

Sistemas Operacionais Campus Anchieta

Prof. Dr. João Carlos Lopes Fernandes

Joao.fernandes1@docente.unip.br

2022 – 1



Material de apoio didático

A primeira e mais fácil maneira de fazer matemática básica no Linux CLI é usando parênteses duplos. Aqui estão alguns exemplos em que usamos valores armazenados em variáveis:

```
$ ADD=$(( 1 + 2 ))
```

```
$ echo $ADD
```

```
$ MUL=$(( $ADD * 5 ))
```

```
$ echo $MUL
```

```
$ SUB=$(( $MUL - 5 ))
```

```
$ echo $SUB
```

```
$ DIV=$(( $SUB / 2 ))
```

```
$ echo $DIV
```

```
$ MOD=$(( $DIV % 2 ))
```

```
$ echo $MOD
```

Usando o Comando expr

O comando `expr` avalia as expressões e imprime o valor da expressão fornecida na saída padrão. Veremos diferentes maneiras de usar `expr` para fazer matemática simples, fazer comparações, incrementar o valor de uma variável e encontrar o comprimento de uma string.

```
$ expr 3 + 5
```

```
$ expr 15 % 3
```

```
$ expr 5 \* 3
```

```
$ expr 5 - 3
```

```
$ expr 20 / 4
```

Quando uma expressão é avaliada como falsa, expr imprimirá um valor 0, caso contrário, imprimirá 1.

Vejam os alguns exemplos:

```
$ expr 5 = 3
```

```
$ expr 5 = 5
```

```
$ expr 8 != 5
```

```
$ expr 8 \> 5
```

```
$ expr 8 \< 5
```

```
$ expr 8 \<= 5
```

Você também pode usar o comando `expr` para incrementar o valor de uma variável. Dê uma olhada no exemplo a seguir (da mesma forma, você também pode diminuir o valor de uma variável).

```
$ NUM=$(( 1 + 2 ))
```

```
$ echo $NUM
```

```
$ NUM=$((expr $NUM + 2))
```

```
$ echo $NUM
```

Veamos também como encontrar o comprimento de uma string usando:

```
$ expr length "This is Unip.br"
```


Usando o comando bc

bc (Calculadora Básica) é um utilitário de linha de comando que fornece todos os recursos que você espera de uma calculadora científica ou financeira simples. É especialmente útil para fazer matemática de ponto flutuante.

Se o comando bc não estiver instalado, você pode instalá-lo usando:

```
$ sudo apt install bc      #Debian/Ubuntu
```

```
$ sudo yum install bc      #RHEL/CentOS
```

```
$ sudo dnf install bc      #Fedora 22+
```

Depois de instalado, você pode executá-lo no modo interativo ou não interativamente passando argumentos para ele - examinaremos os dois casos. Para executá-lo interativamente, digite o comando bc no prompt de comando e comece a fazer algumas contas, como mostrado.

Os exemplos a seguir mostram como usar bc de forma não interativa na linha de comando.

```
$ echo '3+5' | bc
```

```
$ echo '15 % 2' | bc
```

```
$ echo '15 / 2' | bc
```

```
$ echo '(6 * 2) - 5' | bc
```


O sinalizador **-1** é usado para a escala padrão (dígitos após o ponto decimal) para 20, por exemplo:

```
$ echo '12/5' | bc
```

```
$ echo '12/5' | bc -l
```

Usando o comando Awk

Awk é um dos programas de processamento de texto mais proeminentes no GNU/Linux. Ele suporta os operadores aritméticos de adição, subtração, multiplicação, divisão e módulo. Também é útil para fazer matemática de ponto flutuante.

Você pode usá-lo para fazer matemática básica, conforme mostrado.

```
$ awk 'BEGIN { a = 6; b = 2; print "(a + b) = ", (a + b) }'
```

```
$ awk 'BEGIN { a = 6; b = 2; print "(a - b) = ", (a - b) }'
```

```
$ awk 'BEGIN { a = 6; b = 2; print "(a * b) = ", (a * b) }'
```

```
$ awk 'BEGIN { a = 6; b = 2; print "(a / b) = ", (a / b) }'
```

```
$ awk 'BEGIN { a = 6; b = 2; print "(a % b) = ", (a % b) }'
```

Usando fator de comando

O comando fator é usado para decompor um número inteiro em fatores primos. Por exemplo:

```
$ factor 10
```

```
$ factor 127
```

```
$ factor 222
```

```
$ factor 110
```

