

# Introdução à Ciência da Computação

Shell Script – parte III

---

Professor Iago Augusto de Carvalho  
[iago.carvalho@unifal-mg.edu.br](mailto:iago.carvalho@unifal-mg.edu.br)

# Usando a calculadora do Bash - bc

---

A calculadora do bash é na verdade uma linguagem de programação que permite executar expressões de ponto flutuante no terminal.

Ela reconhece:

Números inteiros e ponto flutuante

Variáveis simples e arrays

Comentários estilo na linguagem C (`/* */`)

Expressões matemáticas

Declarações condicionais if-then, while

Funções

# Usando a calculadora do Bash - bc

---

Para acessar a calculadora do bash no shell digite o comando:

**bc**

Assim, entrará no modo interativo da calculadora. Digite então as expressões que deseja calcular e pressione Enter.

Para sair da calculadora, digite:

**quit**

# Usando a calculadora do Bash - bc

---

```
adriana@adriana-VirtualBox: ~  
adriana@adriana-VirtualBox:~$ bc  
bc 1.07.1  
Copyright 1991-1994, 1997, 1998, 2000, 2004, 2006, 2008, 2012-2017 Free Software  
Foundation, Inc.  
This is free software with ABSOLUTELY NO WARRANTY.  
For details type `warranty'.
```

```
adriana@adriana-VirtualBox: ~  
adriana@adriana-VirtualBox:~$ bc  
bc 1.07.1  
Copyright 1991-1994, 1997, 1998, 2000, 2004, 2006, 2008, 2012-2017 Free Software  
Foundation, Inc.  
This is free software with ABSOLUTELY NO WARRANTY.  
For details type `warranty'.  
2 + 2  
4  
quit  
adriana@adriana-VirtualBox:~$
```

# Usando a calculadora do Bash - bc

---

```
adriana@adriana-VirtualBox:~$ bc
bc 1.07.1
Copyright 1991-1994, 1997, 1998, 2000, 2004, 2006, 2008, 2012-2017 Free Software
Foundation, Inc.
This is free software with ABSOLUTELY NO WARRANTY.
For details type `warranty'.
2 * (3 + 5)
16
sqrt(9)
3
█
```

```
adriana@adriana-VirtualBox:~$ bc -q
2 + 2
4
sqrt(49)
7
3 * (4 - 2)
6
quit
adriana@adriana-VirtualBox:~$
```

```
adriana@adriana-VirtualBox:~$ man bc
```

## Manual bc

```
adriana@adriana-VirtualBox: ~
bc(1)                                General Commands Manual                                bc(1)

NAME
    bc - An arbitrary precision calculator language

SYNTAX
    bc [ -hlwsqv ] [long-options] [ file ... ]

DESCRIPTION
    bc is a language that supports arbitrary precision numbers with interactive execution of statements. There are some similarities in the syntax to the C programming language. A standard math library is available by command line option. If requested, the math library is defined before processing any files. bc starts by processing code from all the files listed on the command line in the order listed. After all files have been processed, bc reads from the standard input. All code is executed as it is read. (If a file contains a command to halt the processor, bc will never read from the standard input.)

    This version of bc contains several extensions beyond traditional bc implementations and the POSIX draft standard. Command line options can cause these extensions to print a warning or to be rejected. This document describes the language accepted by this processor. Extensions will be identified as such.

OPTIONS
    -h, --help
        Print the usage and exit.

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

# Aritmética de Ponto Flutuante na calculadora - bc

É controlada pela variável especial **scale**. Você configura seu valor para o número desejado de casas decimais que necessita nos resultados.

Por padrão, a variável `scale` vem definida com o valor zero(0).

```
adriana@adriana-VirtualBox:~$ bc
bc 1.07.1
Copyright 1991-1994, 1997, 1998, 2000, 2004, 2006, 2008, 2012-2017 Free Software
Foundation, Inc.
This is free software with ABSOLUTELY NO WARRANTY.
For details type `warranty'.
9/2
4
```

# Aritmética de Ponto Flutuante na calculadora - bc

---

```
adriana@adriana-VirtualBox:~$ bc -q
2 / 4
0
9 / 2
4
scale=2
2 / 4
.50
9 / 2
4.50
quit
adriana@adriana-VirtualBox:~$
```

```
adriana@adriana-VirtualBox:~$ bc -q
scale=8
9 / 2
4.50000000
2 / 4
.50000000
scale=4
7 / 3
2.3333
9.5 / 2
4.7500
quit
adriana@adriana-VirtualBox:~$
```



# Aritmética de Ponto Flutuante na calculadora - bc

---

```
adriana@adriana-VirtualBox:~$ bc -q
scale=5
a=4
b=6
a / b
.66666
quit
adriana@adriana-VirtualBox:~$
```

Cálculo com  
variáveis

# Usando a calculadora bc em scripts

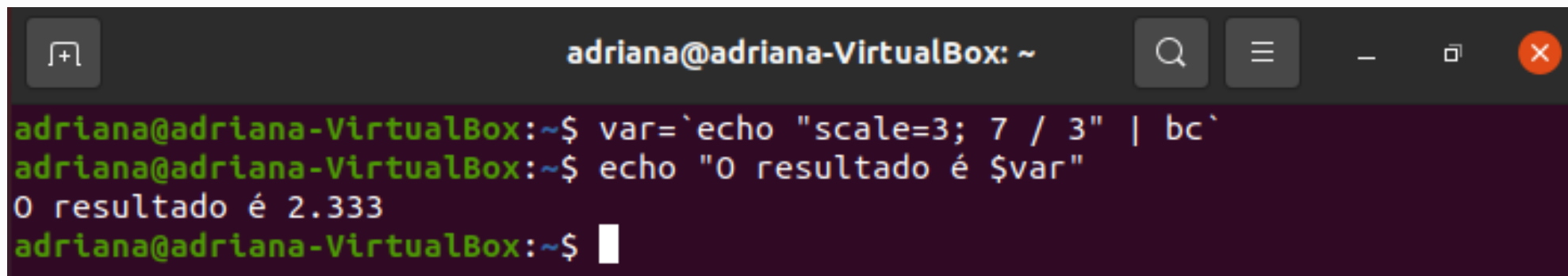
Para usar a bc em um script, use a crase (backtick`) para rodar o comando e atribuir seu valor a uma variável:

**variável=`echo “variáveis; expressão” | bc`**

Exemplo – Digite os comandos a seguir no terminal:

**var=`echo “scale=2; 2 / 5” | bc`**

**echo “Resultado: \$var”**



```
adriana@adriana-VirtualBox: ~  
adriana@adriana-VirtualBox:~$ var=`echo "scale=3; 7 / 3" | bc`  
adriana@adriana-VirtualBox:~$ echo "O resultado é $var"  
O resultado é 2.333  
adriana@adriana-VirtualBox:~$
```

# Usando a calculadora bc em scripts

---

1. Criando script para cálculos simples:

```
#!/bin/bash  
var1=100  
var2=30  
var3=`echo "scale=4; $var1 / $var2" | bc`  
echo "Resultado: $var3"
```

Essa técnica funciona bem para cálculos simples, mas em cálculos mais complexos é recomendável usar a técnica do redirecionamento de entrada inline.

# Usando a calculadora bc em scripts

```
Abrir  *scriptbc.sh  Salvar  -  X
1 #!/bin/bash
2 #Cálculo com script
3 var1=100
4 var2=30
5 var3=`echo "scale=4; $var1 / $var2" | bc`
6 echo "Resultado: $var3"
```

```
adriana@adriana-VirtualBox:~$ gedit scriptbc.sh
adriana@adriana-VirtualBox:~$ chmod 755 scriptbc.sh
adriana@adriana-VirtualBox:~$ ls
'Área de Trabalho'  Downloads  Modelos  Público  snap
Documentos          Imagens   Música   scriptbc.sh  Vídeos
adriana@adriana-VirtualBox:~$ ./scriptbc.sh
Resultado: 3.3333
adriana@adriana-VirtualBox:~$
```

# Usando a calculadora bc em scripts

---

2. Criando script para cálculos mais complexos:

