Iteração

João Marcelo Uchôa de Alencar

Universidade Federal do Ceará - Quixadá

19 de Setembro de 2019

A estrutura do *for*

```
for var in valor1 valor2 ... valorN
do
    <comando1>
    <comando2>
    <...>
    <comandoM>
done
```

A variável ${var} assume um valor diferente para cada iteração.
Um comando pode ser usado para gerar a lista de valores.
Exemplo: **seq** permite gerar uma sequência de valores inteiros.

```
$ seq 10
$ seq 1 10
$ seq 1 2 10
```

A opção $-s$ permite mudar o caractere que separa os números gerados. Se não existir a palavra **in** e a sequência de valores, ${var} irá percorrer a lista de parâmetros.

Exemplos

- Vamos criar um *script* chamado **somaTotal.sh** que receba um parâmetro e calcule a soma de todos os inteiros de zero até o valor do parâmetro.
- Agora vamos fazer um *script* chamado **maiorDeTodos.sh** que receba vários parâmetros inteiros e devolva o maior deles.

O Separador Entre os Campos - IFS

Comandos como o **sort**, o **cut** e o próprio **for** na geração de sua lista de valores usam o espaço em branco como separador padrão.

```
usuario=$1
grep $usuario /etc/passwd
for info in $(grep $usuario /etc/passwd)
do
   echo $info
done
OLDIFS=$IFS
IFS=":"
for info in $(grep $usuario /etc/passwd)
do
   echo $info
done
IFS=$OLDIFS
```

A estrutura do *while*

```
while <comando>
do
    <comando1>
    <comando2>
    <...>
    <comandoN>
done
```

O **comando** é sempre executado no início. Se o resultado for sucesso, então os outros comandos começam a ser executados. Enquanto **comando** for sucesso, o laço se repete.

A estrutura do *until*

```
until <comando>
do
    <comando1>
    <comando2>
    <...>
    <comandoN>
done
```

Só começa a executar os comandos caso o **comando** da condição seja fracasso. Se após a execução do bloco e **comando** ainda for fracasso o laço se repete. Só irá terminar quando **comando** executar sucesso.
Para finalizar, só lembrando que tanto o **break** e o **continue** das outras linguagens funcionam no *Shell*.

Exemplos

- Faça um *script* que fique em execução sem parar, sendo que cada vez que um usuário entrar ou sair do sistema, ele exiba um aviso de mudança no número de usuários.
- Incremente o *script* anterior. Faça com que a mensagem exibide informa se o usuário entrou ou saiu, e qual o nome do usuário.
- Faça um *script* que receba como parâmetro o caminho de um diretório que só tem arquivos de texto como conteúdo. O *script* deve imprimir em ordem crescente uma lista com os arquivos de acordo com a quantidade de linhas de cada arquivo. Lembrando que o arquivo com mais linhas não é necessariamente o arquivo com tamanho maior em *bytes*.

Atividades

- Crie o diretório *atividades/atividade06*
- Você vai desenvolver o *script latencia.sh*
- Esse *script* vai receber como parâmetro o nome de um arquivo de texto, contendo um endereço IP por linha.
- O *script* deve usar o comando **ping** para enviar dez pacotes ICMP para cada endereço do arquivo, calculando o valor médio do tempo de resposta.
- O *script* deve imprimir uma lista de IP ordenada do menor para o maior tempo médio de resposta, informando além do endereço, o tempo de resposta médio.

No **próximo slide** tem um exemplo de como utilizar o *script*.

Atividades

```
# os IPs abaixo são fantasia.
$ cat enderecos_ip.txt
8.8.8.8
192.168.0.1
54.230.57.207
$ ./latencia.sh enderecos_ip.txt
192.168.0.1 11.1ms
54.230.57.207 55.4ms
8.8.8.8 94.0ms
```