Em um terminal Linux, caso você deseje somar 20 e 5 digitando simplesmente "20+5", isso resulta em um "comando não encontrado". Para fazer um cálculo na linha de comando é necessário usar o comando "expr" ou a sintaxe \$((operações)). Veja esses exemplos, alguns usando variáveis:

```
$ a=5
```

```
$ num=20
```

```
$ expr 20 + 05
```

```
$ expr 20 \* 5 # Asterisco é caractere especial, por isso a barra antes
```

```
$ expr "$a" + "$num"
```

```
$ expr length "Monolito Nimbus" # Conta o número de caracteres do que  
tiver entre aspas
```

```
$ var=$((20+5))
```

```
$ soma=$((var+2-outra_var))
```

```
$ echo $soma
```

O *Bash* (assim como o *sh*, *ash*, *csh*) não possui suporte nativo para operações com ponto flutuante, sendo necessário o uso de comandos externos. Um desses comandos é o **bc**: uma linguagem para cálculos, permitindo realizar cálculos matemáticos através do terminal – e, assim, automatizar alguns processos.

- ✓ adição (+)
- ✓ subtração (-)
- ✓ multiplicação (*)
- ✓ divisão (/)
- ✓ resto da divisão (%)
- ✓ raiz quadrada (sqrt)
- ✓ potência (^)
- ✓ seno (s(x))
- ✓ cosseno (c(x))
- ✓ logaritmo natural ln (l(x))
- ✓ função exponencial (e(x))

Usando casas decimais

```
$ a=1000
```

```
$ b=132
```

```
$ echo "scale=3;$b/$a*100" | bc
```

```
$ 13.200
```

```
$ a=1000
```

```
$ b=132
```

```
$ echo "scale=2;$b/$a*100" | bc
```

```
$ 13.20
```

```
$ a=1000
```

```
$ b=132
```

```
$ echo "scale=1;$b/$a*100" | bc
```

```
$ 13.2
```


Veja alguns exemplos, alguns usando variáveis – “scale” define o número de casas decimais:

```
$ echo "(20+5)/5" | bc
```

```
$ echo '100.098*(1+0.22)' | bc
```

```
$ echo "scale=2;($lat1+$lat2)/2" | bc # onde lat1=-  
23.6855443 e lat2=-23.6555284
```

```
$ lat_med=$(bc <<< "scale=7;($lat1+$lat2)/2")
```

Às vezes, pode aparecer um *erro* do tipo “valor muito grande para esta base de numeração (token com erro é “008”)”. Isso acontece porque o bash considera números que iniciam com zero como estando na base 8 (portanto, não podem ter dígitos maiores que 7). Para indicar que o valor é decimal, use “10#” na frente da variável (incluir “\$” mesmo quando estiver dentro de parênteses). Por exemplo:

```
$ echo $((10#$a+10#$b))
```

Caso apareça um erro do tipo "erro de sintaxe: operador aritmético inválido (token com erro é" ou "(standard_in) 1: illegal character: ^M", é porque tem algum caractere não-numérico na variável utilizada (^M, por exemplo). Para isso, imprima essa variável e dê um pipe em sua saída para o seguinte comando e eliminar esses caracteres:

```
echo $a | sed $'s/^[[:print:]]\t//g'
```

Os ângulos são dados em radianos. É possível criar uma função que converta de graus para radianos:

```
$ pi = 3.14159265
```

```
$ define d2r(n) { return n * (pi/180); }
```

Da mesma forma, pode-se definir uma função para passar o logaritmo da base e para base 10:

```
$ define l10(x) {return l(x)/l(10)}
```



```
#!/bin/bash
```

```
echo "Seu nome de usuário é:"
```

```
whoami
```

```
echo "Info de hora atual e tempo que o computador está ligado:"
```

```
uptime
```

```
echo "O script está executando do diretório:"
```

```
Pwd
```

```
#Este é um comentário
```

```
#Este é outro comentário
```

```
echo "Este script contém comentários."
```




```
#!/bin/bash
```

```
site=www.unip.br
```


```
meu_numero_favorito=22
```

```
_cidade="São Paulo"
```

```
echo "Um ótimo site para você aprender a programar e  
se manter atualizado é: $site"
```

```
echo "Meu número favorito é: $meu_numero_favorito"
```

```
echo "Minha cidade natal é: $_cidade"
```



```
#!/bin/bash
```

```
echo "Digite um número qualquer:"  
read numero;  
if [ "$numero" -gt 20 ];  
then  
    echo "Este número é maior que 20!"  
fi
```

