do arquivo.
chmod a+x teste.txt	Inclui (+) a permissão de execução do arquivo teste.txt para o dono, grupo e outros usuários.
chmod a=rw teste.txt	Define a permissão de todos os usuários exatamente (=) para leitura e gravação do arquivo teste.txt.

Vamos ver um pouquinho sobre Modo de Permissão Octal! *Que diabos é isso, Diego?* Galera, é possível utilizar o modo octal para se alterar a permissão de acesso a um arquivo, ao invés de utilizar os modos de permissão +r, -r, +w, -w, entre outros. **O modo octal é um conjunto de oito números onde cada número define um tipo de acesso diferente.** *Professor, por que utilizar o modo octal em vez do modo tradicional?* Porque ele é mais flexível para gerenciar permissões de acesso!

Você pode especificar diretamente a permissão do dono, grupo e outros ao invés de gerenciar as permissões de cada um separadamente. Vejamos a lista de permissões de acesso octal:

VALOR OCTAL	VALOR BINÁRIO	CARACTERES	DESCRIÇÃO
0	000	---	Nenhuma permissão de acesso – equivalente a -rwx.
1	001	--x	Permissão de execução (x).
2	010	-w-	Permissão de gravação (w).
3	011	-wx	Permissão de gravação e execução (wx) – equivalente à permissão 2+1.
4	100	r--	Permissão de leitura (r).
5	101	r-x	Permissão de leitura e execução (rx) – equivalente à permissão 4+1.
6	110	rw-	Permissão de leitura e gravação (rw) – equivalente à permissão 4+2.
7	111	rwx	Permissão de leitura, gravação e execução – equivalente a +rwx (4+2+1).

Você entendeu? Se entendeu mesmo, responda quais são as permissões de acesso do arquivo **teste** dado o seguinte comando:

chmod 764 teste

Os números são interpretados da direita para a esquerda como permissão de acesso aos outros usuários (4), grupo (6), e dono (7). O exemplo acima significa que os outros usuários (4) terem acesso somente leitura (r) ao arquivo teste; o grupo (6) ter a permissão de leitura e gravação (rw); e o dono (7) ter permissão de leitura, gravação e execução (rwx) ao arquivo teste. Se você ficou em dúvida, consulte a tabela anterior. Vamos ver outro exemplo:

chmod 40 teste

Lembre-se novamente que os números são interpretados da direita para a esquerda. Ele define a permissão de acesso dos outros usuários (0) como nenhuma e define a permissão de acesso do grupo (4) como somente leitura (r). **Note que utilizei somente dois números, logo isso significa que a permissão de acesso do dono do arquivo não será modificada.** *Pegadinha, né?* Vamos ver mais um exemplo para fechar o entendimento:

chmod 777 teste

Define-se a permissão de acesso a outros usuários (7), grupo (7) e dono (7) como leitura, gravação e execução (rwx), logo todos podem ler, escrever e executar. Observação: existe também o comando chown, que permite mudar o dono de um arquivo/diretório e, opcionalmente, pode também ser utilizado para modificar o grupo (Ex: **chown diego teste.txt** muda o dono do arquivo **teste.txt** para **diego**). *Fechado?*

Comando grep

INCIDÊNCIA EM PROVA: MÉDIA

Esse comando permite procurar por um texto dentro de um ou mais arquivos ou no dispositivo de entrada padrão. Vejamos sua sintaxe:

Sintaxe: grep [expressão] [arquivo] [opções]

PARÂMETROS COMUNS	DESCRIÇÃO
-f	Especifica que o texto que será localizado está em um arquivo (-f = file).
-i	Ignora a diferença entre maiúsculas e minúsculas.