```
#!/bin/bash
var1=6
var2=5
var3=4
var4=`bc << EXP
scale=4
a=($var1 * $var2)
b=($var3 + $var1)
a+b
EXP
`

echo "Resultado: $var4"
```

# Usando a calculadora bc em scripts


```
Abrir  *exemplo.sh  Salvar  -  X
1 #!/bin/bash
2 var1=6
3 var2=5
4 var3=4
5 var4=`bc << EXP
6 scale=4
7 a=($var1 * $var2)
8 b=($var3 + $var1)
9 a + b
10 EXP
11 `
12 echo "Resultado: $var4"
```

```
adriana@adriana-VirtualBox:~$ gedit exemplo.sh
adriana@adriana-VirtualBox:~$ chmod a+x exemplo.sh
adriana@adriana-VirtualBox:~$ ./exemplo.sh
Resultado: 40
adriana@adriana-VirtualBox:~$
```

# Usando a calculadora bc em scripts

---

```
1 #!/bin/bash
2 var1=6
3 var2=5
4 var3=4
5 var4=`bc << EXP
6 scale=4
7 a=($var1 * $var2)
8 b=($var3 + $var1)
9 a + b
10 EXP
11 `
12 echo "Resultado: $a"
```



```
adriana@adriana-VirtualBox:~$ gedit exemplo.sh
adriana@adriana-VirtualBox:~$ ./exemplo.sh
Resultado:
adriana@adriana-VirtualBox:~$
```

# Status de Saída de Comandos

---

Cada comando que roda no shell usa um valor de status de saída para indicar ao shell que o processamento terminou.

O status de saída é um inteiro entre 0 e 255.

A variável especial `$?` armazena o valor do status de saída do último comando executado.

O status de saída de um comando executado com sucesso é 0 (zero). Se houver erro, será um inteiro positivo.



# Status de Saída de Comandos

---

Código	Significado
0	Comando completado com sucesso
1	Erro geral desconhecido
126	O comando não pode ser executado (permissões)
127	Comando não encontrado
130	Comando finalizado com Ctrl + C

Para ver o código de status de um comando, digite **echo \$?** logo após o término de sua execução.

# Status de Saída de Comandos

---

```
adriana@adriana-VirtualBox:~$ ls
'Área de Trabalho'  Downloads  Imagens  Música  scriptbc.sh  Vídeos
Documentos          exemplo.sh Modelos   Público  snap
adriana@adriana-VirtualBox:~$ echo $?
0
adriana@adriana-VirtualBox:~$
```

```
adriana@adriana-VirtualBox:~$ aaa

Comando 'aaa' não encontrado, você quis dizer:

comando 'aha' do deb aha (0.5-1)
comando 'jaaa' do deb jaaa (0.9.2-1)
comando 'aa' do deb astronomical-almanac (5.6-6)

Experimente: sudo apt install <deb name>

adriana@adriana-VirtualBox:~$ echo $?
127
adriana@adriana-VirtualBox:~$
```

```
adriana@adriana-VirtualBox:~$ cat > teste
Adriana

^C
adriana@adriana-VirtualBox:~$ echo $?
130
adriana@adriana-VirtualBox:~$
```

# Comando exit

---

Por padrão, seu shell script finaliza com o status de saída do último comando executado no script.

É possível alterar esse comportamento para retornar seu próprio código de status.

O comando exit permite especificar um status de saída quando o script finaliza.

Exemplo:

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

```
var1=10
```

```
var2=2
```

```
var3=$((var1 * var2))
```

```
echo $var3
```

```
exit 6 #Retorna o código de saída 6
```

E também podemos usar variáveis como parâmetro do comando exit:

```
exit $var3
```

# Comando exit

```
adriana@adriana-VirtualBox:~$ gedit exitteste.sh
adriana@adriana-VirtualBox:~$ ./exitteste.sh
bash: ./exitteste.sh: Permissão negada
adriana@adriana-VirtualBox:~$ echo $?
126
adriana@adriana-VirtualBox:~$
```

```
1 #!/bin/bash
2 var1=10
3 var2=2
4 var3=$((var1 * var2))
5 echo $var3
```

```
1 #!/bin/bash
2 var1=10
3 var2=2
4 var3=$((var1 * var2))
5 echo $var3
6 exit 6
```

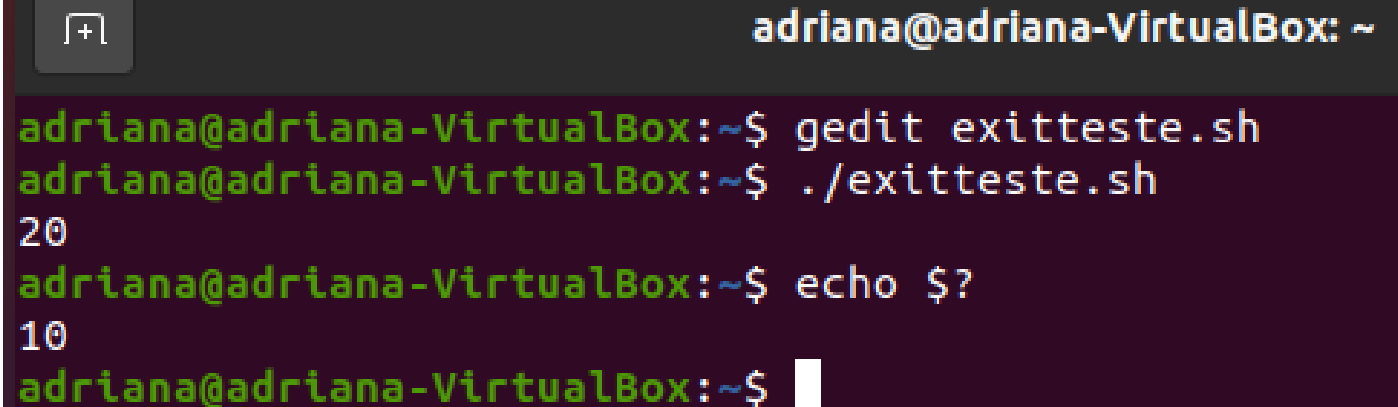
```
adriana@adriana-VirtualBox:~$ chmod 755 exitteste.sh
adriana@adriana-VirtualBox:~$ ./exitteste.sh
20
adriana@adriana-VirtualBox:~$ echo $?
0
adriana@adriana-VirtualBox:~$
```

```
adriana@adriana-VirtualBox:~$ gedit exitteste.sh
adriana@adriana-VirtualBox:~$ ./exitteste.sh
20
adriana@adriana-VirtualBox:~$ echo $?
6
adriana@adriana-VirtualBox:~$
```

# Comando exit

---

```
1 #!/bin/bash
2 var1=10
3 var2=2
4 var3=$((var1 * var2))
5 echo $var3
6 exit $var1
```



A terminal window titled "adriana@adriana-VirtualBox: ~" with a window icon in the top-left corner. The terminal shows the following commands and output:

```
adriana@adriana-VirtualBox:~$ gedit exitteste.sh
adriana@adriana-VirtualBox:~$ ./exitteste.sh
20
adriana@adriana-VirtualBox:~$ echo $?
10
adriana@adriana-VirtualBox:~$
```

# Decisão condicional: If-Then

---

Sintaxe:

```
if comando  
then  
  comandos  
fi
```

ou

```
if comando; then  
  comandos  
fi
```

A declaração if do shell bash executa o comando definido na linha if.

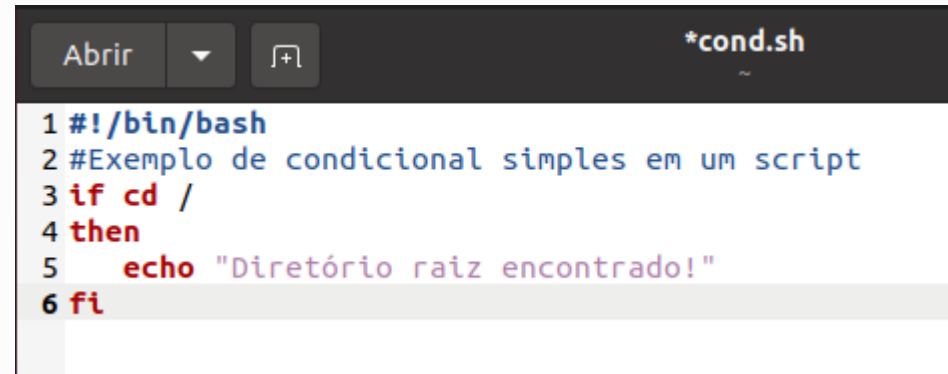
Se o status de saída do comando for zero, os comandos listados após a seção then serão executados.

Caso contrário, esses comandos serão ignorados.

# Decisão condicional: If-Then

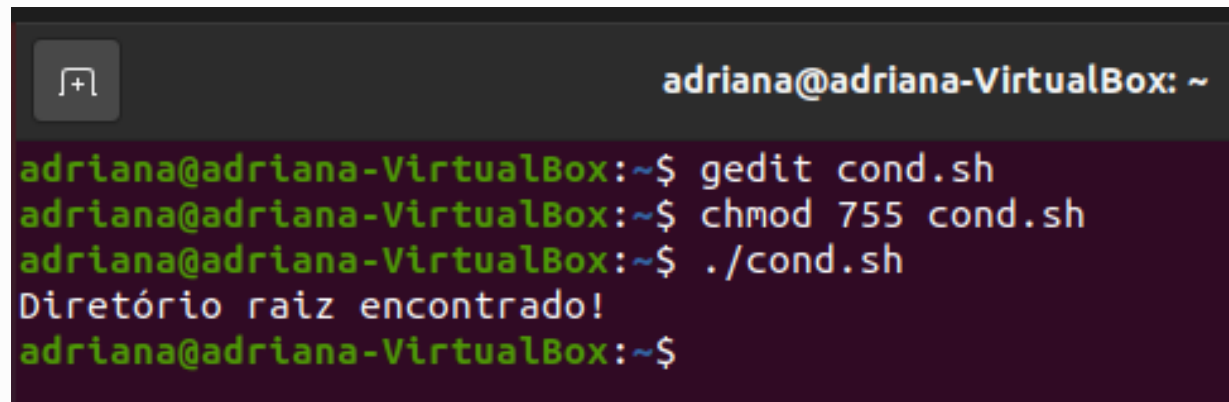
Exemplo:

```
#!/bin/bash
#Exemplo de condicional simples em um script
if cd /
then
    echo "Diretório raiz encontrado!"
fi
```



```
Abrir ▼ [+]
```

```
*cond.sh
~
1 #!/bin/bash
2 #Exemplo de condicional simples em um script
3 if cd /
4 then
5     echo "Diretório raiz encontrado!"
6 fi
```



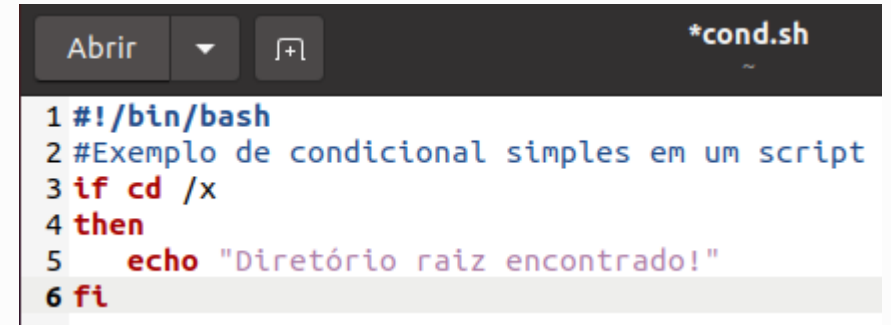
```
adriana@adriana-VirtualBox: ~
```

```
adriana@adriana-VirtualBox:~$ gedit cond.sh
adriana@adriana-VirtualBox:~$ chmod 755 cond.sh
adriana@adriana-VirtualBox:~$ ./cond.sh
Diretório raiz encontrado!
adriana@adriana-VirtualBox:~$
```

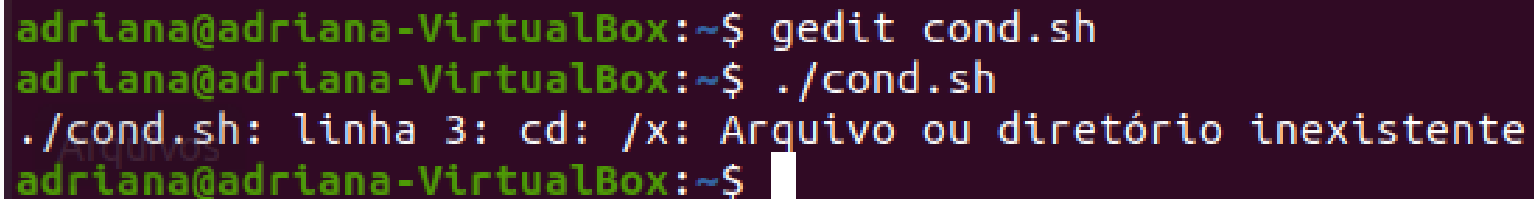
# Decisão condicional: If-Then

Exemplo:

```
#!/bin/bash
#Exemplo de condicional simples em um script
if cd /x
then
    echo "Diretório raiz encontrado!"
fi
```



```
1 #!/bin/bash
2 #Exemplo de condicional simples em um script
3 if cd /x
4 then
5     echo "Diretório raiz encontrado!"
6 fi
```



```
adriana@adriana-VirtualBox:~$ gedit cond.sh
adriana@adriana-VirtualBox:~$ ./cond.sh
./cond.sh: linha 3: cd: /x: Arquivo ou diretório inexistente
adriana@adriana-VirtualBox:~$
```



# Decisão condicional: If-Then-Else

---

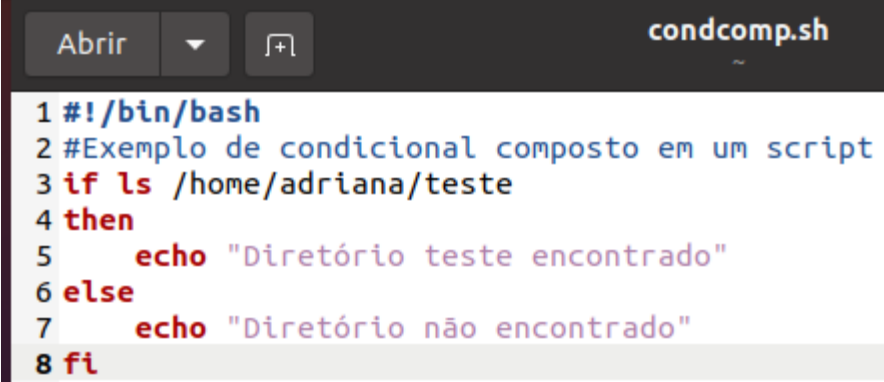
```
if comando
then
    comandos
else
    outros comandos
fi
```

O condicional composto permite executar um bloco de código caso o comando testado retorne código de status zero, e outro bloco de código caso retorne status diferente de zero.

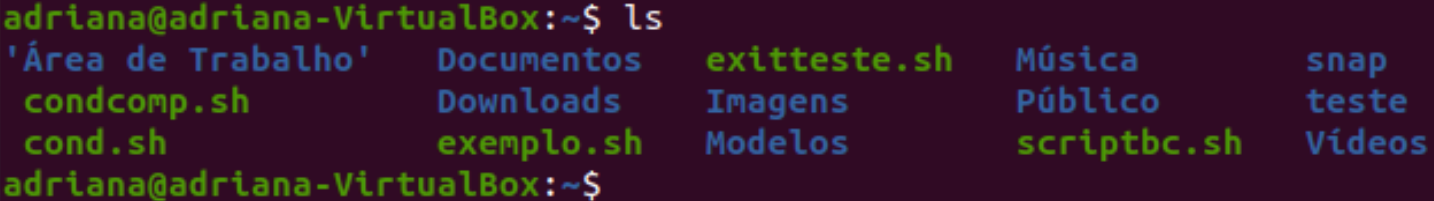
# Decisão condicional: If-Then-Else

Exemplo:

```
#!/bin/bash
#Exemplo de condicional composto em um script
if ls /home/adriana/teste
then
    echo "Diretório teste encontrado!"
else
    echo "Diretório teste não encontrado!"
fi
```

A screenshot of a terminal window with a dark background. The title bar shows 'condcomp.sh'. The script content is as follows:

```
1 #!/bin/bash
2 #Exemplo de condicional composto em um script
3 if ls /home/adriana/teste
4 then
5     echo "Diretório teste encontrado"
6 else
7     echo "Diretório não encontrado"
8 fi
```

A screenshot of a terminal window showing the execution of the script. The prompt is 'adriana@adriana-VirtualBox:~\$'. The command 'ls' is entered, and the output is a grid of files and directories. The prompt then changes to 'adriana@adriana-VirtualBox:~\$' again.

```
adriana@adriana-VirtualBox:~$ ls
'Área de Trabalho'  Documentos  exitteste.sh  Música      snap
condcomp.sh         Downloads  Imagens       Público     teste
cond.sh             exemplo.sh Modelos        scriptbc.sh Vídeos
adriana@adriana-VirtualBox:~$
```

# Decisão condicional: If-Then-Else

---

```
adriana@adriana-VirtualBox: ~  
adriana@adriana-VirtualBox:~$ gedit condcomp.sh  
adriana@adriana-VirtualBox:~$ chmod a+x condcomp.sh  
adriana@adriana-VirtualBox:~$ ./condcomp.sh  
Diretório teste encontrado  
adriana@adriana-VirtualBox:~$
```

```
adriana@adriana-VirtualBox: ~  
adriana@adriana-VirtualBox:~$ ls  
'Área de Trabalho'  Documentos  exitteste.sh  Música  snap  
condcomp.sh         Downloads  Imagens      Público  Vídeos  
cond.sh             exemplo.sh Modelos       scriptbc.sh  
adriana@adriana-VirtualBox:~$ ./condcomp.sh  
ls: não foi possível acessar '/home/adriana/teste': Arquivo ou diretório inexis  
tente  
Diretório não encontrado  
adriana@adriana-VirtualBox:~$
```

# Condicional if aninhado

---

Às vezes é necessário verificar várias situações relacionadas em ao script.

Em vez de ter que escrever declarações if-then separadas, você pode usar uma versão alternativa da seção else, chamada elif.

O **elif** continua uma seção else com outra declaração if-then.

O shell bash executará as declarações if em ordem, e apenas a primeira que retornar status de saída zero terá a seção then correspondente executada.

```
if comando1
then
    comandos
elif comando2
then
    comandos 2
elif comando3
then
    comandos 3
else
    comandos-else
fi
```

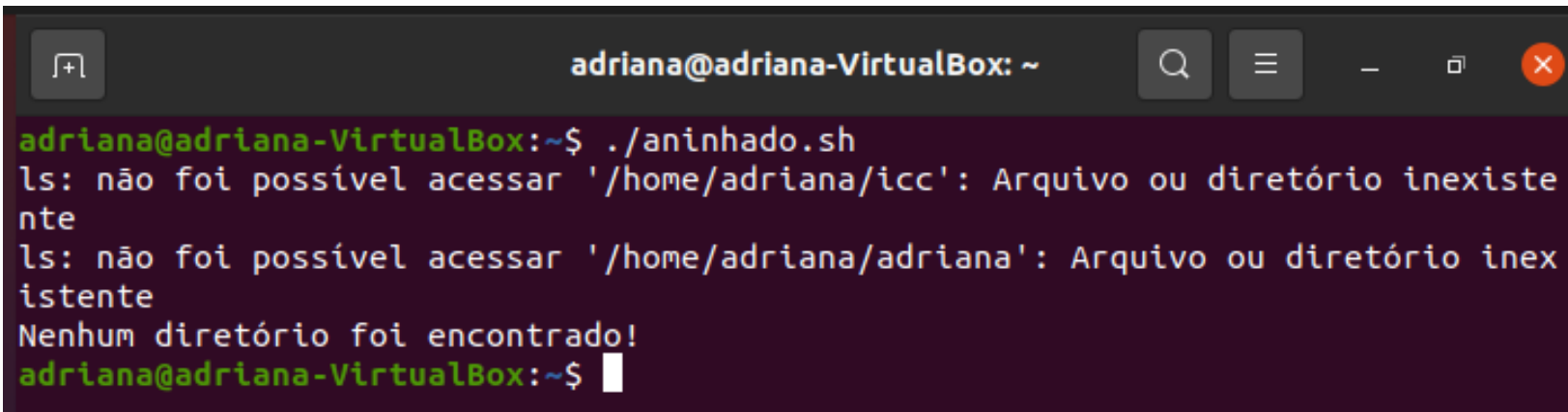
# Condicional if aninhado

```
Abrir  aninhado.sh
1 #!/bin/bash
2 #Teste de if-then aninhado
3 var1="icc"
4 var2="adriana"
5
6 if ls /home/adriana/$var1
7 then
8     echo "Diretório $var1 encontrado!"
9 elif ls /home/adriana/$var2
10 then
11     echo "Diretório $var2 é o que foi encontrado!"
12 else
13     echo "Nenhum diretório foi encontrado!"
14 fi
```

```
adriana@adriana-VirtualBox:~$ gedit aninhado.sh
adriana@adriana-VirtualBox:~$ chmod 755 aninhado.sh
adriana@adriana-VirtualBox:~$ ./aninhado.sh
Diretório icc encontrado!
adriana@adriana-VirtualBox:~$ ls
aninhado.sh      Documentos      icc             Público
'Área de Trabalho' Downloads      Imagens        scriptbc.sh
condcomp.sh      exemplo.sh      Modelos        snap
cond.sh          exitteste.sh    Música          Vídeos
adriana@adriana-VirtualBox:~$
```

# Condicional if aninhado

```
adriana@adriana-VirtualBox:~$ ./aninhado.sh
ls: não foi possível acessar '/home/adriana/icc': Arquivo ou diretório inexiste
nte
Diretório adriana é o que foi encontrado!
adriana@adriana-VirtualBox:~$ ls
adriana          cond.sh          exitteste.sh    Público
aninhado.sh      Documentos      Imagens         scriptbc.sh
'Área de Trabalho' Downloads      Modelos         snap
condcomp.sh      exemplo.sh      Música          Vídeos
adriana@adriana-VirtualBox:~$
```



```
adriana@adriana-VirtualBox:~$ ./aninhado.sh
ls: não foi possível acessar '/home/adriana/icc': Arquivo ou diretório inexiste
nte
ls: não foi possível acessar '/home/adriana/adriana': Arquivo ou diretório inex
istente
Nenhum diretório foi encontrado!
adriana@adriana-VirtualBox:~$
```

# Comando test

---

A declaração if-then não consegue avaliar nenhuma condição que não seja o código de status de saída de um comando.

Porém, é possível avaliar outras condições usando o comando test em uma declaração if-then.

O comando **test** avalia uma condição, e se ela retornar true(V), o comando test retorna o código de status de saída igual a zero. Caso contrário, retorna status de saída igual a 1.

# Uso do test com if-then

---

