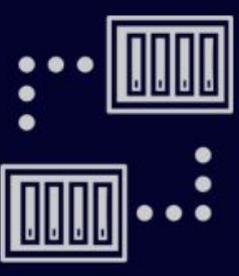


Linux is not Unix

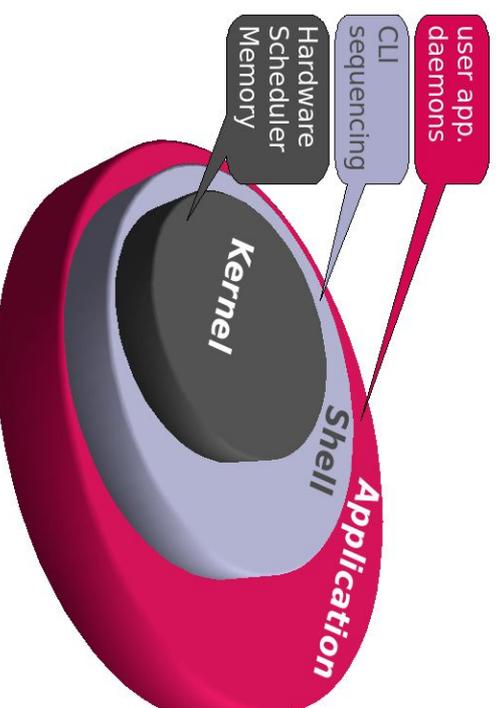


O que é Linux?



O que é Linux?

- Existem duas respostas para esta pergunta:
- **A resposta formal:**
 - Linux é o núcleo de um sistema operacional;
 - É o conjunto de componentes essenciais que coordena hardware, software e interações com o usuário;
 - Kernel.



O que é Linux?

- Existem duas respostas para esta pergunta:
- **A segunda resposta envolve o senso comum...**
 - Quando as pessoas falam “Linux”, elas estão tratando de sistemas operacionais baseados no Linux;
 - Em outras palavras, sistemas operacionais desenvolvidos utilizando o kernel do Linux;
 - Nós normalmente chamamos esses sistemas operacionais de **distribuições Linux**, ou **distros Linux**.

O que é Linux?

- Distribuições Linux
 - Uma distribuição Linux típica consiste em:
 - Kernel do Linux
 - Inclui a maior parte dos drivers.
 - Ferramentas e bibliotecas do GNU
 - Utilitários de linha de comando para facilitar o uso e gerenciamento do sistema.
 - Gerenciador de pacotes
 - O modo pelo qual o seu sistema gerencia a instalação, atualização e remoção de software.

O que é Linux?

- Distribuições Linux
 - Uma distribuição Linux típica consiste em:
 - Documentação
 - Documentos em texto que explicam como o software funciona e seus mais variados recursos.
 - Ambiente de trabalho
 - É um software que permite a você utilizar um mouse e gera uma interface gráfica de usuário.
 - Aplicativos de usuário
 - Software como editores de texto, navegadores, reprodutores de música e etc.

O que é Linux?

- Distribuições Linux
 - Existem várias distribuições Linux diferentes
 - Ubuntu
 - Fedora
 - Arch Linux
 - OpenSUSE
 - elementary OS
 - Linux Mint
 - etc...



O que é Linux?

- Distribuições Linux
 - Por que existem tantas distribuições Linux?
 - Porque o kernel do Linux e maioria do software para Linux é livre.
 - Qualquer um pode desenvolver uma distribuição Linux para suprir seus gostos e necessidades.
 - Todo o ecossistema do Linux está sob sua escolha.
 - Você pode escolher como você quer que seu computador se comporte pois você pode instalar o software que quiser.
 - Você pode personalizar seu visual e muito mais.

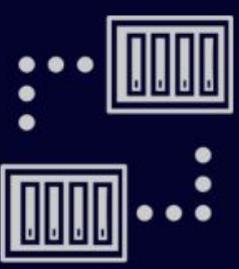
O que é Linux?

- Distribuições Linux
 - Com uma grande escolha vem uma grande responsabilidade... Mas só se você quiser!
 - Existe uma tonelada de opções para quem é iniciante.
 - Ubuntu, Linux Mint e elementary OS são alguns exemplos.
 - Permitem fácil instalação e uma fácil experiência de uso.

O que é Linux?

- **Distribuições Linux**
 - **As vantagens do Linux são enormes.**
 - **Linux é livre: Ambos em termos de custo e de liberdade.**
 - **Você pode fazer o que quiser sem pedir permissão.**
 - **Linux é estável**
 - **No Linux não há a “Tela azul da morte” :)**
 - **Não há erros de registro**
 - **Linux é seguro**
 - **É muito difícil ser atingido por vírus ou malware.**
 - **Linux é versátil**
 - **Está presente em TVs, aparelhos de TV a cabo, roteadores, carros, geladeiras e telefones.**

Modo texto x Modo gráfico



Modo texto x Modo gráfico

- **Interface:**
 - Método de interação com o usuário.
 - No mundo dos sistemas operacionais existem dois tipos de interface:
 - Interface gráfica;
 - Interface de texto (ou linha de comando).

Modo texto x Modo gráfico

- Interface gráfica:
 - Oferece uma experiência mais intuitiva e amigável.
 - Utiliza elementos gráficos como janelas, botões, caixas de texto, ícones.
 - Podem não oferecer uma experiência simples.
 - “Simplicidade é qualquer coisa baseada no minimalismo”.

Modo texto x Modo gráfico

- Interface de texto:
 - Experiência menos amigável e intuitiva;
 - Permite uma interação mais simples e direta;
 - Baseia-se em comandos de texto que realizam tarefas;
 - Economiza tempo e reduz a complexidade.

Modo texto x Modo gráfico

- Interface de texto:

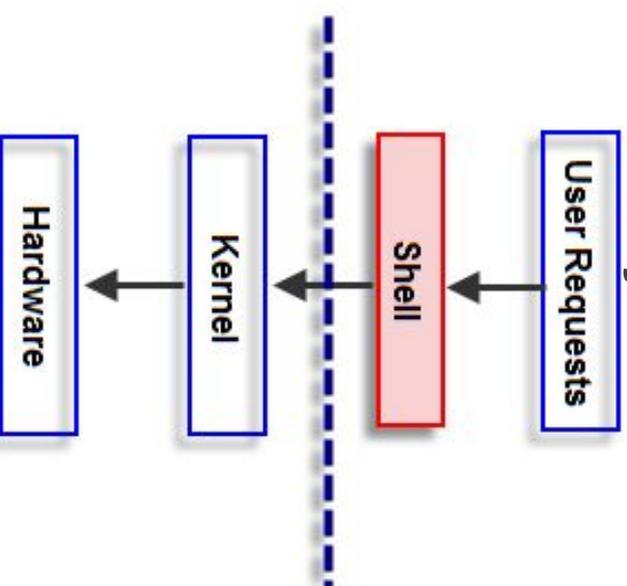
```
Debian GNU/Linux 6.0 Shothoth tty2
Shothoth login: lovecraft
Password:
Last login: Wed Aug 31 23:38:26 EST 2011 on tty2
Linux Shothoth 2.6.32-5-amd64 #1 SMP Wed Jan 12 03:40:32 UTC 2011 x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
No mail.
lovecraft@Shothoth ~ $ fbgrab console.png
```

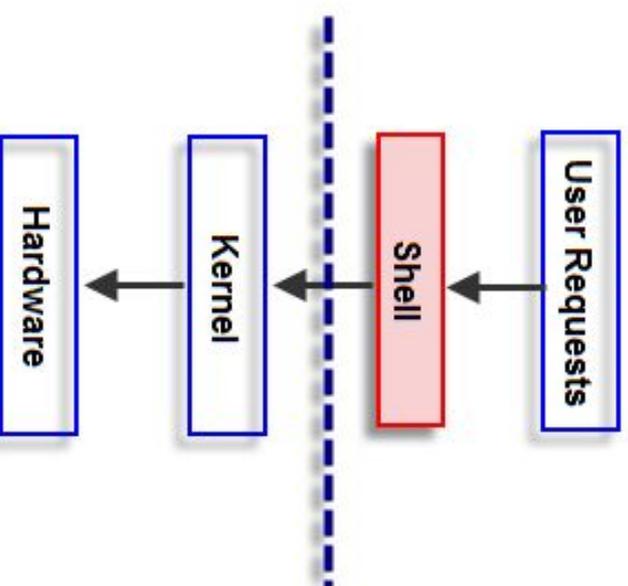
Modo texto x Modo gráfico

- Interface de texto:
 - Utiliza uma Shell:
 - É a ligação entre o usuário e o sistema em uma interface de texto;
 - Interpreta os comandos introduzidos pelo usuário para outros aplicativos ou chamadas do sistema;
 - Permite a automatização de tarefas.



Modo texto x Modo gráfico

- Interface de texto:
 - BASH (**B**ourne **A**gain **S**hell):
 - Shell mais utilizada entre as distros Linux;
 - Fácil utilização e interpretação.

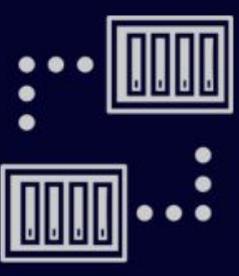


Modo texto x Modo gráfico

- BASH:
 - Permite o uso de atalhos do teclado para realização de algumas funções.

