

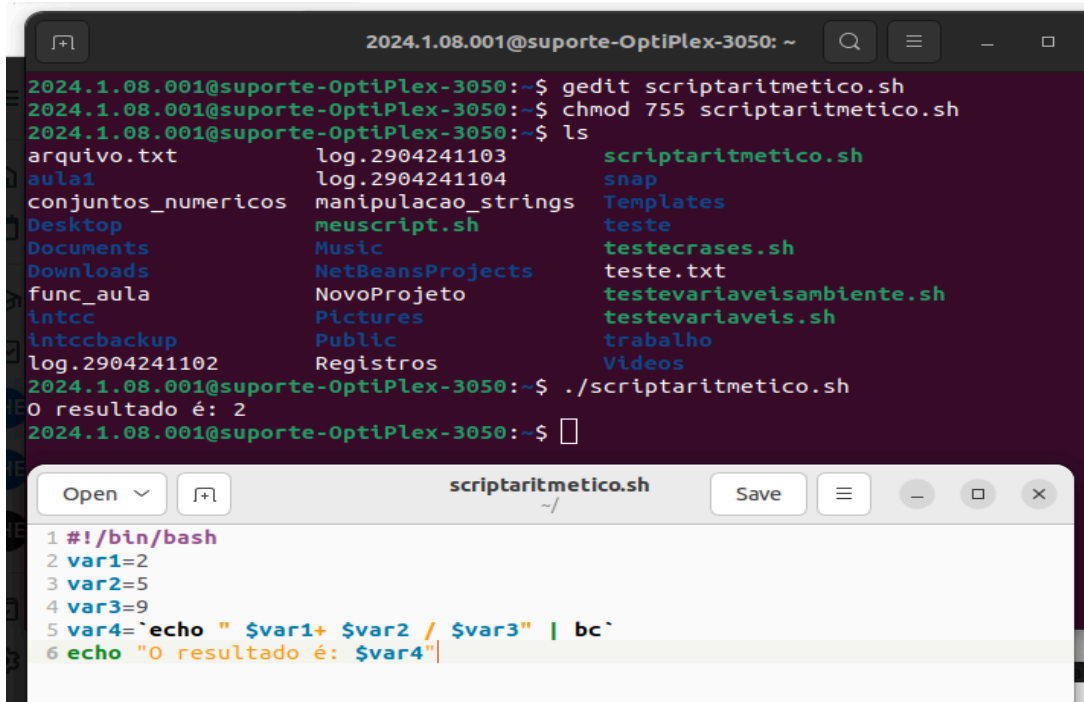
Introdução à Ciência da Computação – Lista 6

Shell script – parte 3

Nome: Ana Flávia Freiria Rodrigues

RA: 2024.1.08.001

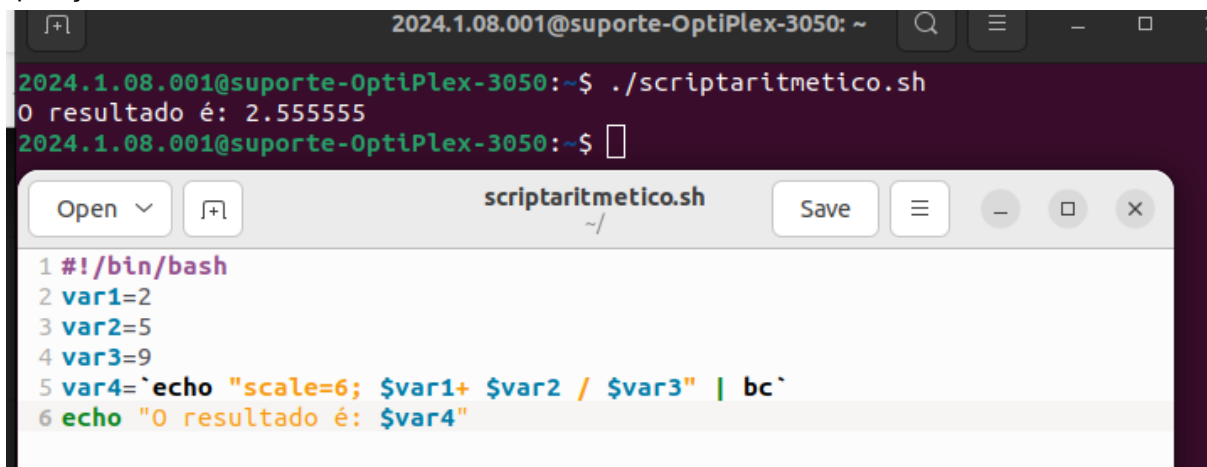
1) Crie um script chamado scriptaritmetico, com uma operação aritmética arbitrária usando pelo menos 4 variáveis, realizando uma operação de divisão cujo resultado não seja um número inteiro. Execute o script e mostre o resultado. Qual o recurso a ser utilizado caso você queira que o valor não inteiro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



```
2024.1.08.001@suporte-OptiPlex-3050: ~  
2024.1.08.001@suporte-OptiPlex-3050:~$ gedit scriptaritmetico.sh  
2024.1.08.001@suporte-OptiPlex-3050:~$ chmod 755 scriptaritmetico.sh  
2024.1.08.001@suporte-OptiPlex-3050:~$ ls  
arquivo.txt          log.2904241103      scriptaritmetico.sh  
aula1                log.2904241104      snap  
conjuntos_numericos manipulacao_strings  Templates  
Desktop              meuscript.sh         teste  
Documents            Music                testecrases.sh  
Downloads            NetBeansProjects    teste.txt  
func_aula            NovoProjeto          testevariaveisambiente.sh  
intcc                Pictures             testevariaveis.sh  
intccbackup          Public               trabalho  
log.2904241102       Registros            Videos  
2024.1.08.001@suporte-OptiPlex-3050:~$ ./scriptaritmetico.sh  
O resultado é: 2  
2024.1.08.001@suporte-OptiPlex-3050:~$  
  
scriptaritmetico.sh  
1 #!/bin/bash  
2 var1=2  
3 var2=5  
4 var3=9  
5 var4=`echo " $var1+ $var2 / $var3" | bc`  
6 echo "O resultado é: $var4"
```

