Uma Brevíssima Introdução ao Linux

MCTA026-13 - Sistemas Operacionais

Emilio Francesquini e Fernando Teubl Ferreira
e.francesquini@ufabc.edu.br / fernando.teubl@ufabc.edu.br

2019.Q1

Centro de Matemática, Computação e Cognição
Universidade Federal do ABC

Disclaimer

UFABC

- Estes slides foram preparados para o curso de **Sistemas Operacionais na UFABC**.
- Este material pode ser usado livremente desde que sejam mantidos, além deste aviso, os créditos aos autores e instituições.
- Contém materiais obtidos a partir de `http://www.ee.surrey.ac.uk/Teaching/Unix/`

Introdução

Do que se trata tudo isso?

UFABC

- Uma classe de sistemas operacionais se desenvolveu a partir do UNIX
  - Mais conhecidos são:
    - Solaris
    - GNU/Linux
    - MacOS X
- Estes sistemas compartilham diversas características comuns como, por exemplo, APIs de programação

UFABC

- Sistemas UNIX se dividem em 3 partes principais
  - Kernel/Núcleo
  - Shell
  - Programas

O núcleo

UFABC

- O núcleo é o centro nervoso de um SO
  - Aloca e faz a arbitragem de tempo e recursos do hardware aos processos
  - Age como uma camada de abstração para o hardware
  - Provê serviços de alto nível às aplicações (sistemas de arquivos, comunicação interprocessos, ...)

O shell

UFABC

- O shell age como uma interface entre o usuário e o núcleo do SO.
- Após o login, o Linux executa um shell pré-configurado para o usuário
- O shell é um interpretador de comandos que pode ser:
  - Modo texto: Bash, Tcsh, Zsh, ...
    - Equivalente ao `command.com` no Windows
  - Modo gráfico: Gnome, KDE, Cinnamon, Mate, XMonad, ...

Arquivos e processos

UFABC

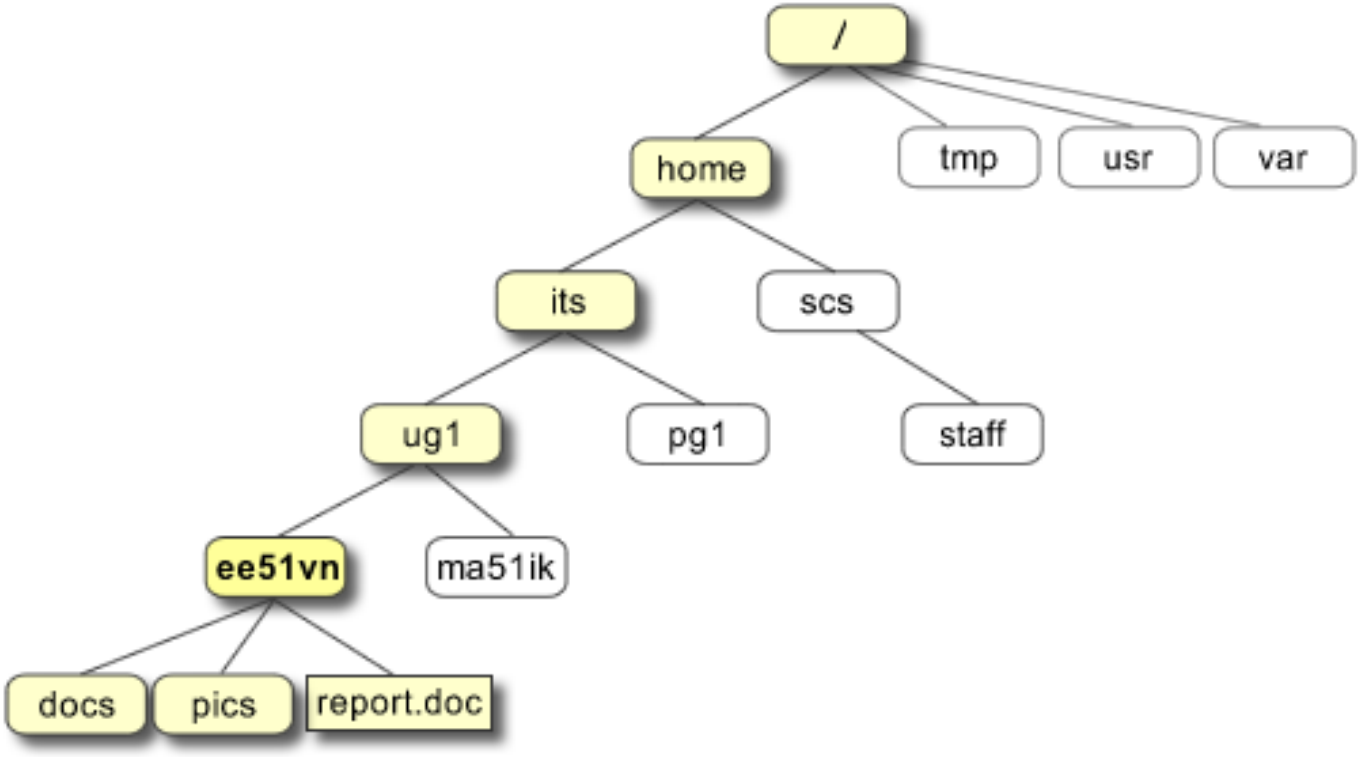
- Um processo é um programa em execução. Ele é identificado por um número `PID`
- Um arquivo é uma coleção de dados. Pode-se criar arquivos usando editores de texto, compiladores, ...
  - Exemplos:
    - Um documento de texto
    - O código de um programa
    - Um programa compilado
    - Um **diretório**

