

**INSTITUTO FEDERAL**

Farroupilha

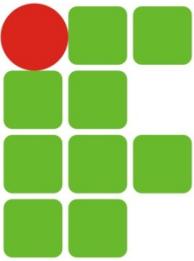
Campus Avançado Uruguaiana

# Linux Comandos Básicos

---

Prof. Leandro Martins Dallanora

[leandro.dallanora@iffarroupilha.edu.br](mailto:leandro.dallanora@iffarroupilha.edu.br)

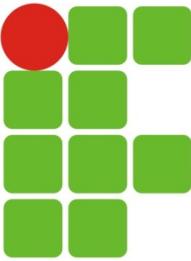


# Introdução

- Todo sistema operacional possui um modo em linha de comando que nos permite executar diversas tarefas.
- No linux a linha de comando tem o seguinte aspecto:



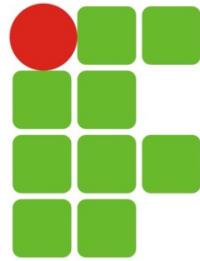
The screenshot shows a Linux desktop interface with a terminal window open. The terminal window has a title bar with the text "aluno@IFFAR: ~". Below the title bar is a menu bar with options: Arquivo, Editar, Ver, Pesquisar, Terminal, and Ajuda. The main area of the terminal window displays the command prompt "aluno@IFFAR:~\$". The background of the desktop is light gray, and there are several green decorative rounded rectangles at the bottom.



# Introdução

---

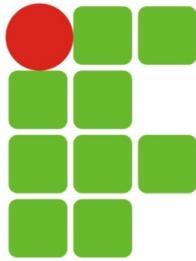
- Assim como um prompt do DOS ou o terminal do Mac OS, a linha de comando do Linux dá acesso a um processador de comandos chamado Bash, que permite que você controle o sistema ao fornecer instruções baseadas em texto.
- Ao abrir a linha de comando, você verá o prompt ***usuario@maquina:\$*** para usuários comuns ou ***root@maquina:#*** caso você esteja utilizando Root.
- Root corresponde ao superusuário em sistemas Linux, e esse usuário tem controle completo sobre o sistema operacional.



# Sistema de arquivos do Linux

- No mundo Linux, tudo é um arquivo: teclados, impressoras, dispositivos de rede – tudo.
- Todos os arquivos podem ser visualizados, editados, apagados, criados, movidos.
- O sistema de arquivos do Linux é constituído de uma série de diretórios que se originam na raiz do sistema e arquivos ( / ).
- Para ver seu diretório atual, digite **pwd** no terminal.

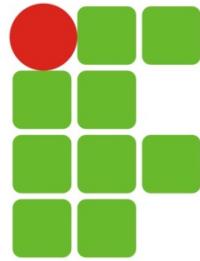
```
aluno@IFFAR:~$ pwd  
/home/aluno  
aluno@IFFAR:~$ █
```



# Mudando de diretório

- Para ir para outro diretório, digite **cd diretório** usando o caminho *absoluto* ou *relativo* do novo diretório, de acordo com a sua localização corrente.
- O **caminho absoluto** corresponde ao caminho de um arquivo em relação ao diretório raiz ( / ).
- Por exemplo, para ir para o seu diretório de downloads a partir de qualquer local, você deve fornecer o **caminho absoluto** do diretório, usando:

```
aluno@IFFAR:~$ cd /home/aluno/Downloads/  
aluno@IFFAR:~/Downloads$ █
```



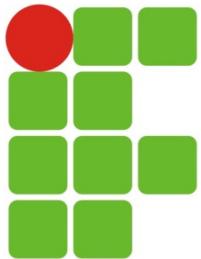
# Mudando de diretório

- Se você estiver no diretório do seu usuário ( exemplo /home/aluno ), você poderá utilizar o **caminho relativo**, ou seja, relativo à sua localização corrente, digitando apenas:

```
aluno@IFFAR:~$ cd Downloads/  
aluno@IFFAR:~/Downloads$
```