Para que o valor fracionário apareça no resultado, devo usar o recurso “scale”.

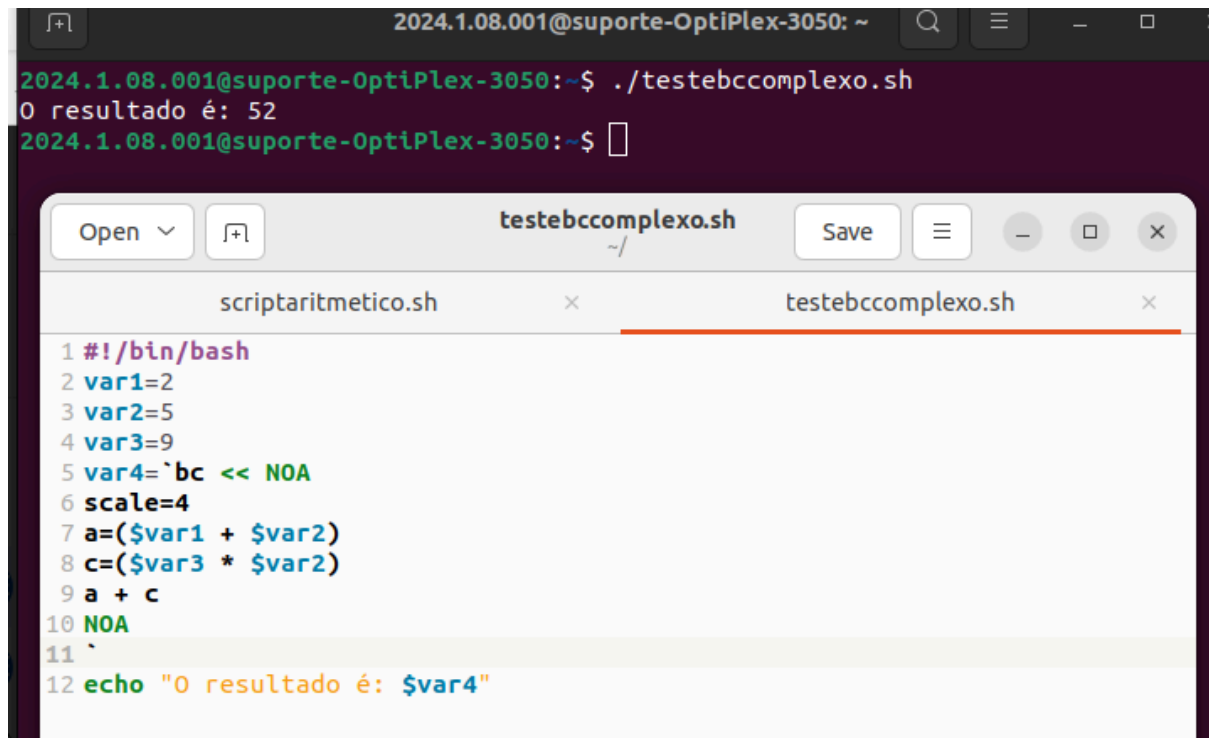
2) Ponha em execução a calculadora bc. Mostre o uso da variável scale, exibindo um resultado de operação aritmética com 6 casas decimais.



```
2024.1.08.001@suporte-OptiPlex-3050: ~  
2024.1.08.001@suporte-OptiPlex-3050:~$ ./scriptaritmetico.sh  
O resultado é: 2.555555  
2024.1.08.001@suporte-OptiPlex-3050:~$  
  
scriptaritmetico.sh  
1 #!/bin/bash  
2 var1=2  
3 var2=5  
4 var3=9  
5 var4=`echo "scale=6; $var1+ $var2 / $var3" | bc`  
6 echo "O resultado é: $var4"
```

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.

