



## Introdução à Ciência da Computação – Lista 6

### Shell script – parte 3

Nome: Gabriel Ribeiro

RA: 2024.1.08.010

- 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  -  +  x
1 #!/bin/bash
2 var1=14
3 var2=9
4 var3=8
5 var4=`bc << EXP
6 scale=2
7 a=($var1 * $var3)
8 a / $var2
9 EXP
10 `
11 echo "Resultado: $var4"
```

```
2024.1.08.010@suporte-OptiPlex-3050: ~
2024.1.08.010@suporte-OptiPlex-3050:~$ gedit scriptaritmetico.sh
2024.1.08.010@suporte-OptiPlex-3050:~$ chmod a+x scriptaritmetico.sh
2024.1.08.010@suporte-OptiPlex-3050:~$ ./scriptaritmetico.sh
Resultado: 12.44
2024.1.08.010@suporte-OptiPlex-3050:~$
```

Para mostrar resultados não inteiros é necessário usar a calculadora do bash, o bc. No comando bc tem uma variável especial chamada scale, que controla a quantidade de casas depois da vírgula mostradas.

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

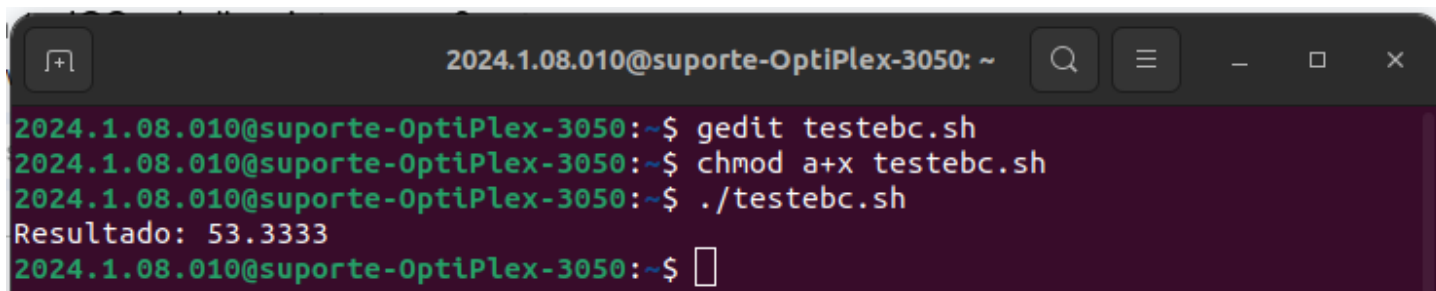
```
2024.1.08.010@suporte-OptiPlex-3050: ~
2024.1.08.010@suporte-OptiPlex-3050:~$ bc -q
scale=6
7 / 3
2.333333
quit
2024.1.08.010@suporte-OptiPlex-3050:~$
```

- 3) Crie um script simples chamado testeabc, 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.

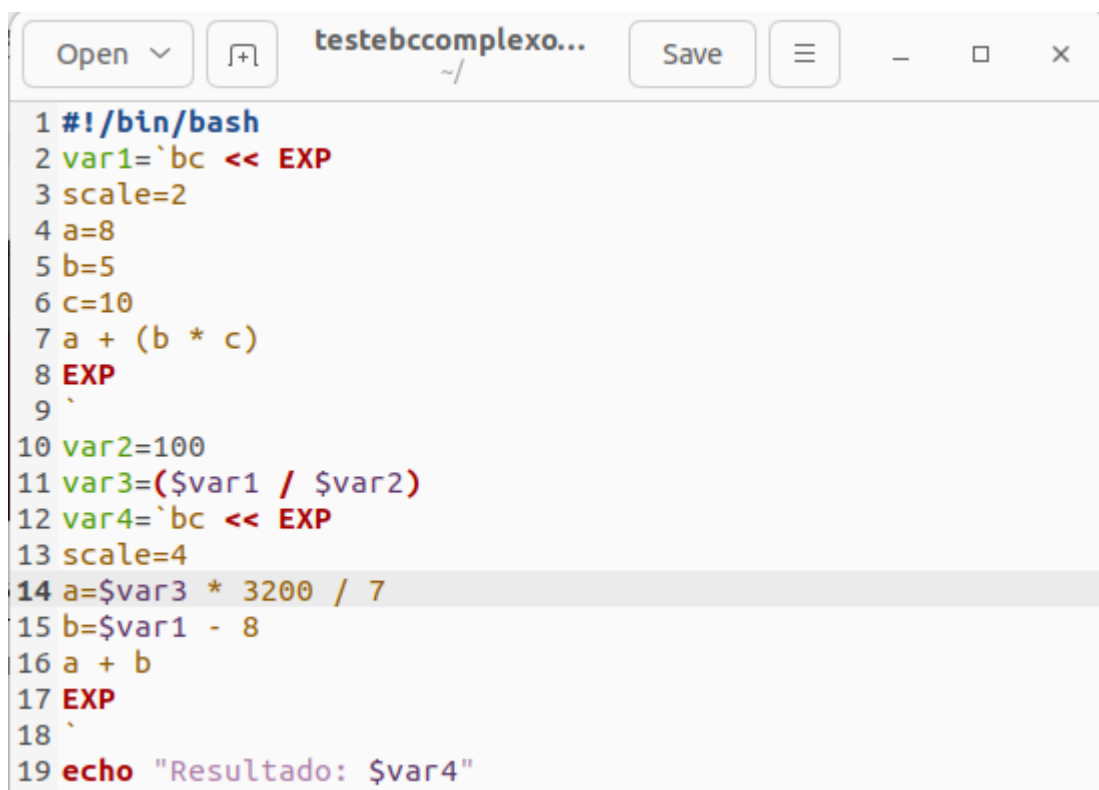