```
if test condição  
then  
    comandos  
fi
```

○ shell bash fornece uma maneira alternativa de declarar o comando test com if-then:

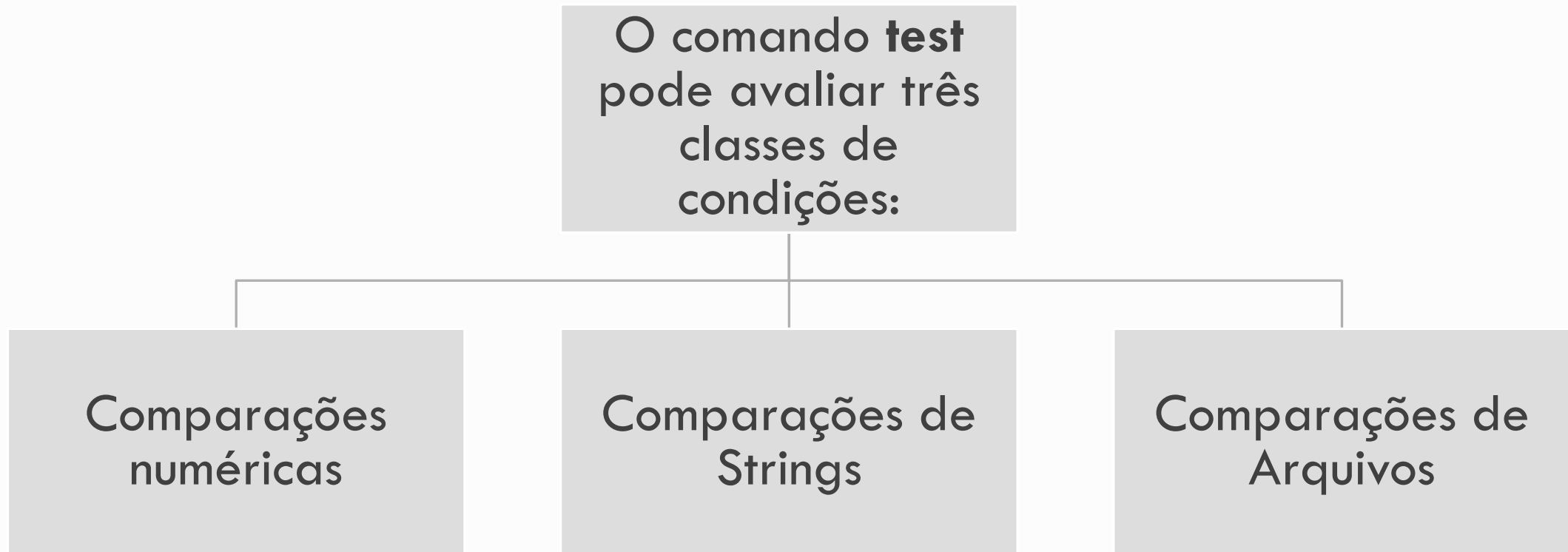
```
if [ condição ]  
then  
    comandos  
fi
```

Os colchetes definem a condição usada pelo test. Deve haver espaços antes e depois da condição.



# Comando test – classes de condições

---



# Comparações numéricas com test

---

Comparação	Descrição
<code>n1 -eq n2</code>	Verifica se n1 é igual a n2
<code>n1 -ge n2</code>	Verifica se n1 é maior ou igual a n2
<code>n1 -gt n2</code>	Verifica se n1 é maior do que n2
<code>n1 -le n2</code>	Verifica se n1 é menor ou igual a n2
<code>n1 -lt n2</code>	Verifica se n1 é menor do que n2
<code>n1 -ne n2</code>	Verifica se n1 é diferente de n2

Bóson Treinamentos

Avaliam tanto números quanto variáveis.

O comando test não suporta aritmética de ponto flutuante.

# Comparações numéricas com test

---

```
#!/bin/bash
#Comparações numéricas com test e if-then
var1=10
var2=15
if [ $var1 -gt 9 ]
then
    echo "A variável de valor $var1 é maior que 9"
fi
if [ $var1 -eq $var2 ]
then
    echo "Os valores são iguais"
else
    echo "Os valores são diferentes"
fi
```

# Comparações numéricas com test

```
Abrir  *test.sh
1 #!/bin/bash
2 #Comparações numéricas com teste e if-then
3 var1=10
4 var2=15
5 if [ $var1 -gt 9 ]
6 then
7     echo "A variável de valor $var1 é maior que 9"
8 fi
9
10 if [ $var1 -eq $var2 ]
11 then
12     echo "Os valores são iguais"
13 else
14     echo "Os valores são diferentes"
15 fi
```

```
adriana@adriana-VirtualBox: ~
adriana@adriana-VirtualBox:~$ gedit test.sh
adriana@adriana-VirtualBox:~$ chmod a+x test.sh
adriana@adriana-VirtualBox:~$ ./test.sh
A variável de valor 10 é maior que 9
Os valores são diferentes
adriana@adriana-VirtualBox:~$
```

# Comparações numéricas com test

```
Abrir  ▾  [+]
```

```
*test.sh
~
1 #!/bin/bash
2 #Comparações numéricas com teste e if-then
3 var1=6
4 var2=6
5 if [ $var1 -gt 9 ]
6 then
7     echo "A variável de valor $var1 é maior que 9"
8 fi
9
10 if [ $var1 -eq $var2 ]
11 then
12     echo "Os valores são iguais"
13 else
14     echo "Os valores são diferentes"
15 fi
```

```
adriana@adriana-VirtualBox:~$ gedit test.sh
adriana@adriana-VirtualBox:~$ ./test.sh
Os valores são iguais
adriana@adriana-VirtualBox:~$
```

# Comparações de string com test

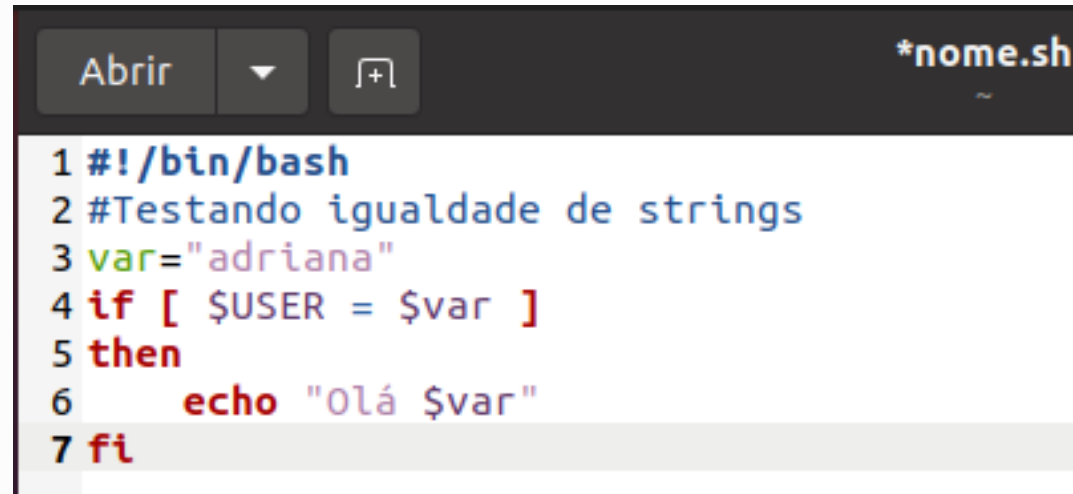
O comando test também permite realizar comparações entre valores de strings.

Comparação	Descrição
<code>str1 = str2</code>	Verifica se <code>str1</code> é idêntica a <code>str2</code>
<code>str1 != str2</code>	Verifica se <code>str1</code> é diferente de <code>str2</code>
<code>str1 &lt; str2</code>	Verifica se <code>str1</code> é “menor” que <code>str2</code>
<code>str1 &gt; str2</code>	Verifica se <code>str1</code> é “maior” que <code>str2</code>
<code>-n str1</code>	Verifica se <code>str1</code> tem comprimento maior que zero
<code>-z str1</code>	Verifica se <code>str1</code> tem comprimento zero

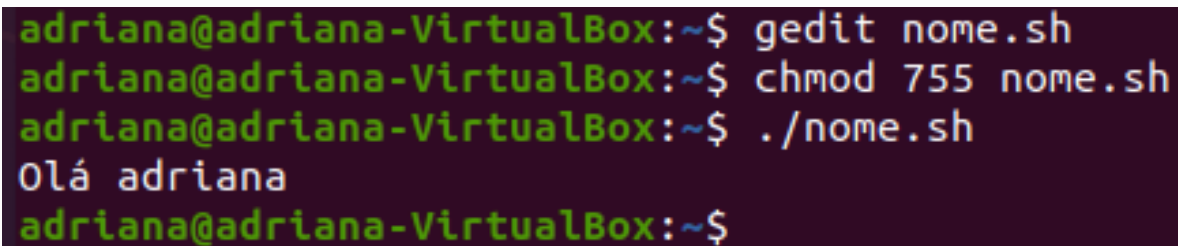
Bóson Treinamentos

# Comparações de string com test