- **n** string1: o comprimento de string1 é diferente de 0;
- **z** string1: o comprimento de string1 é zero;
- string1 = string2: string1 e string2 são idênticas;
- string1 != string2: string1 e string2 são diferentes;
- inteiro1 **-eq** inteiro2: inteiro1 possui o mesmo valor que inteiro2;
- inteiro1 **-ne** inteiro2: inteiro1 não possui o mesmo valor que inteiro2;
- inteiro1 **-gt** inteiro2: inteiro1 é maior que inteiro2;
- inteiro1 **-ge** inteiro2: inteiro1 é maior ou igual a inteiro2;
- inteiro1 **-lt** inteiro2: inteiro1 é menor que inteiro2;
- inteiro1 **-le** inteiro2: inteiro1 é menor ou igual a inteiro2;
- **e** nome_do_arquivo: verifica se nome_do_arquivo existe;
- **d** nome_do_arquivo: verifica se nome_do_arquivo é um diretório;
- **f** nome_do_arquivo: verifica se nome_do_arquivo é um arquivo regular (texto, imagem, programa, docs, planilhas).



```
#!/bin/bash
```

```
echo "Digite um número qualquer:"  
read numero;  
if [ "$numero" -ge 0 ];  
then  
    echo "O número $numero é positivo!"  
else  
    echo "O número $numero é negativo!"  
fi
```

```
#!/bin/bash
```

```
echo "Selecione uma opção:"
echo "1 - Exibir data e hora do sistema"
echo "2 - Exibir o resultado da divisão 10/2"
echo "3 - Exibir uma mensagem"
read opcao;
if [ $opcao == "1" ];
then
    data=$(date +"%T, %d/%m/%y, %A")
    echo "$data"
elif [ $opcao == "2" ];
then
    result=$((10/2))
    echo "divisao de 10/2 = $result"
elif [ $opcao == "3" ];
then
    echo "Informe o seu nome:"
    read nome;
    echo "Bem-vindo ao mundo do shell script, $nome!"
fi
```



```
#!/bin/bash
```

```
echo "Selecione uma opção:"
```

```
echo "1 - Exibir data e hora do sistema"
```

```
echo "2 - Exibir o resultado da divisão 10/2"
```

```
echo "3 - Exibir uma mensagem"
```

```
read opcao;
```

```
case $opcao in
```

```
  "1")
```

```
    data=$(date +"%T, %d/%m/%y, %A")
```

```
    echo "$data"
```

```
    ;;
```

```
  "2")
```

```
    result=$((10/2))
```

```
    echo "divisao de 10/2 = $result"
```

```
    ;;
```

```
  "3")
```


```
    echo "Informe o seu nome:"
```

```
    read nome;
```

```
    echo "Bem-vindo ao mundo do shell script, $nome!"
```

```
    ;;
```

```
esac
```

```
#!/bin/bash
```

```
echo "Testando o comando seq"  
for i in $(seq 1 5 100);  
do  
    echo "$i"  
done
```

```
#!/bin/bash
```

```
echo "Testando o comando seq"  
for i in $(seq 1 100);  
do  
    echo "$i"  
done
```



```
#!/bin/bash
```

```
echo "Informe o que você quiser, -1 para  
sair"
```

```
read dado;  
while [ $dado != "-1" ];  
do  
    echo "Você digitou $dado"  
    read dado;  
done
```



```
#!/bin/bash
```

```
echo "Informe até que valor positivo e maior que zero contar:"
```

```
read valor;
```

```
i=1
```

```
while [ $i -le $valor ];
```

```
do
```

```
    echo "$i"
```

```
    ((i=$i+1))
```

```
done
```

```
#!/bin/bash
main()
{
    echo "Escolha uma opção:"
    echo "1 - Esvaziar a lixeira"
    echo "2 - Calcular fatorial"
    read opcao;
    case $opcao in
        "1")
            esvaziar_lixeira
            ;;
        "2")
            calcular_fatorial
            ;;
    esac
}
esvaziar_lixeira()
{
```

```
    echo "Esvaziando a lixeira..."
    path="${HOME}/.local/share/Trash/files"
    cd "$path"
    for file in *
    do
        rm -rf "$file"
    done
    echo "Done!"
}
calcular_fatorial()
{
    echo "Informe um número:"
    read numero;
    i=1
    fat=1
    while [ $i -le $numero ]
    do
        fat=$((fat*i))
        i=$((i+1))
    done
    echo "fatorial de $numero é $fat"
}
main
```



```
#!/bin/bash
```

```
if [ $# -lt 1 ];
```

```
then
```

```
    echo "Precisa fornecer pelo menos 1 argumento!"
```

```
    exit 1
```

```
fi
```

```
echo "Número de argumentos passados: $#"
```

```
i=0
```

```
for argumento in $*
```

```
do
```

```
    i=$((i+1))
```

```
    echo "Argumento $i passado: $argumento"
```

```
done
```