- O comando **cd ..** faz você retroceder um nível no sistema de arquivos, nos levando de volta ao diretório home do usuário, como mostrado aqui:

```
aluno@IFFAR:~/Downloads$ cd ..  
aluno@IFFAR:~$ █
```



# Listando um diretório

- Para listar o conteúdo de um diretório podemos utilizar o comando **ls**

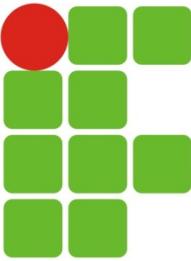
```
aluno@IFFAR:~$ ls  
'Área de Trabalho'  
Documentos  
aluno@IFFAR:~$ █
```

Downloads  
Imagens

Modelos  
Música

Público  
Vídeos

- Podemos passar alguns argumentos para esse comando, como por exemplo **ls -l** ( para listar com mais detalhes ), ou ainda, **ls -a** ( para listar arquivos ocultos também ), ou ainda **ls -la** ( para mostrar todos os arquivos, incluindo os ocultos, com maiores detalhes ).



# Conhecendo os comandos: man pages

- Para conhecer melhor um comando e suas opções e argumentos, você pode consultar a sua documentação (ou seja, sua *página de manual* ou *man page*) por meio do comando ***man comando***.
- Por exemplo, para saber mais a respeito do comando **ls**, digite **man ls**.

```
aluno@IFFAR:~$ man ls
```

LS(1) General Commands Manual

LS(1)

**NOME**

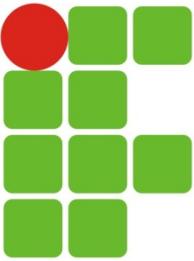
ls, dir, vdir - lista o conteúdo do diretório

**SINOPSE**

ls [opções] [arquivo...]  
dir [arquivo...]  
vdir [arquivo...]

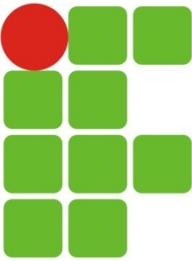
Opções POSIX: [-CFRacdilqrstu1]

Manual page ls(1) line 1 (press h for help or q to quit)



# Conhecendo os comandos: man pages

- A *man page* oferece informações úteis (embora não tenha uma aparência muito amigável) sobre o comando **ls**, incluindo o seu uso, a descrição e as opções disponíveis.
- Como você pode ver na seção de descrição, o comando **ls** lista todos os arquivos do diretório de trabalho corrente por padrão, porém **ls** também pode ser usado para obter informações sobre um arquivo em particular.
- Por exemplo, de acordo com a *man page*, você pode utilizar a opção **-a**, juntamente com o **ls**, para mostrar todos os arquivos, incluindo os *diretórios ocultos* – diretórios que não são apresentados na listagem padrão do **ls**.



# Criando um novo arquivo ou diretório

- Para criar um arquivo novo e vazio chamado *meuarquivo*, utilize o comando **touch**.

```
aluno@IFFAR:~$ touch meuarquivo
```

- Para criar um diretório novo em seu diretório de trabalho corrente, digite **mkdir diretório**, conforme mostrado aqui:

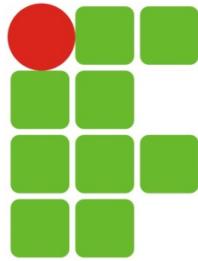
```
aluno@IFFAR:~$ mkdir meudir
```

```
aluno@IFFAR:~$ ls
```

```
'Área de Trabalho' Documentos Downloads Imagens  
meuarquivo meudir Modelos Música Público  
Vídeos
```

```
aluno@IFFAR:~$ cd meudir/
```

```
aluno@IFFAR:~/meudir$ █
```



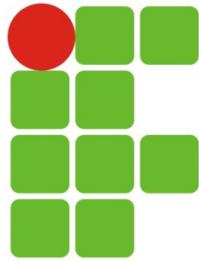
# Copiando, movendo e apagando arquivos

- Para copiar um arquivo, utilize o comando **cp**, como mostrado aqui:

```
aluno@IFFAR:~/meudir$ cp /home/aluno/meuarquivo meuarquivo2
```