Diretórios

UFABC

- Diretórios também são arquivos no Linux
- Contudo são arquivos que recebem um tratamento especial
- Arquivos (inclusive diretórios) são agrupados em diretórios
- Todos os arquivos pertencem/estão contidos em um diretório com exceção do diretório raiz (*root*)
  - No Linux o diretório raiz é representado por /

UFABC



O caminho completo para o arquivo `report.doc` é `/home/its/ug1/ee51vn/report.doc`

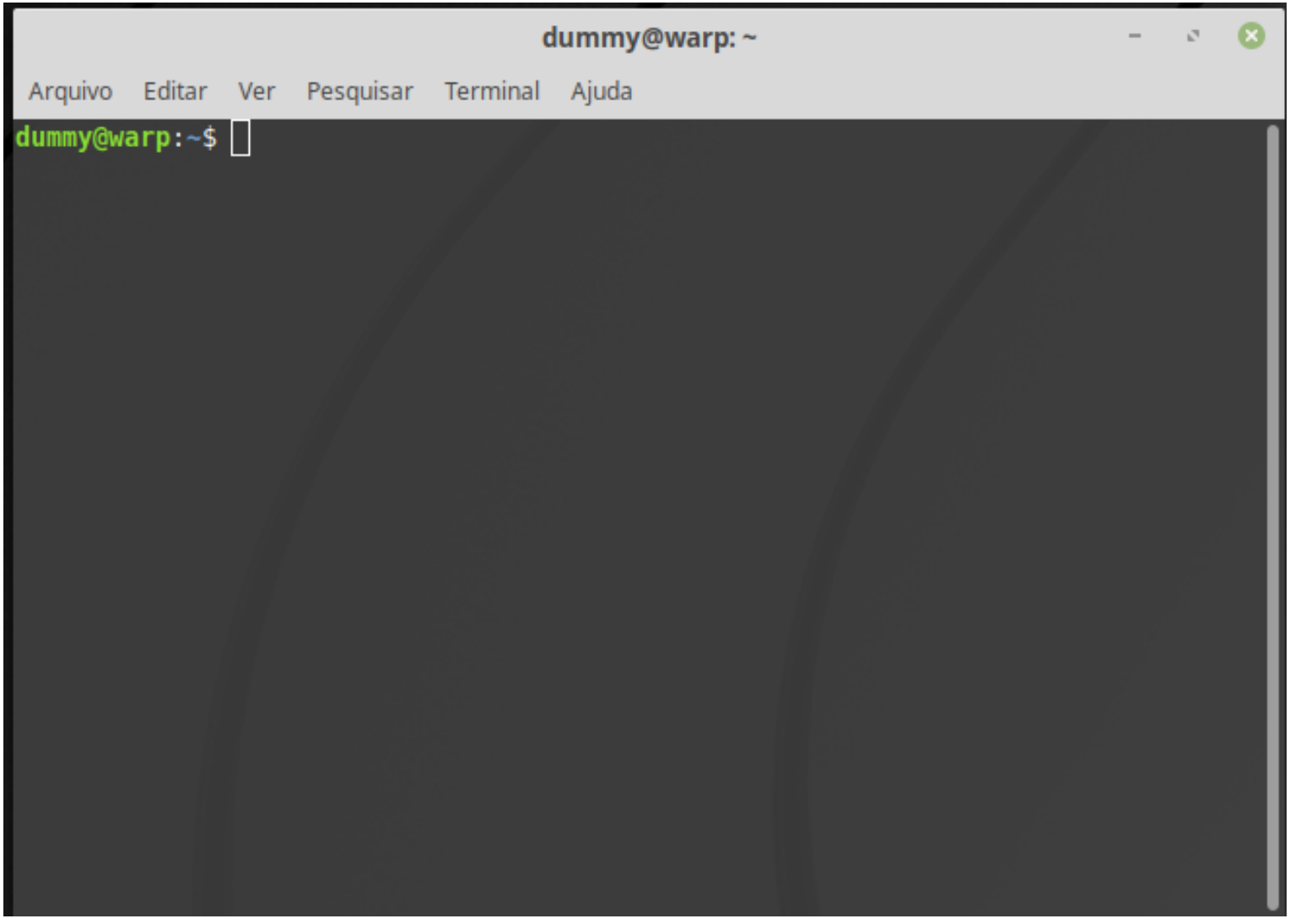
Mão na massa

UFABC