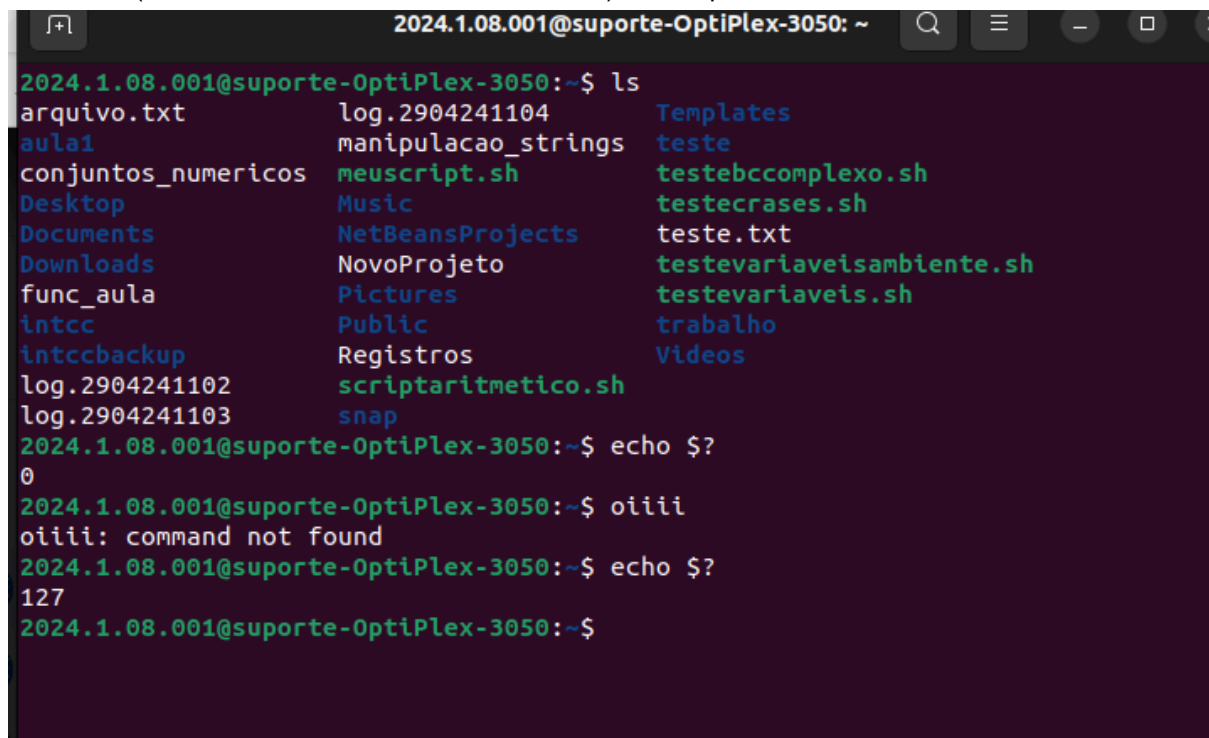
4) Crie um script chamado `testebccomplexo`, em que você utilize operações aritméticas diversas com a calculadora `bc` (pelo menos duas), armazenando os resultados em variáveis, como mostrado na aula. Neste caso, utilize a técnica de redirecionamento de entrada inline. Execute o script, mostrando o resultado.



The image shows a terminal window and a script editor. The terminal window displays the command `./testebccomplexo.sh` and its output: `0 resultado é: 52`. The script editor shows the contents of `testebccomplexo.sh`, which is a bash script that uses the `bc` calculator to perform arithmetic operations on variables `var1`, `var2`, and `var3`.

```
2024.1.08.001@suporte-OptiPlex-3050: ~  
2024.1.08.001@suporte-OptiPlex-3050:~$ ./testebccomplexo.sh  
0 resultado é: 52  
2024.1.08.001@suporte-OptiPlex-3050:~$  
  
testebccomplexo.sh  
1 #!/bin/bash  
2 var1=2  
3 var2=5  
4 var3=9  
5 var4=`bc << NOA  
6 scale=4  
7 a=($var1 + $var2)  
8 c=($var3 * $var2)  
9 a + c  
10 NOA  
11 `  
12 echo "0 resultado é: $var4"
```