EXEMPLOS	DESCRIÇÃO
grep "diego" texto.txt	Busca a palavra "diego" no arquivo texto.txt
grep -i command grep	Busca a palavra "command" em um arquivo chamado grep (ignorando a capitalização).

```
root@vps40618123:/# grep -i command grep
How to Use Grep Command in Linux? + some Useful Grep Examples
In this post, we will teach you how to use the grep command in Linux, and reinforce the learning with a useful exam
ple for your daily workflow.
   1. How to Use the grep Command?
The grep command belonging to the Unix family is one of the most versatile and useful tools available. It looks in
a text file for a pattern that we define. In other words, with grep you can search for a word or pattern and the li
ne or lines that have it will be printed.
At first sight, it can be a command of little utility, however, sysadmins that handle many services with various co
nfiguration files, use it to query or search for specific lines within those files.
```

Comando kill

INCIDÊNCIA EM PROVA: MÉDIA

Este comando permite enviar um sinal a um processo em execução. Caso seja usado sem parâmetros, o kill enviará um sinal de término ao processo sendo executado (fechará o programa). Para "matar" um programa ou processo, é necessário saber de antemão o seu PID (**P**rocess **ID**entification Number). Imagine que cada processo/programa em um computador possua um número de identidade – esse número é o PID!

Sintaxe: kill [opções] [sinal] [número]

PARÂMETROS COMUNS	DESCRIÇÃO
-9	Mata o processo imediatamente sem permitir salvar dados.

Caso o usuário deseje finalizar um processo em execução por meio do seu nome ao invés do seu número de identificação, pode utilizar o comando **killall** (Ex: **killall firefox**).

EXEMPLOS	DESCRIÇÃO
kill 500	Mata o processo com PID = 500.
kill -9 500	Mata o processo com PID = 500 sem permitir salvar dados.
kill 123 4567	Mata os processos com PID = 123 e PID = 4567.

Comando free

INCIDÊNCIA EM PROVA: BAIXA

Este comando é utilizado para mostrar detalhes sobre a **utilização da memória principal** do sistema operacional. Vejamos a sua sintaxe:

```
Sintaxe: free [opções]
```

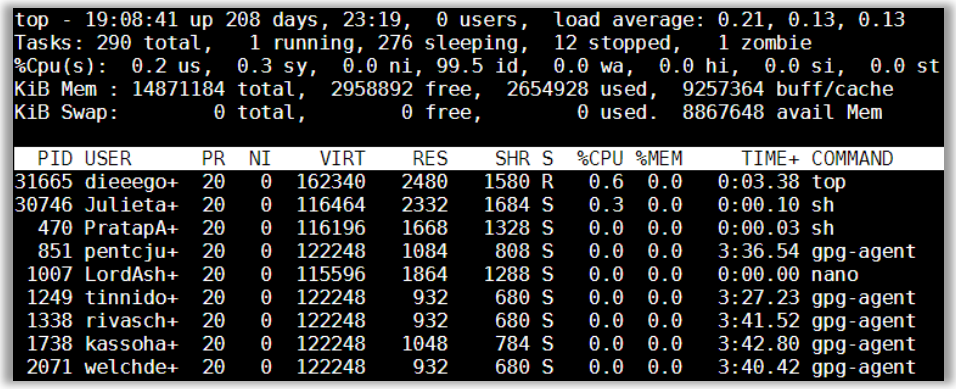
EXEMPLOS	DESCRIÇÃO
free	Exibe um quadro mostrará diversas informações sobre uso da memória.

Comando top

INCIDÊNCIA EM PROVA: BAIXA

Este comando permite mostrar os programas em execução ativos, parados, tempo usado na CPU, detalhes sobre o uso da Memória RAM, Memória Swap, disponibilidade para execução de programas no sistema, entre outros. O top apresenta os resultados da execução de processos em tempo real, exibindo continuamente os processos que estão rodando em seu computador e os recursos utilizados por eles. Para sair do top, basta pressionar a tecla Q.

```
Sintaxe: top [opções]
```



EXEMPLOS	DESCRIÇÃO
top	Mostra resultado semelhante ao apresentado acima.