CTRL + A	Move o cursor para o início da linha de comando
CTRL + C	Interrompe a execução de um programa e retorna a linha de comando
CTRL + D	Faz logout da sessão, equivale aos comandos exit ou logout
CTRL + E	Move o cursor para o fim da linha de comando
CTRL + H	Gera o caractere backspace
CTRL + L	Limpa o terminal
CTRL + R	Faz uma busca no histórico de comandos
CTRL + Z	Suspende um programa
Seta direita ou esquerda	Move o cursor para direita ou esquerda
Seta para baixo ou para cima	Navega pelo histórico de comandos
Shift + PageUp ou Shift + PageDown	Navega pelo buffer do terminal
Tab	Completa um comando ou nome de arquivo
Tab Tab	Mostra as possibilidades de completar um comando ou arquivo

Pedindo ajuda



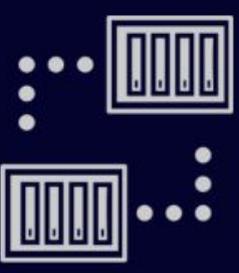
Pedindo ajuda

- A distribuições Linux contam com várias ferramentas de ajuda ao usuários.
- São manuais, descrições de programas e etc.
- Ferramentas mais utilizadas:
 - man
 - Leitor de manuais.
 - Acessa os diversos manuais do sistema.
 - Bibliotecas, jogos, programas e utilitários de sistema são exemplos de programas e aplicações que possuem manuais.

Pedindo ajuda

- Ferramentas mais utilizadas:
 - info
 - Leitor de páginas de informações.
 - Informações são como manuais resumidos.
 - apropos
 - Procura e exibe a descrição de arquivos e diretórios do sistema, se existirem.
 - --help
 - Acessar os arquivos de ajuda ou principais utilizações diretamente no comando.
 - ls --help

Arquivos e diretórios



Arquivos e diretórios

- Arquivos
 - Em sistemas Linux tudo é arquivo, se não for um arquivo, é um processo.
 - Existem arquivos comuns aos quais denominamos arquivos **regulares** (prefixo **-**).
 - Outros tipos de arquivos:
 - **Diretórios** (prefixo **d**): arquivos que são listas de outros arquivos.
 - **Arquivos especiais** (prefixo **c**): são mecanismos de entrada e saída. A maior parte destes arquivos está localizado em */dev*.

Arquivos e diretórios

- Arquivos
 - Outros tipos de arquivos:
 - **Links** (prefixo **l**): um sistema para disponibilizar um arquivo ou diretório em várias partes do sistema.
 - **Sockets** (prefixo **s**): um tipo de arquivo especial que provém operações de rede entre processos de forma protegida.
 - **Named pipes** (prefixo **p**): agem como os sockets, porém não provém operações de rede.

Arquivos e diretórios

- Para a criação de um arquivo existem algumas formas simples:
 - Utilizando o comando **touch** para criar um arquivo vazio:
 - `touch arquivo_novo.txt`
 - Utilizando o comando **cat**:
 - `cat > arquivo_novo.txt`
 - Para finalizar, aperte Ctrl+D.
 - Ou simplesmente escrever um nome depois de um editor de texto:
 - `vim arquivo_novo.txt`

Arquivos e diretórios

- Como verificar o tipo de arquivo?
 - Comando `ls -l`.
 - Mostra uma lista longa dos arquivos presentes no diretório atual.

```
llsilva@LAR-03:/$ ls -l
```

```
total 128
```

```
drwxr-xr-x 162 root root 12288 Abr  7 18:36 etc
```

```
drwxr-xr-x  4 root root    0 Mai  2 14:20 home
```

```
lrwxrwxrwx  1 root root   33 Ago  6 2015 initrd.img -> boot/initrd.img-
```

```
3.19.0-25-generic
```

```
-rw-r--r--  1 llsilva Grupos 8159857 Mar 17 21:02 wordpress-4.4.2-pt_BR.zip
```

Arquivos e diretórios

- Localizando-se na árvore de diretórios
 - Ao iniciar o terminal (*bash*), o usuário geralmente se encontra dentro de seu diretório *home*.
 - O diretório */home* contém todos os diretórios pessoais de usuários do sistema.
 - Pasta saber qual em qual diretório você se encontra, utilize o comando *pwd*.

```
llsilva@LAR-03:~$ pwd
```

```
/home/llsilva
```

Arquivos e diretórios

- Criando diretórios
 - Para criar um diretório utilizamos o comando *mkdir*.
 - Exemplo: **mkdir pasta1**
- Acessando diretórios
 - Acessaremos diretórios utilizando o comando *cd*.
 - Exemplo: **cd pasta1**

Arquivos e diretórios

- Listando o conteúdo de diretórios
 - Para listar os conteúdos de diretórios, utilize o comando `/s`.
 - Exemplo: **ls** para listar os arquivos do diretório onde você está; **ls nome-do-diretorio** para listar os arquivos de um outro diretório.
 - O comando `/s` possui um grande número de opções.

Arquivos e diretórios

- Algumas opções interessantes do **ls** são:
 - **ls -l** : lista arquivos um por linha, incluindo suas propriedades.
 - **ls -a** : lista todos os arquivos, inclusive os ocultos.
 - **ls -h** : mostra o tamanho dos arquivos para fácil leitura. (Ex.: 4Kb, 6Mb...). *Não faz sentido se usado sozinho.*
 - **ls -R** : mostra os arquivos dentro dos diretórios.
 - Essas opções podem ser combinadas.

Arquivos e diretórios

- Para mover arquivos e diretórios existe o comando **mv**:
 - `mv aula_1.txt Documentos/`
 - `mv pasta_1/ Documentos/pasta_2/`
- O comando **mv** também é utilizado para renomear arquivos e pastas no linux.
 - `mv nome_1.txt nome_2.txt`
- O comando **mv** pode gerar cópias de arquivos movidos antes de mover.
 - `mv --backup=simple arquivo1.txt outraPasta/`
 - Ele vai gerar um arquivo oculto com o conteúdo do arquivo original da *outraPasta*.

Arquivos e diretórios

- Para cópia de documentos e pastas é utilizado o comando **cp**.
 - Para cópia de arquivos:
 - `cp aula_1.txt aula_1_renomeado.txt`
 - Para cópia de diretórios:
 - `cp -r pasta_1/ Documentos/outra_pasta/`

Arquivos e diretórios

- Para visualizar o conteúdo de arquivos existem alguns comandos como:
 - **cat** - exibe no terminal o conteúdo inteiro de um arquivo.
 - **head** - exibe as primeiras linhas de um arquivo.
 - O parâmetro **-n** permite escolher a quantidade de linhas a serem exibidas.
 - **tail** - exibe as últimas linhas de um arquivo.
 - O parâmetro **-n** permite escolher a quantidade de linhas a serem exibidas.
 - **less** - Exibe o conteúdo de uma maneira fácil e por blocos.

Arquivos e diretórios

- Um arquivo precisa de premissões para ser acessado. No linux essas permissões podem ser divididas para três entidades diferentes:
 - Dono (u - user): quem criou o arquivo.
 - Grupo (g - group): a todos os usuários de um grupo.
 - Outros (o - others): todos os outros usuários que não estão relacionados acima.

Arquivos e diretórios

- Existem 3 permissões possíveis:
 - Leitura (r - read): permite a leitura dos arquivos;
 - Escrita (w - write): permite a edição dos arquivos ou criação de arquivos em um diretório;
 - Execução (x - execute): permite a execução de arquivos ou a listagem de diretórios;
- As permissões podem ser alteradas de várias formas. Por exemplo: Octal.

Arquivos e diretórios

- 0 : --- (nenhuma permissão)
- 1 : --X (somente execução)
- 2 : -w- (somente escrita)
- 3 : -wx (escrita e execução)
- 4 : r-- (somente leitura)
- 5 : r-x (leitura e execução)
- 6 : rw- (leitura e escrita)
- 7 : rwx (leitura, escrita e execução)

Arquivos e diretórios

- **Exemplos:**

```
chmod 755 arquivo1.txt
```

```
chmod 644 arquivo.txt
```

```
chmod 703 diretorio -R
```

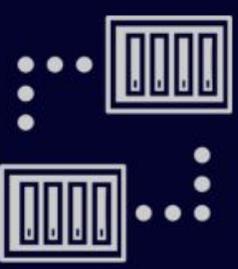
Arquivos e diretórios

- Atribuindo permissões individuais:
 - Acrescentar a permissão de escrita ao grupo:
 - `chmod g+w arquivo.txt`
 - Acrescentar a permissão de execução a outros:
 - `chmod o+x arquivo.txt`
 - Remover permissão de execução a outros:
 - `chmod o-x arquivo.txt`
- Atribuindo múltiplas permissões:
 - Acrescentar leitura e escrita ao dono:
 - `chmod u+rw arquivo.txt`
 - Acrescentando todas as permissões ao grupo:
 - `chmod g=rwx arquivo.txt`

Arquivos e diretórios

- Para trocar o dono ou o grupo que o arquivo pertence pode-se utilizar o comando **chown**. Esse comando só pode ser utilizado com usuários administradores do sistema.
 - `chown novo_usuario:novo_grupo arquivo.txt`
- Para trocar apenas o usuário:
 - `chown novo_usuario arquivo.txt`
- Para trocar apenas o grupo:
 - `chown :novo_grupo arquivo.txt`

Redireccionamiento



Redirecionamento

- O que é entrada padrão e saída padrão?
 - A maior parte dos comandos do Linux leem uma entrada, esta pode ser um arquivo ou um atributo para o comando, e escrevem uma saída.
 - Por padrão, a entrada é o teclado (*stdin*) e saída é a sua tela (*stdout*).

Redirecionamento

- Operadores de redirecionamento
 - Redirecionamento de saída com `>` e `|`
 - Algumas vezes você vai querer redirecionar a saída de um comando para um arquivo.
 - Ou você vai querer redirecionar a saída de um comando para outro comando.
 - Isso é conhecido como redirecionamento de saída.
 - O redirecionamento de saída pode ser realizado com o operador `>` (maior que) ou `|` (pipe), que envia a saída de um comando para outro comando.

