

Sistemas Embarcados

Shell script

# Introdução ao desenvolvimento com Linux

- Conteúdo:
  - Metodologia/Filosofia de desenvolvimento com o Linux/UNIX;
  - Licenças de software livre (GPL e LGPL) ;
  - Comandos básicos do Linux;
  - Organização da estrutura de diretórios;
  - Obtendo informações sobre o sistema;
  - Instalação de programas;
  - Ferramentas de desenvolvimento em linguagem C

# O que é um shell script?

- Arquivo para automação de chamadas a programas no terminal Linux;
- Oferece um ambiente de programação, com variáveis, laços, testes etc.;
- Não é compilado, e sim interpretado.

# Formato

Arquivo *Ex0.sh*

```
#!/bin/bash  
echo Ola shell script!  
# Comentário  
echo Adeus shell script!
```

# Formato

Arquivo *Ex0.sh*

```
#!/bin/bash  
echo Ola shell script!  
# Comentário  
echo Adeus shell script!
```

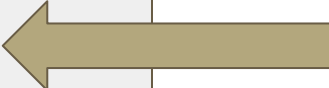
Indica que este arquivo deve ser executado pela programa *bash*, dentro da pasta */bin*

# Formato

Arquivo *Ex0.sh*

```
#!/bin/bash  
echo Ola shell script!  
# Comentário  
echo Adeus shell script!
```

Executa uma  
chamada ao  
comando *echo*




# Formato

Arquivo *Ex0.sh*

```
#!/bin/bash  
echo Ola shell script!  
# Comentário  
echo Adeus shell script!
```

Tudo depois do  
símbolo # é  
ignorado pelo  
terminal




# Formato

Arquivo *Ex0.sh*

```
#!/bin/bash  
echo Ola shell script!  
# Comentário  
echo Adeus shell script!
```

Executa outra  
chamada ao  
comando *echo*

A horizontal arrow points from the text box on the right to the line 'echo Adeus shell script!' in the script code block on the left.



# Formato

Arquivo *Ex0.sh*

```
#!/bin/bash  
echo Ola shell script!  
# Comentário  
echo Adeus shell script!
```

Para criar e executar este script, digite:

# Formato

## Arquivo *Ex0.sh*

```
#!/bin/bash  
echo Ola shell script!  
# Comentário  
echo Adeus shell script!
```

Para criar e executar este script, digite:

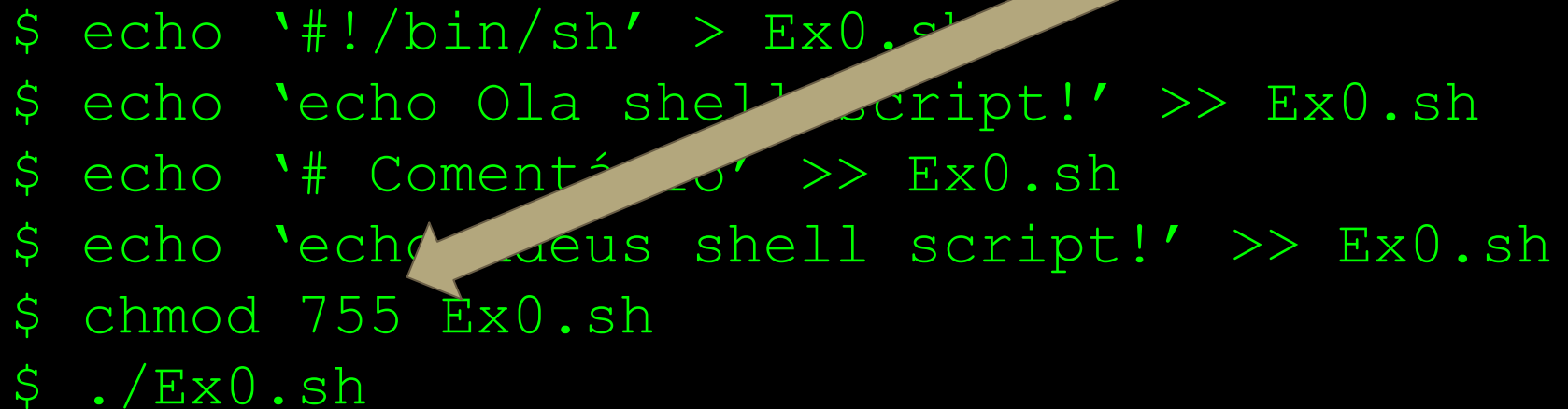
```
$ echo `#!/bin/sh` > Ex0.sh  
$ echo `echo Ola shell script!` >> Ex0.sh  
$ echo `# Comentário` >> Ex0.sh  
$ echo `echo Adeus shell script!` >> Ex0.sh  
$ chmod 755 Ex0.sh  
$ ./Ex0.sh
```