- Vamos iniciar um terminal. Procure por: `terminal` em seu Shell gráfico
- Se tudo der certo você deve ver uma tela parecida com a tela abaixo

Analizando o terminal

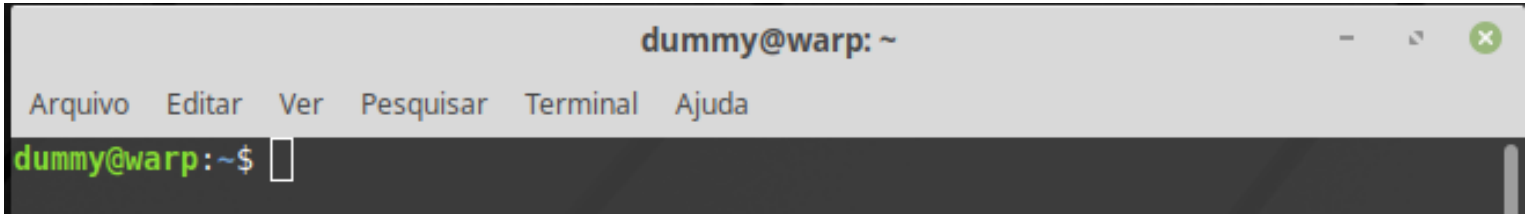
UFABC



- `dummy` é o nome do usuário e `warp` é o nome da máquina

Analizando o terminal

UFABC



- ~ Indica o diretório atual. ~ é o equivalente ao seu diretório padrão ou home directory
- O shell é um interpretador de comandos
  - Lê o comando digitado
  - Interpreta e executa
  - Espera pelo próximo comando

