MINISTÉRIO DA EDUCAÇÃO
Universidade Federal de Alfenas
UNIFAL-MG
Rua Gabriel Monteiro da Silva, 700 – Centro. Alfenas/MG. CEP: 37130-001

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

Nome: Gabriela Mazon Rabello de Souza **RA:2025.1.08.006**

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?

Recurso utilizado: scale=x; variável: x.

```
#!/bin/bash
a=10
b=3
c=4
d=2
resultado=$[($a+$b)/($c+$d)]
echo "resultado:$resultado"
```

```
2025.1.08.006@suporte-OptiPlex-3050:~$ gedit scriptaritmetico.sh
2025.1.08.006@suporte-OptiPlex-3050:~$ chmod 755 scriptaritmetico.sh
2025.1.08.006@suporte-OptiPlex-3050:~$ ./scriptaritmetico.sh
resultado:2
```

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

```
2025.1.08.006@suporte-OptiPlex-3050:~$ 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'.
scale=6
10/3
3.333333
quit
```

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 ~/

```
#!/bin/bash
#testebc
x=7
y=2
resultado=`echo "scale=3; $x / $y" | bc`
echo "Resultado:$resultado!"
```

```
2025.1.08.006@suporte-OptiPlex-3050:~$ gedit testebc.sh
2025.1.08.006@suporte-OptiPlex-3050:~$ chmod 755 testebc.sh
2025.1.08.006@suporte-OptiPlex-3050:~$ ./testebc.sh
Resultado:3.500!
```

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.

```
#!/bin/bash

a=5
b=3
c=2
resultado=`bc << EOF
scale=4
x=$a + $b
y=$a * $c
x + y
EOF
`
echo "Resultado:$resultado"

```

```
2025.1.08.006@suporte-OptiPlex-3050:~$ gedit testebccomplexo.sh
2025.1.08.006@suporte-OptiPlex-3050:~$ chmod 755 testebccomplexo.sh
2025.1.08.006@suporte-OptiPlex-3050:~$ ./testebccomplexo.sh
Resultado:18
2025.1.08.006@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.

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.

```
2025.1.08.006@suporte-OptiPlex-3050:~$ ls
'AEDS I'                          Public
 Desktop                          scriptaritmetico.sh
 Documents                        script.sh
 Downloads                        snap
 exerc2.sh                        Templates
 lista_2025-05-20_11-06-00.txt    testebccomplexo.sh
 lista_2025-05-20_11-06-01.txt    testebc.sh
 lista_2025-05-20_11-06-02.txt    testecrases.sh
 log.2005251113                   teste.txt
 log.2005251114                   testevariaveisambiente.sh
 log.2005251116                   testevariaveis.sh
 Music                           'Trabalho Vetores.cpp'
 Pictures                         Videos
2025.1.08.006@suporte-OptiPlex-3050:~$ echo $?
0
2025.1.08.006@suporte-OptiPlex-3050:~$ comandoInvalido
comandoInvalido: command not found
2025.1.08.006@suporte-OptiPlex-3050:~$ echo $?
127
```

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 permite especificar um status de saída quando o script finaliza.

```
#!/bin/bash
# Exercicio 6
echo " Caso 1, o script será encerrado com status 42."
exit 42
```

```
2025.1.08.006@suporte-OptiPlex-3050:~$ gedit exerc6.sh
2025.1.08.006@suporte-OptiPlex-3050:~$ chmod a+x exerc6.sh
2025.1.08.006@suporte-OptiPlex-3050:~$ ./exerc6.sh
 Caso 1, o script será encerrado com status 42.
2025.1.08.006@suporte-OptiPlex-3050:~$ echo $?
42
```

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.