# Formato

## Arquivo *Ex0.sh*

```
#!/bin/bash  
echo Ola shell script!  
# Comentário  
echo Adeus shell script!
```

O comando *chmod 755* é fundamental para tornar o *script* executável (rwxr-xr-x)



```
$ echo `#!/bin/sh` > Ex0.sh  
$ echo `echo Ola shell script!` >> Ex0.sh  
$ echo `# Comentário` >> Ex0.sh  
$ echo `echo Adeus shell script!` >> Ex0.sh  
$ chmod 755 Ex0.sh  
$ ./Ex0.sh
```

# Mais exemplos

Arquivo *Ex1.sh*

```
#!/bin/bash  
echo "Seu nome de usuário é:"  
whoami  
echo "Hora atual e tempo ligado:"  
uptime  
echo "O script está executando do diretório:"  
pwd
```

# Mais exemplos

## Arquivo *Ex2.sh*

```
#!/bin/bash
#!/bin/sh
# This is a comment!
echo "1 Hello    World"    # This is a comment, too!
echo "2 Hello World"
echo "3 Hello * World"
echo 4 Hello * World
echo 5 Hello    World
echo "6 Hello" World
echo 7 Hello "    " World
echo "8 Hello "*" World"
echo 9 `hello` world
echo '10 hello' world
```

# Variáveis

Arquivo *Ex3.sh*

```
#!/bin/bash
nome=Fulano
echo Ola $nome
echo Vamos procurar arquivos com seu nome:
ls -l *$nome*.*
echo Vamos conferir o conteúdo da variável:
echo \$nome = $nome
```

# Variáveis

Arquivo *Ex3.sh*

```
#!/bin/bash
```

```
nome=Fulano
```

```
echo Ola $nome
```

```
echo Vamos procurar arquivos com seu nome:
```

```
ls -l *$nome*.*
```

```
echo Vamos conferir o conteúdo da variável:
```

```
echo \ $nome = $nome
```



Definição da  
variável *nome*

# Variáveis

Arquivo *Ex3.sh*

```
#!/bin/bash
nome=Fulano
echo Ola $nome
echo Vamos procurar ar
ls -l *$nome*.
echo Vamos conferir o conteúdo da variável:
echo \$nome = $nome
```

Não pode haver espaço antes e depois da igualdade:

- *nome=Fulano* é válido;
- *nome = Fulano* não é.



# Variáveis

Arquivo *Ex3.sh*

```
#!/bin/bash
```

```
nome=Fulano
```

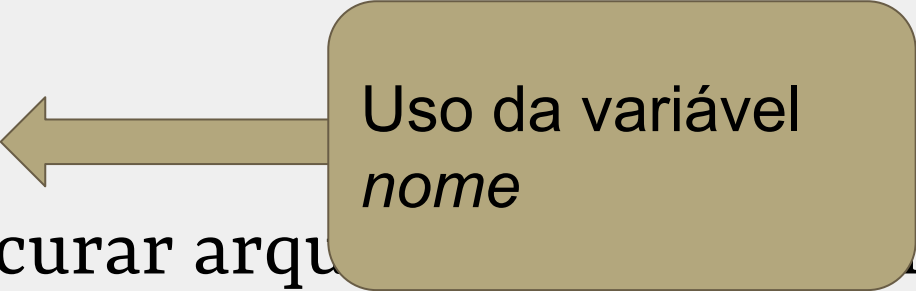
```
echo Ola $nome
```

```
echo Vamos procurar arquivos com o nome:
```

```
ls -l *$nome*.*
```

```
echo Vamos conferir o conteúdo da variável:
```

```
echo \ $nome = $nome
```

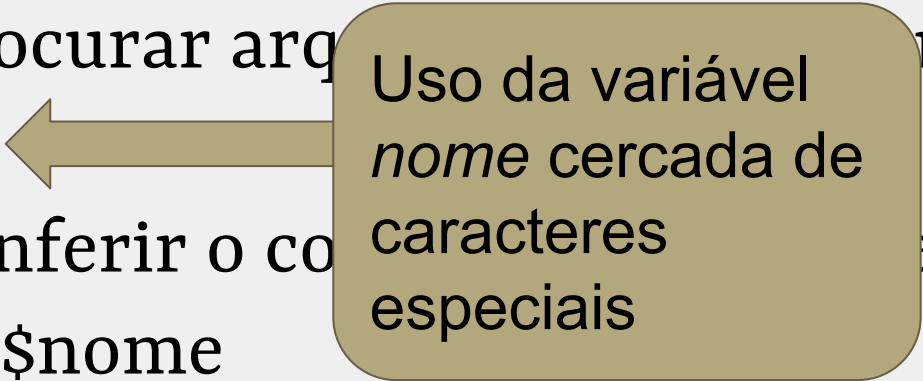


Uso da variável  
*nome*

# Variáveis

Arquivo *Ex3.sh*

```
#!/bin/bash
nome=Fulano
echo Ola $nome
echo Vamos procurar arquivos com o nome:
ls -l *$nome*. *
echo Vamos conferir o conteúdo:
echo \ $nome = $nome
```