Redirecionamento

llsilva@LAR-03:~\$ cat arquivo1

Eu amo

llsilva@LAR-03:~\$ cat arquivo2

chocolate

llsilva@LAR-03:~\$ cat arquivo1 arquivo2 > arquivo3

llsilva@LAR-03:~\$ cat arquivo3

Eu amo

chocolate

Redirecionamento

- Operadores de redirecionamento
 - Redirecionar nada para um arquivo é a mesma coisa de apagar seu conteúdo.
 - Esse processo é chamado de *truncamento*.

```
llsilva@LAR-03:~$ cat arquivo3
```

Eu amo

chocolate

```
llsilva@LAR-03:~$ > arquivo3
```

```
llsilva@LAR-03:~$ cat arquivo3
```

```
llsilva@LAR-03:~$
```

Redirecionamento

- Operadores de redirecionamento
 - O mesmo processo para um arquivo não existente criará o arquivo.

```
llsilva@LAR-03:~$ ls -l arquivo
```

ls: não é possível acessar arquivo: Arquivo ou diretório não encontrado

```
llsilva@LAR-03:~$ > arquivo
```

```
llsilva@LAR-03:~$ ls -l arquivo
```

```
-rw-r--r-- 1 llsilva Grupos 0 Mai 16 12:58 arquivo
```

Redirecionamento

- Operadores de redirecionamento
 - Exemplo de uso de pipes:
 - Busca de um arquivo ou diretório em particular em uma lista de arquivos e diretórios:

```
lilsilva@LAR-03:~$ ls -l | grep "arquivo"
```

```
-rw-r--r-- 1 lilsilva Grupos 0 Mai 16 12:58 arquivo
```

```
-rw-r--r-- 1 lilsilva Grupos 7 Mai 16 12:51 arquivo1
```

```
-rw-r--r-- 1 lilsilva Grupos 11 Mai 16 12:51 arquivo2
```

```
-rw-r--r-- 1 lilsilva Grupos 0 Mai 16 12:56 arquivo3
```

Redirecionamento

- Operadores de redirecionamento
 - Exemplo de uso de pipes:
 - Mostrar a saída de uma listagem de diretórios em páginas:
 - `ls -la | less`

Redirecionamento

- Redirecionamento de entrada:
 - Operador < (menor que).

```
llsilva@LAR-03:~$ cat arquivo_desordenado.txt
```

```
Eu
```

```
amo
```

```
chocolate
```

```
llsilva@LAR-03:~$ sort < arquivo_desordenado.txt
```

```
amo
```

```
chocolate
```

```
Eu
```

Redirecionamento

- Combinação de operadores de redirecionamento
 - É possível utilizar mais de um operador de redirecionamento para obter saídas melhor filtradas:

```
llsilva@LAR-03:~$ less --help | grep -i "examine" > examine.txt
```

```
llsilva@LAR-03:~$ cat examine.txt
```

```
:e [file]    Examine a new file.
```

```
:n          * Examine the (N-th) next file from the command line.
```

```
:p          * Examine the (N-th) previous file from the command line.
```

```
:x          * Examine the first (or N-th) file from the command line.
```

```
+cmd       Execute the less cmd each time a new file is examined.
```

```
llsilva@LAR-03:~$ sort < arquivo_desordenado.txt > arquivo_ordenado.txt
```

```
llsilva@LAR-03:~$ cat arquivo_ordenado.txt
```

amo

chocolate

Eu

Redirecionamento

- O operador >>
 - Ao invés sobrescrever os dados de um arquivo de texto, ele adiciona novos dados no final do arquivo.

Redirecionamento

```
llsilva@LAR-03:~$ cat arquivo_desordenado.txt
```

```
Eu
```

```
amo
```

```
chocolate
```

```
llsilva@LAR-03:~$ date >> arquivo_desordenado.txt
```

```
llsilva@LAR-03:~$ cat arquivo_desordenado.txt
```

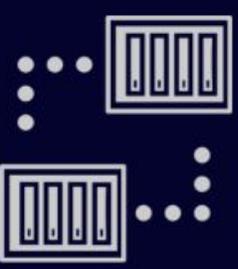
```
Eu
```

```
amo
```

```
chocolate
```

```
Seg Mai 16 13:15:25 BRT 2016
```

Exercícios



Exercícios

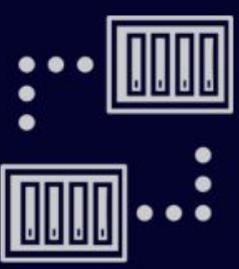
Abra uma janela de Terminal, para abrir uma *shell* do Linux, e faça o que se pede:

1. Feche a sessão atual da *shell* e inicie uma nova sessão.
2. Abra o manual (`man`) do comando `top`. Qual é a função do comando `top`?
3. No seu diretório *home*, ou pasta pessoal, escreva os comandos para criar os diretórios `pastaa1` e `pastaa2`.
4. No seu diretório *home*, liste o conteúdo do diretório.
5. Entre no diretório `pastaa1` e mostre o caminho do diretório corrente.
6. Crie o arquivo `texto1.txt` dentro do diretório `pastaa1`, e o arquivo `texto2.txt` dentro do diretório `pastaa2`.
7. Preencha os arquivos `texto1.txt` e `texto2.txt` com conteúdo diferente.
8. Copie o arquivo `texto1.txt` para `pastaa2`.
9. Renomeie o arquivo `texto1.txt` para `texto2.txt`. O que aconteceu com os arquivos?
10. Remova os diretórios `pastaa1` e `pastaa2`.

Exercícios

11. Limpe o terminal.
12. Mostre a data atual do sistema.
13. Navegue para a pasta “Minicurso” presente em sua pasta pessoal. Liste seu conteúdo.
14. Veja o arquivo `nomes.txt`. Ele possui diversos nomes, cada um por linha. Utilize comandos para criar um outro arquivo `nomes_ordenados.txt` com os nomes em ordem alfabética.
15. Una os arquivos `part1.txt`, `part2.txt` e `part3.txt`, em outro arquivo chamado `lero.txt`. Use os redirecionadores para realizar esta tarefa.
16. Verifique se o arquivo `nomes.txt` ou o `nomes_ordenados.txt` possui o nome “Lemão” utilizando o comando `grep`. Teste com outros nomes. E teste também com pedaços de nomes, ou até letras.
17. Liste o conteúdo do diretório “Minicurso” novamente, porém com as informações de permissões e usuários e filtre a saída para capturar apenas arquivos `.txt`.

Recapitulando...



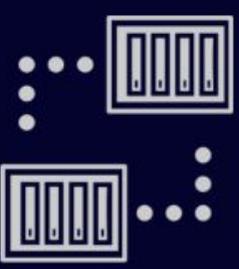
Recapitulando

- O que é o Linux?
 - Kernel
 - Distribuições Linux
 - Características
- Modo texto x Modo gráfico
 - Interface
 - Interface gráfica
 - Interface de texto
 - Shell
 - Bash

Recapitulando

- Arquivos e diretórios
 - No Linux, tudo é arquivo. Arquivos podem ser especiais, possuindo um identificador.
 - Exemplo: diretório (**d**)
- Comandos
 - man, info, apropos, --help
 - touch, ls, pwd, mkdir, cd, mv, cp, rm
 - cat, head, tail, less
 - chmod, chown
 - grep, sort, date
- Redirecionadores
 - >, <, |, >>

Sistema de archivos



Sistema de arquivos

- **Particionamento**
 - Por que particionar?
 - Um dos objetivos de se particionar o disco rígido é dividir os dados em várias partes;
 - Quando um desastre acontece, apenas uma parte dos dados são afetados;

Sistema de arquivos

- **Particionamento**
 - Existem dois tipos principais de partições no Linux:
 - *Partição de dados*: contém dados normais do sistema Linux, incluindo a partição root que contém dados de execução do sistema;
 - *Partição de swap*: é a expansão da memória física do computador; memória extra no disco rígido.
 - A maior parte dos sistemas contém uma partição root, representada pelo caractere /
 - Uma partição root padrão contém arquivos de configuração do sistema, os programas mais básicos, bibliotecas do sistema, espaço temporário e o diretório *home* do usuário administrador do sistema.

