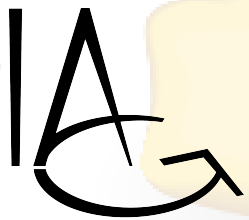
Introdução ao Linux

Estruturas de repetição e decisão

Israel Dragone

24 de outubro de 2018



INSTITUTO DE ASTRONOMIA,
GEOFÍSICA E CIÊNCIAS
ATMOSFÉRICAS

Atribuição de variáveis

Em shell não é necessário declarar o tipo nem o tamanho da varíavel, basta fazer a atribuição.

- var1=1
- var2=2.5
- var3="hello world"

Imprimindo na tela

```
$ echo $var1
```

```
1
```

```
$ echo "$var3, var1=$var1 e var2=$var2"
```

```
hello world, var1=1 e var2=2.5 (aspas duplas reconhecem $, \ e `)
```

```
$ echo 'var1=$var1 e var2=$var2'
```

```
var1=$var1 e var2=$var2
(aspas simples, protegem completamente a string)
```

echo

Opções Importates

- **echo** -n: Não pula linha automaticamente ao final do echo
- **echo** -e: Reconhece caracteres de escape com \
  - **n:** pula linha
  - **t:** tabulação horizontal
  - **v:** tabulação vertical
- $ **echo** -e "Alunos do 2° ano: \n Astolfo \t Beatriz \v Camila ..."

  $\longrightarrow$

  Alunos do 2° ano:
  Astolfo Beatriz

  Camila

Receber informações do teclado

Lendo uma variável

```
$ echo "Digite seu nome"
$ read nome                                          Henrique
```

```
$ echo $nome
```

```
Henrique
```

Lendo mais de uma variável

```
$ echo "Digite seu nome, idade e curso"
$ read v1 v2 v3                                      Ana 21 direito
```

Suprimir o echo e limitar o número de caracteres a ser registrado

```
$ read -p "Cor dos olhos: " cor
$ read -p "Digite o ano no formato aa"-n2 ano                2018
```

Operadores aritméticos

- **Adição:** $a+1, $a+$b, 1+2
- **Subtração:** $a-1, $a-$b, 1-2
- **Multiplicação:** $a*1, $a*$b, 1*2
- **Divisão:** $a/1, $a/$b, 1/2
- **Módulo:** $a%1, $a%$b, 1%2
- **Exponenciação:** $a**1, $a^$b, 1**2
- **Incremento:** **let** i++, **let** i+=1

Observações

- $ soma=$((a+b))
- $ soma=$[a+b]
- $ **echo** $((a+b))
- $ **echo** $[a+b]

Operação com floats

```
$ echo $[10/3]
```

```
3
```

```
$ echo $[10/1.5]
```

```
syntax error: invalid arithmetic operator
```

```
$ echo $[2.2*2]
```

```
syntax error: invalid arithmetic operator
```

```
$ echo $[2.2+3]
```

```
syntax error: invalid arithmetic operator
```

O shell não sabe trabalhar com números decimais!!

Bash calculator (bc)

```
$ echo "scale=2; 10/3" | bc
```
$\#\frac{10}{3}$

```
3.33
```

```
$ bc <<< "scale=4; (10/1.5)+3"
```
$\#(\frac{10}{1.5}) + 3$

```
9.6666
```

```
$ bc <<< "scale=2; sqrt(96)"
```
$\#(\sqrt{96})$

```
9.79
```

```
$ bc -l <<< "scale=3; s(3.14/4)"
```
$\#\text{sen}(\frac{\pi}{4})$

```
0.706
```

```
$ var=$(echo "scale=2; 10/3" | bc)
$ var=`bc <<<"scale=4; 10/1.5"`
```