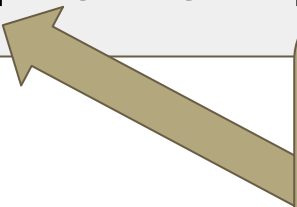


Uso da variável  
*nome* cercada de  
caracteres  
especiais

# Variáveis

Arquivo *Ex3.sh*

```
#!/bin/bash
nome=Fulano
echo Ola $nome
echo Vamos procurar arquivos com seu nome:
ls -l *$nome*.
echo Vamos conferir o conteúdo da variável:
echo \ $nome = $nome
```



O símbolo \\$ escreve o cifrão na tela, desconsiderando o nome da variável que vem depois

# Variáveis

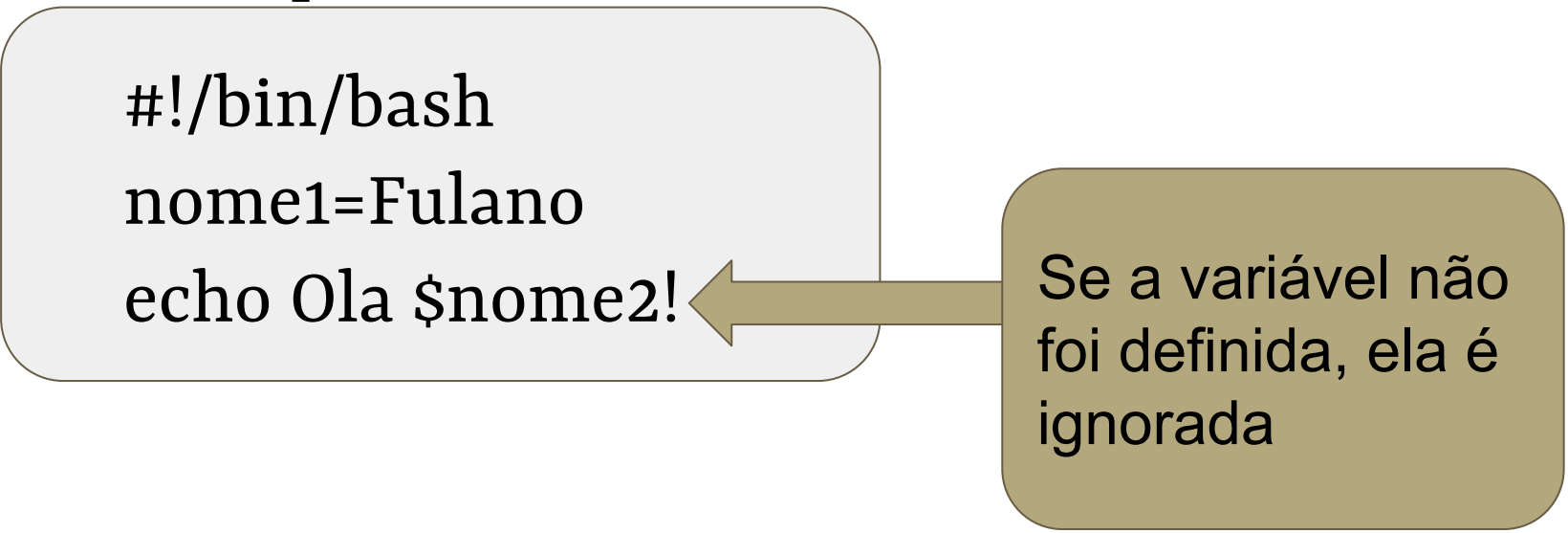
Arquivo *Ex4.sh*

```
#!/bin/bash  
nome1=Fulano  
echo Ola $nome2!
```

# Variáveis

Arquivo *Ex4.sh*

```
#!/bin/bash  
nome1=Fulano  
echo Ola $nome2!
```



Se a variável não  
foi definida, ela é  
ignorada

# Variáveis

Arquivo *Ex5.sh*

```
#!/bin/bash  
saida1=`ls -l`  
echo $saida1  
echo -----  
saida2=$(ls -l)  
echo $saida2
```

# Variáveis

Arquivo *Ex5.sh*

```
#!/bin/bash
```

```
saida1=`ls -l`
```

```
echo $saida1
```

```
echo -----
```

```
saida2=$(ls -l)
```

```
echo $saida2
```