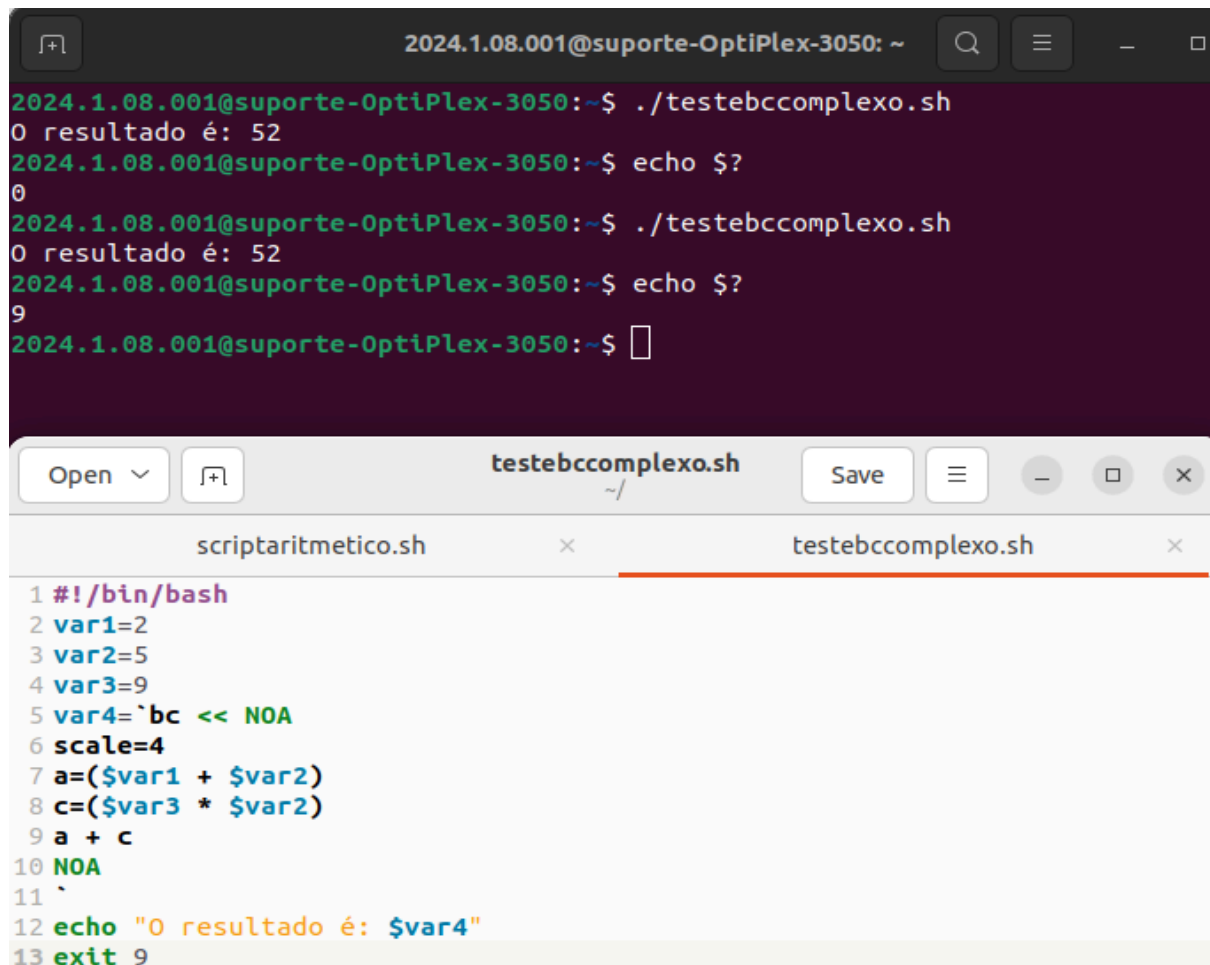
5) O que consiste o status de saída de um programa? Mostre um exemplo de execução de dois comandos (um com sucesso e outro desconhecido) e verifique esse status. Mostre em tela.



The image shows a terminal window where the user lists files in the current directory, then runs `echo $?` to check the status of the previous command, followed by `oiii` (which is not found) and another `echo $?` to check the status of the previous command.

```
2024.1.08.001@suporte-OptiPlex-3050: ~  
2024.1.08.001@suporte-OptiPlex-3050:~$ ls  
arquivo.txt      log.2904241104  Templates  
aula1            manipulacao_strings  teste  
conjuntos_numericos  meuscript.sh      testebccomplexo.sh  
Desktop          Music            testecrases.sh  
Documents        NetBeansProjects  teste.txt  
Downloads        NovoProjeto      testevariaveisambiente.sh  
func_aula        Pictures          testevariaveis.sh  
intcc            Public           trabalho  
intccbackup      Registros        Videos  
log.2904241102   scriptaritmetico.sh  
log.2904241103   snap  
2024.1.08.001@suporte-OptiPlex-3050:~$ echo $?  
0  
2024.1.08.001@suporte-OptiPlex-3050:~$ oiii  
oiii: command not found  
2024.1.08.001@suporte-OptiPlex-3050:~$ echo $?  
127  
2024.1.08.001@suporte-OptiPlex-3050:~$
```

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.