```
#!/bin/bash
#Testar igualdade de strings
var="adriana"
if [ $USER = $var ]
then
    echo "Olá $var"
fi
```

A screenshot of a code editor window titled '\*nome.sh'. The editor has a dark theme with a toolbar at the top containing buttons for 'Abrir', a dropdown arrow, and a file icon. The script content is displayed with syntax highlighting: line 1 is '#!/bin/bash', line 2 is '#Testando igualdade de strings', line 3 is 'var="adriana"', line 4 is 'if [ \$USER = \$var ]', line 5 is 'then', line 6 is ' echo "Olá \$var"', and line 7 is 'fi'.

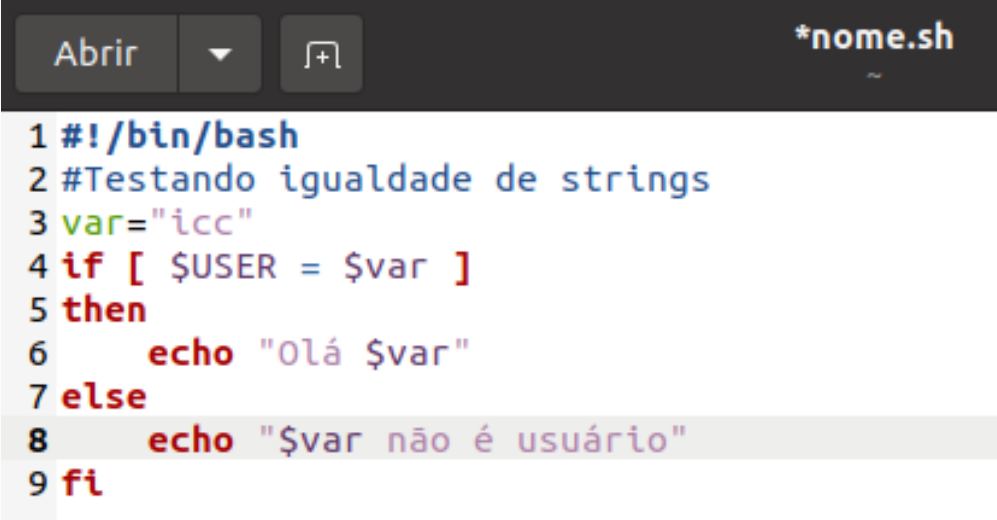
```
1 #!/bin/bash
2 #Testando igualdade de strings
3 var="adriana"
4 if [ $USER = $var ]
5 then
6     echo "Olá $var"
7 fi
```

A screenshot of a terminal window with a dark background. It shows the user 'adriana' at a 'VirtualBox' prompt. The user runs 'gedit nome.sh', then 'chmod 755 nome.sh', and finally './nome.sh'. The output of the script is 'Olá adriana'.

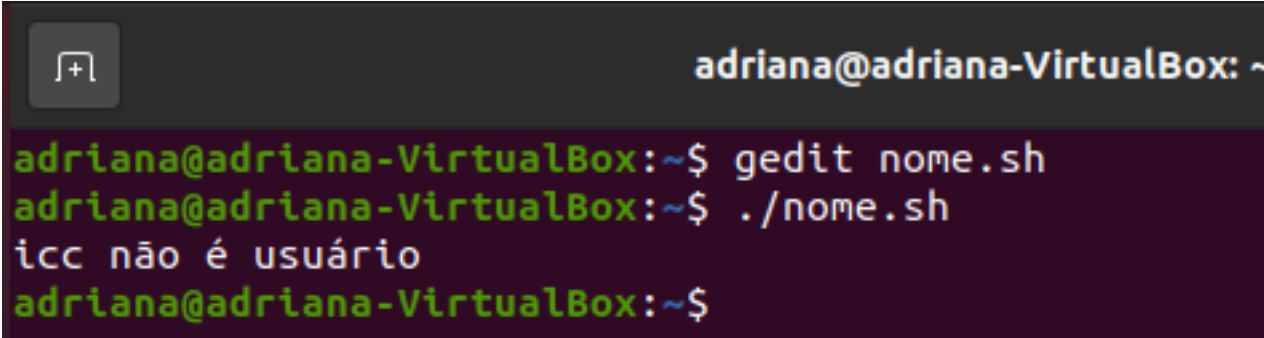
```
adriana@adriana-VirtualBox:~$ gedit nome.sh
adriana@adriana-VirtualBox:~$ chmod 755 nome.sh
adriana@adriana-VirtualBox:~$ ./nome.sh
Olá adriana
adriana@adriana-VirtualBox:~$
```

# Comparações de string com test

```
#!/bin/bash
#Testar igualdade de strings
var="icc"
if [ $USER = $var ]
then
    echo "Olá $var"
else
    echo "$var não é usuário"
fi
```



```
1 #!/bin/bash
2 #Testando igualdade de strings
3 var="icc"
4 if [ $USER = $var ]
5 then
6     echo "Olá $var"
7 else
8     echo "$var não é usuário"
9 fi
```

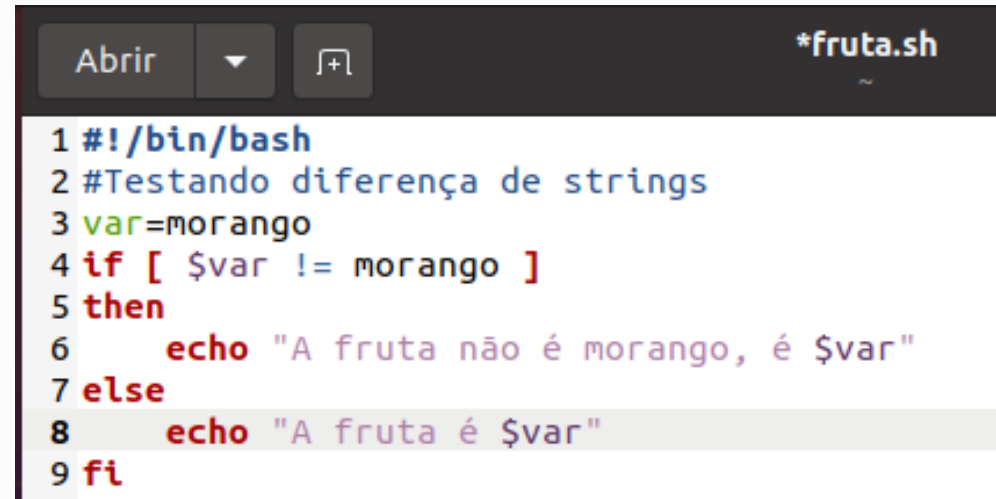