Navegação

O diretório home

UFABC

- Local de trabalho do usuário
- Tem direito para criar, modificar e apagar arquivos e diretórios
- Sua localização é, tipicamente:
  - `/root` para o usuário `root` (superusuário)
  - `/home/<nome_do_usuario>` para os usuários comuns

Determinando onde você está

UFABC

- O comando **pwd** imprime na tela o diretório de trabalho atual
- Quando você acaba de se logar, o seu diretório de trabalho é o seu home

```
dummy@warp:~$ pwd
/home/dummy
dummy@warp:~$
```

- Atenção: O shell diferencia minúsculas e maiúsculas

UFABC

- O comando **ls** lista diretórios e arquivos existentes
- Sintaxe: `ls [parâmetros]`

```
dummy@warp:~$ ls
Desktop   Documentos   Downloads   Imagens   Modelos   Música   Públic
dummy@warp:~$
```

`ls` - Parâmetros

UFABC

- `-l` Exibe em forma de lista longa (única lista com toda a descrição dos arquivos)
- `-d` (directory name) Exibe apenas o nome do diretório ao invés de seu conteúdo
- `-a` (all) Exibe todos os arquivos, inclusive ocultos
  - Arquivos ocultos – iniciam com (`.`)

ls - Exemplo

UFABC

```
emilio@warp:~$ ls -l
total 29644
-rwxrwxr-x  1 emilio emilio     8512 Oct 14 21:36  a.out
-rw-rw-r--  1 emilio emilio       16 Dec  9 09:17  avaliacao_auto.txt
drwx------ 16 emilio emilio    53248 Jan 29 20:19  Downloads
-rw-rw-r--  1 emilio emilio 10169495 Apr  9  2018  ds5thedn.pdf
-rw-rw-r--  1 emilio emilio       10 Nov 26 17:33  lico
drwxr-xr-x  2 emilio emilio     4096 Dec 22  2017  Music
drwxrwxr-x  3 emilio emilio     4096 May  3  2018  NetBeansProjects
-rw-rw-r--  1 emilio emilio   115637 Dec  3 18:50  org-caldav-d54da94.
-rw-rw-r--  1 emilio emilio        0 Dec  3 18:42  org-caldav-inbox.or
drwxrwxr-x  2 emilio emilio     4096 Apr 18  2018  personal
drwxr-xr-x  4 emilio emilio    12288 Feb 12 23:19  Pictures
drwxr-xr-x  2 emilio emilio     4096 Dec 22  2017  Public
drwxrwxr-x  3 emilio emilio     4096 Apr  3  2018  R
-rw-rw-r--  1 emilio emilio      578 Nov 26 17:14  teste.c
-rw-rw-r--  1 emilio emilio     3096 Dec  3 18:44  teste.el
drwxrwxr-x  4 emilio emilio     4096 Sep 11 17:10  texmf
drwxr-xr-x  3 emilio emilio     4096 Jan 29 20:23  Videos
dummy@warp:~$
```

`ls` - Outras opções

UFABC

- `-F` (formatted)
  - Exibe um caractere especial demarcando o tipo de arquivo
    - `*` - executável
    - `/` - diretório
    - `@` - links
    - `|` - filas
    - `=` - sockets (conexões de redes)

`ls` - Outras opções

UFABC

- `--color` - Exibe em cores diferenciadas arquivos/programas/ diretórios/links
- `-g` - (group only) - Semelhante ao –l, porém não exibe o proprietário do arquivo
- `-o` - (owner only) - Semelhante ao –l, porém não exibe o grupo do qual o arquivo pertence
- `-h` - (Human readable) - Exibe o tamanho em formato amigável (ex.: 1K, 234M, 2G)

