Marisangila Alves, MSc
marisangila.alves@udesc.br
marisangila.com.br

UDESC
UNIVERSIDADE DO ESTADO DE SANTA CATARINA
JOINVILLE
CENTRO DE CIÊNCIAS TECNOLÓGICAS

UDESC
Universidade do Estado de Santa Catarina

2025/1

Sistemas Operacionais

Shell

Sumário

Shell

Shell é um programa que interpreta comandos digitados no terminal de sistemas operacionais derivados do UNIX. Ele funciona como uma interface entre o usuário e o sistema operacional. É um termo genérico. Existem vários tipos de shells:

- `bash` — Bourne Again Shell (padrão no Linux)
- `zsh` — Z Shell (usado no macOS moderno)
- `sh` — Shell original (Unix)
- `fish` — Friendly Interactive Shell
- `PowerShell` — Shell moderno da Microsoft

O Bash é um interpretador de comandos que pode ser executado em diversos terminais, dependendo do sistema operacional.

Exemplos:

- **Linux**: GNOME Terminal, Konsole, XTerm, Tilix, Alacritty
- **macOS**: Terminal.app, iTerm2
- **Windows**: Git Bash, Windows Terminal (com WSL), Cygwin, MobaXterm

`Dica:` Use `echo $SHELL` para verificar o shell atual.

Listar conteúdo do diretório: `ls`

- `ls` — lista o conteúdo do diretório atual.
- `ls nome_pasta/` — lista o conteúdo de uma pasta específica.
- `ls -a` — exibe arquivos ocultos.
- `ls -l` — exibe em formato de lista.
- `ls -r` — exibe em ordem reversa.

Código 1: Listar arquivos

```
1 ls
2 ls pasta/
3 ls -alr
```

- `pwd` — mostra o caminho completo (absoluto) do diretório atual.
- **Caminho relativo**: inicia a partir do diretório atual, sem a barra no início.
- **Caminho absoluto**: inicia com barra (/), representando o caminho completo.

Código 2: Exibir caminho atual

```
1 pwd
```

- `cd pasta1` — entra na pasta especificada.
- `cd ..` — volta ao diretório anterior.
- `cd /` — vai para o diretório raiz.
- `cd ~` — vai para o diretório pessoal do usuário.
- `cd pasta1/pasta2` — navega por múltiplos diretórios.

Código 3: Exemplos com `cd`

```
cd ..
cd ~/Downloads
cd /etc
cd pasta1/pasta2
```

Manipulação de Arquivos

- `cp arquivo.txt pasta/` — copia o arquivo para a pasta.
- `cp arquivo1.txt arquivo2.txt` — cria uma cópia com outro nome.
- `cp -r pasta1/ pasta2/` — cópia recursiva (diretórios).
- `cp -v` — exibe detalhes da cópia.

Código 4: Exemplo com cp

```
1 cp arquivo.txt pasta/
2 cp -rv pasta1/ pasta2/
```

Mover ou renomear arquivos: `mv`

- `mv arquivo.txt pasta1/` — move arquivo.
- `mv pasta1 pasta2` — move ou renomeia pastas.
- `mv arquivo.txt novo_nome.txt` — renomeia arquivo.

Código 5: Exemplo com `mv`

```
1 mv arquivo.txt pasta1/
2 mv arquivo.txt novo_nome.txt
```

- `mkdir pasta` — cria uma nova pasta.
- `mkdir -p pasta1/pasta2` — cria estrutura de diretórios.

Código 6: Exemplo com `mkdir`

```
1 mkdir projetos
2 mkdir -p curso/scripts/bash
```

Remover arquivos e pastas: `rm`

- `rm arquivo.txt` — remove um arquivo.
- `rm -r pasta1` — remove uma pasta e seu conteúdo.
- `rm -f` — força a remoção.
- `rm -v` — exibe informações.
- `rm -i` — pede confirmação.

Código 7: Exemplo com `rm`

```
1 rm arquivo.txt
2 rm -rf pasta1/
```

Criar arquivos vazios: `touch`

- `touch nome.txt` — cria um novo arquivo vazio.
- Também pode ser usado para atualizar a data de modificação de um arquivo.

Código 8: Exemplo com `touch`

```
1 touch novo_arquivo.txt
```

Concatenar e exibir conteúdo: `cat`

- `cat arquivo.txt` — exibe o conteúdo do arquivo.
- `cat arquivo1 arquivo2` — concatena arquivos.

Código 9: Exemplo com `cat`

```
1 cat texto.txt
2 cat cabecalho.txt corpo.txt rodape.txt > completo.txt
```

- `echo "mensagem"` — exibe texto no terminal.
- Pode ser usado com variáveis e redirecionamentos.

Código 10: Exemplo com `echo`

```
1 echo "Olá mundo!"
2 echo "Usuário: $USER"
```

No shell, é possível redirecionar a entrada e a saída de comandos para arquivos ou entre programas.

- `>` — redireciona saída padrão (sobrescreve)
- `»` — redireciona saída padrão (acrescenta ao final)
- `<` — redireciona arquivo como entrada
- `«` — entrada de múltiplas linhas (aqui-documento)
- `2>` — redireciona saída de erro
- `&>` — redireciona saída padrão e de erro
- `|` — envia a saída de um comando para outro (pipe)

Código 11: Redirecionar saídas e entradas

```
echo "Olá mundo!" > mensagem.txt      # Cria ou sobrescreve
echo "Mais uma linha" >> mensagem.txt  # Acrescenta

sort < nomes.txt                       # Entrada por arquivo
cat << EOF
Texto de várias
linhas aqui
EOF

ls pasta_inexistente 2> erro.txt      # Somente erros
ls pasta 1> out.txt 2> err.txt         # Saída e erro separados
comando &> tudo.txt                    # Saída + erro
```

Identificando usuários: `whoami` e `who`

Os comandos `whoami` e `who` são utilizados para obter informações sobre os usuários no sistema.

