Introdução à Ciência da Computação

Shell Script – parte II

Professor Iago Augusto de Carvalho

iago.carvalho@unifal-mg.edu.br

1

Criando scripts simples Múltiplos comandos

É possível encadear múltiplos comandos em sequência no bash, bastando para isso separá-los com um ponto-e-vírgula:

**cd /; ls -l**

Essa sequência pode ser considerada como um script do shell ultra simples.

Claro que toda vezes que for necessária a execução dessa sequência, deve-se digitar todos os comandos novamente.

Para evitar esse problema, criaremos arquivos de script contendo os comandos que queremos executar.

2

Criando scripts simples Arquivo de Script

Para criar um arquivo de script, use um editor de texto de sua preferência (vi, nano, emacs, gedit) e crie um arquivo.

A primeira tarefa é especificar qual shell será usado na primeira linha do arquivo: **#!/bin/bash**

O símbolo **#** representa uma linha de comentário, exceto na primeira linha do arquivo, onde a combinação **#!** indica o shell que rodará o script.

3

Criando scripts simples Arquivo de Script

Após especificar o shell podemos entrar com os comandos a serem executados, juntamente com os comentários necessários:

**#!/bin/bash**

**# Primeiro script no bash**

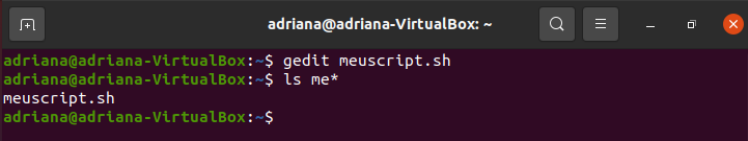
**cd /**

**ls -**l

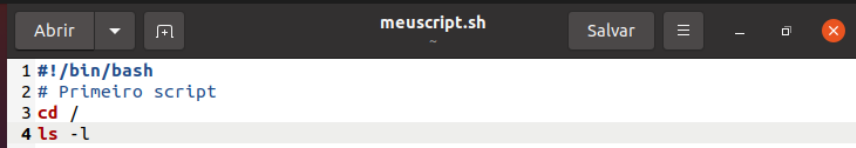
O shell processará os comandos na ordem em que aparecem no arquivo.

4

Criando scripts simples Arquivo de Script

1 

3

2 Ainda não executa.

5

Criando scripts simples Arquivo de Script

Após digitar os comandos desejados no arquivo de script, salve-o com o nome desejado.

É necessário dar permissão de execução ao script, caso contrário ele não roda: **chmod a+x meuscript.sh**

Opcionalmente, adicione também o script a algum diretório presente na variável PATH, ou adicione o diretório que contém a variável.

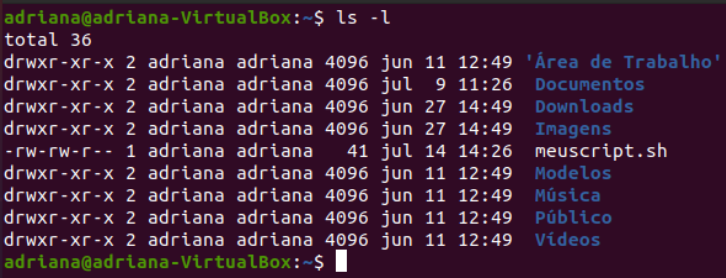
Rode o script chamando-o pelo nome:

**meuscript.sh**

6

Criando scripts simples

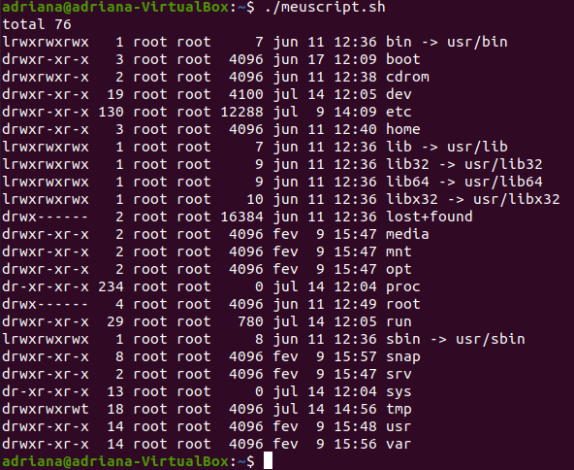
Arquivo de Script

7

Criando scripts simples

Arquivo de Script

8

Criando scripts 

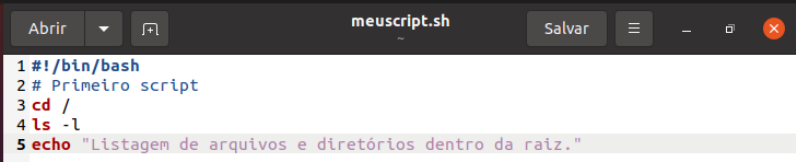
simples

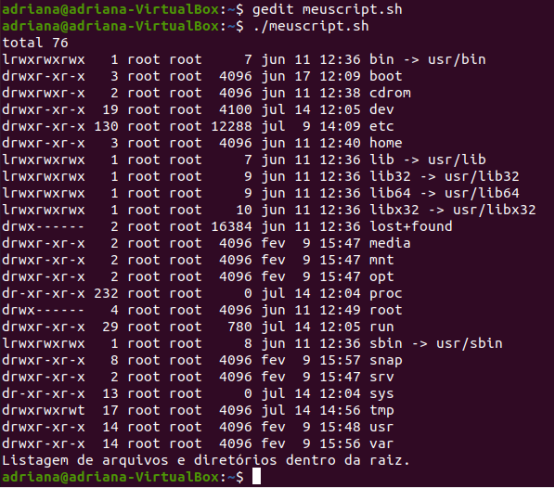
Arquivo de Script

9

Criando scripts simples

Arquivo de Script

10

Criando scripts 

simples

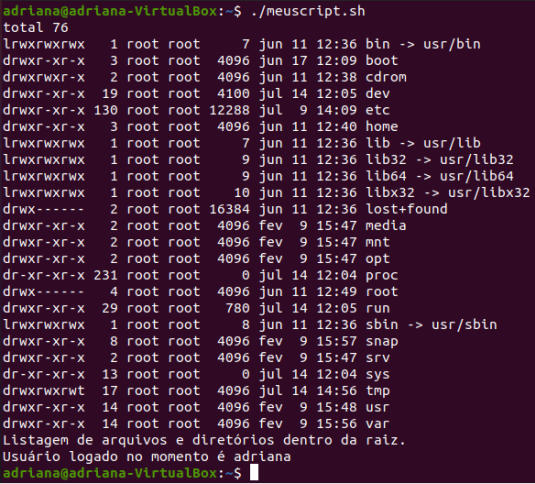
Arquivo de Script

11

Criando scripts simples

Arquivo de Script

12

Criando scripts 

simples

Arquivo de Script

13

Usando variáveis de Ambiente

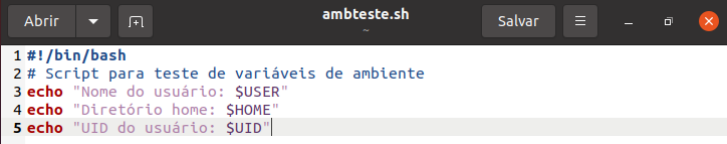
#!/bin/bash

# Informações sobre o usuário:

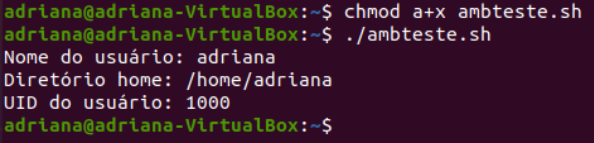
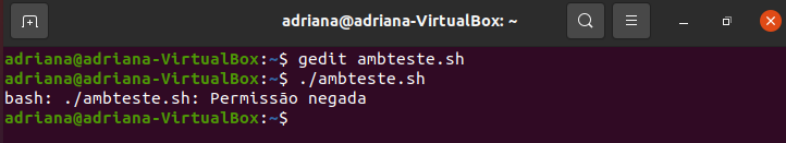
echo “Usuário: $USER”

echo “Diretório home: $HOME”

echo “UID do usuário: “$UID”

14

Usando variáveis de Ambiente

15

Usando variáveis de Usuário

Variáveis próprias podem ser criadas dentro de um script.

Variáveis de usuário podem conter até 20 caracteres no nome e são case-sensitive. O shell determina automaticamente o tipo das variáveis criadas.

As variáveis criadas pelo usuário são desalocadas quando termina a execução do script.