```
1 #!/bin/bash
2 var1=10
3 var2=16
4 var3=3
5 var4=`echo "scale=4; $var1 * $var2 / $var3" | bc`
6 echo "Resultado: $var4"
```



```
2024.1.08.010@suporte-OptiPlex-3050: ~
2024.1.08.010@suporte-OptiPlex-3050:~$ gedit testebc.sh
2024.1.08.010@suporte-OptiPlex-3050:~$ chmod a+x testebc.sh
2024.1.08.010@suporte-OptiPlex-3050:~$ ./testebc.sh
Resultado: 53.3333
2024.1.08.010@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=`bc << EXP
3 scale=2
4 a=8
5 b=5
6 c=10
7 a + (b * c)
8 EXP
9 `
10 var2=100
11 var3=($var1 / $var2)
12 var4=`bc << EXP
13 scale=4
14 a=$var3 * 3200 / 7
15 b=$var1 - 8
16 a + b
17 EXP
18 `
19 echo "Resultado: $var4"
```

```
2024.1.08.010@suporte-OptiPlex-3050: ~  
2024.1.08.010@suporte-OptiPlex-3050:~$ gedit testebccomplexo.sh  
2024.1.08.010@suporte-OptiPlex-3050:~$ chmod a+x testebccomplexo.sh  
2024.1.08.010@suporte-OptiPlex-3050:~$ ./testebccomplexo.sh  
Resultado: 92850.0000  
2024.1.08.010@suporte-OptiPlex-3050:~$ gedit testebccomplexo.sh  
2024.1.08.010@suporte-OptiPlex-3050:~$ chmod a+x testebccomplexo.sh  
2024.1.08.010@suporte-OptiPlex-3050:~$ ./testebccomplexo.sh  
Resultado: 26564.2857  
2024.1.08.010@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 que rodam no shell usam um valor de status para indicar que o processamento terminou.

```
2024.1.08.010@suporte-OptiPlex-3050: ~  
2024.1.08.010@suporte-OptiPlex-3050:~$ pwd  
/home/2024.1.08.010  
2024.1.08.010@suporte-OptiPlex-3050:~$ echo $?  
0  
2024.1.08.010@suporte-OptiPlex-3050:~$ abrir  
abrir: command not found  
2024.1.08.010@suporte-OptiPlex-3050:~$ echo $?  
127  
2024.1.08.010@suporte-OptiPlex-3050:~$
```