A saída do comando *ls -l* é atribuída à variável *saida1*

# Variáveis

Arquivo *Ex5.sh*

```
#!/bin/bash
```

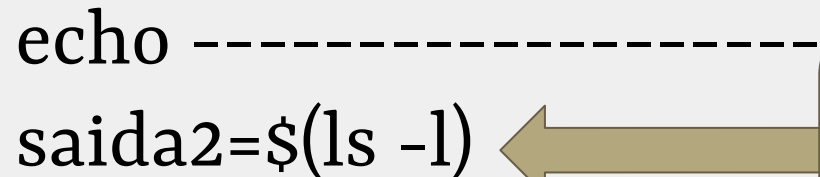
```
saida1=`ls -l`
```

```
echo $saida1
```

```
echo -----
```

```
saida2=$(ls -l)
```

```
echo $saida2
```



Este formato também é válido  
para atribuir a saída de um  
comando a uma variável



# Entrada do usuário


Arquivo *Ex6.sh*

```
#!/bin/bash  
echo 'Qual é o seu nome?'  
read nome;  
echo Olá $nome!
```

# Entrada do usuário

Arquivo *Ex6.sh*

```
#!/bin/bash  
echo 'Qual é o seu nome'  
read nome;  
echo Olá $nome!
```



Aquilo que for digitado pelo usuário será atribuído à variável *nome*

# If

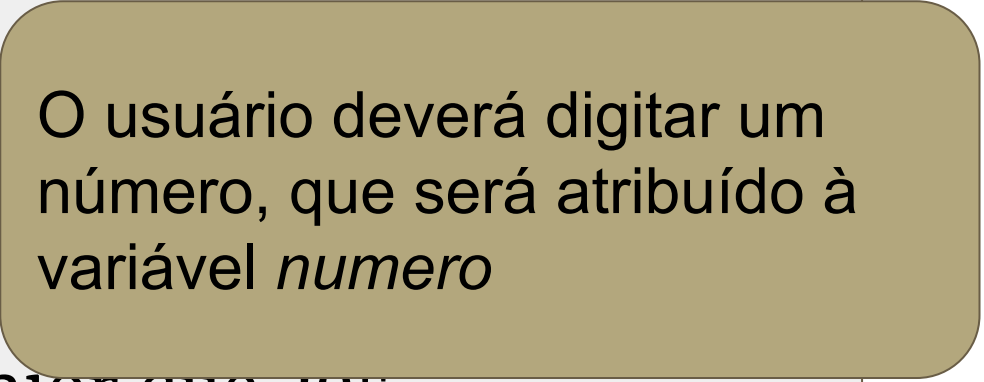
Arquivo *Ex7.sh*

```
#!/bin/bash
echo 'Digite um número qualquer:'
read numero
if [ $numero -gt 20 ]; then
    echo 'Este número é maior que 20!'
fi
```

# If

## Arquivo *Ex7.sh*

```
#!/bin/bash  
echo 'Digite um número'  
read numero  
if [ $numero -gt 20 ]; then  
    echo 'Este número é maior que 20!'  
fi
```




O usuário deverá digitar um número, que será atribuído à variável *numero*

# If

Arquivo *Ex7.sh*

```
#!/bin/bash  
echo 'Digite um número qualquer:'  
read numero  
if [ $numero -gt 20 ]; then  
    echo 'Este número é maior que 20!'  
fi
```



Se *numero*>20, escreva o texto abaixo.

# If

Arquivo *Ex7.sh*

```
#!/bin/bash  
echo 'Digite um número qualquer:'  
read numero  
if [ $numero -gt 20 ]; then  
    echo '  
fi
```

← Repare como o *if* foi finalizado.

# If

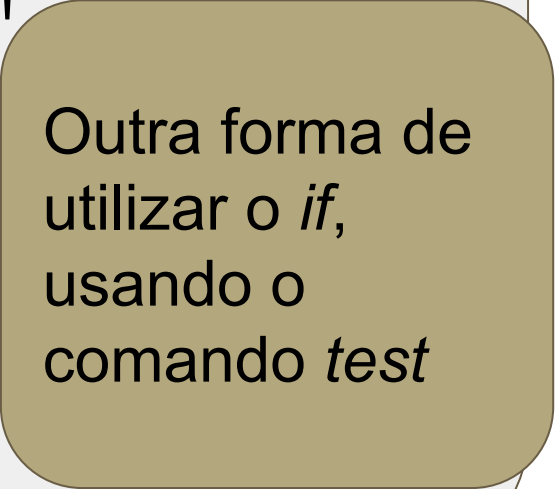
Arquivo *Ex8.sh*

```
#!/bin/bash  
echo 'Digite um número qualquer:'  
read numero  
if test $numero -gt 20; then  
    echo 'Este número é maior que 20!'  
fi
```