Comando test

Comparações Numéricas

- **Igual:** -eq (EQual)
- **Diferente:** -ne (Not Equal)
- **Maior:** -gt (Greater Than)
- **Maior ou igual:** -ge (Greater or Equal)
- **Menor:** -lt (Less Than)
- **Menor ou igual:** -le (Less or Equal)
- **Operadores lógicos:** && (AND), || (OR) ! (Not)

Exemplos

- [$a -eq $b], [$a -eq 10]
- [$a -gt $b], [$a -gt 10]
- [$a -le $b] && [$a -gt 10]
- [[($a -gt 3 && $a -lt 10) || ($b -eq 3)]]

Comando test

Comparação de Strings

- **Igual:** “$a” = “$b”, “$a” = “linux”
- **Diferente:** “$a” != “$b”, “$a” != “curso linux”
- **Não Nula:** [-n “$a”]
- **Nula:** [-z “$a”]

Teste com arquivos

- **Arquivo existe:** [-e “$filename”]
- **Arquivo existe e tem tamanho maior que zero:** [-s “$filename”]
- **Permissão de leitura:** [-r “$filename”]
- **Permissão de escrita:** [-w “$filename”]
- **Permissão de execução:** [-x “$filename”]

Como usar essas comparações para estruturar ações?

```
if [ condição ]; then
    ações
fi
```

```
cor="azul"
```

```
if [ "$cor" = "azul" ]; then
    echo "A cor escolhida foi azul"
fi
```

```
cor="vermelho"
```

```
if [ "$cor" = "azul" ]; then
    echo "A cor escolhida foi azul"
else
    echo "A cor escolhida não foi azul"
fi
```

Estruturando condições

```
if [ "$cor" = "azul" ]; then
    echo "A cor escolhida foi azul"

elif [ "$cor" = "verde" ]; then
    echo "A cor escolhida foi verde"

elif [ "$cor" = "rosa" ]; then
    echo "A cor escolhida foi rosa"

elif [ "$cor" = "vermelho" ]; then
    echo "A cor escolhida foi vermelho"

else
    echo "Cor inválida"
fi
```

Testando várias opções - case

```
case "$cor" in

    azul) echo "A cor escolhida foi azul" ;;

    verde) echo "A cor escolhida foi verde" ;;

    rosa) echo "A cor escolhida foi rosa" ;;

    vermelho) echo "A cor escolhida foi vermelho" ;;

    *) echo "Opção inválida" ;;

esac
```

Criando scripts - Hello World

No terminal

```
$ gedit meu_script.sh &
```

No arquivo

```
echo "Hello World"
```

No terminal

```
$ bash meu_script.sh
```

```
Hello World
```

Exercícios

1. Faça um script que receba um número inteiro do teclado (n) e exiba tela: (i) o número lido (n); (ii) n+5; (iii) se n é par ou ímpar e (iv) n^2.

2. Faça um script que receba dois números inteiros do teclado (n_1 e n_2) e exiba na tela: (i) qual foi o número maior lido; (ii) se n_1 e n_2 são pares ou ípares ou se um é par e outro é ímpar; (iii) calcule a raiz quadrada do menor número e (iv) e^{n_2}.

3. Faça um script que dê a opção de selecionar um conjunto cmp ou de tiro do arquivo su, pelo número ep ou cdp. Adicione as seguintes opções ao script: (i) visualização do dado; (ii) balanceamento das energias e (iii) visualização do espectro de frequência

Laços de repetição - lista

```
for i in abacate morango III ç 3 5 goiaba
do
    echo -n "$i "
done
```

```
abacate morango III ç 3 5 goiaba
```

```
for i in  /sismicaII/*.su
do
    echo -n "$i "
done
```

```
1.su 2.su 3.su 4.su 5.su 6.su 7.su 8.su 9.su 10.su
```

```
for i in `cat mute_traces`
do
    sukill < arquivo.su key=tracl min=$i
done
```