Comando ps

INCIDÊNCIA EM PROVA: BAIXA

Este comando é utilizado para visualizar quais processos estão sendo executados em um computador, além de exibir qual usuário executou o programa, a hora que o processo foi iniciado, entre outros (ps = process status). *Diego, qual é a diferença do ps para o top?* Galera, a

principal diferença é que o top apresenta estatísticas em tempo real, já o ps é estático. Ele é bastante utilizado também para descobrir qual é o PID de um determinado processo.

Sintaxe: ps [opções]

PARÂMETROS COMUNS	DESCRIÇÃO
ps -a	Mostra todos os processos existentes.
ps -u	Mostra o nome de usuário e hora que um processo foi iniciado.
ps- x	Mostra processos que não foram iniciados pelo terminal.
ps -e	Mostra variáveis de ambiente no momento da inicialização do processo.
ps -f	Mostra a árvore de execução de processos.

```
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root           1  0.0  0.0  33776  3176 ?        Ss   19:21   0:01 /sbin/init
root           2  0.0  0.0      0     0 ?        S    19:21   0:00 [kthreadd]
root           3  0.0  0.0      0     0 ?        S    19:21   0:00 [ksoftirqd/0]
root           4  0.0  0.0      0     0 ?        S    19:21   0:00 [kworker/0:0]
root           5  0.0  0.0      0     0 ?        S<   19:21   0:00 [kworker/0:0H]
root           7  0.0  0.0      0     0 ?        S    19:21   0:01 [rcu_sched]
root           8  0.0  0.0      0     0 ?        S    19:21   0:01 [rcuos/0]
root           9  0.0  0.0      0     0 ?        S    19:21   0:01 [rcuos/1]
root          10  0.0  0.0      0     0 ?        S    19:21   0:00 [rcuos/2]
```

Comando man

INCIDÊNCIA EM PROVA: BAIXA

Esse é o comando permite consultar o manual do sistema (**man** = **man**ual). As páginas de manual acompanham quase todos os programas GNU/Linux. Elas trazem uma descrição básica do comando/programa e detalhes sobre o funcionamento de opção. É mais comum fazer a visualização de uma página de manual em modo texto, com rolagem vertical. Também documenta parâmetros usados em alguns arquivos de configuração.

Sintaxe: man [seção] [comando/arquivo]

PARÂMETROS COMUNS	DESCRIÇÃO
-a	Mostra todas as páginas para o manual requisitado no comando (**-a** vem de **a**ll).
-f	Apresenta apenas uma pequena descrição do comando.

EXEMPLOS	DESCRIÇÃO
man ls	Exibe o manual do comando ls.
man chmod	Exibe o manual do comando chmod.
man -f mkdir	Exibe uma pequena descrição do comando mkdir (*make directories* = cria diretórios).

```
[dieeego88@webminal.org ~]$man -f mkdir
mkdir (1)            - make directories
[dieeego88@webminal.org ~]$
```

Comando df

INCIDÊNCIA EM PROVA: BAIXA

Este comando permite **exibir informações sobre espaço livre e espaço ocupado** nas partições do sistema operacional, arquivos e diretórios, do sistema de arquivos como um todo (**df** = **d**isk **f**ree).

```
Sintaxe: df [opções]
```

PARÂMETROS COMUNS	DESCRIÇÃO
-k	Lista o tamanho dos blocos em kbytes.
-m	Lista o tamanho dos blocos em Mbytes.

EXEMPLOS	DESCRIÇÃO					
df	Filesystem	1K-blocks	Used	Available	Use%	Mounted on
	/dev/cciss/codop2	78361192	23185840	51130588	32%	/

Comando head

INCIDÊNCIA EM PROVA: BAIXA

Este comando permite apresentar as linhas iniciais de um arquivo de texto. Eventualmente, você possui um arquivo texto imenso e deseja visualizar apenas sua parte inicial (head = cabeça).

```
Sintaxe: head [opções]
```

PARÂMETROS COMUNS	DESCRIÇÃO
-n	Exibe o número de linhas do início de um arquivo (**-n** = **n**umber).