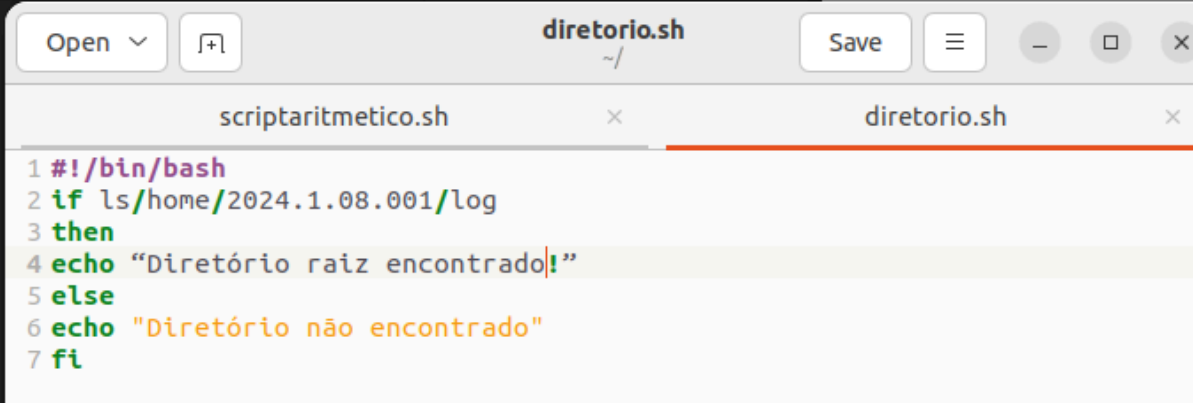


The image shows a terminal window and a code editor. The terminal window at the top displays the execution of a script named `testebccomplexo.sh`. It shows two runs: the first outputs `0 resultado é: 52` and the second outputs `9`. The code editor below shows the script's content, which uses `bc` for arithmetic and `scale=4` for precision. The script calculates `a = 2 + 5 = 7` and `c = 9 * 5 = 45`, then outputs `0 resultado é: 52` (likely a typo for 52 in the original image, but the output is 9).

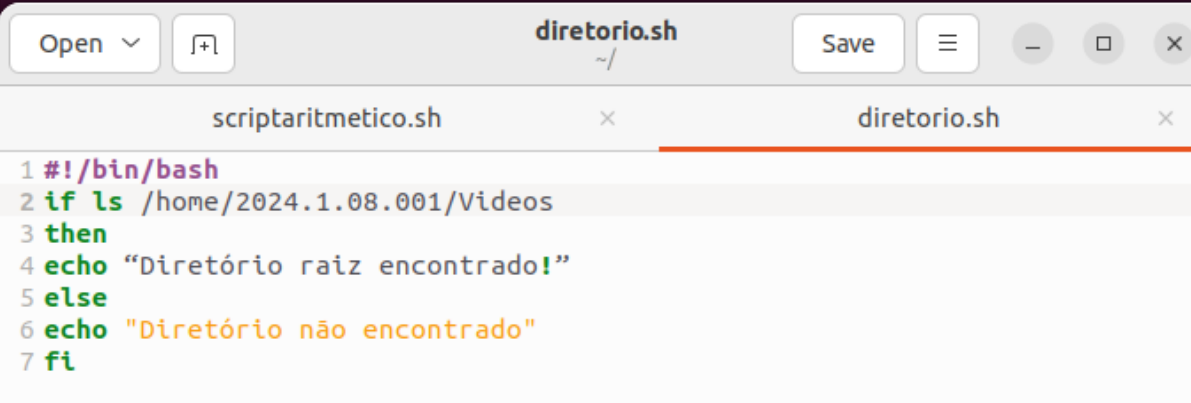
```
2024.1.08.001@suporte-OptiPlex-3050: ~  
2024.1.08.001@suporte-OptiPlex-3050:~$ ./testebccomplexo.sh  
0 resultado é: 52  
2024.1.08.001@suporte-OptiPlex-3050:~$ echo $?  
0  
2024.1.08.001@suporte-OptiPlex-3050:~$ ./testebccomplexo.sh  
0 resultado é: 52  
2024.1.08.001@suporte-OptiPlex-3050:~$ echo $?  
9  
2024.1.08.001@suporte-OptiPlex-3050:~$  
  
testebccomplexo.sh  
1 #!/bin/bash  
2 var1=2  
3 var2=5  
4 var3=9  
5 var4=`bc << NOA  
6 scale=4  
7 a=($var1 + $var2)  
8 c=($var3 * $var2)  
9 a + c  
10 NOA  
11 `  
12 echo "0 resultado é: $var4"  
13 exit 9
```

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.

```
2024.1.08.001@suporte-OptiPlex-3050:~$ echo $?
0
2024.1.08.001@suporte-OptiPlex-3050:~$ ./testebccomplexo.sh
0 resultado é: 52
2024.1.08.001@suporte-OptiPlex-3050:~$ echo $?
9
2024.1.08.001@suporte-OptiPlex-3050:~$ gedit diretorio.sh
2024.1.08.001@suporte-OptiPlex-3050:~$ chmod 755 diretorio.sh
2024.1.08.001@suporte-OptiPlex-3050:~$ ls
arquivo.txt          log.2904241103      snap
aula1                log.2904241104      Templates
conjuntos_numericos  manipulacao_strings teste
Desktop              meuscript.sh        testebccomplexo.sh
diretorio.sh          Music               testecrases.sh
Documents            NetBeansProjects   teste.txt
Downloads            NovoProjeto         testevariaveisambiente.sh
func_aula            Pictures            testevariaveis.sh
intcc                Public              trabalho
intccbackup          Registros           Videos
log.2904241102       scriptaritimetrico.sh
2024.1.08.001@suporte-OptiPlex-3050:~$ ./diretorio.sh
./diretorio.sh: line 2: ls/home/2024.1.08.001/log: No such file or directory
Diretório não encontrado
2024.1.08.001@suporte-OptiPlex-3050:~$
```