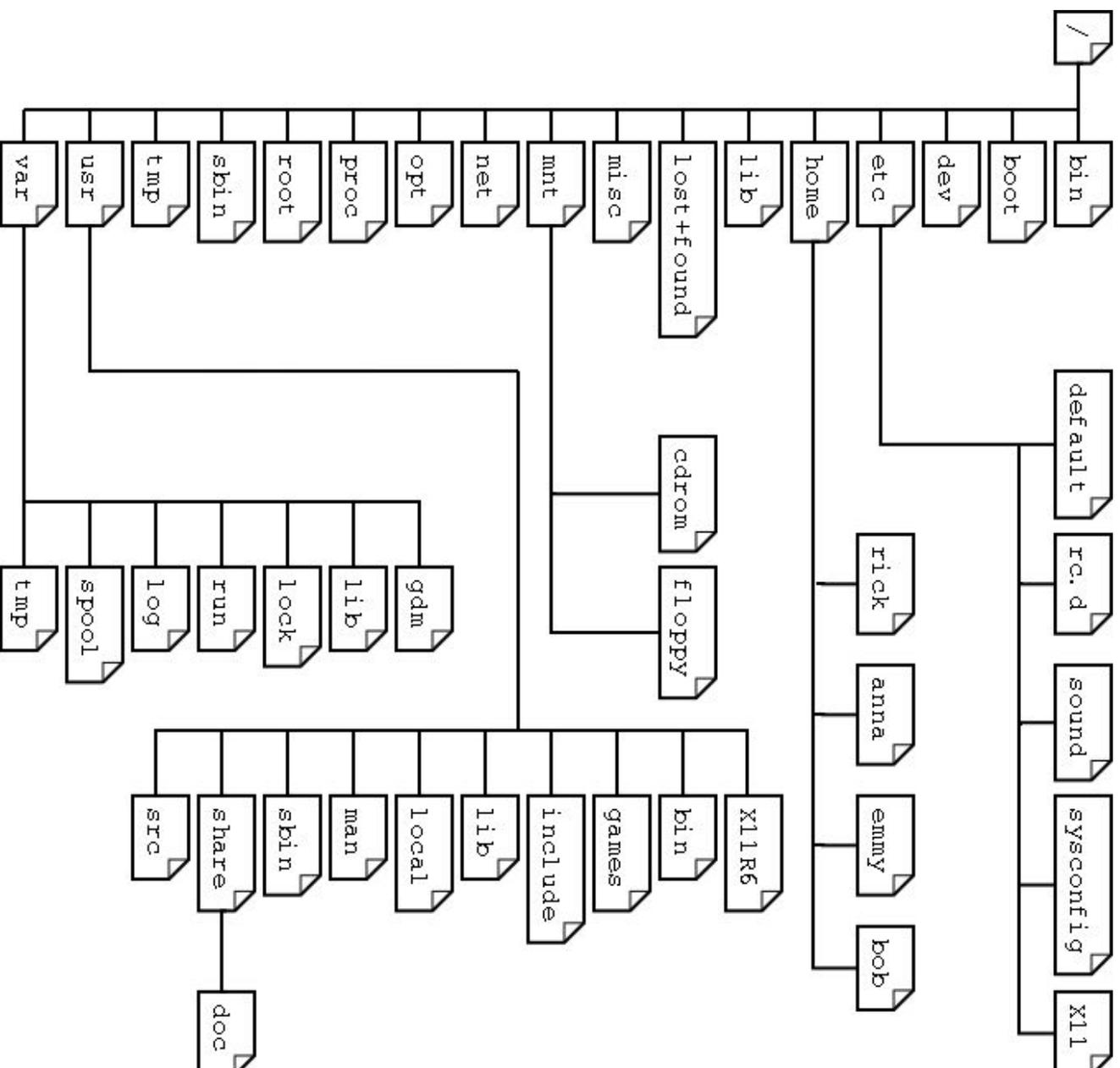
Sistema de arquivos

- **Particionamento**
 - Dentro da partição root podemos encontrar outras partições comuns aos sistemas Linux, tais como:
 - Uma partição para programas de usuários (*/usr*);
 - Uma partição que contém arquivos pessoais dos usuários (*/home*);
 - Uma partição que armazena arquivos temporários como filas de impressão ou log do sistema (*/var*);
 - Uma partição para software extra de terceiros (*/opt*).

Sistema de arquivos

- Organização do sistema de arquivos
 - Existem vários outros diretórios e partições dentro da partição root;
 - A estrutura da partição root pode depender dos desenvolvedores da distribuição Linux e do propósito para o qual foi desenvolvida a distro.
 - Vejamos uma representação gráfica destes diretórios na distribuição Red Hat:

Sistema de arquivos



Sistema de arquivos

- Organização do sistema de arquivos
 - Você pode verificar quais diretórios estão contidos na partição root de seu sistema listando seus arquivos;

```
lilsilva@LAR-04:~$ cd /
```

```
lilsilva@LAR-04:/$ ls
```

```
adminlar bin boot cdrom dev etc home initrd.img  lar lib lib64 lost+found  
media mnt opt proc root run sbin snap srv  sys tmp usr var vmlinuz
```

Sistema de arquivos

- Subdiretórios do diretório /

Diretório	Conteúdo
/bin	Programas comuns, compartilhados pelo sistema e por seus usuários
/boot	Arquivos de inicialização do Linux, como vmlinuz e grub.
/dev	Contém referências à todos os dispositivos, que são arquivos com propriedades especiais.
/etc	Os arquivos de configuração do sistema mais importantes estão no /etc.
/home	Os diretórios home dos usuários do sistema.
/initrd	Contém informações sobre o boot do sistema.
/lib	Contém bibliotecas utilizadas pelo sistema e pelos usuários.
/lost+found	Toda partição contém um diretórios lost+found. Arquivos que são salvos durante falhas se encontram aqui.
/mnt	Diretório padrão para montagem de dispositivos externos. Por exemplo: CD-ROM, Câmera digital.
/opt	Tipicamente contém software de terceiros

Sistema de arquivos

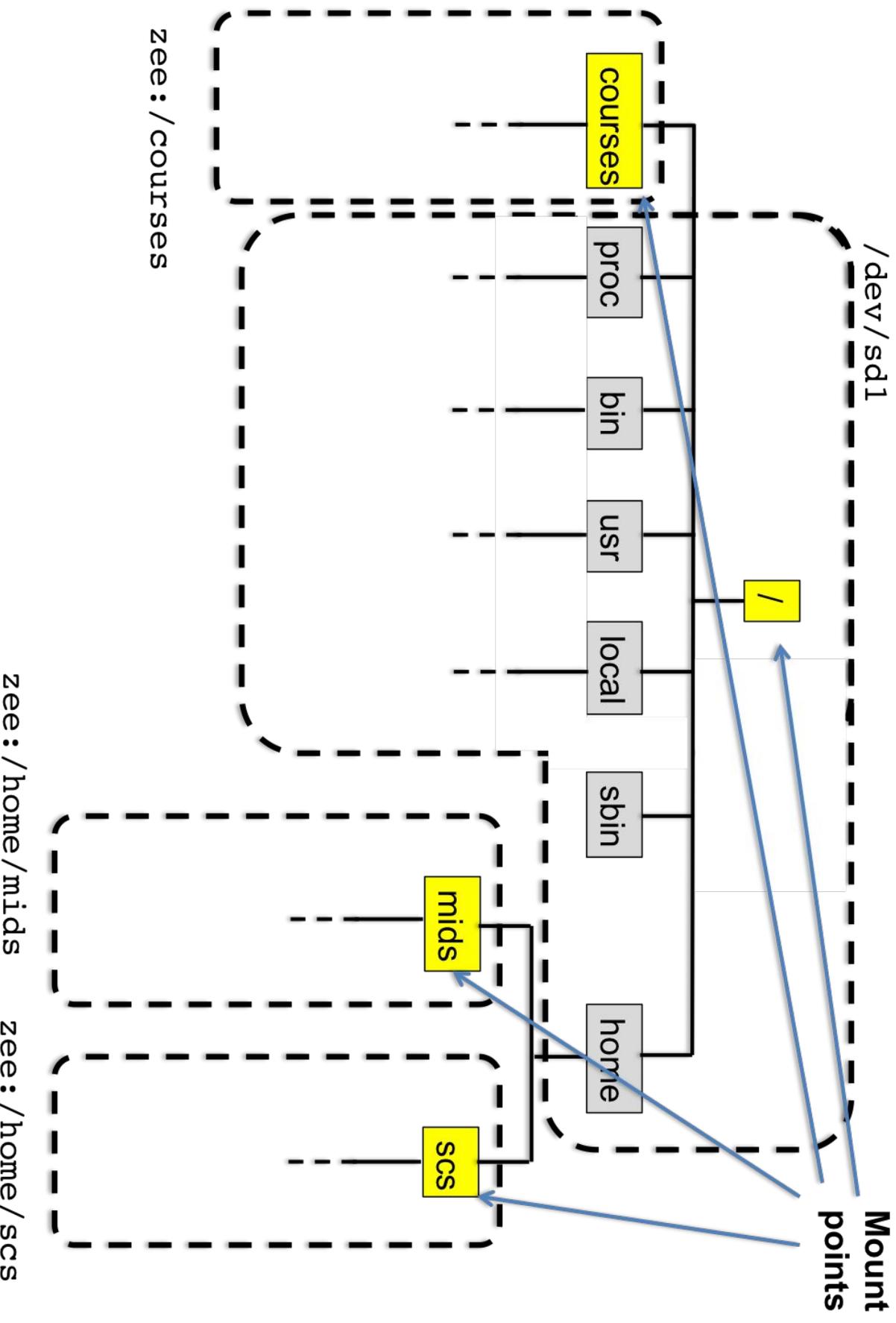
- Subdiretórios do diretório /

Diretório	Conteúdo
/proc	Um sistema de arquivos virtual que contém informações sobre recursos do sistema.
/root	O diretório home do usuário administrador do sistema.
/sbin	Programas para uso do sistema e do administrador do sistema.
/tmp	Diretório de arquivos temporários, ele é limpo a cada reinício do sistema.
/usr	Os diretórios home dos usuários do sistema.

Sistema de arquivos

- Pontos de montagem
 - Todas as partições são relacionadas ao sistema através de pontos de montagem;
 - Um ponto de montagem define um lugar de um conjunto de dados em particular no sistema;
 - Normalmente, todas as partições são conectadas a partir da partição root;
 - Na partição root, diretórios são criados, e estes diretórios são definidos como ponto de partida para acesso à partições montadas neles.

Sistema de arquivos



zee:/home/mids

zee:/home/scs

Sistema de arquivos

- Pontos de montagem
 - Podemos verificar a situação da partições e de seus pontos de montagem utilizando o comando **df**.

