



Introdução à Ciência da Computação – Lista 6 Shell script – parte 3

Nome: Pedro Augusto de Souza Finochio RA: 2024.1.08.020

- 1) Crie um script chamado `scriptaritmetico`, com uma operação aritmética arbitrária usando pelo menos 4 variáveis, realizando uma operação de divisão cujo resultado não seja um número inteiro. Execute o script e mostre o resultado. Qual o recurso a ser utilizado caso você queira que o valor não inteiro apareça no resultado? Qual variável eu uso para isso?

```
Open  [+]
```

```
scriptaritmetico.sh
~/
```

```
Save  [Menu]  [Close]  [X]
```

```
1 #!/bin/bash
2 #Operações matemáticas
3
4 um=8
5 dois=6
6 tres=7
7 quatro=4
8 cinco=$(( $um + $dois + $tres / ($quatro) ))
9 echo "Resposta: $cinco"
```

```
2024.1.08.020@suporte-OptiPlex-3050:~$ gedit scriptaritmetico.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ chmod 755 scriptaritmetico.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ ./scriptaritmetico.sh
Resposta: 5
2024.1.08.020@suporte-OptiPlex-3050:~$
```

Para que o valor não inteiro apareça no resultado, utiliza-se a aritmética do ponto flutuante através da variável `scale`.

- 2) Ponha em execução a calculadora `bc`. Mostre o uso da variável `scale`, exibindo um resultado de operação aritmética com 6 casas decimais:

```
2024.1.08.020@suporte-OptiPlex-3050:~$ bc -q
scale=6
4/9
.444444
```

- 3) Crie um script simples chamado `testebc`, em que você utilize a calculadora `bc` dentro dele, envolvendo o uso de algumas variáveis e a operação de divisão, com o direcionamento via pipe. Execute o script, mostrando o resultado:

```
Open  [+]
```

```
testebc.sh
~/
```

```
Save  [Menu]  [Close]  [X]
```

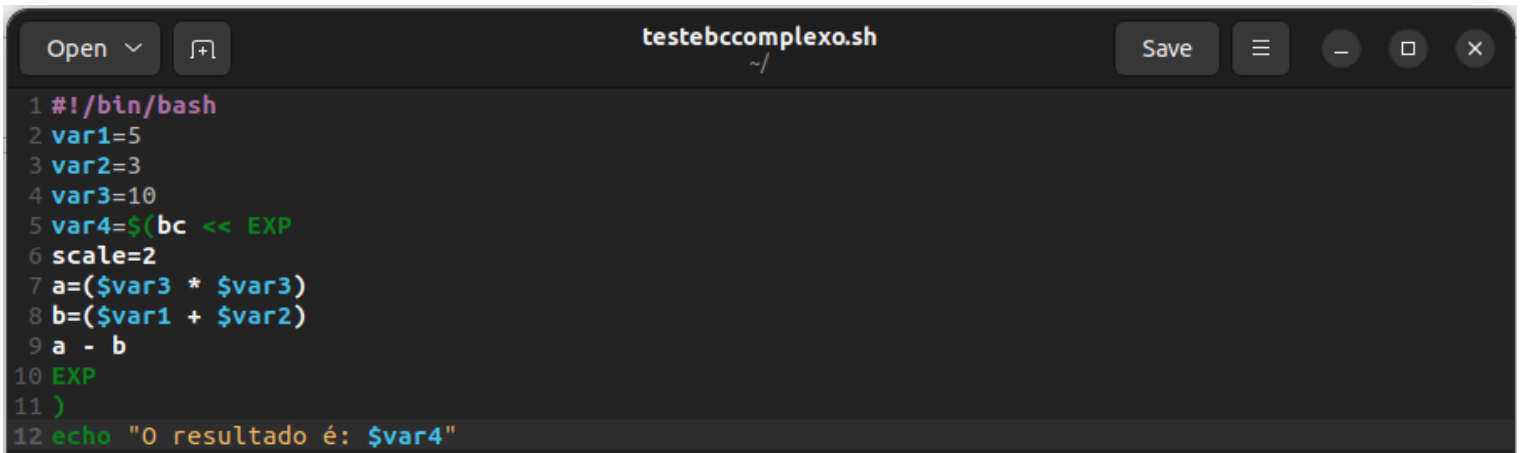
```
1 #!/bin/bash
2 #Cálculo com script
3 var1=90
4 var2=50
5 var3=$(( echo "scale=4; $var2 / $var1" | bc ))
6 echo "O resultado é: $var3"
```

```

2024.1.08.020@suporte-OptiPlex-3050:~$ gedit testebc.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ chmod 755 testebc.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ ls
AEDS1 Desktop diretório Documents Downloads help Music NetBeansProjects Pictures Public scriptaritmetico.sh snap Templates testebc.sh Videos
2024.1.08.020@suporte-OptiPlex-3050:~$ ./testebc.sh
0 resultado é: .5555
2024.1.08.020@suporte-OptiPlex-3050:~$