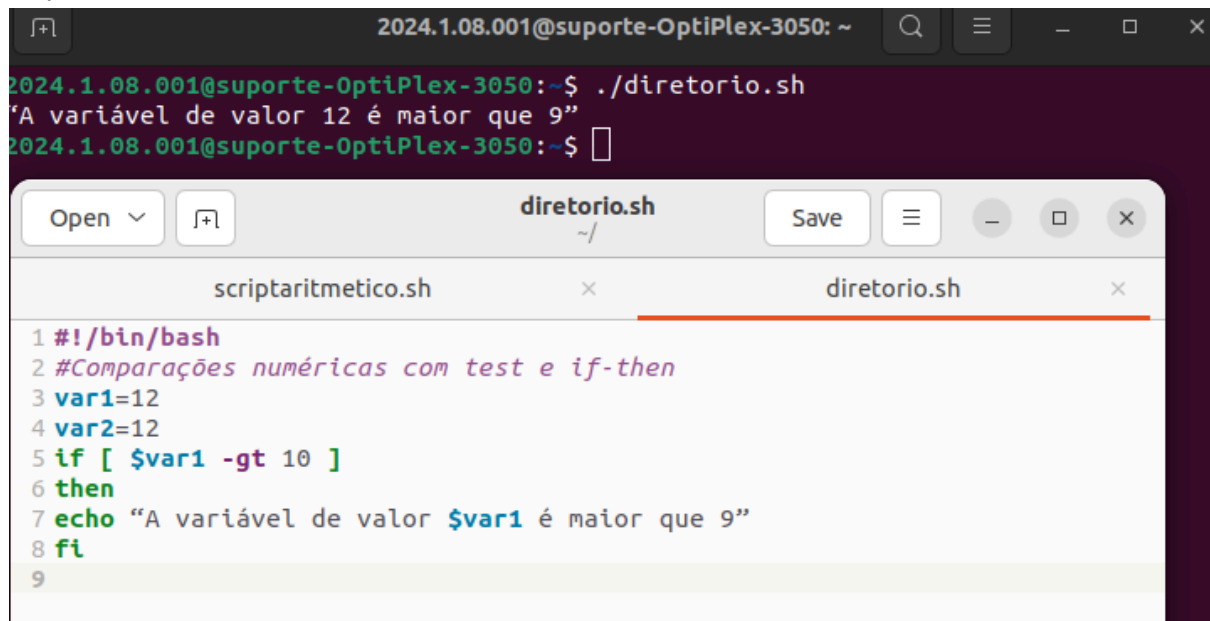


```
2024.1.08.001@suporte-OptiPlex-3050:~$ ./diretorio.sh
"Diretório raiz encontrado!"
2024.1.08.001@suporte-OptiPlex-3050:~$
```



8) Crie um script envolvendo várias condicionais usando a estrutura if then elif else, fazendo

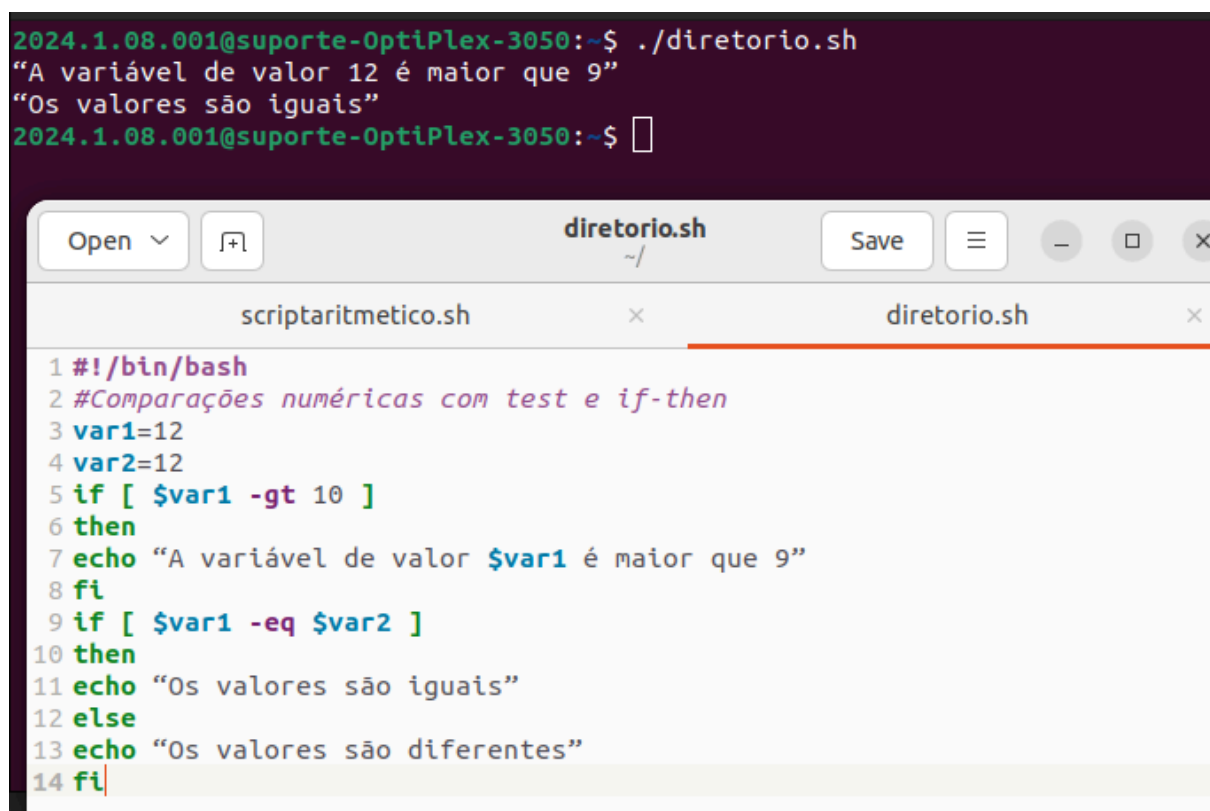
duas operações aritméticas arbitrárias, verificando o valor das variáveis que armazenam essa operação, checando se o valor da primeira é maior, menor ou igual ao valor da segunda. Execute o script e mostre o resultado em tela.