UFABC

- `-s` - Lista ordenando os arquivos pelo tamanho
- `-S` (size) - Imprime o tamanho do arquivo em blocos (geralmente múltiplos de 1024 bytes)
- `-r` (reverse) - Lista em ordem inversa à exibida (depende do critério)
- `-R` (recursive) - Lista recursivamente os subdiretórios e seus respectivos conteúdos
- `-x` - Lista em colunas, ao invés de linhas
- `-X` - Lista em ordem alfabética
- `-m` - Exibe os arquivos separados por vírgulas

O comando man

UFABC

- Não é preciso decorar todas as opções. Na dúvida utilize o comando **man**
- O comando man recebe como parâmetro o nome do programa sobre o qual deseja ver as páginas de manual

```
dummy@warp:~$ man ls
```

```
LS(1)                                   User Commands                                   LS(1)

NAME
       ls - list directory contents

SYNOPSIS
       ls [OPTION]... [FILE]...

DESCRIPTION
       List  information  about  the  FILEs  (the current directory by default).  Sort entries alphabetically if none of -cftuvSUX nor --sort is
       specified.

       Mandatory arguments to long options are mandatory for short options too.

       -a, --all
              do not ignore entries starting with .

       -A, --almost-all
              do not list implied . and ..

       --author
              with -l, print the author of each file

       -b, --escape
              print C-style escapes for nongraphic characters

       --block-size=SIZE
              scale sizes by SIZE before printing them; e.g., '--block-size=M' prints sizes in units of 1,048,576 bytes; see SIZE format below

       -B, --ignore-backups
              do not list implied entries ending with ~

       -c     with -lt: sort by, and show, ctime (time of last modification of file status information); with -l: show ctime and sort  by  name;
              otherwise: sort by ctime, newest first
```

Criando diretórios - `mkdir`

UFABC

- O comando **`mkdir`** cria um diretório

```
dummy@warp:~$ mkdir teste
```

Verifique a criação com o comando **`ls`**

Mudando o diretório de trabalho - cd

UFABC

```
dummy@warp:~$ cd teste
dummy@warp:~/teste$
```

- Note que o bash muda a linha de prompt para refletir a mudança de diretório
- Para retornar ao diretório *home* basta executar o comando **cd** sem nenhum parâmetro
- Há dois diretórios especiais
  - `.` Indica o diretório atual. Ou seja, `cd .` é inócuo.
  - `..` Indica o diretório superior ao atual. `cd ..` retorna ao diretório superior.

Onde estou? - pwd

UFABC

- O comando **pwd** imprime na tela o diretório de trabalho atual.

```
dummy@warp:~/teste$ pwd
/home/dummy/teste
dummy@warp:~/teste$
```

Exercício 1

Use os comandos **cd**, **ls**, e **pwd** para explorar o sistema de arquivos.

Exercício 2

Crie um outro diretório chamado `interior` dentro do diretório `teste` e mude o diretório de trabalho para lá. Observe a mudança no prompt do seu shell.

Exercício 3

Navegue pela estrutura de diretórios que você criou entrando e saindo dos diretórios usando o comando `cd`

Redirecionamentos

Redirecionando a saída

UFABC

- A maior parte dos comandos no Linux escreve na saída padrão
- Contudo é comum querermos que a saída seja escrita em um arquivo, por exemplo.
- No Linux é possível redirecionar a saída padrão, a saída de erro e a entrada padrão de um comando

O comando `cat`

UFABC

- Execute o comando **`cat`** sem nenhum parâmetro
- Após inserir uma quebra de linha o que ocorre?
- Saia com **`Ctrl + D`** (EOF)

Redirecionamento da saída

UFABC

- O comando `cat` lê da entrada padrão e quando acaba uma linha ele imprime o texto que recebeu na saída padrão
- Vamos usar o `cat` redirecionando tanto a sua entrada quanto a sua saída
- `>` - usado para redirecionar a saída padrão para um arquivo

```
dummy@warp:~$ cat > arquivo1.txt
abacate
banana
coco
damasco
^D
dummy@warp:~$
```

Redirecionamento de entrada

UFABC

- Podemos também redirecionar a entrada do comando `cat`
- `<` Redireciona a entrada de um comando a partir do arquivo passado como parâmetro

```
dummy@warp:~$ cat < arquivo1.txt
abacate
banana
coco
damasco
dummy@warp:~$
```

- O comando `cat` também aceita receber o nome do arquivo diretamente (`cat arquivo1.txt`)
- Ele também aceita receber 2 (ou mais) nomes de arquivos. Um é impresso na tela após o outro.