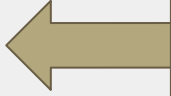
# If

Arquivo *Ex8.sh*

```
#!/bin/bash
echo 'Digite um número qualquer:'
read numero
if test $numero -gt 20; then
    echo 'Este número é maior que 20'
fi
```



Outra forma de  
utilizar o *if*,  
usando o  
comando *test*





# If

Arquivo *Ex8.sh*

```
#!/bin/bash
```

**Parâmetros mais comuns utilizados com o comando test:**

**n string1:** o comprimento de string1 é diferente de 0;

**z string1:** o comprimento de string1 é zero;

**string1 = string2:** string1 e string2 são idênticas;

**string1 != string2:** string1 e string2 são diferentes;

**int1 -eq int2:** int1 possui o mesmo valor que int2;

# If

Arquivo *Ex8.sh*

```
#!/bin/bash
```

```
echo 'Digite um número qualquer:'
```

**Parâmetros mais comuns utilizados com o comando test:**

**int1 -ne int2:** int1 não possui o mesmo valor que int2;

**int1 -gt int2:** int1 é maior que int2;

**int1 -ge int2:** int1 é maior ou igual a int2;

**int1 -lt int2:** int1 é menor que int2;

**int1 -le int2:** int1 é menor ou igual a int2;

# If

## Arquivo *Ex8.sh*

```
#!/bin/bash
```

**Parâmetros mais comuns utilizados com o comando test:**

**e nome\_do\_arquivo:** verifica se nome\_do\_arquivo existe;

**d nome\_do\_arquivo:** verifica se nome\_do\_arquivo é um diretório;

**f nome\_do\_arquivo:** verifica se nome\_do\_arquivo é um arquivo regular (texto, imagem, programa, docs, planilhas).

# If

## Arquivo *Ex9.sh*

```
#!/bin/bash
echo 'Digite um número qualquer:'
read numero
if [ $numero -gt 20 ]; then
    echo 'Este número é maior que 20!'
elif [ $numero -gt 0 ]; then
    echo 'Este número é maior que 0!'
else
    echo 'Este número é negativo!'
fi
```

# If

## Arquivo *Ex9.sh*

```
#!/bin/bash
echo 'Digite um número qualquer:'
read numero
if [ $numero -gt 20 ]; then
    echo 'Este número é maior que 20!'
elif [ $numero -gt 0 ]; then
    echo 'Este número é maior que 0!'
else
    echo 'Este número é negativo!'
fi
```



Else if

# If

## Arquivo *Ex9.sh*

```
#!/bin/bash
echo 'Digite um número qualquer:'
read numero
if [ $numero -gt 20 ]; then
    echo 'Este número é maior que 20!'
elif [ $numero -gt 0 ]; then
    echo 'Este número é maior que 0!'
else
    echo 'Este número é negativo!'
fi
```



Else

# Case

## Arquivo *Ex10.sh*

```
#!/bin/bash
echo 'Selecione uma opção:'
echo '1 - Exibir data e hora do sistema'
echo '2 - Calcular 10/2'
echo '3 - Exibir uma mensagem'
read opcao
```

```
case $opcao in
'1')
    date +"%T, %d/%m/%y, %A"
    ;;
'2')
    result=$((10/2))
    echo "10/2 = $result"
    ;;
'3')
    echo "Informe o seu nome:"
    read nome;
    echo "Olá $nome!"
    ;;
esac
```

# Case

## Arquivo *Ex10.sh*

```
#!/bin/bash
echo 'Selecione uma opção:'
echo '1 - Exibir data e hora do sistema'
echo '2 - Calcular 10/2'
echo '3 - Exibir uma mensagem'
read opcao
```

(Exemplo quebrado em duas partes para caber no slide)

```
case $opcao in
'1')
    date +"%T, %d/%m/%y, %A"
    ;;
'2')
    result=$((10/2))
    echo "10/2 = $result"
    ;;
'3')
    echo "Informe o seu nome:"
    read nome;
    echo "Olá $nome!"
    ;;
esac
```




# Case

## Arquivo *Ex10.sh*

O *case* é equivalente  
ao *If-Elif-Else*

```
#!/bin/bash
echo 'Selecione uma opção:'
echo '1 - Exibir data e hora do sistema'
echo '2 - Calcular 10/2'
echo '3 - Exibir uma mensagem'
read opcao
```



```
case $opcao in
'1')
    date +"%T, %d/%m/%y, %A"
    ;;
'2')
    result=$((10/2))
    echo "10/2 = $result"
    ;;
'3')
    echo "Informe o seu nome:"
    read nome;
    echo "Olá $nome!"
    ;;
esac
```