The image shows a terminal window and a code editor. The terminal window displays the command `./diretorio.sh` and its output: `'A variável de valor 12 é maior que 9'`. The code editor shows the contents of `diretorio.sh`, which is a bash script that sets `var1=12` and `var2=12`, and uses an `if` statement to check if `var1` is greater than 10. If true, it prints `'A variável de valor $var1 é maior que 9'`.

```
2024.1.08.001@suporte-OptiPlex-3050: ~  
2024.1.08.001@suporte-OptiPlex-3050:~$ ./diretorio.sh  
'A variável de valor 12 é maior que 9'  
2024.1.08.001@suporte-OptiPlex-3050:~$  
  
diretorio.sh  
1 #!/bin/bash  
2 #Comparações numéricas com test e if-then  
3 var1=12  
4 var2=12  
5 if [ $var1 -gt 10 ]  
6 then  
7 echo "A variável de valor $var1 é maior que 9"  
8 fi  
9
```

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.



The image shows a terminal window and a code editor. The terminal window displays the command `./diretorio.sh` and its output: `'A variável de valor 12 é maior que 9'` and `'Os valores são iguais'`. The code editor shows the contents of `diretorio.sh`, which is a bash script that sets `var1=12` and `var2=12`, and uses `if` statements to check if `var1` is greater than 10 and if `var1` is equal to `var2`. If both conditions are true, it prints `'Os valores são iguais'`.

```
2024.1.08.001@suporte-OptiPlex-3050:~$ ./diretorio.sh  
'A variável de valor 12 é maior que 9'  
'Os valores são iguais'  
2024.1.08.001@suporte-OptiPlex-3050:~$  
  
diretorio.sh  
1 #!/bin/bash  
2 #Comparações numéricas com test e if-then  
3 var1=12  
4 var2=12  
5 if [ $var1 -gt 10 ]  
6 then  
7 echo "A variável de valor $var1 é maior que 9"  
8 fi  
9 if [ $var1 -eq $var2 ]  
10 then  
11 echo "Os valores são iguais"  
12 else  
13 echo "Os valores são diferentes"  
14 fi
```

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.

The image consists of two screenshots of a terminal window and a code editor, illustrating the process of creating and debugging a shell script.

Top Screenshot:

The terminal window shows the following commands and output:

```
2024.1.08.001@suporte-OptiPlex-3050: ~  
2024.1.08.001@suporte-OptiPlex-3050:~$ gedit fruta.sh  
2024.1.08.001@suporte-OptiPlex-3050:~$ chmod 755 fruta.sh  
2024.1.08.001@suporte-OptiPlex-3050:~$ ./fruta.sh  
./fruta.sh: line 4: [: missing `']  
"A variavel não é fruta"  
2024.1.08.001@suporte-OptiPlex-3050:~$
```

The code editor shows the script `fruta.sh` with the following content:

```
1 #!/bin/bash  
2 #Testando diferença de strings  
3 var=casa  
4 if [ $var = fruta]  
5 then  
6 echo "A variavel é $var"  
7 else  
8 echo "A variavel não é $var"  
9 fi
```

Bottom Screenshot:

The terminal window shows the following commands and output:

```
2024.1.08.001@suporte-OptiPlex-3050: ~  
2024.1.08.001@suporte-OptiPlex-3050:~$  
2024.1.08.001@suporte-OptiPlex-3050:~$ ./fruta.sh  
./fruta.sh: line 4: [: missing `']  
"A variavel é fruta"  
2024.1.08.001@suporte-OptiPlex-3050:~$
```

The code editor shows the script `fruta.sh` with the following content:

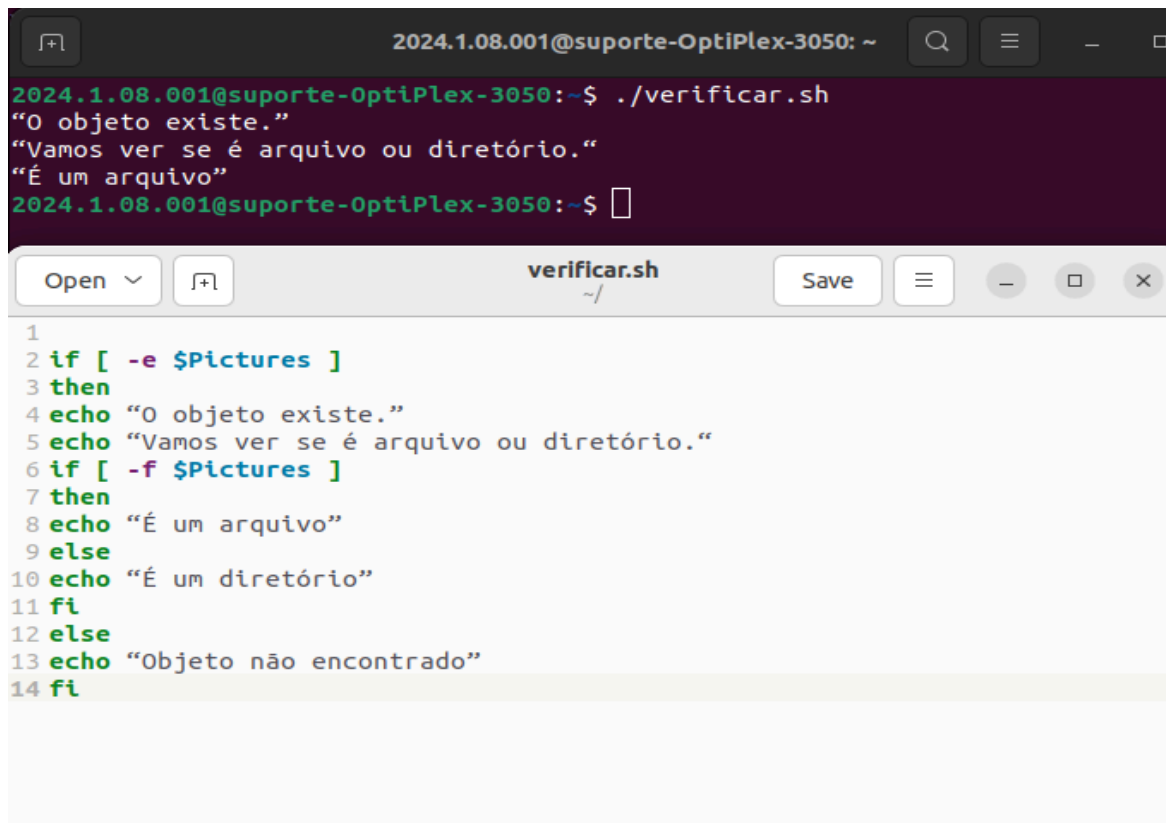
```
1 #!/bin/bash  
2 #Testando diferença de strings  
3 var=fruta  
4 if [ $var != fruta]  
5 then  
6 echo "A variavel nao é $var"  
7 else  
8 echo "A variavel é $var"  
9 fi
```

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).

```
2024.1.08.001@suporte-OptiPlex-3050: ~  
2024.1.08.001@suporte-OptiPlex-3050:~$ chmod 755 vazio.sh  
2024.1.08.001@suporte-OptiPlex-3050:~$ ./vazio.sh  
"A variável não está vazia, contém o valor mangueira"  
"Variável não está vazia"  
2024.1.08.001@suporte-OptiPlex-3050:~$  
  
vazio.sh  
Save  
fruta.sh x vazio.sh x  
1 #!/bin/bash  
2 #Testar se variável possui conteúdo  
3 var=mangueira  
4 var2='' #variável vazia aspas simples  
5 if [ -n $var ]  
6 then  
7 echo "A variável não está vazia, contém o valor $var"  
8 else  
9 echo "A variável está vazia"  
10 fi  
11  
12 if [ -z $var2 ]  
13 then  
14 echo "Variável está vazia"  
15 else  
16 echo "Variável não está vazia"  
17 fi
```

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)
- e arquivo (Verifica se o arquivo existe)
- f arquivo (verifica se é um arquivo)
- s arquivo (Verifica se o arquivo existe e não esta vazio)
- x arquivo (Verifica se o arquivo existe e tem permissão de execução)



The image displays a terminal window and a code editor. The terminal window at the top shows the execution of a script named `verificar.sh`. The prompt is `2024.1.08.001@suporte-OptiPlex-3050: ~`. The user enters `./verificar.sh`, and the script outputs three lines: `"O objeto existe."`, `"Vamos ver se é arquivo ou diretório."`, and `"É um arquivo"`. The code editor below shows the source code of `verificar.sh`, which is a shell script that checks if a file or directory exists and prints a message accordingly.

```
2024.1.08.001@suporte-OptiPlex-3050: ~  
2024.1.08.001@suporte-OptiPlex-3050:~$ ./verificar.sh  
"O objeto existe."  
"Vamos ver se é arquivo ou diretório."  
"É um arquivo"  
2024.1.08.001@suporte-OptiPlex-3050:~$  
  
1  
2 if [ -e $Pictures ]  
3 then  
4 echo "O objeto existe."  
5 echo "Vamos ver se é arquivo ou diretório."  
6 if [ -f $Pictures ]  
7 then  
8 echo "É um arquivo"  
9 else  
10 echo "É um diretório"  
11 fi  
12 else  
13 echo "Objeto não encontrado"  
14 fi
```