- 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 serve para alterar o comportamento do status de saída dos comandos, retornando assim um código de status personalizado.

```
Open  ▾  testebccomplexo...  Save  ▮  -  □  ×

1 #!/bin/bash
2 var1=`bc << EXP
3 scale=2
4 a=8
5 b=5
6 c=10
7 a + (b * c)
8 EXP
9 `
10 var2=100
11 var3=($var1 / $var2)
12 var4=`bc << EXP
13 scale=4
14 a=$var3 * 3200 / 7
15 b=$var1 - 8
16 a + b
17 EXP
18 `
19 echo "Resultado: $var4"
20 exit 14
```

```
2024.1.08.010@suporte-OptiPlex-3050: ~  🔍  ▮  -  □  ×

2024.1.08.010@suporte-OptiPlex-3050:~$ gedit testebccomplexo.sh
2024.1.08.010@suporte-OptiPlex-3050:~$ chmod a+x testebccomplexo.sh
2024.1.08.010@suporte-OptiPlex-3050:~$ ./testebccomplexo.sh
Resultado: 26564.2857
2024.1.08.010@suporte-OptiPlex-3050:~$ echo $?
14
2024.1.08.010@suporte-OptiPlex-3050:~$ █
```

```
Open  ▾  testebc.sh  Save  ▮  -  □  ×

1 #!/bin/bash
2 var1=10
3 var2=16
4 var3=3
5 var4=`echo "scale=4; $var1 * $var2 / $var3" | bc`
6 echo "Resultado: $var4"
7 exit $var1
```

```
2024.1.08.010@suporte-OptiPlex-3050: ~  🔍  ▮  -  □  ×

Resultado: 26564.2857
2024.1.08.010@suporte-OptiPlex-3050:~$ gedit testebc.sh
2024.1.08.010@suporte-OptiPlex-3050:~$ chmod a+x testebc.sh
2024.1.08.010@suporte-OptiPlex-3050:~$ ./testebc.sh
Resultado: 53.3333
2024.1.08.010@suporte-OptiPlex-3050:~$ echo $?
10
2024.1.08.010@suporte-OptiPlex-3050:~$ █
```

- 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.

```
2024.1.08.010@suporte-OptiPlex-3050: ~  
2024.1.08.010@suporte-OptiPlex-3050:~$ gedit procuradiretorio.sh  
2024.1.08.010@suporte-OptiPlex-3050:~$ chmod a+x procuradiretorio.sh  
2024.1.08.010@suporte-OptiPlex-3050:~$ ./procuradiretorio.sh  
ls: cannot access '/home/Backup/Gabriel': No such file or directory  
Diretório não encontrado  
2024.1.08.010@suporte-OptiPlex-3050:~$
```

```
2024.1.08.010@suporte-OptiPlex-3050: ~  
2024.1.08.010@suporte-OptiPlex-3050:~$ gedit procuradiretorio.sh  
2024.1.08.010@suporte-OptiPlex-3050:~$ chmod a+x procuradiretorio.sh  
2024.1.08.010@suporte-OptiPlex-3050:~$ ./procuradiretorio.sh  
Screenshots  
Diretório encontrado  
2024.1.08.010@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.