Sistema de arquivos

```
llsilva@LAR-04:~$ df -h
```

```
Sist. Arq.          Tam. Usado Disp. Uso% Montado em
```

```
udev              1,7G    0 1,7G  0% /dev
```

```
tmpfs             345M  5,8M 340M  2% /run
```

```
/dev/sda1        140G  11G 122G  9% /
```

```
tmpfs            1,7G  2,0M 1,7G  1% /dev/shm
```

```
tmpfs            5,0M  4,0K 5,0M  1% /run/lock
```

```
200.137.66.2:/export/llsilva  2,0T  479G 1,4T 26% /home/llsilva
```

```
tmpfs           345M  148K 345M  1% /run/user/10214
```

```
/dev/sdg1       15G  5,4G 9,6G 36% /media/llsilva/UBUNTU-
```

```
MATE
```

Máscara do usuário

- Mais conhecido como *umask* é a permissão padrão para a criação de arquivo e diretórios de um usuário.
- É basicamente o que as entidades não podem fazer no arquivo.

```
arthurcosmi@abcNoteUbu:~$ umask
```

```
0002
```

Máscara do usuário

- Exemplo: Para o usuário anterior com *umask 0002* deseja-se saber quais as permissões para o dono, grupo e outros de uma pasta que será criada.
- O primeiro zero pode ser descartado;
- O segundo representa as permissões do dono. Logo, deve-se diminuir a permissão máxima do número do *umask*. Assim:
$$\text{umask} = 0 \rightarrow \text{perm} = 7 - 0 \rightarrow \text{perm} = 7 = \text{rwx}$$
- Fazendo isso para os outros números encontramos os valores do grupo e outros:

Máscara do usuário

- Grupo:

`umask = 0` → `perm = 7 - 0` → `perm = 7 = rwx`

- Outros:

`umask = 2` → `perm = 7 - 2` → `perm = 5 = r-x`

```
arthurcosmi@abcNoteUbu:~$ ls -lh | grep lalala
```

```
drwxrwxr-x 2 arthurcosmi arthurcosmi 4,0K Mai 24 12:48 lalala
```

Máscara do usuário

- Para arquivos regulares existe uma exceção:
 - Quando o *umask* for 0,2,4 ou 6 ao invés de usar a permissão máxima igual a 7 usa-se 6. Assim, para um arquivo criado com a *umask* anterior (0002), temos:
- **Dono:**
 $umask = 0 \rightarrow perm = 6 - 0 \rightarrow perm = 6 = rw-$
- **Grupo:**
 $umask = 0 \rightarrow perm = 6 - 0 \rightarrow perm = 6 = rw-$

Máscara do usuário

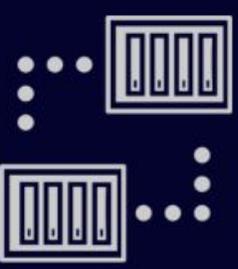
- Outros:

`umask = 0` → `perm = 6 - 2` → `perm = 4 = r--`

```
arthurcosmi@abcNoteUbu:~$ ls -lh | grep hehe
```

```
-rw-rw-r-- 1 arthurcosmi arthurcosmi 0 Mai 24 12:57 hehe
```

Processos



Processos

- O que é um processo?
 - Processo é uma instância de um programa computacional que está sendo executado. Um processo contém o código do programa e o status de sua execução.
 - No Linux, como no Unix, múltiplos processos podem ser executados simultaneamente por múltiplos usuários.
 - Em alguns casos, um processo iniciado por um usuário pode persistir mesmo quando o usuário se desloga.

- **Tipos de processos:**
 - Interativos
 - Automáticos
 - Daemons

Processos

- **Processo interativo**
 - É iniciado e controlado manualmente por um usuário através de uma sessão de terminal.
 - Normalmente iniciados em *foreground*.
 - Todos comandos que vimos até agora iniciam um processo interativo quando chamados.

```
labgrad / # ls
```

```
bin dev initrd.img lib64      mnt root srv usr
```

```
boot etc labgrad  lost+found opt run sys var
```

```
cdrom home lib    media  proc sbin tmp vmlinuz
```

```
labgrad / #
```

Processos

- **Processo interativo**
 - Alguns processos podem demorar e não requerer entrada de dados via *shell*. Neste caso, o *prompt* do seu terminal ficará preso esperando a finalização do processo.

```
2014100468@labgrad ~ $ gedit
```

...

- **Processo interativo**
 - Para evitar isso, podemos executar o processo em *background*.
2014100468@labgrad ~ \$ gedit &
[1] 26601
2014100468@labgrad ~ \$

- Controlando processos

(parte do) comando	Significado
comando_normal	Executa este comando em <i>foreground</i>
comando &	Executa este comando em <i>background</i>
jobs	Mostra os comandos em <i>background</i>
Ctrl+z	Suspende um processo de <i>foreground</i>
Ctrl+c	Interrompe um processo de <i>foreground</i>
bg	Reativa um processo suspenso
fg	Coloca um processo para <i>foreground</i>
kill	Envia um sinal para um processo (normalmente usado para finalizar)

Processos

- **Processos automáticos**
 - Também conhecidos como batch, são processos desconectados do terminal.

Processos

- **Daemons**
 - Daemons são processos não-interativos de sistema que rodam continuamente.
 - Normalmente são iniciados junto ao sistema e esperam em *background* até que seu serviço seja necessário.
 - Exemplo: *networking*
 - Este daemon é iniciado junto ao sistema e espera um programa cliente requisitar uma conexão, como um cliente FTP.

Processos

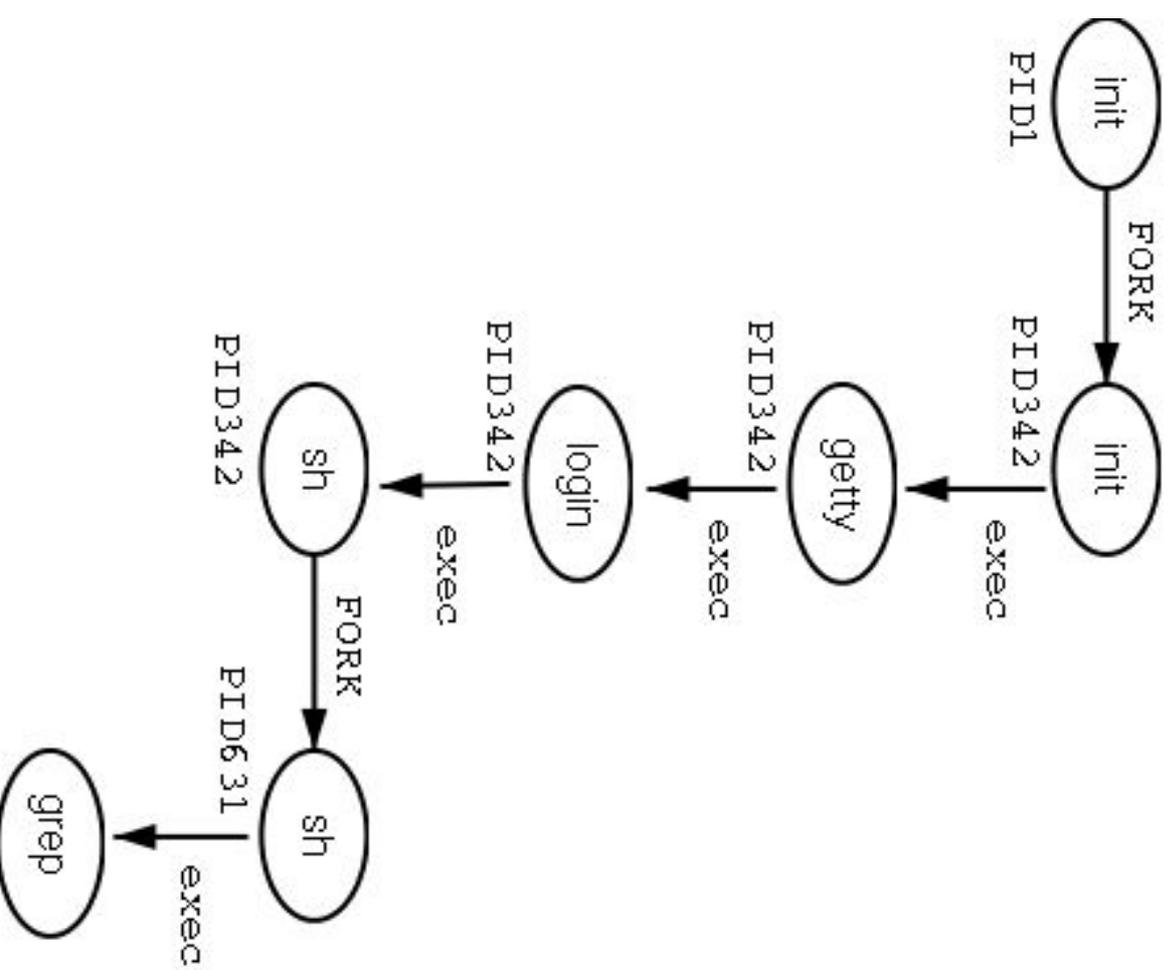
- Atributos de processos
 - **pid** : identificador único do processo.
 - **ppid** : identificador (pid) do pai que criou este processo.
 - **nice** : grau de “amigabilidade” do processo. Isto é usado para calcular a prioridade do processo. Quanto maior o *nice*, menos prioritário este processo será.
 - **Terminal ou TTY** : terminal ao qual este processo está conectado.
 - **RUID e EUID** : dono do processo.
 - **RGIO e EGIO** : grupo ao qual o processo pertence.

Processos

- Mostrando informações de processos
 - **ps** : visualiza processos
 - `ps -aux | grep username`
 - `ps -aux | grep processname`
 - **top** : apresenta os processos em ordem de maior consumo do sistema, atualizando a cada poucos segundos.
 - **pstree** : mostra árvore de processos, indicando relação de pai-filho entre os processos do sistema

Processos

- Criação de processos



Processos

- Terminando processos
 - Um processo pode terminar normalmente: não foi morto nem interrompido. Este retornará um valor ao pai indicando o resultado do processo.
 - Um processo pode ser finalizado através de um **signal**.
 - Existem uma variedade de sinais que você pode enviar a um processo.
 - Use o comando **kill** para terminar um processo.
 - O comando **kill -l** exibe todos os sinais possíveis para se utilizar.

