



## Introdução à Ciência da Computação – Lista 6

### Shell script – parte 3

**Nome:** Érik Alexandre Vieira Peres **RA:** 2025.1.08.005

- 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?

```
scriptaritmetico.sh
1 #!/bin/bash
2 one=5
3 two=4
4 tree=2
5 for=$((($one + $two)/$tree)
6 echo "Resultado: $for"

2025.1.08.005@suporte-OptiPlex-3050:~$ chmod a+x scriptaritmetico.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ gedit scriptaritmetico.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ ./scriptaritmetico.sh
Resultado: 4
2025.1.08.005@suporte-OptiPlex-3050:~$
```

Pode-se utilizar a variável “scale”, ela permite exibir os valores que estão após a casa decimal, além de definir o número de casas decimais com o qual se quer exibir.

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

```
2025.1.08.005@suporte-OptiPlex-3050:~$ bc
bc 1.07.1
Copyright 1991-1994, 1997, 1998, 2000, 2006
Foundation, Inc.
This is free software with ABSOLUTELY NO WARRANTY.
For details type 'warranty'.
10 / 6
1
scale=6
10 / 6
1.666666
quit
2025.1.08.005@suporte-OptiPlex-3050:~$
```

- 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.

```
testebc.sh
1 #!/bin/bash
2 var1=500
3 var2=10
4 var3=5
5 var4=`echo "scale=2; ($var1 * $var2) / $var3" | bc`
6 echo "Resultado: $var4"

2025.1.08.005@suporte-OptiPlex-3050:~$ gedit testebc.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ chmod 755 testebc.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ ls
1.zip  DEADJOE  NetBeansProjects  testebc.sh
2.zip  Desktop  Pictures          teste.cpp
3.zip  Documents projeto_vscode    testecrashes.sh
4.zip  Downloads Public           teste.exe
5.zip  log.2005251220 scriptaritmetico.sh testevariaveisambiente.sh
6.zip  log.2005251221 snap             testevariaveis.sh
arq1.sh log.2005251222 Templates        Videos
arq.txt Music       teste

2025.1.08.005@suporte-OptiPlex-3050:~$ ./testebc.sh
"Resultado: 1000.00"
2025.1.08.005@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.

```
Open ▾ [icon] testebccomplexo.sh ~/
1 #!/bin/bash
2 var1=1
3 var2=3
4 var3=5
5 var4=6
6 var5=`bc << EXP
7 scale=2
8 a=($var1 + $var3)
9 b=($var2 * $var4)
10 a+b
11 EXP
12 `
13 var6=`bc << EXP
14 scale=2
15 c=($var2 * $var1)
16 d=($var4 + $var3)
17 c+d
18 EXP
19 `
20 echo "Resultado1: $var5"
21 echo "Resultado2: $var6"

2025.1.08.005@suporte-OptiPlex-3050:~$ gedit testebccomplexo.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ chmod 755 testebccomplexo.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ ls
1.zip      DEADJOE      NetBeansProjects  testebccomplexo.sh
2.zip      Desktop      Pictures           testebc.sh
3.zip      Documents    projeto_vscode     teste.cpp
4.zip      Downloads    Public             testecrases.sh
5.zip      log.2005251220 scriptaritmetico.sh teste.exe
6.zip      log.2005251221 snap             testevariaveisambiente.sh
arq1.sh    log.2005251222 Templates         testevariaveis.sh
arq.txt    Music        teste              Videos
2025.1.08.005@suporte-OptiPlex-3050:~$ ./testebccomplexo.sh
"Resultado1: 24"
"Resultado2: 14"
2025.1.08.005@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.

Os comandos shell possuem valores de status únicos de saída, servem para indicar ao shell que o processamento terminou. O status de saída de sucesso ao executar um comando é "0", enquanto "números positivos" indicam erros. O comando para visualizar o status é "echo \$?" logo após o término da execução, sendo que "\$?" é a variável que armazena o valor.

```
2025.1.08.005@suporte-OptiPlex-3050:~$ ls
1.zip      DEADJOE      NetBeansProjects  testebccomplexo.sh
2.zip      Desktop      Pictures           testebc.sh
3.zip      Documents    projeto_vscode     teste.cpp
4.zip      Downloads    Public             testecrases.sh
5.zip      log.2005251220 scriptaritmetico.sh teste.exe
6.zip      log.2005251221 snap             testevariaveisambiente.sh
arq1.sh    log.2005251222 Templates         testevariaveis.sh
arq.txt    Music        teste              Videos
2025.1.08.005@suporte-OptiPlex-3050:~$ echo $?
0
2025.1.08.005@suporte-OptiPlex-3050:~$ lss
Command 'lss' not found, but there are 15 similar ones.
2025.1.08.005@suporte-OptiPlex-3050:~$ echo $?
127
2025.1.08.005@suporte-OptiPlex-3050:~$
```

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.