EXEMPLOS	DESCRIÇÃO
head teste.txt	Apresenta as linhas iniciais do arquivo teste.txt.
head -n 20 teste.txt	Apresenta as 20 primeiras linhas do início do arquivo teste.txt.

Comando tail

INCIDÊNCIA EM PROVA: BAIXA

Este comando permite apresentar as linhas finais de um arquivo de texto. Eventualmente, você possui um arquivo texto imenso e deseja visualizar apenas sua parte final (tail = calda).

```
Sintaxe: tail [opções]
```

PARÂMETROS COMUNS	DESCRIÇÃO
-n	Exibe o número de linhas do final de um arquivo (**-n** = **n**umber).

EXEMPLOS	DESCRIÇÃO
tail teste.txt	Apresenta as linhas finais do arquivo teste.txt.
tail -n 20 teste.txt	Apresenta as 20 últimas linhas do final do arquivo teste.txt.

Comando sort

INCIDÊNCIA EM PROVA: BAIXÍSSIMA

Este comando permite organizar as linhas de um arquivo texto ou da entrada padrão. A organização é feita por linhas e por meio de delimitadores (normalmente um espaço). É importante mencionar que esse comando trata o arquivo como um conjunto de caracteres onde a ordem crescente é: espaços, números, letras maiúsculas [A-Z] e letras minúsculas [a-z]. Primeiro as maiúsculas e depois as minúsculas (isso cai em prova!).

```
Sintaxe: sort [opções] [arquivo]
```

PARÂMETROS COMUNS	DESCRIÇÃO
sort -r	Inverte a ordem apresentada (-r = reverse).
sort -f	Ignora a diferença entre maiúsculas e minúsculas.

EXEMPLOS	DESCRIÇÃO
sort texto.txt	Organiza o conteúdo do arquivo texto.txt em ordem crescente.
sort texto.txt -r	Organiza o conteúdo do arquivo texto.txt em ordem decrescente.
sort -f texto.txt	Organiza o conteúdo em ordem crescent ignorando maiúsculas e minúsculas.

Comando more

INCIDÊNCIA EM PROVA: BAIXÍSSIMA

Este comando permite fazer a paginação de arquivos ou da entrada padrão. O comando more pode ser usado como comando para leitura de arquivos que ocupem mais de uma tela. Quando toda a tela é ocupada, o more efetua uma pausa e permite que você pressione ENTER ou ESPAÇO para continuar avançando no arquivo sendo visualizado. Para sair do more pressione q. Vejamos a sua sintaxe padrão:

```
Sintaxe: more [arquivo]
```

EXEMPLOS	DESCRIÇÃO
more /etc/passwd	Permite paginar o arquivo /etc/passwd.

Comando less

INCIDÊNCIA EM PROVA: BAIXÍSSIMA

Este comando permite fazer a paginação de arquivos ou da entrada padrão. O comando less pode ser usado como comando para leitura de arquivos que ocupem mais de uma tela. Quando toda a tela é ocupada, o less efetua uma pausa (semelhante ao more) e permite que você pressione Seta para Cima e Seta para Baixo ou PgUP/PgDown para fazer o rolamento da página. Para sair do less, pressione Q. A diferença para o more é que o less é mais rápido.

Sintaxe: less [arquivo]

EXEMPLOS	DESCRIÇÃO
less /etc/passwd	Permite paginar o arquivo /etc/passwd.

Comando rmdir

INCIDÊNCIA EM PROVA: BAIXÍSSIMA

Este comando é responsável por **remover diretórios vazios**, no entanto o diretório deve estar vazio antes de ser excluído (**rmdir** = **r**e**m**ove **dir**ectory).

Sintaxe: rmdir [caminho/diretório] [caminho1/diretório1]

PARÂMETROS COMUNS	DESCRIÇÃO
-p	Remove uma hierarquia de diretórios.
-v	Exibe informações de progresso do processamento (**-v** = **v**erbose).