```
adriana@adriana-VirtualBox: ~
adriana@adriana-VirtualBox:~$ gedit nome.sh
adriana@adriana-VirtualBox:~$ ./nome.sh
icc não é usuário
adriana@adriana-VirtualBox:~$
```

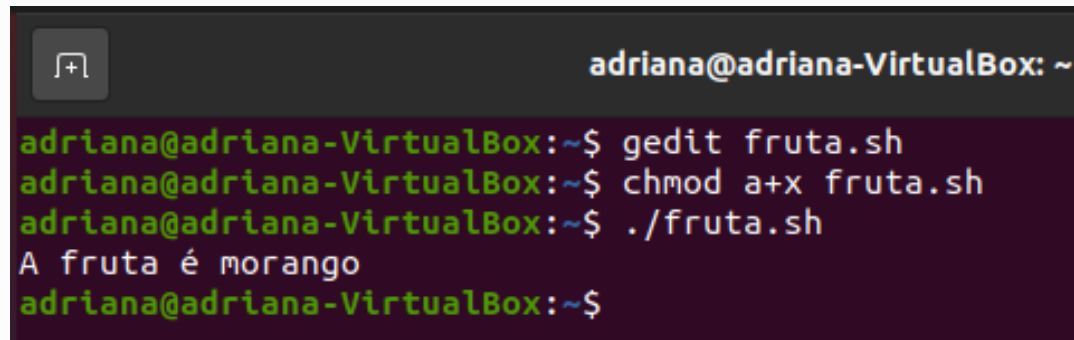


# Comparações de string com test

```
#!/bin/bash
#Testando diferença de strings
var=morango
if [ $var != morango ]
then
    echo "A fruta não é morango, é $var"
else
    echo "A fruta é $var"
fi
```



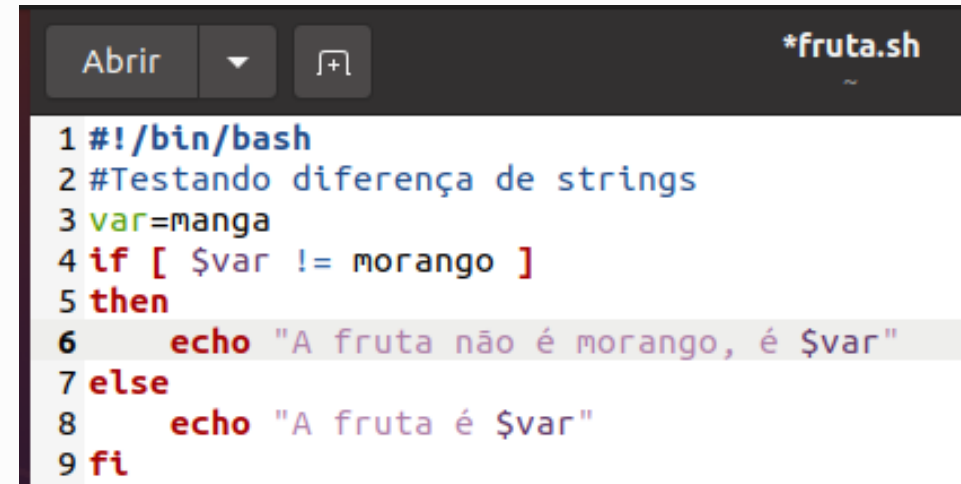
```
1 #!/bin/bash
2 #Testando diferença de strings
3 var=morango
4 if [ $var != morango ]
5 then
6     echo "A fruta não é morango, é $var"
7 else
8     echo "A fruta é $var"
9 fi
```



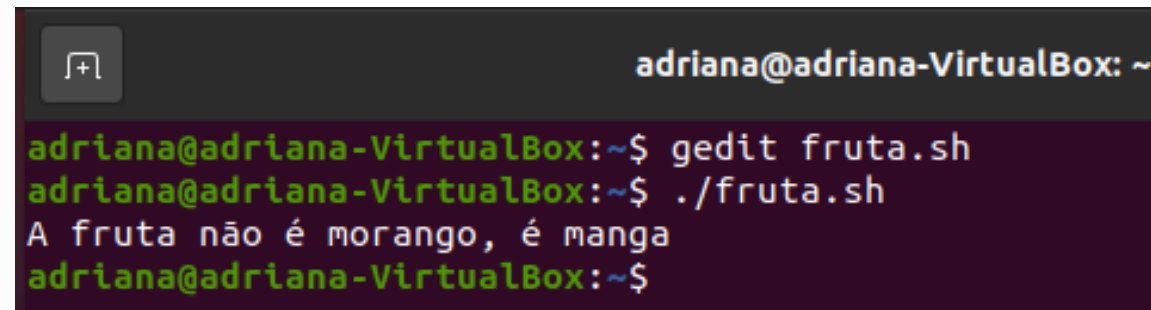
```
adriana@adriana-VirtualBox: ~
adriana@adriana-VirtualBox:~$ gedit fruta.sh
adriana@adriana-VirtualBox:~$ chmod a+x fruta.sh
adriana@adriana-VirtualBox:~$ ./fruta.sh
A fruta é morango
adriana@adriana-VirtualBox:~$
```

# Comparações de string com test

```
#!/bin/bash
#Testando diferença de strings
var=manga
if [ $var != morango ]
then
    echo "A fruta não é morango, é $var"
else
    echo "A fruta é $var"
fi
```



```
Abrir ▼ [ícone] *fruta.sh
1 #!/bin/bash
2 #Testando diferença de strings
3 var=manga
4 if [ $var != morango ]
5 then
6     echo "A fruta não é morango, é $var"
7 else
8     echo "A fruta é $var"
9 fi
```



```
adriana@adriana-VirtualBox: ~
adriana@adriana-VirtualBox:~$ gedit fruta.sh
adriana@adriana-VirtualBox:~$ ./fruta.sh
A fruta não é morango, é manga
adriana@adriana-VirtualBox:~$
```

# Comparações de string com test

---

```
#!/bin/bash
#Testar se variável possui conteúdo
var=abacaxi
var2=""      #variável vazia aspas simples
if [ -n $var ]
then
    echo "A variável não está vazia, contém o valor $var"
else
    echo "A variável está vazia"
fi
```

```
if [ -z $var2 ]
then
    echo "Variável está vazia"
else
    echo "Variável não está vazia"
fi
```

# Comparações de string com test

```
Abrir  *varfruta.sh  Sa

1 #!/bin/bash
2 #Testar se variável possui conteúdo
3 var=abacaxi
4 var2=''
5 if [ -n $var ]; then
6     echo "Variável não está vazia, contém o valor $var"
7 else
8     echo "Variável está vazia"
9 fi
10
11 if [ -z $var2 ]; then
12     echo "Variável está vazia"
13 else
14     echo "Variável não está vazia"
15 fi
```

```
adriana@adriana-VirtualBox: ~

adriana@adriana-VirtualBox:~$ gedit varfruta.sh
adriana@adriana-VirtualBox:~$ chmod 755 varfruta.sh
adriana@adriana-VirtualBox:~$ ./varfruta.sh
Variável não está vazia, contém o valor abacaxi
Variável está vazia
adriana@adriana-VirtualBox:~$
```

# Comparações de Arquivos

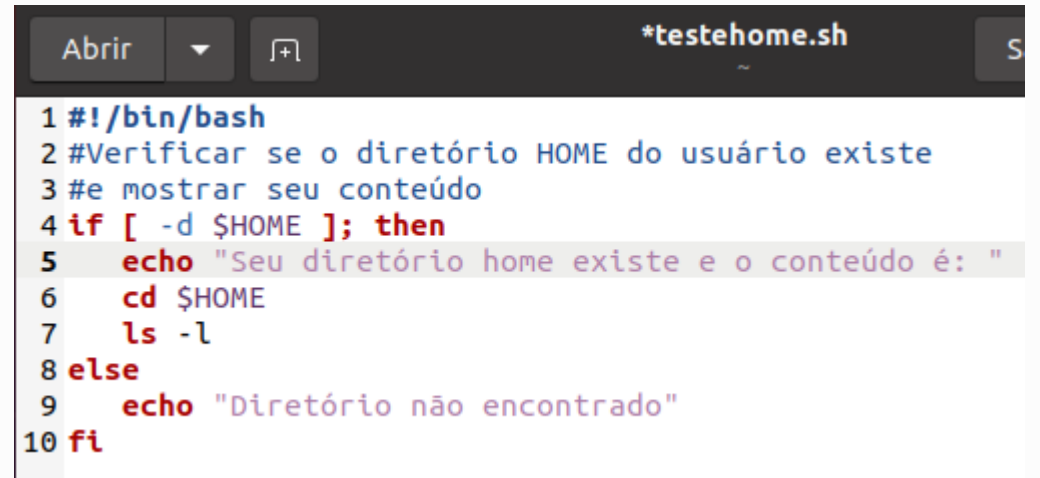
As comparações de arquivos são o tipo de comparações mais poderosas e mais usadas em shell scripting.

O comando test permite testar o status de arquivos e diretórios no sistema de arquivos Linux.

Comparação	Descrição
-d arquivo	Verifica se o arquivo existe e se é um diretório
-e arquivo	Verifica se o arquivo existe
-f arquivo	Verifica se o arquivo existe e se é um arquivo
-r arquivo	Verifica se o arquivo existe e se possui permissão de leitura para o usuário atual
-s arquivo	Verifica se o arquivo existe e não está vazio
-w arquivo	Verifica se o arquivo existe e tem permissão de escrita
-x arquivo	Verifica se o arquivo existe e tem permissão de execução
-O arquivo	Verifica se o arquivo existe e é propriedade do usuário atual
-G arquivo	Verifica se o arquivo existe e seu grupo padrão é o mesmo do usuário atual
<u>arq1</u> -nt <u>arq2</u>	Verifica se <u>arq1</u> é mais novo que <u>arq2</u>
<u>arq1</u> -ot <u>arq2</u>	Verifica se <u>arq1</u> é mais antigo que <u>arq2</u>

# Comparações de arquivos