```

4) Crie um script chamado `testebccomplexo`, em que você utilize operações aritméticas diversas com a calculadora `bc` (pelo menos duas), armazenando os resultados em variáveis, como mostrado na aula. Neste caso, utilize a técnica de redirecionamento de entrada inline. Execute o script, mostrando o resultado:



```

1 #!/bin/bash
2 var1=5
3 var2=3
4 var3=10
5 var4=$(bc << EXP
6 scale=2
7 a=($var3 * $var3)
8 b=($var1 + $var2)
9 a - b
10 EXP
11 )
12 echo "0 resultado é: $var4"

```

```

2024.1.08.020@suporte-OptiPlex-3050:~$ gedit testebccomplexo.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ chmod 755 testebccomplexo.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ ls
AEDS1 Desktop diretório Documents Downloads help Music NetBeansProjects Pictures Public scriptaritmetico.sh snap Templates testebccomplexo.sh testebc.sh Videos
2024.1.08.020@suporte-OptiPlex-3050:~$ ./testebccomplexo.sh
0 resultado é: 92
2024.1.08.020@suporte-OptiPlex-3050:~$

```

5) O que consiste o status de saída de um programa? Mostre um exemplo de execução de dois comandos (um com sucesso e outro desconhecido) e verifique esse status. Mostre em tela:

O status de saída de um programa em um script shell refere-se a um valor numérico que o programa retorna para o ambiente do shell após sua execução. Esses valores são conhecidos como "códigos de saída" ou "códigos de retorno", e esses valores variam entre 0 e 255, sendo o 0 retornado quando o programa é executado com sucesso.

```

2024.1.08.020@suporte-OptiPlex-3050:~$ cat > teste
Exemplo
2024.1.08.020@suporte-OptiPlex-3050:~$ echo $?
0
2024.1.08.020@suporte-OptiPlex-3050:~$ cat > teste
Exemplo
^C
2024.1.08.020@suporte-OptiPlex-3050:~$ echo $?
130
2024.1.08.020@suporte-OptiPlex-3050:~$

```

6) Qual a função do comando exit? Mostre um exemplo do uso do comando exit dentro de um script, mudando o valor padrão do status de saída. Mostre tanto o uso do exit exibindo um número qualquer até 255, quanto o valor de uma variável que você utilize no script. Execute o script e mostre o valor do status de saída em cada caso:

O comando “exit” é usado em shell scripts para encerrar a execução do script e retornar um código de saída específico para o ambiente do shell que invocou o script. Sua função principal é permitir que você indique ao sistema operacional se o script foi executado com sucesso ou se ocorreu algum erro durante a execução.

```
teste.sh
~/
Open  Save  -  □  ×
1 #!/bin/bash
2 var1=6
3 var2=4
4 var3=$((var1 + var2))
5 echo $var3
```

```
2024.1.08.020@suporte-OptiPlex-3050:~$ gedit teste.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ chmod 755 teste.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ ./teste.sh
10
2024.1.08.020@suporte-OptiPlex-3050:~$ echo $?
0
2024.1.08.020@suporte-OptiPlex-3050:~$
```