# Case

Arquivo *Ex10.sh*

Separador de casos

```
#!/bin/sh
echo
echo
echo
echo
read
```

stema'

```
case $opcao in
'1')
    date +"%T, %d/%m/%y, %A"
    ;;
'2')
    result=$((10/2))
    echo "10/2 = $result"
    ;;
'3')
    echo "Informe o seu nome:"
    read nome;
    echo "Olá $nome!"
    ;;
esac
```

# Case

Arquivo *Ex10.sh*

```
#!/bin  
echo  
echo  
echo  
echo  
read
```

**Término de cada  
caso**

ema'

```
case $opcao in
```

```
'1')
```

```
date +"%T, %d/%m/%y, %A"
```

```
;;
```

```
'2')
```

```
result=$((10/2))
```

```
echo "10/2 = $result"
```

```
;;
```

```
'3')
```

```
echo "Informe o seu nome:"
```

```
read nome;
```

```
echo "Olá $nome!"
```

```
;;
```

```
esac
```

# Case

## Arquivo *Ex10.sh*

```
#!/bin/bash
echo 'Selecione uma opção:'
echo '1 - Exibir data e hora do sistema'
echo '2 - Calcular 10/2'
echo '3 - Exibir uma mensagem'
read opcao
```

Término do *case*

```
case $opcao in
'1')
    date +"%T, %d/%m/%y, %A"
    ;;
'2')
    result=$((10/2))
    echo "10/2 = $result"
    ;;
'3')
    echo "Informe o seu nome:"
    read nome;
    echo "Olá $nome!"
    ;;
esac
```

# For

Arquivo *Ex11.sh*

```
#!/bin/bash
echo Testando o loop for
for i in 10 9 8 7 6 5 4 3 2 1
do
    echo $i
    sleep 0.5
done
echo BOOM!
```

# For

Arquivo *Ex11.sh*

```
#!/bin/bash
echo Testando o loop for
for i in 10 9 8 7 6 5 4 3 2 1
do
    echo $i
    sleep 0.5
done
echo BOOM!
```

```
for VARIAVEL in VALORES
do
    AÇÕES
done
```

# For

Arquivo *Ex12.sh*

```
#!/bin/bash
echo Testando o loop for
for i in {10..1}
do
    echo $i
    sleep 0.5
done
echo BOOM!
```

# For

Arquivo *Ex12.sh*

```
#!/bin/bash
echo Testando o loop for
for i in {10..1}
do
    echo $i
    sleep 0.5
done
echo BOOM!
```



Outra forma de indicar valores



# For

Arquivo *Ex13.sh*

```
#!/bin/bash  
echo Testando o loop for  
for i in 1 2 OK "Ola mundo" *  
do  
    echo $i  
done
```

# For

Arquivo *Ex13.sh*

```
#!/bin/bash  
echo Testando o loop for  
for i in 1 2 OK "Ola mundo" *  
do  
    echo $i  
done
```



Os valores podem ser de qualquer tipo

# For

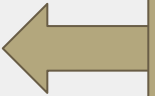
Arquivo *Ex14.sh*

```
#!/bin/bash  
echo Testando o loop for  
for i in {0..100..5}  
do  
    echo i=$i  
done
```

# For

Arquivo *Ex14.sh*

```
#!/bin/bash
echo Testando o loop for
for i in {0..100..5}
do
    echo i=$i
done
```



Outra forma de indicar valores  
(de 0 a 100 em passos de 5)

# For


Arquivo *Ex15.sh*

```
#!/bin/bash  
echo Testando o loop for  
for i in {100..0..-5}  
do  
    echo i=$i  
done
```

# For

Arquivo *Ex15.sh*

```
#!/bin/bash
echo Testando o loop for
for i in {100..0..-5}
do
    echo i=$i
done
```



Outra forma de indicar valores  
(de 100 a 0 em passos de -5)

# While

Arquivo *Ex16.sh*

```
#!/bin/bash
echo 'Informe o que você quiser, -1 para sair'
read dado
while [ $dado != '-1' ]
do
    echo 'Você digitou' $dado
    read dado
done
echo SAIU!
```

# While

Arquivo *Ex16.sh*

```
#!/bin/bash
echo 'Informe o que você quiser, -1 para sair'
read dado
while [ $dado != '-1' ]
do
    echo 'Você digitou' $dado
    read dado
done
echo SAIU!
```

```
while CONDIÇÃO
do
    AÇÕES
done
```



# While

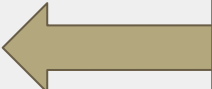
Arquivo *Ex17.sh*

```
#!/bin/bash
echo Informe valor para contagem
read valor
i=1
echo Contagem:
while [ $i -le $valor ]
do
    echo $i
    i=$((i+1))
done
```