Comando ln

INCIDÊNCIA EM PROVA: BAIXÍSSIMA

Este comando permite criar links para arquivos e diretórios no sistema. O link é um mecanismo que faz referência a outro arquivo ou diretório em outra localização. O link em sistemas GNU/Linux faz referência reais ao arquivo/diretório podendo ser feita cópia do link (será copiado o arquivo alvo), entrar no diretório (caso o link faça referência a um diretório), entre outros. No entanto, também é possível criar links simbólicos. *Professor, o que é um link simbólico?*

Em um link comum (também chamado hardlink), mesmo que o arquivo original seja deletado, o link continuará com o conteúdo da referência original, uma vez que ele funciona como uma cópia. **Já o link simbólico apenas armazena o endereço de referência do arquivo original.** Logo, caso o arquivo original seja deletado, o link simbólico perderá seu valor, uma vez que ele apontará para uma referência que já não existe mais.

```
Sintaxe: ln [opções] [origem] [link]
```

PARÂMETROS COMUNS	DESCRIÇÃO
-s	Cria um link simbólico ao invés de copiar (**-s** = **s**imbolic).

EXEMPLOS	DESCRIÇÃO
ln -s /dev/arq2 arq1	Cria um link simbólico com nome **arq1** do arquivo **/dev/arq2**.

Comando wc

INCIDÊNCIA EM PROVA: BAIXÍSSIMA

Este comando permite contar o número de palavras, bytes e linhas em um arquivo ou entrada padrão (**wc** = **w**ord **c**ount).

```
Sintaxe: wc [opções] [arquivo]
```

PARÂMETROS COMUNS	DESCRIÇÃO
wc -w	Mostra a quantidade de palavras (**-w** = **w**ords).
wc -l	Mostra a quantidade de linhas (**-l** = **l**ines).

EXEMPLOS	DESCRIÇÃO
wc /etc/passwd	Mostra a quantidade de linhas, palavras e letras (bytes) no arquivo /etc/ passwd.
wc -w /etc/passwd	Mostra a quantidade de palavras.
wc -l /etc/passwd	Mostra a quantidade de linhas.

Comando sudo

INCIDÊNCIA EM PROVA: BAIXÍSSIMA

Esse comando é responsável por permitir que usuários comuns obtenham privilégios de outro usuário, em geral o superusuário (*root*), para executar tarefas específicas.

```
Sintaxe: sudo [opções] comando
```

PARÂMETROS COMUNS	DESCRIÇÃO
-u	O sudo executa o comando com os privilégios do usuário especificado.

Comando apt-get

INCIDÊNCIA EM PROVA: BAIXÍSSIMA

Este comando é utilizado para permitir a instalação, reinstalação, atualização e remoção de pacotes do sistema. Há sintaxes diferentes:

```
Sintaxe 1: apt-get [options] command
Sintaxe 2: apt-get [options] install | remove pkg1 [pkg2 ...]
Sintaxe 3: apt-get [options] source pkg1 [pkg2 ...]
```

PARÂMETROS COMUNS	DESCRIÇÃO
-h	Fornece as opções do utilitário.
--reinstall	Permite reinstalar um determinado pacote.

Comando logout

INCIDÊNCIA EM PROVA: BAIXÍSSIMA

Este comando é utilizado para sair de uma sessão do terminal. É bastante similar ao comando exit e pode ser executado também por meio do comando CTRL+D.

```
Sintaxe: logout
```

Comando shutdown

INCIDÊNCIA EM PROVA: BAIXÍSSIMA

Este comando permite **desligar ou reiniciar o computador** imediatamente ou após determinado tempo (programável) de forma segura – usuários são avisados que o computador será desligado.

```
Sintaxe: shutdown [opções] [hora] [mensagem]
```

PARÂMETROS COMUNS	DESCRIÇÃO
shutdown -r	Reinicia o sistema operacional imediatamente ou após um período em minutos.
shutdown -h	Desliga o sistema operacional imediatamente ou após um período em minutos.
shutdown -k	Simula o desligamento/reinicio do sistema, enviando mensagem aos usuários.
shutdown -c	Cancela a execução do shutdown.