Processos

- Sinais mais comuns

Nome do sinal	Número do sinal	Significado
SIGTERM	15	Termina o processo de um modo gentil
SIGINT	2	Interrompe um processo. O processo pode ignorar este sinal
SIGKILL	9	Interrompe o processo. O processo não pode ignorar este sinal
SIGHUP	1	Para daemons: relê o arquivo de configuração

Processos

- Agendando processos
 - O comando **sleep**:
 - O comando **sleep** aguarda uma quantidade de tempo em segundos.
 - É possível usar o comando **sleep** para atrasar a execução de outro comando.

```
llsilva@LAR-03:~$ sleep 10; echo "Passaram-se 10 segundos..."
```

```
Passaram-se 10 segundos...
```

- Agendando processos
 - Crontab:
 - Tabela de execução de processos agendados.
 - **crontab -e**

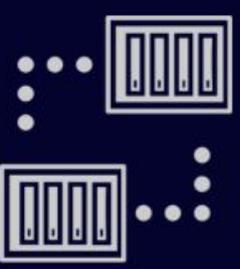
Exercícios

1. Em qual partição está o seu diretório home?
2. Em quantas partições estão o seu sistema?
3. Qual é o total em tamanho da sua instalação do Linux?
4. Crie um novo diretório em seu diretório home
5. Você pode mover este diretório para o mesmo nível de seu diretório home?
6. Copie todos os arquivos do diretório /usr/share/pixmaps para o novo diretório.
7. Liste todos os arquivos em ordem alfabética reversa.
8. Mude para o seu diretório home. Crie um novo diretório e copie todos os arquivos do diretório /etc para ele. Tenha certeza de que o você também copiou os diretórios que são subdiretórios de /etc (cópia recursiva).
9. Mude para um novo diretório e faça um diretório para arquivos começando com uma letra maiúscula e um para arquivos começando com uma letra minúscula. Mova todos os arquivos para seus diretórios apropriados. Use o menor número de comandos possível.
10. Remova os arquivos restantes.
11. Detele o diretório e todo o seu conteúdo usando um só comando.
12. Onde está o programa **grep**?
13. Faça link simbólico no seu diretório home para o diretório /var/tmp. Cheque se ele realmente funciona.
14. Faça outro link simbólico no seu diretório home para este link. Verifique se funciona. Remova o primeiro link e liste o conteúdo do diretório. O que aconteceu com o link?

Exercícios

15. Abre o **top** em um terminal enquanto você realiza os exercícios em outro.
16. Execute o comando **ps**.
17. Leia as páginas de manual (*man*) para saber como listar todos os seus processos.
18. Execute o comando `find /`. Qual efeito que ele tem no uso do sistema? Interrompa este comando.
19. O que faz o **kill -9**?
20. Execute o **xclock** em *foreground*. Agora passe-o para *background*. Pare este programa com o comando **kill**.
21. Execute o **xcalc** diretamente no *background*. Passe-o para *foreground*.
22. Quanto tempo demora para executar **ls** no seu diretório atual?
23. Qual é o seu *TTY* atual?
24. Diga qual é o comando que está causando maior consumo no seu sistema.

Editores de texto



Editores de texto

- Por que usar um editor de texto?
 - Escrever:
 - Scripts;
 - Programas;
 - Websites;
 - Livros;
 - etc.
 - Dominar um editor de texto favorece:
 - Independência;
 - Produtividade;
 - Eficiência.

Editores de texto

- Qual editor de texto devo usar?
 - Editor em modo gráfico:
 - Interface amigável;
 - Fácil de usar;
 - Software complexo (pesado);
 - Depende do modo gráfico.
 - Editor em modo texto:
 - Interface pouco amigável;
 - Uso menos intuitivo;
 - Software simples (leve);
 - Independe do modo gráfico.
- Pode ser facilmente usado remotamente

Editores de texto



The screenshot displays a Linux desktop environment with two windows open. The top window is a code editor titled 'heap' showing C++ code for a heap sort algorithm. The bottom window is a text editor titled 'texto.txt' displaying a Portuguese text document.

Code Editor (heap):

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int max (int *a, int n, int i, int j, int
5     int m = i;
6     if (j < n && a[j] > a[m]) {
7         m = j;
8     }
9     if (k < n && a[k] > a[m]) {
10        m = k;
11    }
12    return m;
13 }
14
15 void downheap (int *a, int n, int i) {
16     while (1) {
17         int j = max(a, n, i, 2 * i + 1, 2
18             if (j == i) {
19                 break;
20             }
21         int t = a[i];
22         a[i] = a[j];
23         a[j] = t;
24         i = j;
25     }
26 }

```

Text Editor (texto.txt):

```

1
2
3
4 a apreciação da importância dos conhecimentos estratégicos para atingir a excelência.
5
6 No mundo atual, o desenvolvimento contínuo de distintas formas de atuação assume importantes posições no estabelecimento das
7 direções preferenciais no sentido do progresso. A prática cotidiana prova que a hegemonia do ambiente político prepara-nos
8 para enfrentar situações atípicas decorrentes do sistema de formação de quadros que corresponde às necessidades. A nível
9 organizacional, o novo modelo estrutural aqui preconizado garante a contribuição de um grupo importante na determinação de
10 todos os recursos funcionais envolvidos.
11
12 Do mesmo modo, a consulta aos diversos militantes talvez venha a ressaltar a relatividade dos índices pretendidos. Assim
13 mesmo, a contínua expansão de nossa atividade estimula a padronização dos níveis de motivação departamental. Não obstante, o
14 entendimento das metas propostas estende o alcance e a importância dos modos de operação convencionais. O que temos que ter
15 sempre em mente é que o desafiador cenário globalizado desafia a capacidade de equalização dos relacionamentos verticais entre
16 as hierarquias. O incentivo ao evanço tecnológico, assim como a estrutura atual da organização ainda não demonstrou
17 convincentemente que vai participar na mudança das regras de conduta normativas.
18
19 O cidadão em identificar pontos críticos na execução dos pontos do programa oferece uma interessante oportunidade para
20 verificação do remanejamento dos quadros funcionais. Nunca é demais lembrar o peso e o significado destes problemas, uma vez
21 que o surgimento do comércio virtual cumpre um papel essencial na formulação do sistema de participação geral. Por outro lado,
22 a constante divulgação das informações pode nos levar a considerar a reestruturação das posturas dos órgãos dirigentes com
23 relação às suas atribuições.
24
25 Todavia, o início de atividade geral de formação de atitudes nos obriga à análise do orçamento setorial. Todas estas questões,
26 devidamente ponderadas, levantam dúvidas sobre se a adoção de políticas descentralizadoras afeta positivamente a correta
27 previsão dos métodos utilizados na avaliação de resultados. O empenho em analisar a mobilidade dos capitais internacionais é
28 uma das consequências de alternativas às soluções ortodoxas.
29
30 Percebemos, cada vez mais, que a expansão dos mercados mundiais possibilita uma melhor visão global dos procedimentos
31 normalmente adotados. Pensando mais a longo prazo, a revolução dos costumes aponta para a melhoria das formas de ação. É
32 importante questionar o quanto o acompanhamento das preferências de consumo auxilia a preparação e a composição das novas
33 proposições. No entanto, não podemos esquecer que o comprometimento entre as equipes acarreta um processo de reformulação e
34 modernização do retorno esperado a longo prazo.
35
36 Por conseguinte, o julgamento imparcial das eventualidades faz parte de um processo de gerenciamento das condições financeiras
37 e administrativas exigidas. Evidentemente, a necessidade de renovação processual deve passar por modificações
38 independentemente do investimento em reciclagem técnica. Costaria de enfatizar que o fenômeno da Internet não pode mais se
39 dissociar do processo de comunicação como um todo. Podemos já vislumbrar o modo pelo qual a percepção das dificuldades agrega
40 valor ao estabelecimento das condições Inagavelmente apropriadas.
41
42 Desta maneira, a valorização de fatores subjetivos causa impacto indíreto na reavaliação do levantamento das variáveis
43 envolvidas. Neste sentido, o aumento do diálogo entre os diferentes setores produtivos apresenta tendências no sentido de
44
45
46

```

Desktop Environment:

- Window titles: heap, texto.txt - /home/2014100468 - Atom
- System tray: 2014100468, 10:10, 100%
- Taskbar: C -> Largura das tabelagens: 4 -> Lin 10, Col 15 -> IN\$

Editores de texto

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int max (int *a, int n, int i, int j, int k) {
5     int m = i;
6     if (j < n && a[j] > a[m]) {
7         m = j;
8     }
9     if (k < n && a[k] > a[m]) {
10        m = k;
11    }
12    return m;
13 }
14
15 void downheap (int *a, int n, int i) {
16    while (i) {
17        int j = max(a, n, i, 2 * i + 1, 2 * i + 2);
18        if (j == i) {
19            break;
20        }
21        int t = a[i];
22        a[i] = a[j];
23        a[j] = t;
24        i = j;
25    }
26 }
27
28 void heapsort (int *a, int n) {
29    int i;
30    for (i = (n - 2) / 2; i >= 0; i--) {
31        downheap(a, n, i);
32    }
33    for (i = 0; i < n; i++) {
34        int t = a[n - i - 1];
35        a[n - i - 1] = a[0];
36        a[0] = t;
37        downheap(a, n - i - 1, 0);
38    }
39 }
40
41 int main () {
    "heapsort.c" 51L, 1067C

```

GNU nano 2.2.6
Terminal

im, vi e venci.