```
#!/bin/bash
#Verificar se o diretório HOME do usuário existe
# e mostrar seu conteúdo
if [ -d $HOME ]
then
    echo "Seu diretório home existe e o conteúdo é:"
    cd $HOME
    ls -l
else
    echo "Diretório não encontrado"
fi
```



```
1 #!/bin/bash
2 #Verificar se o diretório HOME do usuário existe
3 #e mostrar seu conteúdo
4 if [ -d $HOME ]; then
5     echo "Seu diretório home existe e o conteúdo é: "
6     cd $HOME
7     ls -l
8 else
9     echo "Diretório não encontrado"
10 fi
```

## Comparações de arquivos

```
adriana@adriana-VirtualBox:~$ gedit testehome.sh
adriana@adriana-VirtualBox:~$ chmod 755 testehome.sh
adriana@adriana-VirtualBox:~$ ./testehome.sh
Seu diretório home existe e o conteúdo é:
total 80
-rwxr-xr-x 1 adriana adriana 289 jul 27 19:00 aninhado.sh
drwxr-xr-x 2 adriana adriana 4096 jun 11 12:49 'Área de Trabalho'
-rwxrwxr-x 1 adriana adriana 181 jul 27 18:36 condcomp.sh
-rwxr-xr-x 1 adriana adriana 126 jul 27 18:26 cond.sh
drwxr-xr-x 2 adriana adriana 4096 jul 9 11:26 Documentos
drwxr-xr-x 2 adriana adriana 4096 jun 27 14:49 Downloads
-rwxrwxr-x 1 adriana adriana 135 jul 22 18:28 exemplo.sh
-rwxr-xr-x 1 adriana adriana 136 jul 22 19:45 exitteste.sh
-rwxrwxr-x 1 adriana adriana 166 jul 27 19:43 fruta.sh
drwxr-xr-x 2 adriana adriana 4096 jun 27 14:49 Imagens
drwxr-xr-x 2 adriana adriana 4096 jun 11 12:49 Modelos
drwxr-xr-x 2 adriana adriana 4096 jun 11 12:49 Música
-rwxr-xr-x 1 adriana adriana 149 jul 27 19:37 nome.sh
drwxr-xr-x 2 adriana adriana 4096 jun 11 12:49 Público
-rwxr-xr-x 1 adriana adriana 116 jul 21 22:06 scriptbc.sh
drwxr-xr-x 3 adriana adriana 4096 jul 15 21:37 snap
-rwxr-xr-x 1 adriana adriana 233 jul 27 20:12 testehome.sh
-rwxrwxr-x 1 adriana adriana 260 jul 27 19:27 test.sh
-rwxr-xr-x 1 adriana adriana 291 jul 27 19:58 varfruta.sh
drwxr-xr-x 2 adriana adriana 4096 jun 11 12:49 Vídeos
adriana@adriana-VirtualBox:~$
```

# Comparações de arquivos

---

```
#!/bin/bash
#Verificar se um objeto é um arquivo
if [ -e $HOME ]
then
    echo "O objeto existe. Vamos ver se é arquivo ou diretório"
    if [ -f $HOME ]
    then
        echo "É um arquivo"
    else
        echo "É um diretório"
    fi
else
    echo "Objeto não encontrado"
fi
```



# Comparações de arquivos

```
Abrir  *comparquiv.sh  Salvar  ≡  
1 #!/bin/bash  
2 #Verificar se um objeto é um arquivo  
3  
4 if [ -e $HOME ]; then  
5     echo "O objeto existe. Vamos ver se é arquivo ou diretório"  
6     if [ -f $HOME ]; then  
7         echo "É um arquivo"  
8     else  
9         echo "É um diretório"  
10    fi  
11 else  
12     echo "Objeto não encontrado"  
13 fi
```

```
adriana@adriana-VirtualBox: ~  
adriana@adriana-VirtualBox:~$ gedit comparquiv.sh  
adriana@adriana-VirtualBox:~$ chmod 755 comparquiv.sh  
adriana@adriana-VirtualBox:~$ ./comparquiv.sh  
O objeto existe. Vamos ver se é arquivo ou diretório  
É um diretório  
adriana@adriana-VirtualBox:~$
```

# Comparações de arquivos

---

```
#!/bin/bash
#Verificar permissão de leitura em um arquivo
arquivo=/etc/passwd
#testar se o arquivo existe
if [ -f $arquivo]
then
    #Existe. Testar se o usuário tem permissão de leitura
    if [ -r $arquivo ]
    then
        echo "Possui permissão de leitura. Mostrando as 5 últimas linhas:"
        tail -5 $arquivo
    else
        echo "Sem permissão de leitura"
    fi
else
    echo "Arquivo não encontrado"
fi
```

# Comparações de arquivos

```
Abrir  *permarq.sh  Salvar

1 #!/bin/bash
2 #Verificar permissão de leitura em um arquivo
3 arquivo=/etc/passwd
4 #Testar se o arquivo existe:
5
6 if [ -f $arquivo ]; then
7     echo "Existe. Testar se o usuário tem permissão de leitura"
8     if [ -r $arquivo ]; then
9         echo "Possui permissão de leitura. Mostrando as 5 últimas linhas:"
10        tail -5 $arquivo
11    else
12        echo "Sem permissão de leitura"
13    fi
14 else
15     echo "Arquivo não encontrado"
16 fi
```

```
adriana@adriana-VirtualBox: ~

adriana@adriana-VirtualBox:~$ gedit permarq.sh
adriana@adriana-VirtualBox:~$ chmod 755 permarq.sh
adriana@adriana-VirtualBox:~$ ./permarq.sh
Existe. Testar se o usuário tem permissão de leitura
Possui permissão de leitura. Mostrando as 5 últimas linhas:
pulse:x:123:128:PulseAudio daemon,,,:/var/run/pulse:/usr/sbin/nologin
gnome-initial-setup:x:124:65534::/run/gnome-initial-setup:/bin/false
gdm:x:125:130:Gnome Display Manager:/var/lib/gdm3:/bin/false
adriana:x:1000:1000:Adriana,,,:/home/adriana:/bin/bash
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
adriana@adriana-VirtualBox:~$
```

# Referências

---

PRITCHARD, S.; PESSANHA, B. G.; LANGFELDT, N.; STANGER, J.; DEAN, J. 2007. **Certificação Linux LPI Rápido e Prático. Guia de Referência nível 1: Exames 101 e 102.** 2ª Ed. Rio de Janeiro: Editora Alta Books.

**Curso de Shell Scripting – Bóson Treinamentos**

<http://www.bosontreinamentos.com.br/curso-de-shell-scripting/>