Laços de repetição (seq)

```
for i in $(seq 1 10)
do
    echo -n "$i "
done
```

```
1 2 3 4 5 6 7 8 9 10
```

```
for i in $(seq 1 2 10)
do
    echo -n "$i "
done
```

```
1 3 5 7 9
```

```
for i in $(seq 1 -1 -10)
do
    echo -n "$i "
done
```

```
1 0 -1 -2 -3 -4 -5 -6 -7 -8 -9 -10
```

Laços de repetição - {}

```
for i in {1..10}
do
    echo -n "$i "
done
```

```
1 2 3 4 5 6 7 8 9 10
```

```
for i in {1..10..2}
do
    echo -n "$i "
done
```

```
1 3 5 7 9
```

```
for i in {A..Z}
do
    echo -n "$i "
done
```

```
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
```

Laços de repetição - padrão C

```
for ((i=1;i<10;i++))
do
    echo -n "$i "
done
```

```
1 2 3 4 5 6 7 8 9 10
```

```
for ((i=1;i<10;i+=2))
do
    echo -n "$i "
done
```

```
1 3 5 7 9
```

```
for ((i=1;i>-10;i--))
do
    echo -n "$i "
done
```

```
1 0 -1 -2 -3 -4 -5 -6 -7 -8 -9 -10
```

Laços de repetição - while

```
while [ $var -ne 0 ];
do
    acões
done
```

```
var="s"
while [ "$var" != "n" ];
do
    read -p "Digite o nome do arquivo a ser filtrado" filename
    read -p "Digite as frequências desejadas (f1,f2,f3,f4)" f1 f2 f3 f4
    sufilter < filename f=$f1,$f2,$f3,$f4 amps=0,1,1,0 > filtrado.su
    suspecfx < filtrado.su | suxgraph style=normal
    echo "Deseja aplicar o filtro novamente? (s/n)" var
done
```

Funções

```
function nomeFuncao( ) {
    código
}
```

```
function filtroFreq( ) {
var=“s”
while [ “$var” != “n” ];
do
    read -p “Digite o nome do arquivo a ser filtrado” filename
    read -p “Digite as frequências desejadas (f1,f2,f3,f4)” f1 f2 f3 f4
    sufilter < filename f=$f1,$f2,$f3,$f4 amps=0,1,1,0 > filtrado.su
    suspecfx < filtrado.su | suxgraph style=normal
    echo “Deseja aplicar o filtro novamente? (s/n)” var
done
}
```

Funções

```
function operacoes( ) {
    n1=$1 n2=$2
    echo "n1*n2 = $[n1*n2]"
    echo "n1^n2 = $[n1**n2]"
    echo `bc -l <<< "scale=2; sqrt(n1) * l(n2)"`
}
```

```
operacoes 8 3
```

```
24
512
3.07
```

- **$0:** Armazena o nome da função
- **$1 . . . $n:** Armazena os argumentos de 1 a n passados a função
- **$#:** Armazena o número de argumentos passados à função
- **$*:** Exibe todos os argumentos passados à função

Exercícios

4. Crie um script que dado um número inteiro calcula sua tabuada de 1 até 20. Salve a saída num arquivo de texto chamado tabuada_do_$num.dat.
5. Sabendo que a letra A ocupa a posição 1 no alfabeto, B a 2, C a 3 ... Faça um código que exiba as letras em posições pares ou ímpares de acordo com a escolha do usuário.
6. O arquivo ex_aula2.su contém dados coletados na praça do relógio concatenados seguidas vezes. Aplique um sukill nos traços que contém a assinatura na fonte.
7. Crie um script para realizar todo o fluxograma de processamento aprendido até agora (balanceamento, ganho, filtro de frequência, filtro FK ...), sempre mostrando o resultado para o usuário e questionando se deseja repetir ou não o passo realizado. Podem usar os dados adquiridos na praça do Relógio. Obs.: Lembre-se que é possível chamar outros scripts de dentro do seu script, basta usar o mesmo comando inserido no terminal