Acima de tudo, é fundamental ressaltar que a complexidade dos estudos efetuados facilita a criação do fluxo de informações. Ainda assim, existem dois mundos atuais, o desenvolvimento contínuo de distintas formas de atuação assume importantes posições no estabelecimento das direções preferenciais do mesmo modo, a consulta aos diversos militantes talvez venha a ressaltar a relatividade dos índices pretendidos. Assim mesmo, a contínua expansão do cuidado em identificar pontos críticos na execução dos pontos do programa oferece uma interessante oportunidade para verificação do remanejamento. Todavia, o início da atividade geral de formação de atitudes nos obriga à análise do orçamento setorial. Todas estas questões, devidamente ponderadas percebemos, cada vez mais, que a expansão dos mercados mundiais possibilita uma melhor visão global dos procedimentos normalmente adotados. Pensando por conseguinte, o julgamento imparcial das eventualidades faz parte de um processo de gerenciamento das condições financeiras e administrativas existentes. Desta maneira, a valorização de fatores subjetivos causa impacto indireto na reavaliação do levantamento das variáveis envolvidas. Neste sentido, as experiências acumuladas demonstram que o consenso sobre a necessidade de qualificação auxilia a preparação e a composição das diversas correntes. O cuidado em identificar pontos críticos na hegemonia do ambiente político estimula a padronização dos conhecimentos estratégicos para atingir a expressão. No entanto, não podemos esquecer que a consulta aos diversos militantes obstaculiza a apreensão da importância dos relacionamentos verticais entre. Nunca é demais lembrar o peso e o significado destes problemas, uma vez que a complexidade dos estudos efetuados cumpre um papel essencial na formulação e na implementação de estratégias. Percebemos, cada vez mais, que a mobilidade dos capitais internacionais acarreta um processo de reformulação e modernização do retorno esperado a longo prazo. No mundo atual, o fenômeno da Internet não pode mais se dissociar das novas proposições. A prática cotidiana prova que a percepção das dificuldades e o empenho em analisar a consolidação das estruturas maximiza as possibilidades por conta do fluxo de informações. Por conseguinte, a consolidação dos dados, é fundamental ressaltar que a determinação clara de objetivos não pode mais se dissociar do orçamento setorial. Assim mesmo, o compromisso em avanço tecnológico, assim como o desenvolvimento contínuo de distintas formas de atuação pode nos levar a considerar a reestruturação das atividades.

14, 1
Topo

[O] Obter Ajuda	[G] Gravar	[L] Ler o Arq	[P] Pág Anter	[R] Recort Txt	[C] Pos Atual
[X] Sair	[J] Justificar	[W] Onde está?	[N] Próx Pág	[M] Colar Txt	[A] Para Spell

Editores de texto

- Nano
 - Editor de texto voltado à simplicidade.
 - Pros:
 - Interface simples e mais amigável.
 - Conjunto de comandos compactos.
 - Cons:
 - Poucas funcionalidades.
 - Atalhos pouco intuitivos (CTRL-O para salvar?)

Editores de texto

- GNU Emacs
 - Pros:
 - Extensivo.
 - Customizável.
 - Bem documentado.
 - *Syntax coloring*.
 - Muitas funcionalidades.
 - Cons:
 - Curva de aprendizado íngreme.
 - Interface-usuário menos amigável.

Editores de texto

- Vi(m)
 - Editor de texto modal.
 - Pros:
 - Disponível em quase qualquer distribuição UNIX (vi).
 - Extensivo.
 - Customizável.
 - Bem documentado.
 - *Syntax coloring*.
 - Muitas funcionalidades.
 - Cons:
 - Curva de aprendizado *muito* íngreme.
 - Interface-usuário pouco amigável.

- **Controle modal**
 - **Comando**
 - Neste modo você se move pelo texto, busca, substitui, marca blocos de texto e realiza edições.
 - As teclas pressionadas são interpretadas como comandos.
 - Alguns comandos entram no modo de **Inserção**.
 - **Inserção**
 - Modo para escrita de texto.
 - As teclas são inseridas como texto.
 - **Visual**
 - Modo de seleção de texto.
 - Comandos afetarão todo o texto selecionado.

- Comandos básicos
 - Movendo-se pelo texto
 - **h** move o cursor para a esquerda.
 - **l** move para a direita.
 - **k** move pra cima.
 - **j** move pra baixo.

^

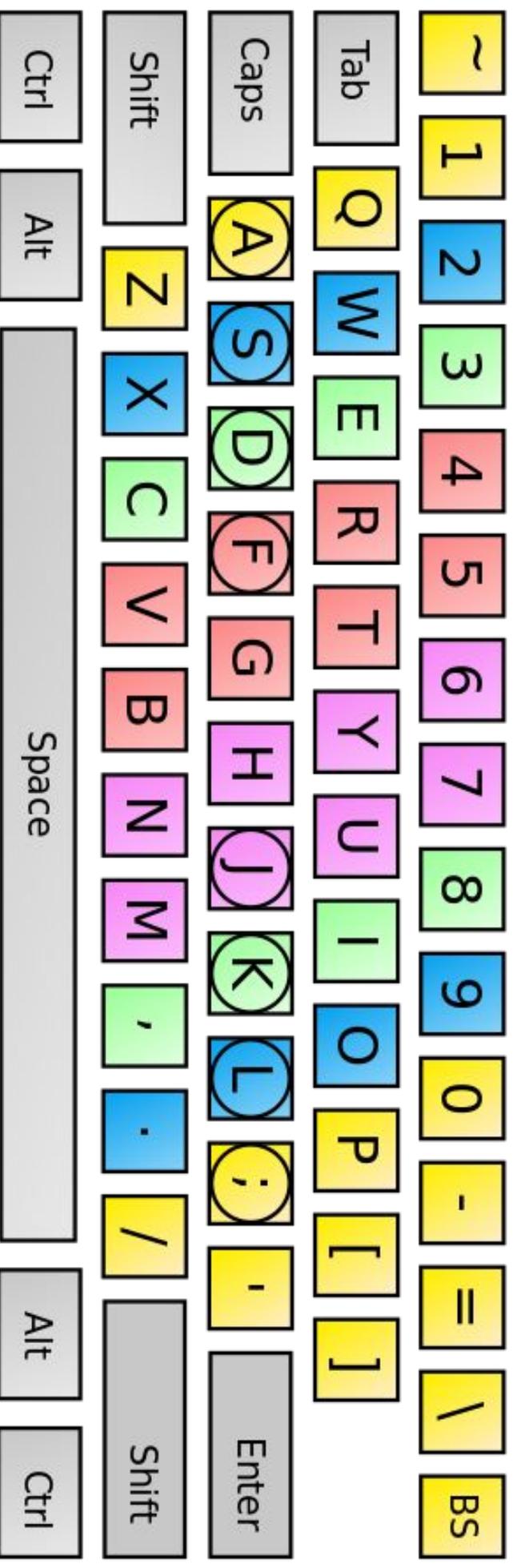
k

< h l >

j

v

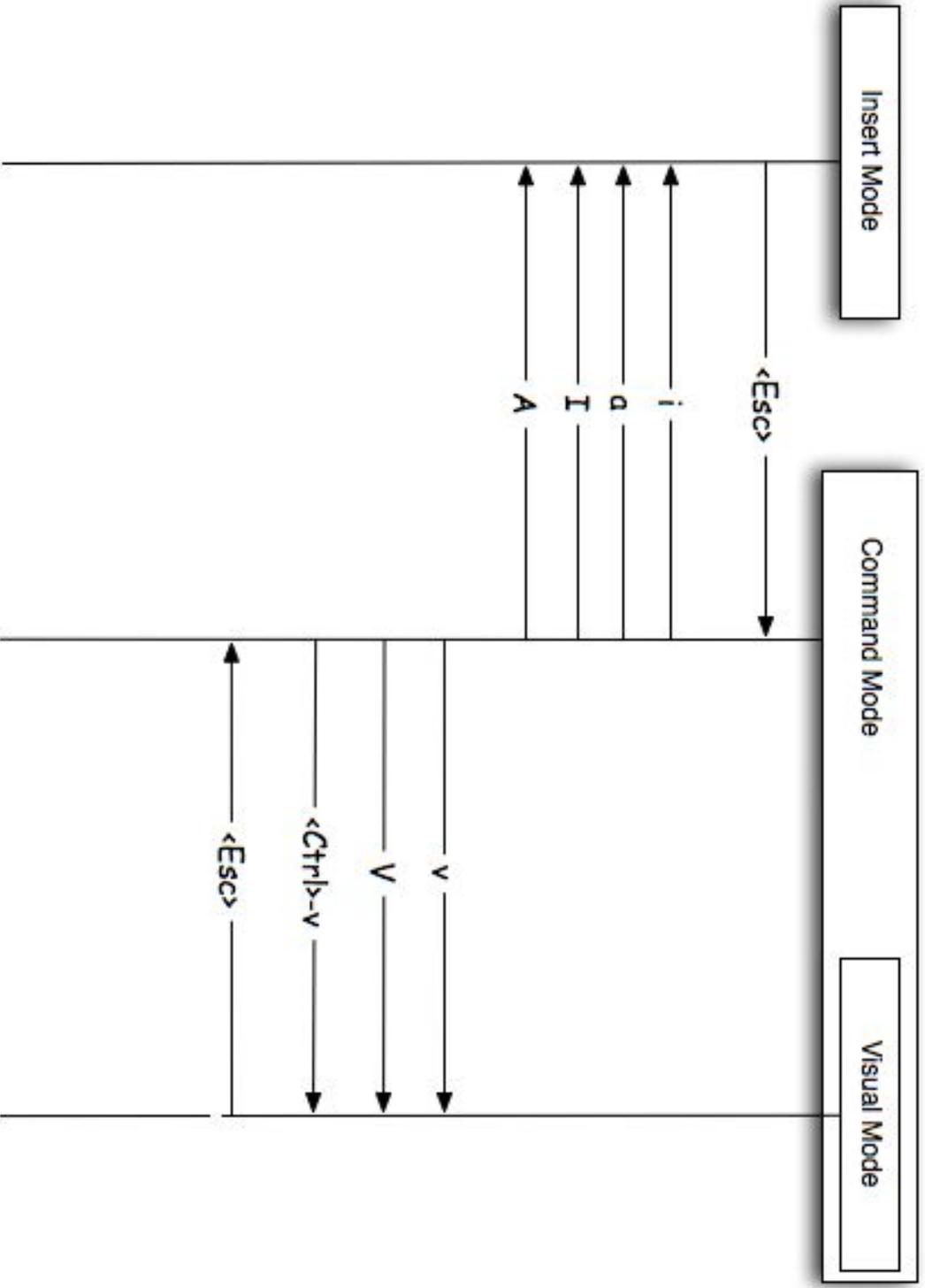
- Comandos básicos
 - Movendo-se pelo texto
 - **h l k j** ???
 - *Touch typing*



- Comandos básicos
 - Salvando e saindo do vim
 - **:w** salva o arquivo (*write*).
 - **:q** sai do arquivo (*quit*).
 - **:q!** força a saída, descartando as alterações feitas.
 - **:wq** salva e sai.
 - **:w newfile** salva no arquivo *newfile*.
 - **:w!** sobrepõe permissão *read-only*.