- O comando tem a seguinte sintaxe: *cp origem destino*. Ao usar o **cp**, o arquivo original permanece inalterado e uma cópia é criada no destino desejado.
- De modo semelhante, você pode mover um arquivo de um local para outro por meio do comando **mv**. A sintaxe é idêntica à de **cp**, porém, dessa vez, o arquivo é transferido do local de origem para o destino.

```
aluno@IFFAR:~/meudir$ mv meuarquivo2 /home/aluno/
```



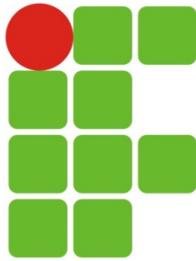
# Copiando, movendo e apagando arquivos

- Um arquivo pode ser apagado do sistema de arquivos por meio do comando **rm arquivo**.
- Para apagar arquivos *recursivamente*, utilize o comando **-r**.

```
aluno@IFFAR:~$ rm meuarquivo
```

## AVISO

- Tome cuidado ao apagar arquivos, particularmente de forma recursiva! Alguns brincam que o primeiro comando a ser ensinado aos iniciantes em Linux é *rm -rf* a partir do diretório root, que apaga necessariamente todo o sistema de arquivos. Isso mostra aos novos usuários o poder de executar ações como root. Não tente fazer isso em casa!



# Adicionando texto a um arquivo

- O comando **echo** ecoa o que quer que você digite no terminal, conforme mostrado aqui:

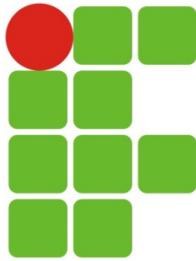
```
aluno@IFFAR:~$ echo Bem Vindo  
Bem Vindo
```

- Para salvar texto em um arquivo, você pode redirecionar a sua saída para um arquivo por meio do símbolo **>**, em vez de enviá-la para o terminal.

```
aluno@IFFAR:~$ echo Bem Vindo > texto
```

- Para ver o conteúdo de seu novo arquivo, utilize o comando **cat**.

```
aluno@IFFAR:~$ cat texto  
Bem Vindo
```

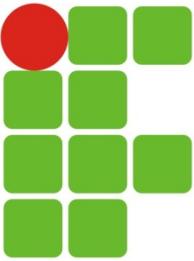


# Adicionando texto a um arquivo

- Agora envie um linha de texto diferente para *texto*, como mostrado a seguir:

```
aluno@IFFAR:~$ echo Seja Bem Vindo > texto  
aluno@IFFAR:~$ cat texto  
Seja Bem Vindo
```

- O comando **>** sobrescreve o conteúdo anterior do arquivo. Se você enviar outra linha para *texto*, essa nova linha sobrescreverá a saída do comando anterior.
- Como você pode ver, o conteúdo de *texto* agora é igual a ***Seja Bem Vindo.***

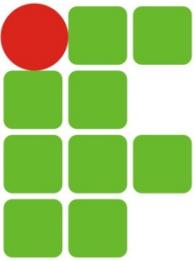


# Concatenando texto a um arquivo

- Para concatenar texto a um arquivo, utilize `>>`, conforme mostrado aqui:

```
aluno@IFFAR:~$ echo Seja Bem Vindo Novamente >> texto  
aluno@IFFAR:~$ cat texto  
Seja Bem Vindo  
Seja Bem Vindo Novamente
```

- Como podemos notar, a concatenação preserva o conteúdo anterior do arquivo.