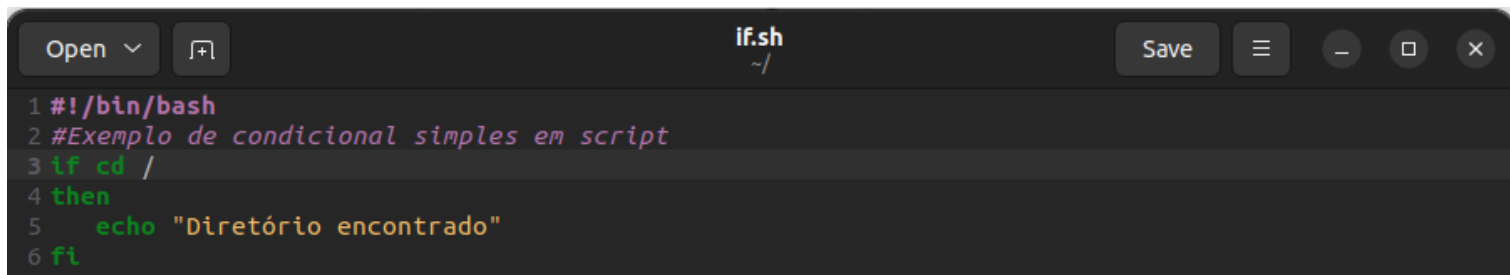
```
teste.sh
~/
Open  Save  -  □  ×
1 #!/bin/bash
2 var1=6
3 var2=4
4 var3=$((var1 + var2))
5 echo $var3
6 exit 7
```

```
2024.1.08.020@suporte-OptiPlex-3050:~$ gedit teste.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ ./teste.sh
10
2024.1.08.020@suporte-OptiPlex-3050:~$ echo $?
7
2024.1.08.020@suporte-OptiPlex-3050:~$
```

7) Crie um script simples envolvendo comandos condicionais if then else, para verificar a existência de um diretório específico no seu home. Primeiro procure um diretório inexistente, depois um diretório existente e exiba as mensagens específicas de acordo com o resultado. Execute o script e mostre em tela:

```
if.sh
~/
Open  Save  -  □  ×
1 #!/bin/bash
2 #Exemplo de condicional simples em script
3 if cd /x
4 then
5     echo "Diretório encontrado"
6 fi
```


```
2024.1.08.020@suporte-OptiPlex-3050:~$ gedit if.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ chmod 755 if.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ ./if.sh
./if.sh: line 3: cd: /x: No such file or directory
2024.1.08.020@suporte-OptiPlex-3050:~$
```



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

```
2024.1.08.020@suporte-OptiPlex-3050:~$ gedit if.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ ./if.sh
Diretório encontrado
2024.1.08.020@suporte-OptiPlex-3050:~$
```

8) Crie um script envolvendo várias condicionais usando a estrutura if then elif else, fazendo duas operações aritméticas arbitrárias, verificando o valor das variáveis que armazenam essa operação, checando se o valor da primeira é maior, menor ou igual ao valor da segunda. Execute o script e mostre o resultado em tela:



```
1 #!/bin/bash
2 # Exemplo de condicional composto em script
3
4 var1=6
5 var2=2
6 var3=4
7 var4=7
8
9 var6=$((var4 - var3))
10 var5=$((var1 / var2))
11
12 if ((var6 > var5)); then
13     echo "$var6 é maior do que $var5"
14 else
15     if ((var6 == var5)); then
16         echo "$var6 é igual a $var5"
17     else
18         echo "$var5 é maior do que $var6"
19     fi
20 fi
```

```
2024.1.08.020@suporte-OptiPlex-3050:~$ gedit ifconta.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ chmod 755 ifconta.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ ./ifconta.sh
3 é igual a 3
2024.1.08.020@suporte-OptiPlex-3050:~$
```

9) Crie um script envolvendo condicionais usando a estrutura if then else, criando duas variáveis string arbitrárias e verificando seus valores, checando se o conteúdo das variáveis é igual. Execute o script e mostre o resultado em tela:

```
scriptvalor.sh
~/
Open Save [Menu] [Window] [Close] [Exit]

1 #!/bin/bash
2 #Exemplo de script com string arbitrária
3 var1="Futebol"
4 var2="Basquete"
5 if (( $var1 == $var2 )); then
6 echo "Futebol é melhor do que basquete"
7 else
8 echo "Basquete é melhor do que futebol"
9 fi
```

```
2024.1.08.020@suporte-OptiPlex-3050:~$ gedit scriptvalor.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ chmod 755 scriptvalor.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ ./scriptvalor.sh
Futebol é melhor do que basquete
2024.1.08.020@suporte-OptiPlex-3050:~$
```