Concatenando em um arquivo

UFABC

- O > sobrescreve todo o conteúdo de um arquivo.
- Caso desejemos adicionar ao final do arquivo podemos utilizar >>

```
dummy@warp:~$ cat >> arquivo1.txt
esfregadinha
figo
goiaba
dummy@warp:~$ cat <arquivo1.txt
abacate
banana
coco
damasco
esfregadinha
figo
goiaba
dummy@warp:~
```

Pipes - |

UFABC

- Podemos conectar a saída de um programa com a entrada do próximo utilizando pipes |
- O comando `shuf` gera uma sequência de números aleatórios
  - `-i` indica a faixa de números desejados
  - `-n` indica a quantidade desejada

```
dummy@warp:~$ shuf -i 0-9 -n 5
6
2
0
1
8
dummy@warp:~$
```

Pipes - |

UFABC

- O comando `sort` ordena as linhas recebidas pela entrada padrão
- Podemos encadear a saída do `shuf` com a entrada do `sort`

```
dummy@warp:~ shuf -i 0-9 -n 5 | sort
0
3
5
6
9
dummy@warp:~$
```

UFABC

Exercício 4

Use o programa `cat` e o redirecionamento de arquivos para criar um arquivo com uma lista de pelo menos 5 nomes de carros e outro arquivo com o nome de pelo menos 5 motos.

Exercício 5

Utilizando redirecionamento de entradas e saídas em uma única linha de comando imprima a lista de todos os carros e todos as motos contidos nos dois arquivos criados no exercício anterior. Sua solução deve imprimir a lista de nomes em ordem alfabética.

Manipulando arquivos

Copiando arquivos - cp

UFABC

- O comando **cp** cria uma cópia de um arquivo
  - Primeiro parâmetro é a origem e o segundo é o destino

```
dummy@warp:~$ cp teste.txt teste_copia.txt
dummy@warp:~$
```

- Você pode usar `.`, `..`, e `~` como parte do caminho

Movendo/Renomeando arquivos - mv

UFABC

- O comando **mv** move (ou renomeia) um arquivo
  - Primeiro parâmetro é a origem e o segundo é o destino

```
dummy@warp:~$ mv teste.txt novo_nome.txt
dummy@warp:~$
```

- Você pode usar `.`, `..`, e `~` como parte do caminho

Apagando arquivos - `rm`

UFABC

- O comando **`rm`** apaga um arquivo
  - Para apagar um diretório é preciso indicar que deve ter um coportamento recursivo com a opção `-r`
    - Veja a *man page* para mais detalhes
  - Recebe como parâmetro o arquivo a ser apagado
  - Atenção: a ação de apagar é irreversível (não há uma lixeira!)

```
dummy@warp:~$ rm novo_nome.txt
dummy@warp:~$
```

- Você pode usar `.`, `..`, e `~` como parte do caminho

Limpando a tela - `clear`

UFABC

- Caso queira limpar a tela, utilize o comando **`clear`**
- Tipicamente o comando clear está associado às teclas `Ctrl + L`

Mostrando o conteúdo de um arquivo - `cat`

UFABC

- O comando **`cat`** imprime na tela o conteúdo de um arquivo

```
dummy@warp:~$ cat teste.txt
Conteúdo do arquivo teste. Linha 1.
Conteúdo do arquivo teste. Linha 2.
...
dummy@warp:~$
```