EXEMPLOS	DESCRIÇÃO
shutdown -h now	Desligar o computador imediatamente.
shutdown -r now	Reinicia o computador imediatamente.
shutdown -r 20	Faz o sistema ser reiniciado após 20 minutos.

Comando touch

INCIDÊNCIA EM PROVA: BAIXÍSSIMA

Este comando é utilizado para mudar a data e hora que um arquivo foi criado. Caso ele seja usado com arquivos que não existam, por padrão, ele criará estes arquivos.

```
Sintaxe: touch [opções] [arquivos]
```

PARÂMETROS COMUNS	DESCRIÇÃO
touch -t	Usa Mês (MM), Dias (DD), Horas (hh), minutos (mm) e opcionalmente o ANO e segundos para modificação do(s) arquivos ao invés da data e hora atual.

EXEMPLOS	DESCRIÇÃO
touch teste	Cria o arquivo teste caso ele não existir.
touch -t 10011230 arq1	Altera da data e hora do arquivo para 01/10 e 12:30.

Comando date

INCIDÊNCIA EM PROVA: BAIXÍSSIMA

Este comando permite visualizar/modificar a Data e Hora do sistema. Você precisa estar como usuário root para modificar a data e hora. Muitos programas do sistema, arquivos de registro (log) e tarefas agendadas funcionam com base na data e hora fornecidas pelo sistema, assim esteja consciente das modificações que a data/hora pode trazer a estes programas (principalmente em se tratando de uma rede com muitos usuários).

```
Sintaxe: date MesDiaHoraMinuto[AnoSegundos]
```

PARÂMETROS COMUNS	DESCRIÇÃO
%d	Dia do mês (1-31)
%m	Mês do Ano (1-12)
%y	Ano (dois dígitos)

EXEMPLOS	DESCRIÇÃO
date	Exibe a data atual: 16/05/2020
Data 12250815	Muda a data para 25/12 e a hora para 08:15

Comando diff

INCIDÊNCIA EM PROVA: BAIXÍSSIMA

Este comando permite comparar dois arquivos e mostrar as diferenças entre eles. Esse comando é utilizado somente para a comparação de arquivos em formato texto.

```
Sintaxe: diff [diretório1/arquivo1] [diretório2/arquivo2] [opções]
```

PARÂMETROS COMUNS	DESCRIÇÃO
-b	Ignora espaços em branco como diferenças
-B	Ignora linhas em branco inseridas ou apagadas nos arquivos.

EXEMPLOS	DESCRIÇÃO
diff arq1.txt arq2.txt	Compara o arquivo arq1.txt com arq2.txt e exibe suas diferenças na tela.

Comando gzip

INCIDÊNCIA EM PROVA: BAIXÍSSIMA

Este comando permite compactar (diminuir o tamanho) um arquivo com ótima taxa de compactação e velocidade. A extensão dos arquivos compactados pelo gzip é a .gz.

```
Sintaxe:
```

PARÂMETROS COMUNS	DESCRIÇÃO
-d	Descompacta uma arquivo.
-r	Compacta diretórios e sub-diretórios.

EXEMPLOS	DESCRIÇÃO
gzip texto.txt	Compacta o arquivo texto.txt para texto.txt.gz.
gzip -d texto.txt.gz	Descompacta o arquivo texto.txt.gz para texto.txt.

Comando du

INCIDÊNCIA EM PROVA: BAIXÍSSIMA

Este comando é utilizado para **exibir o espaço ocupado** por arquivos e sub-diretórios do diretório atual do sistema operacional (df exibe o espaço como um todo, du exibe o espaço de um diretório).

```
Sintaxe: du [opções]
```

PARÂMETROS COMUNS	DESCRIÇÃO
-h	Permite visualizar o tamanho de forma legível por humanos (Kb, Mb) em vez de blocos.

EXEMPLOS	DESCRIÇÃO
du /home	Permite visualizar o tamanho dos arquivos e subdiretórios do diretório /home em blocos.

du -h /home	Permite visualizar em kilobyte e megabyte.

