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: Pedro Ferreira Prado RA: 2025.1.08.028

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?

Uso do utilitário bc com scale para permitir números com ponto flutuante (decimais), pois o Bash só faz contas com inteiros por padrão.

```
  GNU nano 6.2                        scriptaritmetico.sh *
#!/bin/bash

a=10
b=3
c=1
d=1

#((a+c)/(b+d))

resultado=$(echo "scale=2; ($a+$c)/($b+$d)" | bc)
echo "Resultado da divisao: $resultado"
```

```
2025.1.08.028@suporte-OptiPlex-3050:~$ nano scriptaritmetico.sh
2025.1.08.028@suporte-OptiPlex-3050:~$ ./scriptaritmetico.sh
Resultado da divisao: 2.75
```

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

```
  GNU nano 6.2                          calculadora.sh
#!/bin/bash

echo "scale=6; 10/3" | bc
```

```
2025.1.08.028@suporte-OptiPlex-3050:~$ nano calculadora.sh
2025.1.08.028@suporte-OptiPlex-3050:~$ chmod +x calculadora.sh
2025.1.08.028@suporte-OptiPlex-3050:~$ ./calculadora.sh
3.333333
```

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.

```
  GNU nano 6.2                          testebc
#!/bin/bash

num1=22
num2=7
precisao=6

echo "scale=$precisao;$num1/$num2" | bc
```

```
2025.1.08.028@suporte-OptiPlex-3050:~$ chmod +x testebc
2025.1.08.028@suporte-OptiPlex-3050:~$ ./testebc
3.142857
```

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.

```
  GNU nano 6.2                       testebccomplexo
#!/bin/bash

resultado1=$(bc << EOF
scale=4
a=12.3450
b=7.8910
a+b
EOF
)

resultado2=$(bc << EOF
scale=6
x=15
y=7
x/y
EOF
)

echo "Resultado da soma com 4 casas decimais: $resultado1"
echo "Resultado da soma com 6 casas decimais: $resultado2"
```

```
2025.1.08.028@suporte-OptiPlex-3050:~$ nano testebccomplexo
2025.1.08.028@suporte-OptiPlex-3050:~$ ./testebccomplexo
Resultado da soma com 4 casas decimais: 20.2360
Resultado da soma com 6 casas decimais: 2.142857
```

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.

```
  GNU nano 6.2                          statussaida.sh
#!/bin/bash

ls > /dev/null
echo "Status do comando 'ls': $?"

comandonaoexiste > /dev/null 2>&1
echo "Status do comando 'comandonaoexiste': $?"
```

```
2025.1.08.028@suporte-OptiPlex-3050:~$ nano statussaida.sh
2025.1.08.028@suporte-OptiPlex-3050:~$ chmod +x statussaida.sh
2025.1.08.028@suporte-OptiPlex-3050:~$ ./statussaida.sh
Status do comando 'ls': 0
Status do comando 'comandonaoexiste': 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.

```
echo "Executando"


exit 42

echo "Isso não será mostrado"
```

```
2025.1.08.028@suporte-OptiPlex-3050:~$ nano exit_example.sh
2025.1.08.028@suporte-OptiPlex-3050:~$ chmod +x exit_example.sh
2025.1.08.028@suporte-OptiPlex-3050:~$ ./exit_example.sh
Executando
2025.1.08.028@suporte-OptiPlex-3050:~$
```

```
  GNU nano 6.2                          exit_example.sh
#!/bin/bash

status=100
echo "Saindo com status armazenado: $status"
exit $status
```

```
2025.1.08.028@suporte-OptiPlex-3050:~$ nano exit_example.sh
2025.1.08.028@suporte-OptiPlex-3050:~$ ./exit_example.sh
Saindo com status armazenado: 100
2025.1.08.028@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.

```
  GNU nano 6.2                    verifica_diretorio.sh
#!/bin/bash


dir_inexistente="$HOME/dir_que_nao_existe"
dir_existente="$HOME/Documents"


if [ -d "$dir_inexistente" ]; then
  echo "Diretório $dir_inexistente existe."
else
  echo "Diretório $dir_inexistente NÃO existe."
fi


if [ -d "$dir_existente" ]; then
  echo "Diretório $dir_existente existe."
else
  echo "Diretório $dir_existente NÃO existe."
fi
```

```
2025.1.08.028@suporte-OptiPlex-3050:~$ chmod +x verifica_diretorio.sh
2025.1.08.028@suporte-OptiPlex-3050:~$ ./verifica_diretorio.sh
Diretório /home/2025.1.08.028/dir_que_nao_existe NÃO existe.
Diretório /home/2025.1.08.028/Documents existe.
2025.1.08.028@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.