```
Open  estruturaisifelse.sh  Save  
1 #!/bin/bash  
2 var1=40 * 12 + 43  
3 var2=60 * 1000 / 100  
4  
5 if [$var1 -gt $var2]  
6 then  
7     echo "A primeira variável é maior que a segunda"  
8 elif [$var1 -lt $var2]  
9 then  
10    echo "A primeira variável é menor que a segunda"  
11 else  
12    echo "A primeira variável é igual a segunda"  
13 fi
```

```
2024.1.08.010@suporte-OptiPlex-3050: ~
2024.1.08.010@suporte-OptiPlex-3050:~$ gedit estruturaifelse.sh
2024.1.08.010@suporte-OptiPlex-3050:~$ chmod a+x estruturaifelse.sh
2024.1.08.010@suporte-OptiPlex-3050:~$ ./estruturaifelse.sh
./estruturaifelse.sh: line 2: Atividade 5 - Gabriel Ribeiro: command not found
./estruturaifelse.sh: line 3: Atividade 5 - Gabriel Ribeiro: command not found
A primeira variável é maior que a segunda
2024.1.08.010@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.

```
Open  ifelsestring.sh  Save
1 #!/bin/bash
2 var1="Mundo"
3 var2="Mundo"
4
5 if [ $var1 == $var2 ]
6 then
7     echo "O conteúdo das variáveis são iguais."
8 else
9     echo "O conteúdo das variáveis não são iguais."
10 fi
```

```
2024.1.08.010@suporte-OptiPlex-3050: ~
2024.1.08.010@suporte-OptiPlex-3050:~$ gedit ifelsestring.sh
2024.1.08.010@suporte-OptiPlex-3050:~$ chmod a+x ifelsestring.sh
2024.1.08.010@suporte-OptiPlex-3050:~$ ./ifelsestring.sh
O conteúdo das variáveis são iguais.
2024.1.08.010@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  checarfruta.sh  Save
1 #!/bin/bash
2 var=fruta
3 if [ $var == fruta ]
4 then
5     echo "$var"
6 else
7     echo "Não é fruta, é $var"
8 fi
```



```

2024.1.08.010@suporte-OptiPlex-3050:~$ gedit checarfruta.sh
2024.1.08.010@suporte-OptiPlex-3050:~$ chmod a+x checarfruta.sh
2024.1.08.010@suporte-OptiPlex-3050:~$ ./checarfruta.sh
fruta
2024.1.08.010@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).



```

1 #!/bin/bash
2 string1=''
3 string2=Bom_dia!
4 if [ -z $string1 ]
5 then
6     echo "A variável 1 está vazia"
7 else
8     echo "A variável contém o valor $string1"
9 fi
10
11 if [ -n $string2 ]
12 then
13     echo "A variável contém o valor $string2"
14 else
15     echo "A variável 2 está vazia"
16 fi

```

```

2024.1.08.010@suporte-OptiPlex-3050:~$ gedit conteudostring.sh
2024.1.08.010@suporte-OptiPlex-3050:~$ chmod a+x conteudostring.sh
2024.1.08.010@suporte-OptiPlex-3050:~$ ./conteudostring.sh
A variável 1 está vazia
A variável contém o valor Bom_dia!
2024.1.08.010@suporte-OptiPlex-3050:~$

```

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

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

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

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

-O arquivo: Verifica se o arquivo existe e é propriedade do usuário atual.

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

```
Open  tetse.sh  Save  ~/  
1 #!/bin/bash  
2 arquivo=/intcc/arquivos/agenda.txt  
3  
4 if [ -f $arquivo ]; then  
5     echo "O arquivo existe"  
6 else  
7     echo "O arquivo não existe"  
8 fi
```

```
2024.1.08.010@suporte-OptiPlex-3050: ~  
2024.1.08.010@suporte-OptiPlex-3050:~$ gedit tetse.sh  
2024.1.08.010@suporte-OptiPlex-3050:~$ chmod a+x tetse.sh  
2024.1.08.010@suporte-OptiPlex-3050:~$ ./tetse.sh  
O arquivo não existe  
2024.1.08.010@suporte-OptiPlex-3050:~$
```