```
#!/bin/bash
# Exercicio 7
if [ -d "$HOME/naoexiste" ]; then
echo "Diretorio encontrado."
else
echo "Diretorio nao encontrado."
fi
if [ -d "$HOME/Documents" ];then
echo "Diretorio Documents encontrado."
else
echo "Diretorio Documents não encontrado."
fi
```

```
2025.1.08.006@suporte-OptiPlex-3050:~$ gedit exerc7.sh
2025.1.08.006@suporte-OptiPlex-3050:~$ chmod a+x exerc7.sh
2025.1.08.006@suporte-OptiPlex-3050:~$ ./exerc7.sh
Diretorio nao encontrado.
Diretorio Documents encontrado.
2025.1.08.006@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.

```bash
#!/bin/bash
a=8
b=5
res1=$((a + b))
res2=$((a * b))
if [ $res1 -gt $res2 ]; then
echo "res1 é maior que res2"
elif [ $res1 -lt $res2 ]; then
echo "res1 é menor que res2"
else
echo "res1 é igual a res2"
fi
```

```
2025.1.08.006@suporte-OptiPlex-3050:~$ gedit exerc8.sh
2025.1.08.006@suporte-OptiPlex-3050:~$ chmod 755 exerc8.sh
2025.1.08.006@suporte-OptiPlex-3050:~$ ./exerc8.sh
res1 é menor que res2
2025.1.08.006@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.

```bash
#!/bin/bash
#exercicio 9
str1="maçã"
str2="maçã"

if [ "str1"="str2" ]; then
echo "As strings são iguais."
else
echo "As strings são diferentes"
fi
```

```
2025.1.08.006@suporte-OptiPlex-3050:~$ gedit exerc9.sh
2025.1.08.006@suporte-OptiPlex-3050:~$ chmod a+x exerc9.sh
2025.1.08.006@suporte-OptiPlex-3050:~$ ./exerc9.sh
As strings são iguais.
2025.1.08.006@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.

```bash
#!/bin/bash
# eXercício 10
palavra="fruta"
if [ "$palavra" = "fruta" ]; then
echo " É fruta."
else
echo "Não é fruta."
fi
```

```
2025.1.08.006@suporte-OptiPlex-3050:~$ gedit exerc10.sh
2025.1.08.006@suporte-OptiPlex-3050:~$ chmod a+x exerc10.sh
2025.1.08.006@suporte-OptiPlex-3050:~$ ./exerc10.sh
 É fruta.
2025.1.08.006@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).

```
#!/bin/bash
# Exercicio11

var1="banana"
var2=""
if [ -n "$var1" ]; then
echo "var1 tem conteúdo: $var1"
else
echo "var1 está vazia"
fi
if [ -n "$var2" ]; then
echo "var2 tem conteúdo"
else
echo "var2 está vazia"
fi
```

```
2025.1.08.006@suporte-OptiPlex-3050:~$ gedit exerc11.sh
2025.1.08.006@suporte-OptiPlex-3050:~$ chmod a+x exerc11.sh
2025.1.08.006@suporte-OptiPlex-3050:~$ ./exerc11.sh
var1 tem conteúdo: banana
var2 está vazia
2025.1.08.006@suporte-OptiPlex-3050:~$
```

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

-e: verifica se o arquivo existe; -f: verifica se é um arquivo comum; -d: verifica se é um diretório; -r: tem permissão de leitura; -s: arquivo não está vazio

```
#!/bin/bash
#exercicio 12
arq="/etc/passwd"
if [ -f "$arq" ]; then
echo "O arquivo $arq existe e é um arquivo comum."
else
echo "O arquivo $arq não existe ou não é um arquivo comum."
fi
```

```
2025.1.08.006@suporte-OptiPlex-3050:~$ gedit exerc12.sh
2025.1.08.006@suporte-OptiPlex-3050:~$ chmod a+x exerc12.sh
2025.1.08.006@suporte-OptiPlex-3050:~$ ./exerc12.sh
O arquivo /etc/passwd existe e é um arquivo comum.
2025.1.08.006@suporte-OptiPlex-3050:~$
```