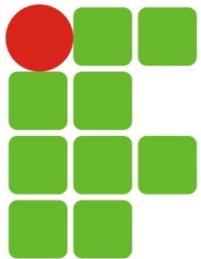


# Permissões de arquivo

- Se você observar a longa saída de `ls -l` em *texto* será possível ver as permissões correntes de *texto*.

```
aluno@IFFAR:~$ ls -l texto  
-rw-r--r-- 1 aluno aluno 40 set 13 15:35 texto
```

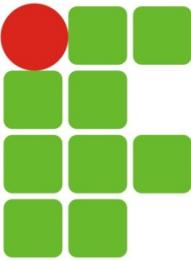
- Da esquerda para a direita, podemos ver o tipo do arquivo e as permissões (-rw-r--r--), a quantidade de links para o arquivo (1), o usuário e o grupo que são os donos do arquivo (aluno), o tamanho do arquivo (40 bytes), a última vez que o arquivo foi alterado (Set 13, 15:35) e, por fim, o nome do arquivo (texto).



# Permissões de arquivo

---

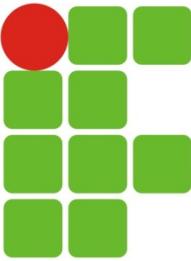
- Os arquivos Linux têm permissões para leitura (**r**), escrita (**w**) e execução (**x**), além de três conjuntos de permissões de usuário: permissões para o dono, para o grupo e para todos os usuários.
- As três primeiras letras representam as permissões para o **dono**, as três letras seguintes representam as permissões para o **grupo**, e as três últimas letras representam as permissões para **todos os usuários**.
- Como texto foi criado a partir da conta de usuário *aluno*, os donos do arquivo são o usuário *aluno* e o grupo *aluno*, como podemos notar na saída, que contém *aluno aluno*.



# Permissões de arquivo

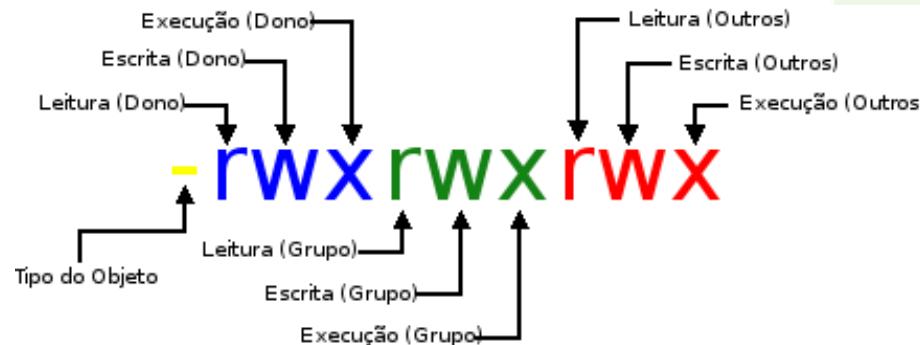
```
aluno@IFFAR:~$ ls -l texto  
-rw-r--r-- 1 aluno aluno 40 set 13 15:35 texto
```

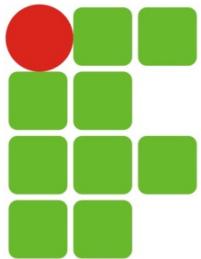
- O usuário *aluno* tem permissões de leitura e escrita no arquivo (**rw**).
- Outros usuários do grupo, se houver, poderão ler o arquivo (**r**), porém não poderão escrever nele, nem executá-lo.
- O último (**r**) mostra que todos os usuários do sistema de arquivos podem ler o arquivo.
- Para alterar as permissões sobre um arquivo, utilize o comando **chmod**. O comando **chmod** pode ser usado para especificar as permissões para o dono, para o grupo e para os demais usuários. Ao especificar as permissões utilize os números de 0 a 7.