10) Crie um script envolvendo condicionais usando a estrutura if then else, criando uma string com um conteúdo, verificando se seu valor é “fruta”. Execute o script e mostre o resultado em tela:

```
scriptfruta.sh
~/
Open Save [Menu] [Window] [Close] [Exit]

1 #!/bin/bash
2 var1="vegetal"
3 if ((var1 = "fruta")); then
4 echo "A string é a variável fruta"
5 else
6 echo "A string não é a variável fruta"
7 fi
```

```
2024.1.08.020@suporte-OptiPlex-3050:~$ gedit scriptfruta.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ chmod 755 scriptfruta.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ ./scriptfruta.sh
A string não é a variável fruta
2024.1.08.020@suporte-OptiPlex-3050:~$
```

11) Crie um script envolvendo condicionais usando a estrutura if then else, criando duas strings, uma vazia, outra com conteúdo e verificando estes resultados (se tem conteúdo em ambos os casos):

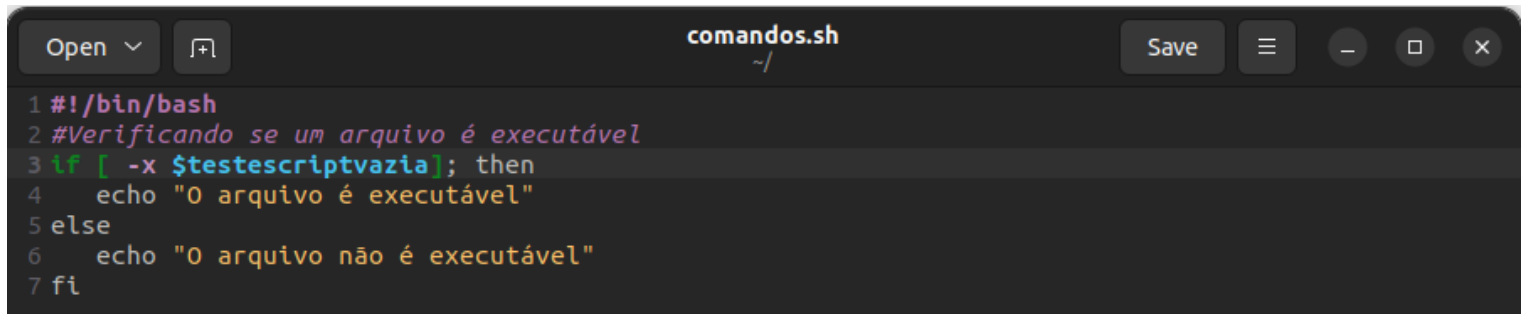
```
scriptvazia.sh
~/
Open Save [Menu] [Window] [Close] [Exit]

1 #!/bin/bash
2
3 var1=" "
4 var2="cheio"
5
6 if [ -z "$var1" ] && [ -z "$var2" ]; then
7 echo "As strings estão vazias."
8 else
9 echo "Pelo menos uma string não está vazia."
10 fi
```

```
2024.1.08.020@suporte-OptiPlex-3050:~$ gedit scriptvazia.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ chmod 755 scriptvazia.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ ./scriptvazia.sh
Pelo menos uma string não está vazia.
2024.1.08.020@suporte-OptiPlex-3050:~$
```

12) Cite 5 opções de comparações envolvendo arquivos. Escolha uma das opções e crie um script envolvendo essa opção:

O comando “-e” verifica se um arquivo existe, o “-d” verifica se um arquivo é um diretório, o “-r” verifica se o arquivo é legível, o “-w” verifica se o arquivo tem permissão para escrever e o “-x” verifica se o arquivo é executável.

A screenshot of a terminal window with a dark background. The window title is 'comandos.sh' with a tilde icon. The terminal shows a script with seven lines: a shebang, a comment, an if statement checking for the -x permission, and echo commands for both true and false conditions. The script is saved and then executed, resulting in the output '0 arquivo é executável'.

```
Open  [icon]  comandos.sh  Save  [icon]  [icon]  [icon]
1 #!/bin/bash
2 #Verificando se um arquivo é executável
3 if [ -x $testescriptvazia ]; then
4     echo "0 arquivo é executável"
5 else
6     echo "0 arquivo não é executável"
7 fi
```

```
2024.1.08.020@suporte-OptiPlex-3050:~$ gedit comandos.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ chmod 755 comandos.sh
2024.1.08.020@suporte-OptiPlex-3050:~$ ./comandos.sh
0 arquivo é executável
2024.1.08.020@suporte-OptiPlex-3050:~$
```