# While

Arquivo *Ex17.sh*

```
#!/bin/bash
echo Informe valor para contagem
read valor
i=1
echo Contagem:
while [ $i -le $valor ]
do
    echo $i
    i=$((i+1))
done
```



Incremento da variável *i*

# Funções

## Arquivo *Ex18.sh*

```
#!/bin/bash
listar_lixeira()
{
    ls -l ~/.local/share/Trash/files
}
elevant_quad()
{
    echo 'Informe um número:'
    read x
    x2=$((x*x))
    echo $x^2 = $x2
}
```

```
echo "Escolha uma opção:"
echo "1 - Listar a lixeira"
echo "2 - Calcular x^2"
read opcao
if [ $opcao -eq 1 ]; then
    listar_lixeira
elif [ $opcao -eq 2 ]; then
    elevant_quad
fi
```

# Funções

## Arquivo *Ex18.sh*

```
#!/bin/bash
listar_lixeira()
{
    ls -l ~/.local/share/Trash/files
}
elevant_quad()
{
    echo 'Informe um número:'
    read x
    x2=$((x*x))
    echo $x^2 = $x2
}
```

```
echo "Escolha uma opção:"
echo "1 - Listar a lixeira"
echo "2 - Calcular x^2"
read opcao
if [ $opcao -eq 1 ]; then
    listar_lixeira
elif [ $opcao -eq 2 ]; then
    elevant_quad
fi
```

(Exemplo quebrado em duas partes para caber no slide)

# Funções

## Arquivo *Ex18.sh*

```
#!/bin/bash
listar_lixeira()
{
    ls -l ~/.local/share/Trash/files
}
e elevar_quad()
{
    echo 'Informe um número:'
    read x
    x2=$((x*x))
    echo $x^2 = $x2
}
```

### Funções

*listar\_lixeira( )*

e

*e elevar\_quad( )*

11

# Funções

## Arquivo *Ex18.sh*

```
#!/bin/bash
listar_lixeira()
{
    ls -l ~/.local/share/Trash/files
}
elevant_quad()
{
    echo 'Informe um número'
    read x
    x2=$((x*x))
    echo $x^2 = $x2
}
```

Chamadas  
das funções

```
echo "Escolha uma opção:"
echo "1 - Listar a lixeira"
echo "2 - Calcular x^2"
read opcao
if [ $opcao -eq 1 ]; then
    listar_lixeira
elif [ $opcao -eq 2 ]; then
    elevant_quad
fi
```

# Argumentos


## Arquivo *Ex19.sh*

```
#!/bin/bash
echo O nome deste script é $0
echo $# argumentos passados pelo usuário
if [ $# -ge 1 ]; then
    echo Os parâmetros de entrada foram:
    echo ' ' $@
    echo Em particular...
    echo ' Arg1 =' $1
fi
if [ $# -ge 2 ]; then
    echo ' Arg2 =' $2
fi
```

# Argumentos

## Arquivo *Ex19.sh*

```
#!/bin/bash
echo O nome deste script é $0
echo $# argumentos passados pelo usuário
if [ $# -ge 1 ]; then
    echo Os parâmetros de entrada foram:
    echo ' ' $@
    echo Em particular...
    echo ' Arg1 =' $1
fi
if [ $# -ge 2 ]; then
    echo ' Arg2 =' $2
fi
```




*\$0* representa o primeiro argumento na chamada do script



# Argumentos

## Arquivo *Ex19.sh*

```
#!/bin/bash
echo 0 nome des
echo $#
if [ $# -ge 1 ]; then
    echo Os parâme
    echo ' ' $@
    echo Em particular...
    echo ' Arg1 =' $1
fi
if [ $# -ge 2 ]; then
    echo ' Arg2 =' $2
fi
```

 **`$#`** representa a quantidade de argumentos na chamada do script

# Argumentos

## Arquivo *Ex19.sh*

```
#!/bin/bash
```

```
echo O nome deste script é $0
```

```
echo $# argumentos passados pelo usuário
```

```
$ ./Ex19.sh
```

```
O nome deste script é ./Ex19.sh
```


```
0 argumentos passados pelo usuário
```

# Argumentos

## Arquivo *Ex19.sh*

```
#!/bin/bash
echo O nome deste script é $0
echo $# argumentos passados pelo usuário
if [ $# -ge 1 ]; then
    echo Os parâmetros são: $@
    echo Em particular...
    echo '  Arg1 =' $1
fi
if [ $# -ge 2 ]; then
    echo '  Arg2 =' $2
fi
```