# Permissões de arquivo

Valor inteiro	Permissões	Representação binária
7	Todas ( r w x )	111
6	Leitura e escrita ( r w - )	110
5	Leitura e execução ( r - x )	101
4	Somente leitura ( r - - )	100
3	Escrita e execução ( - w x )	011
2	Somente escrita ( - w - )	010
1	Somente execução ( - - x )	001
0	Nenhuma ( - - - )	000



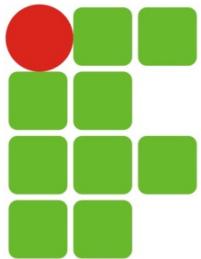


# Permissões de arquivo

- Ao fornecer novas permissões aos arquivos, utilize um dígito para o dono, um para o grupo e um para os demais usuários. Por exemplo, para conceder todas as permissões para o dono, porém nenhuma permissão para leitura, escrita ou execução de um arquivo ao grupo e aos demais, utilize **chmod 700** desta maneira:

```
aluno@IFFAR:~$ chmod 700 texto
aluno@IFFAR:~$ ls -l texto
-rwx----- 1 aluno aluno 40 set 13 15:35 texto
aluno@IFFAR:~$ █
```

- Agora podemos ver que somente o usuário *aluno* tem permissões para leitura, escrita e execução (rwx). Se qualquer outro usuário tentar acessar, teremos um erro de permissão não concedida.

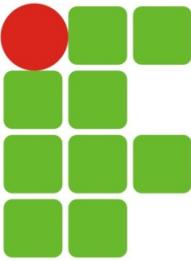


# Editando arquivos

- Através do terminal Linux, podemos editar arquivos utilizando um dos seus editores de texto. Dois dos mais conhecidos são o **vi** e o **nano**.
- Há muita discussão na comunidade linux sobre qual editor é o melhor, assim veremos um pouco sobre o **nano**, por ser meu preferido.

aluno@IFFAR:~\$ nano arquivoTexto.txt



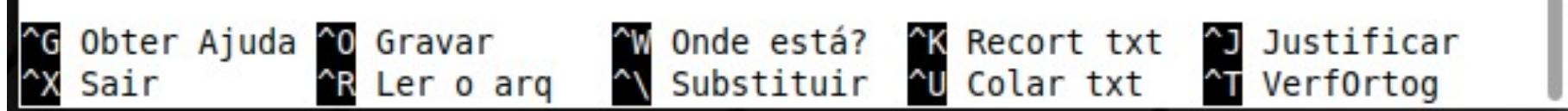


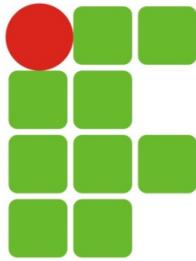
# Editando arquivos

- Podemos criar novos arquivos à partir do nano ou editar um existente. Vamos criar um novo.

```
aluno@IFFAR:~$ nano arquivoTexto.txt
```

- Depois que estiver no nano, você deverá ver um arquivo em branco com informações de ajuda para o nano, mostradas na parte inferior da tela, conforme apresentado aqui:





# Manipulação de dados

---

- Vamos realizar alguns testes com manipulação de dados. Insira o texto abaixo em “**aniversariantes.txt**” usando o editor de texto nano.

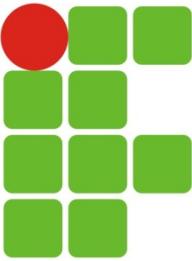
10 Pablo Fevereiro

2 Tyrone Maio

23 Uniqua Abril

14 Tasha Fevereiro

5 Austin Outubro

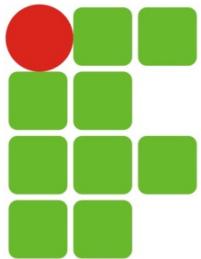


# Manipulação de dados - grep

- O comando grep procura instâncias de uma string de texto em um arquivo. Por exemplo, para pesquisar todas as instâncias da string Fevereiro em nosso arquivo, digite:

```
aluno@IFFAR:~$ grep Fevereiro aniversariantes.txt
10 Pablo Fevereiro
14 Tasha Fevereiro
```

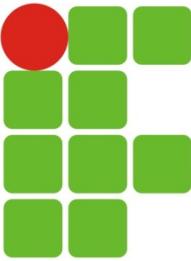
- Como podemos ver, o comando grep nos informa que Pablo e Tasha fazem aniversário em Fevereiro.
- Agora suponha que você queira somente o nome dos aniversariantes, sem o dia e o mês.