16

Criando variáveis de Usuário

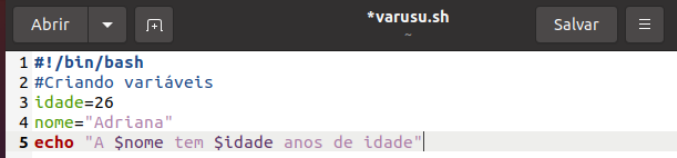
#!/bin/bash

# Criando variáveis

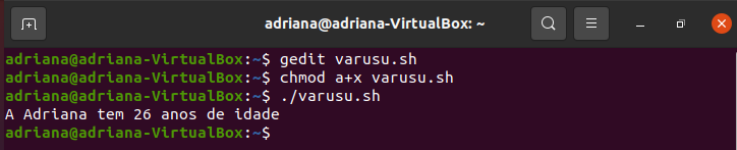
idade=26

nome=“Adriana”

echo “A $nome tem $idade anos de idade”

17

Criando variáveis de Usuário

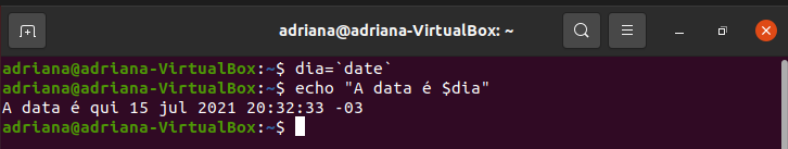
18

Redirecionar saída de comandos e variáveis **Usando crases (backtick`)**

As crases permitem atribuir a saída de um comando do shell a uma variável. O comando todo deve estar entre crases:

**dia=`date`**

**echo “A data é $dia”**

****19

Redirecionar saída de comandos e variáveis 

20

Redirecionar saída de comandos e variáveis 21

Redirecionar saída de comandos e variáveis 22

Redirecionar saída de

comandos e variáveis

23

Redirecionamento de Entrada e Saída

**Redirecionamento de Saída**

O mais comum é enviar a saída de um comando para um arquivo. Para isso, use o símbolo: **>**

**comando > arquivo\_saída**

**ls > arquivo.txt**

Caso queira acrescentar a saída e um comando a um arquivo em vez de sobrescrevê-lo, use o símbolo duplo: **>>**

**comando >> arquivo\_saída**

**date >> arquivo.txt**

24

Redirecionamento de Entrada e Saída 25

Redirecionamento de Entrada e Saída

**Redirecionamento de Entrada**

Em vez de redirecionar a saída de um comando para um arquivo, é possível tomar o conteúdo de um arquivo e redirecioná-lo para um comando.

Para isso, use o símbolo: **<**

**comando < arquivo\_entrada**

**wc < arquivo.txt**

O comando **wc** produz contagem de texto do arquivo. Por padrão mostra três valores: - Número de linhas no texto;

- Número de palavras;

- Número de bytes.

26

Redirecionamento de Entrada e Saída

**Redirecionamento de Entrada Inline**

Com este método é possível especificar os dados para redirecionamento na linha de comando em vez de um arquivo.

O símbolo de redirecionamento de entrada inline é o **<<**

Deve-se especificar um marcador de texto que delimite o inicio e o fim dos dados usados para entrada.

wc << OLA

> primeira frase 

> segunda frase 

> OLA

Prompt secundário –

configurado pela variável de ambiente PS2

Marcador (pode ser qualquer palavra)

27

Redirecionamento de Entrada e Saída **Redirecionamento de Entrada Inline**

****28

Cálculos matemáticos

É possível realizar cálculos matemáticos diretamente no bash, e atribuir o resultado de uma operação a uma variável.

Para isso, englobe a equação usando um cifrão e colchetes:

**variável=$[operação]**

Exemplo:

**primeira=$[2+6]**

**echo $primeira**

**segunda=$[$primeira \* 2]**

**echo $segunda**

29

Cálculos matemáticos

30

Cálculos matemáticos

31

Cálculos 

matemáticos

32

Cálculos matemáticos

33

Cálculos no bash - problema

Porém, há um problema em realizar cálculos com o shell bash, seus operadores matemáticos só suportam aritmética de números inteiros.

var1=9

var2=2

var3=$[$var1 / $var2]

echo “Resultado: $var3”

O resultado deveria ser 4.5, mas o shell retorna 4

34

Cálculos no bash - problema

35

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/

36