**\$@** representa todos  
argumentos na  
chamada do script,  
exceto \$0



# Argumentos

## Arquivo *Ex19.sh*

```
#!/bin/bash
echo O nome deste script é $0
echo $# argumentos passados pelo usuário
if [ $# -ge 1 ]; then
    echo Os parâmetros de entrada foram:
    echo ' ' $@
    echo Em particular...
    echo ' Arg1 =' $1
fi
if [ $# -ge 2 ]; then
    echo ' Arg2 =' $2
fi
```




*\$1* representa o  
segundo argumento  
na chamada do script

# Argumentos

## Arquivo *Ex19.sh*

```
#!/bin/bash
echo O nome deste script é $0
echo $# argumentos passados pelo usuário
if [ $# -ge 1 ]; then
    echo Os parâmetros de entrada foram:
    echo ' ' $@
    echo Em particular...
    echo ' Arg1 =' $1
fi
if [ $# -ge 2 ]; then
    echo ' Arg2 =' $2
fi
```



**\$2** representa o  
terceiro argumento na  
chamada do script

# Argumentos

## Arquivo *Ex19.sh*

```
#!/bin/bash
echo O nome deste script é $0
echo $# argumentos passados pelo usuário
if [ $# -ge 1 ]; then
    echo Os parâmetros de entrada foram:
    echo ' ' $@
    echo Em particular...
    echo ' Arg1 =' $1
fi
if [ $# -ge 2 ]; then
    echo ' Arg2 =' $2
fi
```

```
$ ./Ex19.sh abc
O nome deste script é ./Ex19.sh
1 argumentos passados pelo usuário
Os parâmetros de entrada foram:
    abc
Em particular...
    Arg1 = abc
```

# Argumentos

## Arquivo *Ex19.sh*

```
#!/bin/bash
echo O nome deste script é $0
echo $# argumentos passados pelo usuário
if [ $# -ge 1 ]; then
    echo Os parâmetros de entrada foram:
    echo ' ' $@
    echo Em particular...
    echo ' Arg1 =' $1
fi
if [ $# -ge 2 ]; then
    echo ' Arg2 =' $2
fi
```

```
$ ./Ex19.sh abc 123
O nome deste script é ./Ex19.sh
2 argumentos passados pelo usuário
Os parâmetros de entrada foram:
    abc 123
Em particular...
    Arg1 = abc
    Arg2 = 123
```

# Argumentos

## Arquivo *Ex20.sh*

```
#!/bin/bash
fatorial()
{
    fat=1
    for i in $(seq $1)
    do
        fat=$((fat*(i)))
    done
    echo $fat
}

echo Insira um número:
read n
f=$(fatorial $n)
echo $n! = $f
```



# Argumentos

## Arquivo *Ex20.sh*

```
#!/bin/bash
fatorial()
{
    fat=1
    for i in $(seq $1)
    do
        fat=$((fat*(i)))
    done
    echo $fat
}
```

```
echo Insira um número:
read n
f=$(fatorial $n)
echo $n! = $f
```

Estamos chamando a função *fatorial( )* e passando a variável *\$n* como segundo argumento (*\$1*):

fatorial \$n




# Argumentos

## Arquivo *Ex20.sh*

```
#!/bin/bash
fatorial()
{
    fat=1
    for i in $(seq $1)
    do
        fat=$((fat*(i)))
    done
    echo $fat
}
```

```
echo Insira um número:
read n
f=$(fatorial $n)
echo $n! = $f
```

O que a função *fatorial( )* escrever na tela com o comando *echo* será guardado na variável *\$f*



# Argumentos

## Arquivo *Ex20.sh*

```
#!/bin/bash
fatorial()
{
    fat=1
    for i in $(seq $1)
    do
        fat=$((fat*(i)))
    done
    echo $fat
}
```

```
echo Insira um número:
read n
f=$(fatorial $n)
echo $n! = $f
```

O comando `seq 3` retorna:

1  
2  
3

Usaremos esta saída para definir os valores do laço *for*


# Argumentos

## Arquivo *Ex20.sh*

```
#!/bin/bash
fatorial()
{
    fat=1
    for i in $(seq $1)
    do
        fat=$((fat*(i)))
    done
    echo $fat
}
```

```
echo Insira um número:
read n
f=$(fatorial $n)
echo $n! = $f
```

A mesma lógica de argumentos (\$0, \$1, \$#, @\$ etc.) é válida para funções



# Argumentos

## Arquivo *Ex20.sh*

```
#!/bin/bash
fatorial()
{
    fat=1
    for i in $(seq $1)
    do
        fat=$((fat*(i)))
    done
    echo $fat
}
```

```
echo Insira um número:
read n
f=$(fatorial $n)
echo $n! = $f
```



Cálculo do fatorial

# Referências

- <https://www.devmedia.com.br/introducao-ao-shell-script-no-linux/25778>
- <https://www.shellscript.sh>