```
  GNU nano 6.2                    compara_operacoes.sh
#!/bin/bash

resultado1=$(echo "scale=2; 15 * 3.5" | bc)
resultado2=$(echo "scale=2; 50 / 2" | bc)

echo "Resultado da operação 1 (15 * 3.5): $resultado1"
echo "Resultado da operação 2 (50 / 2): $resultado2"


if (( $(echo "$resultado1 > $resultado2" | bc -l) )); then
  echo "O resultado 1 é maior que o resultado 2."
elif (( $(echo "$resultado1 < $resultado2" | bc -l) )); then
  echo "O resultado 1 é menor que o resultado 2."
else
  echo "O resultado 1 é igual ao resultado 2."
fi
```

```
2025.1.08.028@suporte-OptiPlex-3050:~$ nano compara_operacoes.sh
2025.1.08.028@suporte-OptiPlex-3050:~$ chmod +x compara_operacoes.sh
2025.1.08.028@suporte-OptiPlex-3050:~$ ./compara_operacoes.sh
Resultado da operação 1 (15 * 3.5): 52.5
Resultado da operação 2 (50 / 2): 25.00
O resultado 1 é maior que o resultado 2.
```

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.

```
  GNU nano 6.2                        compara_strings.sh *
#!/bin/bash

str1="OpenAI"
str2="OpenAI"


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

```
2025.1.08.028@suporte-OptiPlex-3050:~$ chmod +x compara_strings.sh
2025.1.08.028@suporte-OptiPlex-3050:~$ ./compara_strings.sh
As strings são iguais.
2025.1.08.028@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.

```
  GNU nano 6.2                        verifica_fruta.sh *
#!/bin/bash


minha_string="fruta"


if [ "$minha_string" = "fruta" ]; then
  echo "A string é 'fruta'."
else
  echo "A string NÃO é 'fruta'."
fi
```

```
2025.1.08.028@suporte-OptiPlex-3050:~$ nano verifica_fruta.sh
2025.1.08.028@suporte-OptiPlex-3050:~$ chmod +x verifica_fruta.sh
2025.1.08.028@suporte-OptiPlex-3050:~$ ./verifica_fruta.sh
A string é 'fruta'.
```

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).

```
  GNU nano 6.2                          verificar.sh *
#!/bin/bash


str_vazia=""
str_com_conteudo="Conteúdo aqui"


if [ -z "$str_vazia" ]; then
  echo "A string vazia está realmente vazia."
else
  echo "A string vazia tem conteúdo."
fi

if [ -z "$str_com_conteudo" ]; then
  echo "A string com conteúdo está vazia."
else
  echo "A string com conteúdo NÃO está vazia."
fi
```

```
2025.1.08.028@suporte-OptiPlex-3050:~$ nano verificar.sh
2025.1.08.028@suporte-OptiPlex-3050:~$ chmod +x verificar.sh
2025.1.08.028@suporte-OptiPlex-3050:~$ ./verificar.sh
A string vazia está realmente vazia.
A string com conteúdo NÃO está vazia.
```

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

e arquivo — verifica se o arquivo existe.

f arquivo — verifica se o arquivo existe e é um arquivo comum (não diretório).

d arquivo — verifica se o arquivo existe e é um diretório.

r arquivo — verifica se o arquivo existe e é legível.

w arquivo — verifica se o arquivo existe e é gravável.

```
  GNU nano 6.2                    verifica_leitura.sh *
#!/bin/bash

arquivo="$HOME/teste.txt"

if [ -r "$arquivo" ]; then
  echo "O arquivo $arquivo existe e é legível."
else
  echo "O arquivo $arquivo NÃO existe ou NÃO é legível."
fi
```

```
2025.1.08.028@suporte-OptiPlex-3050:~$ chmod +x  verifica_leitura.sh
2025.1.08.028@suporte-OptiPlex-3050:~$ ./verifica_leitura.sh
O arquivo /home/2025.1.08.028/teste.txt NÃO existe ou NÃO é legível.
```