# Manipulação de dados - grep

---

- A saída do comando grep pode ser enviada a outro comando por meio de um pipe | para que um processamento adicional seja feito.
- O comando **cut** permite tomar cada linha de entrada, selecionar um delimitador e exibir campos específicos.
- Por exemplo, para obter somente os nomes dos aniversariantes de fevereiro, podemos usar o **grep** para procurar a palavra Fevereiro, como feito anteriormente.
- Em seguida, faça o pipe da saída para **cut**, em que você especificará um espaço em branco como delimitador por meio da opção –d e dirá que você quer o segundo campo por meio da opção de campo ( -f ).

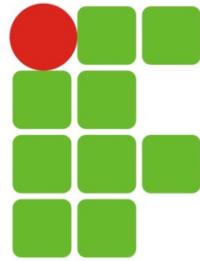


# Manipulação de dados - grep

- O comando e a saída ficariam assim:

```
aluno@IFFAR:~$ grep Fevereiro aniversariantes.txt | cut -d " " -f 2  
Pablo  
Tasha
```

- O resultado, como podemos ver, é que ao efetuar o pipe dos dois comandos, obtemos somente os aniversariantes Pablo e Tasha.



# Comandos de paginação

## more

- Exibe o conteúdo de um arquivo página por página.

`cat arquivo.txt | more`

(Para visualizar os comandos do more, pressione <H>. Para sair do more, pressione <Q>)

## less

- Faz o mesmo que o comando more e permite a navegação com Page UP, Page Down.

`cat arquivo.txt | less`

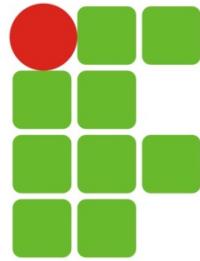
(Para visualizar os comandos do less, pressione <H>.)

Para avançar uma página, pressione <N>

Para retroceder uma página, pressione <P>

Para invocar o editor de texto (vi), pressione <V>

Para sair do less, pressione <Q>)



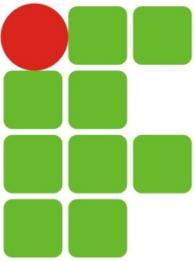
# Comandos de paginação

## head

- Este é um dos primos do cat, ele permite ver apenas as **primeiras** linhas do arquivo, ao invés de exibir a coisa inteira. Basta especificar o número de linhas que devem ser exibidas.

**head -2 texto.txt**

**head -15 arquivo.txt**



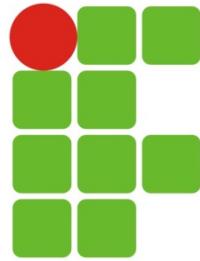
# Comandos de paginação

## tail

- Cauda. Mostra as ***últimas*** linhas do arquivo. O uso é o mesmo do comando head, basta indicar o número de linhas que devem ser mostradas e o nome do arquivo.

`tail -12 texto.txt`

Este comando é muito usado por administradores de sistemas para acompanhar os arquivos de logs de seus sistemas. Como as novas entradas destes arquivos vão sendo inseridas no final do arquivo, o tail permite verificar rapidamente apenas as novas inclusões, sem precisar perder tempo abrindo o arquivo inteiro.



# Iniciar e terminar uma sessão

- Se estivermos no meio de uma sessão e quisermos iniciar ou terminar outra sessão, utilizaremos os seguintes comandos:

## **login**

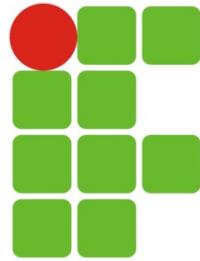
- Cancela a sessão atual e inicia uma nova sessão de usuário.

## **logout** ou **CTRL+D**

- Termina a sessão do usuário.

## **exit**

- Encerra o shell de comandos atual.



# Reiniciar ou desligar o PC

- Os comandos reboot e halt só podem ser executados pelo superusuário root.

## reboot

- Reinicia o computador

## halt ou shutdown (requer root)

- Desliga o computador.