Vendo o conteúdo de um arquivo - `less`

UFABC

- Algumas vezes o conteúdo do arquivo pode ser grande demais para ser visualizado confortavelmente com o comando `cat`
- Nesses casos o comando `less` pode ser mais útil
- A navegação é parecida com aquela utilizada pelo comando `man`

```
dummy@warp:~$ less teste.txt
dummy@warp:~$
```

Vendo o início de um arquivo - head

UFABC

- Outras vezes, contudo, queremos ver apenas as linhas iniciais de um arquivo
- nestes casos o comando **head** pode ser útil
- É possível especificar quantas linhas queremos. `-n`

```
dummy@warp:~$ head -5 teste.txt
Conteúdo do arquivo teste. Linha 1.
Conteúdo do arquivo teste. Linha 2.
Conteúdo do arquivo teste. Linha 3.
Conteúdo do arquivo teste. Linha 4.
Conteúdo do arquivo teste. Linha 5.
dummy@warp:~$
```

Vendo o fim de um arquivo - `tail`

UFABC

- Podemos também querer ver apenas as linhas finais de um arquivo
- Para isto utilizamos o comando **`tail`**
- É possível especificar quantas linhas queremos. `-n`

```
dummy@warp:~$ tail -5 teste.txt
Conteúdo do arquivo teste. Linha 46.
Conteúdo do arquivo teste. Linha 47.
Conteúdo do arquivo teste. Linha 48.
Conteúdo do arquivo teste. Linha 49.
Conteúdo do arquivo teste. Linha 50.
dummy@warp:~$
```

Procurando texto dentro de um arquivo - `grep`

UFABC

- O `grep` é um dos comandos padrão em sistemas UNIX. Ele procura por um determinado texto dentro dos arquivos especificados pela linha de comando.
- O comando `grep "abacate"*` vai procurar pelo texto `abacate` dentro de todos os arquivos do diretório de trabalho atual.
- Atenção: o comando `grep` diferencia maiúsculas de minúsculas, utilize a opção `-i` caso não deseje este comportamento
- Alguns outros parâmetros de interesse
  - `-v` - Mostra as linhas que **não** casam com o padrão dado
  - `-n` - Imprime cada linha encontrada precedida do seu número
  - `-c` - Imprime apenas o número de ocorrências encontradas

Contando caracteres, palavras e linhas - wc

UFABC

- Para contar o número de caracteres, palavras ou linhas **`wc nome_do_arquivo`**
- `-w` conta o número de palavras
- `-l` conta o número de linhas
- `-c` conta o número de bytes (cuidado!)
- `-m` conta o número de caracteres

```
dummy@warp:~$ wc teste.txt
  50  300 1891 teste.txt
dummy@warp:~$ wc -c teste.txt
1891 teste.txt
dummy@warp:~$ wc -m teste.txt
1841 teste.txt
dummy @warp:~$
```

UFABC

Exercício 6

Crie um backup do seu arquivo contendo os nomes de motos e carros para um arquivo `carros.bak` e `motos.bak`

Exercício 7

Crie um diretório `teste_externo` e um diretório `teste_interno` dentro do diretório externo.

Exercício 8

Crie um arquivo `dentro.txt` com qualquer conteúdo dentro do diretório `teste_interno`.

UFABC

Exercício 9

Copie o arquivo `motos.bak` para dentro do diretório `teste_interno` e o arquivo `carros.bak` para dentro do diretório `teste_externo`.

Exercício 10

Renomeie o diretorio `teste_interno` para `Cochabamba`

Exercício 11

Mova o diretório `Cochabamba` para o seu *home* e apague o diretório `teste_externo`.

UFABC

Exercício 12

Conte quantos nomes de motos estão contidos no arquivo `~/Cochabamba/motos.bak` que tenham a letra `a` (maiúscula ou minúscula).

Exercício 13

Liste todas as linhas contidas no arquivo `~/Cochabamba/motos.bak` que contenham a letra `a` (maiúscula ou minúscula) em ordem lexicográfica.