`whoami` — Mostra o nome do usuário atual:

```
1 whoami
```

- Exibe o nome do usuário que executou o comando.
- Útil em scripts e sessões remotas.

`who` — Lista todos os usuários logados:

```
1 who
```

- Mostra quem está logado no sistema.
- Inclui nome do usuário, terminal e horário de login.

O comando sudo (*SuperUser DO*) permite que um usuário autorizado execute comandos com privilégios de outro usuário, geralmente o **root** (superusuário).

Por que usar o sudo?

- Executar tarefas administrativas sem fazer login como root.
- Aumenta a segurança: o usuário precisa estar autorizado.
- Permite registrar (logar) o que foi feito com permissões elevadas.

Código 12: Exemplo de uso do sudo

```
1 sudo apt update
2 sudo shutdown now
```

Será solicitada a senha do usuário atual (não a do root).

Em um sistema operacional Linux, usuários e grupos são fundamentais para o controle de acesso entre o **SO** e o **hardware**.

Criar um usuário:

```
1 adduser "usuario"
```

Criar um grupo:

```
1 addgroup "grupo"
```

Adicionar usuário ao grupo:

```
1 addgroup "usuario" "grupo"
```

Alterar senha:

- Do usuário atual:

```
1 passwd
```

- De outro usuário (somente root):

```
1 passwd "usuario"
```

Visualizar grupos e permissões:

```
1 cat /etc/group
```

É possível alternar entre contas de usuário no terminal usando o comando `su` (*substitute user*).

Mudar para outro usuário:

```
su nome_do_usuario
```

Solicita a senha do usuário alvo e mantém o ambiente do usuário atual.

Mudar para outro usuário com ambiente completo:

```
su - nome_do_usuario
```

Inicia uma sessão como se o usuário tivesse feito login (carrega variáveis, PATH, etc).

Exemplo - mudar para root:

```
su -
```

Inicia uma sessão completa como superusuário (root), se autorizado.

Permissões de Arquivos

- `r` — leitura
- `w` — escrita
- `x` — execução (ou acesso em diretórios)

As permissões são definidas para:

- **Usuário (u)** — dono do arquivo
- **Grupo (g)** — grupo ao qual o dono pertence
- **Outros (o)** — todos os demais usuários

Código 13: Saída do comando `ls -l`

```
-rwxr-xr-- 1 usuario grupo 1234 abr 9 10:00 exemplo.sh
```

- `-` indica tipo (arquivo, diretório, link)
- `rwx` — permissões do usuário
- `r-x` — permissões do grupo
- `r-` — permissões de outros

chmod pode usar dois modos:

Modo simbólico:

- + adiciona permissão
- − remove permissão
- = define permissão exata

Código 14: Exemplo simbólico

```
chmod u+rw arquivo.txt
chmod g-x script.sh
chmod o= arquivo.txt
```

Modificando permissões: modo octal

`chmod` também aceita valores numéricos (octal):

- `r = 4`, `w = 2`, `x = 1`
- `chmod 754 arquivo.txt`
- $\rightarrow$ `7 = rwx (usuário)`, `5 = r-x (grupo)`, `4 = r- (outros)`

Tabela de Permissões (modo octal)

Abaixo estão os valores utilizados no comando `chmod` em formato octal, com suas representações binárias e simbólicas.

Octal	Binário	Caractere	Descrição
0	000	`--`	Sem permissões
1	001	`-x`	Somente execução
2	010	`-w-`	Somente escrita
3	011	`-wx`	Escrita e execução
4	100	`r-`	Somente leitura
5	101	`r-x`	Leitura e execução
6	110	`rw-`	Leitura e escrita
7	111	`rwx`	Leitura, escrita e execução

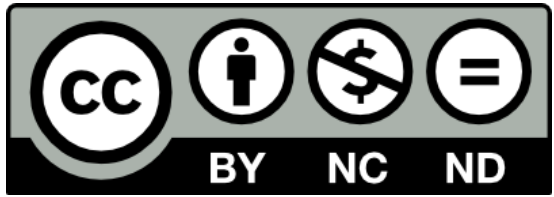
Permissões por tipo de acesso (usuário, grupo, outros)

Código 15: Trocar dono de arquivo

```
1 chown usuario arquivo.txt           # Troca o dono
2 chown usuario:grupo arquivo.txt     # Troca dono e grupo
```

FREE SOFTWARE FOUNDATION. **Bash Reference Manual**. [*S. l.*], 2022. Acesso em: 9 abr. 2025. Disponível em: `https://www.gnu.org/software/bash/manual/bash.html`.

Estes slides estão protegidos por uma licença Creative Commons



Este modelo foi adaptado de Maxime Chupin.

Marisangila Alves, MSc
marisangila.alves@udesc.br
marisangila.com.br

UDESC
UNIVERSIDADE DO ESTADO DE SANTA CATARINA
JOINVILLE
CENTRO DE CIÊNCIAS TECNOLÓGICAS

UDESC
Universidade do Estado de Santa Catarina

2025/1

Sistemas Operacionais

Shell