- Comandos básicos
 - Mudando para o modo de Inserção
 - **i** insere texto antes do cursor.
 - **a** adiciona texto após o cursor.
 - **o** cria uma nova linha abaixo e insere texto nela.
 - **I** insere texto no início da linha do cursor.
 - **A** adiciona texto no final da linha do cursor.
 - **O** cria uma nova linha acima e insere texto nela.
 - Mudando de volta para o modo Comando
 - **<ESC>**

- Comandos básicos
 - Mudando para o modo Visual
 - `v` entra no modo visual.
 - `V` entra no modo visual-linha.
 - `Ctrl-V` entra no modo visual-bloco.
 - Mudando de volta para o modo Comando
 - `<Esc>`



- Operadores populares
 - Movimentação
 - **G** leva o cursor até o final do arquivo.
 - **gg** leva até o começo do arquivo.
 - **:n** move até a linha *n*.
 - **\$** move o cursor ao final da linha atual.
 - **0** move ao início da linha atual.
 - **w** move uma palavra para frente (*word*).
 - **b** move uma palavra para trás (*back*).

- Operadores populares
 - Manipulação de texto
 - **x** apaga o caractere abaixo do cursor.
 - **dd** deleta a linha atual do cursor.
 - **yy** copia a linha do cursor.
 - **p** cola o texto em *buffer* (copiado ou deletado).
 - **np** cola o texto *n* vezes.
 - **u** desfaz (*undo*)
 - **Ctrl-r** refaz (*redo*)

- Operadores populares
 - Busca e substituição
 - `/padrao` busca no arquivo o *padrao* inserido.
 - `/<Enter>` ou `n` realiza a busca anterior novamente.
 - `:%s/padrao/novotexto/g` substitui *padrao* em *novotexto* no arquivo inteiro.
 - `s/padrao/novotexto/g` substitui apenas na linha atual.

- Customizando o Vim
 - Crie o arquivo `~/.vimrc`

```
" Garantir sintaxe colorida
syntax enable

" Barra de números
set number

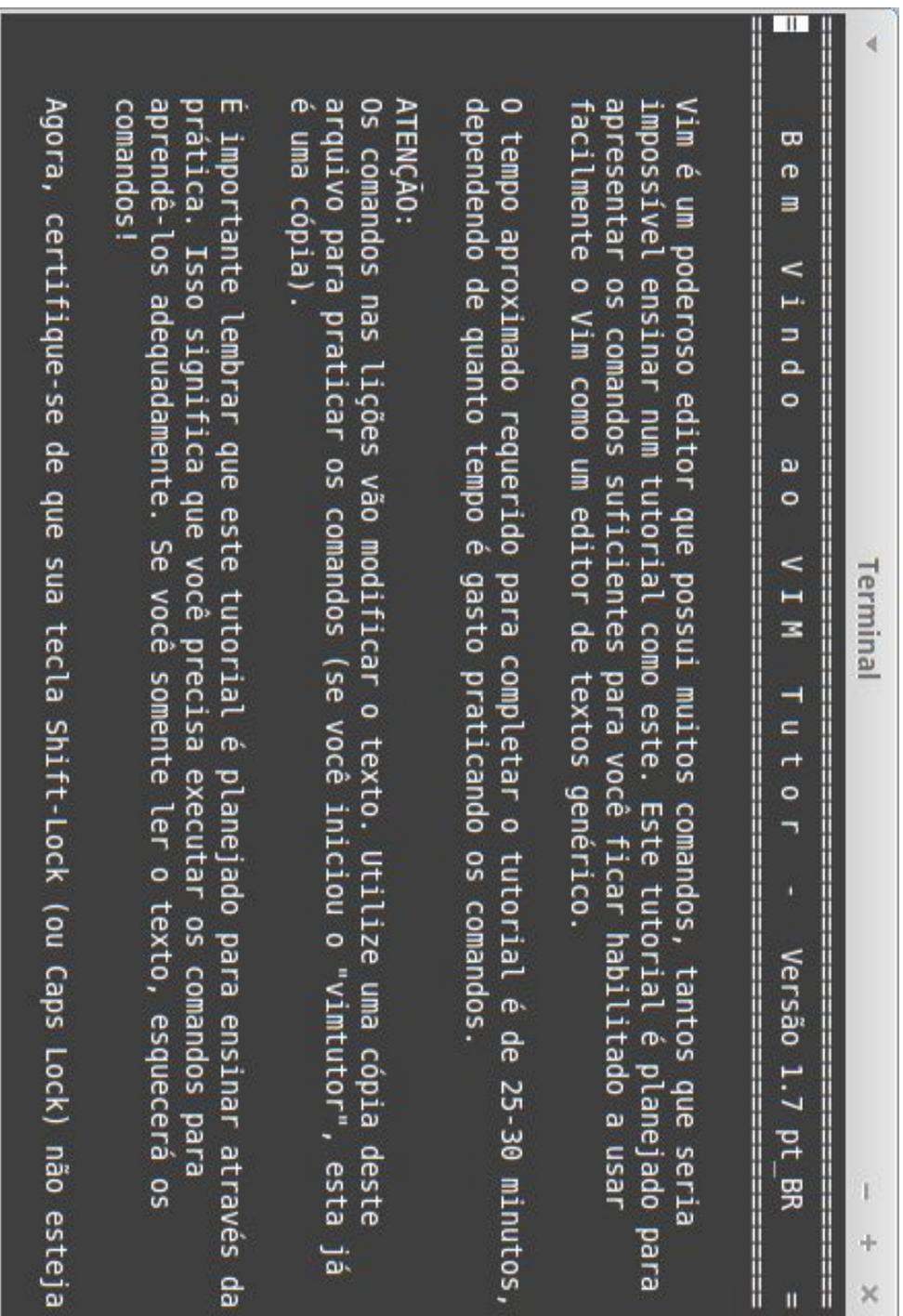
" Recuo automático
set autoindent
set smartindent

" Tamanho do TAB
set softtabstop=4
set shiftwidth=4
set tabstop=4

" Diga ao vim que você usa fundo escuro
set background=dark
```

Vim

- Tutorial para estudo
 - *vimtutor* (digite no terminal)



```
Terminal
-----
 Bem Vindo ao VIM Tutor - Versão 1.7 pt_BR =
-----
 Vim é um poderoso editor que possui muitos comandos, tantos que seria
impossível ensinar num tutorial como este. Este tutorial é planejado para
apresentar os comandos suficientes para você ficar habilitado a usar
facilmente o Vim como um editor de textos genérico.

 O tempo aproximado requerido para completar o tutorial é de 25-30 minutos,
dependendo de quanto tempo é gasto praticando os comandos.

 ATENÇÃO:
 Os comandos nas lições vão modificar o texto. Utilize uma cópia deste
arquivo para praticar os comandos (se você iniciou o "vimtutor", esta já
é uma cópia).

 É importante lembrar que este tutorial é planejado para ensinar através da
prática. Isso significa que você precisa executar os comandos para
aprendê-los adequadamente. Se você somente ler o texto, esquecerá os
comandos!

 Agora, certifique-se de que sua tecla Shift-Lock (ou Caps Lock) não esteja
```

Exercícios

1. Abra no *vim* o arquivo `textao.txt`
2. Vá para a última linha.
3. Vá para a primeira linha.
4. Repita os passos 2 e 3 cinco vezes.
5. Vá para o meio do arquivo com um comando.
6. Apague as linhas 47 e 48 (sem entrar no modo Inserção).
7. Encontre a primeira ocorrência da palavra “desenvolvimento”.
8. Será que existe a palavra “banana” nesse texto?
9. Substitua toda ocorrência de “vim” por “vi(m)”.
10. Substitua toda ocorrência de “vi” por “BOM”.
11. Destaque o passo 10.
12. Destaque e refaça o passo 11 (não pode mais existir “BOM” no texto).
13. Vá para a última linha e comece a escrever algo em uma nova linha abaixo.
14. Copie essa nova linha e cole uma igual antes da primeira linha.
15. Salve esse arquivo editado em `textao-editado.txt`
16. Saia do arquivo atual sem salvar as alterações.

Exercícios

17. Abra no *vim* o arquivo `heapsort.c`
18. Você quer mudar o nome da função `downheap` para `heapbaixo`. Faça isso com um comando e mude todas as ocorrências de `downheap`.
19. Selecione a função `heapsort` inteira e copie-a.
20. Cole a função copiada depois da `main`.
21. Faça o passo 20 *cinquenta* vezes.
22. Destaque o passo 21.
23. Selecione a função `heapsort` que foi colada depois da `main` e aperte **d** (no modo visual).
24. Execute o comando `:%s / . /a/g`. O que aconteceu? O que `“ . ”` significa?
25. Destaque o passo 24.
26. Substitua toda ocorrência de ponto final `“ . ”` em `“PONTO”`. *Dica:* `“ \ ”` anula o efeito especial de um caractere.
27. Faça cada linha do programa virar um comentário ao adicionar `“ // ”` no começo de cada uma. *Dica:* `“ \ ”` representa o começo de uma linha.