Comando chown

INCIDÊNCIA EM PROVA: BAIXÍSSIMA

Este comando muda dono de um arquivo/diretório – opcionalmente pode também ser utilizdo para mudar o grupo. O comando correspondente no MS-DOS é chamado CALCS.

```
Sintaxe: chown [opções] [dono.grupo] [diretório/arquivo]
```

EXEMPLOS	DESCRIÇÃO
chown diego teste.txt	Permite muda o dono do arquivo teste.txt para diego.

QUESTÕES COMENTADAS – CESGRANRIO

1. **(CESGRANRIO / BB – 2014)** O sistema operacional cujas características são utilizar código aberto e interface por linha de comando é o:

a) Mac OS
b) iOS
c) Linux
d) Windows
e) Android

Comentários:

Trata-se do Linux, uma vez que todo esses são de proprietários – exceto o Android. No entanto, sua interface por linha de comando está restrita aos serviços de manutenção e restauração do equipamento, configurada de fábrica em menus prontos. Ele possui ambientes gráficos também!

Gabarito: Letra C

---

2. **(CESGRANRIO / PETROBRÁS – 2011)** Nos sistemas Linux, um processo pode ser interrompido por meio do comando kill. Para isso, é necessário saber o pid desse processo. Qual comando deve ser utilizado para obter esse pid?

a) ps
b) ls
c) du
d) set
e) pwd

Comentários:

O comando que permite listar processos em execução, inclusive do seu número de identificação (Process Identificar – PID) é o ps. A partir dele, um processo pode ser interrompido por meio do kill.

Gabarito: Letra A

LISTA DE QUESTÕES – CEGRANRIO

1. **(CESGRANRIO / BB – 2014)** O sistema operacional cujas características são utilizar código aberto e interface por linha de comando é o:

   a) Mac OS
   b) iOS
   c) Linux
   d) Windows
   e) Android

2. **(CESGRANRIO / PETROBRÁS – 2011)** Nos sistemas Linux, um processo pode ser interrompido por meio do comando kill. Para isso, é necessário saber o pid desse processo. Qual comando deve ser utilizado para obter esse pid?

   a) ps
   b) ls
   c) du
   d) set
   e) pwd

GABARITO – CESGRANRIO

1. LETRA C
2. LETRA A

ESSA LEI TODO MUNDO CONHECE:
PIRATARIA É CRIME.

Mas é sempre bom revisar o porquê e como você pode ser prejudicado com essa prática.

1. 

Professor investe seu tempo para elaborar os cursos e o site os coloca à venda.

2. 

Pirata divulga ilicitamente (grupos de rateio), utilizando-se do anonimato, nomes falsos ou laranjas (geralmente o pirata se anuncia como formador de "grupos solidários" de rateio que não visam lucro).

3. 

Pirata cria alunos fake praticando falsidade ideológica, comprando cursos do site em nome de pessoas aleatórias (usando nome, CPF, endereço e telefone de terceiros sem autorização).

4. 

Pirata compra, muitas vezes, clonando cartões de crédito (por vezes o sistema anti-fraude não consegue identificar o golpe a tempo).

5. 

Pirata fere os Termos de Uso, adultera as aulas e retira a identificação dos arquivos PDF (justamente porque a atividade é ilegal e ele não quer que seus fakes sejam identificados).

6. 

Pirata revende as aulas protegidas por direitos autorais, praticando concorrência desleal e em flagrante desrespeito à Lei de Direitos Autorais (Lei 9.610/98).

7. 

Concurseiro(a) desinformado participa de rateio, achando que nada disso está acontecendo e esperando se tornar servidor público para exigir o cumprimento das leis.

8. 

O professor que elaborou o curso não ganha nada, o site não recebe nada, e a pessoa que praticou todos os ilícitos anteriores (pirata) fica com o lucro.

Deixando de lado esse mar de sujeira, aproveitamos para agradecer a todos que adquirem os cursos honestamente e permitem que o site continue existindo.