Ele é uma espécie de comando \$? com liberdade de manipulação de valores. Dentro de um script, ele é indicado por "exit" seguido do valor a ser definido:

```
Open ▾ [icon] *testebc.sh ~/
1 #!/bin/bash
2 var1=500
3 var2=10
4 var3=5
5 var4=`echo "scale=2; ($var1 * $var2) / $var3" | bc`
6 echo "Resultado: $var4"
7 exit 7

2025.1.08.005@suporte-OptiPlex-3050:~$ gedit testebc.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ chmod 755 testebc.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ ./testebc.sh
"Resultado: 1000.00"
2025.1.08.005@suporte-OptiPlex-3050:~$ echo $?
7
2025.1.08.005@suporte-OptiPlex-3050:~$

Open ▾ [icon] testebc.sh ~/
1 #!/bin/bash
2 var1=500
3 var2=10
4 var3=5
5 var4=`echo "scale=2; ($var1 * $var2) / $var3" | bc`
6 echo "Resultado: $var4"
7 exit $var1

2025.1.08.005@suporte-OptiPlex-3050:~$ gedit testebc.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ chmod 755 testebc.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ ./testebc.sh
Resultado: 1000.00
2025.1.08.005@suporte-OptiPlex-3050:~$ echo $?
244
2025.1.08.005@suporte-OptiPlex-3050:~$
```

- 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.

```
1 #!/bin/bash
2 if ls /home/2025.1.08.005/snap
3 then
4     echo "Diretório snap encontrado!"
5 else
6     echo "Diretório snap não encontrado!"
7 fi
```

```
2025.1.08.005@suporte-OptiPlex-3050:~$ gedit if.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ chmod 755 if.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ ls
1.zip Desktop Pictures teste.cpp
2.zip Documents projeto_vscode testecrises.sh
3.zip Downloads Public teste.exe
4.zip if.sh scriptaritmetico.sh testevariavelambiente.sh
5.zip log.2005251220 snap testevariaveis.sh
6.zip log.2005251221 Templates Videos
arq1.sh log.2005251222 teste
arq.txt Music testeccomplexo.sh
DEADJOE NetheansProjects testebc.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ gedit if.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ chmod 755 if.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ ./if.sh
ls: cannot access '/home/2025.1.08.005/snap': No such file or directory
"Diretório snap não encontrado!"
2025.1.08.005@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 var1=$((1 + 2))
3 var2=$((2 + 4))
4 if [ "$var1" -ne "$var2" ];
5 then
6     if [ "$var1" -gt "$var2" ];
7     then
8         echo "$var1 é > $var2"
9     else
10        echo "$var1 é < $var2"
11    fi
12 else
13    echo "$var1 é = $var2"
14 fi
```

```
2025.1.08.005@suporte-OptiPlex-3050:~$ gedit if.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ chmod 755 if.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ ./if.sh
3 é < 6
2025.1.08.005@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.

```
1 #!/bin/bash
2 var1="Cebola"
3 var2="Cebola"
4 if [ $var1 = $var2 ]
5 then
6     echo "Com $var1"
7 fi
```

```
2025.1.08.005@suporte-OptiPlex-3050:~$ gedit if.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ chmod 755 if.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ ./if.sh
Com Cebola
2025.1.08.005@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.

```
Open  [icon] if.sh ~/
if.sh x
1 #!/bin/bash
2 var=graviola
3 if [ $var != morango ]
4 then
5     echo "A fruta não é morango, é $var"
6 else
7     echo "A fruta é $var"
8 fi
```

```
2025.1.08.005@suporte-OptiPlex-3050:
2025.1.08.005@suporte-OptiPlex-3050:~$ gedit if.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ chmod 755 if.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ ./if.sh
A fruta não é morango, é graviola
2025.1.08.005@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).

```
Open  [icon] if.sh ~/
if.sh x
1 #!/bin/bash
2 var1=Mulher
3 var2=''
4 if [ -n $var1 ]
5 then
6     echo "Não vazia, há: $var1"
7 else
8     echo "Vazia"
9 fi
10
11 if [ -z $var2 ]
12 then
13     echo "Vazia"
14 else
15     echo "Não vazia, há: $var2"
16 fi
```

```
2025.1.08.005@suporte-OptiPlex-3050:
2025.1.08.005@suporte-OptiPlex-3050:~$ gedit if.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ chmod 755 if.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ ./if.sh
Não vazia, há: Mulher
Vazia
2025.1.08.005@suporte-OptiPlex-3050:~$
```

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

1: -d arquivo (Verifica se o arquivo existe e se é um diretório).

2: -e arquivo (Verifica se o arquivo existe).

3: -f arquivo (Verifica se o arquivo existe e se é um arquivo).

4: -r arquivo (Verifica se o arquivo existe e se possui permissão de leitura para o usuário atual).

5: -s arquivo (Verifica se o arquivo existe e se não está vazio).

Opção escolhida: 2

```
Open  [icon] if.sh ~/
if.sh x
1 #!/bin/bash
2 if [ -e $HOME ]
3 then
4     echo "O arquivo existe"
5 else
6     echo "O arquivo não existe"
7 fi
```

```
2025.1.08.005@suporte-OptiPlex-3050:
2025.1.08.005@suporte-OptiPlex-3050:~$ gedit if.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ chmod 755 if.sh
2025.1.08.005@suporte-OptiPlex-3050:~$ ./if.sh
O arquivo existe
2025.1.08.005@suporte-OptiPlex-3050:~$
```