Laboratório de Sistemas Operacionais II – Aula prática 03 – 30/08/2016

Trabalhando com Comando Test e Condições Compostas

1. **Introdução** a Comandos Condicionais Compostos.

   Abaixo está uma tabela com as principais opções para testes de condições de arquivos:

Opções	**Verdadeiro ($?=0) se arquivo existe e:**
-r arquivo	tem permissão de leitura.
-w arquivo	tem permissão de gravação.
-x arquivo	tem permissão de execução.
-f arquivo	é um arquivo regular.
-d arquivo	é um diretório.
-u arquivo	seu bit set-user-ID está ativo.
-g arquivo	seu bit set-group-ID está ativo.
-k arquivo	seu sticky bit está ativo.
-s arquivo	seu tamanho é maior do que zero.

   Obs.: Esta tabela será utilizada a partir do item 5 abaixo.

2. **Criar** uma pasta para a aula de hoje. Chame de Aula3Shell (***mkdir Aula3Shell***). Seguir para a pasta criada Aula3Shell (***cd Aula3Shell***).

3. **Crie** quatro arquivos usando o ***cat*** ou ***vi***, conforme abaixo:

- Primeiro arquivo:

cat > quequeisso ***ou*** ***vi quequeisso***

Fundação Santo André
Faculdade de Filosofia, Ciências e Letras
Curso de Bacharelado em Sistemas de Informação
Disciplina de Laboratório de Sistemas Operacionais

Digite ***ctrl +c*** para salvar o arquivo ou os comandos para salvar o vi.

- Segundo arquivo:

cat > tafechado ***ou*** ***vi tafechado***

A formação da minha nota final é obtida pela soma de P1 e at1 mais as notas da P2 e at2.
Digite ***ctrl +c*** para salvar o arquivo ou os comandos para salvar o vi.
Obs.: Retire todas as permissões de acesso deste arquivo: digite ***chmod 000 tafechado –v***.

- Terceiro arquivo:

cat > talogado ***ou*** ***vi talogado***

Moro no ABC que pertence à Grande São Paulo. Esta pertence ao estado de São Paulo, que por sua vez pertence ao Brasil, que está localizado na América do Sul.
Digite ***ctrl +c*** para salvar o arquivo ou os comandos para salvar o vi.
Obs.: Coloque a permissão de execução para este arquivo: digite ***chmod u+x talogado –v***.

- Quarto arquivo:

cat > tavazio ***ou*** ***vi tavazio***

Digite ***ctrl +c*** para salvar o arquivo ou os comandos para salvar o ***vi***.

4. Agora, vamos ver como ficaram as permissões de acesso desses quatro arquivos. **Mostre** os dados de todos os arquivos desta pasta com o comando ***ls –l***. Veja se as permissões de acesso estão conforme abaixo:

- r w – r w – r - -	***quequeisso***
- - - - - - - - - - -	***tafechado***
- r w x r w – r - -	***talogado***
- r w – r w – r - -	***tavazio***

5. **Fazer** os seguintes testes no seu prompt:

test –f tafechado
echo $?
0
Obs. O comando ***test*** quer saber se tafechado é um arquivo (opção ***–f***).
Condição verdadeira. A resposta foi 0 (sucesso).

test –f talogado
echo $?
0
Obs. O comando ***test*** quer saber se existe arquivo talogado (opção ***–f***).
Condição verdadeira. A resposta foi 0 (sucesso).

test –r tafechado
echo $?
1
Obs. O comando ***test*** quer saber se eu (usuário) tenho direito de leitura sobre o arquivo tafechado (opção ***–r***).
Condição falsa. A resposta foi 1 (insucesso). Veja permissões no item 4.
test –r talogado
echo $?
0

Obs. O comando ***test*** quer saber se eu (usuário) tenho direito de leitura sobre o arquivo talogado (opção ***–r***).
Condição verdadeira. A resposta foi 0 (sucesso).

test –x talogado
echo $?
0

Obs. O comando ***test*** quer saber se eu (usuário) posso executar talogado (opção ***–x***).
Condição verdadeira. A resposta foi 0 (sucesso).

test –x quequeisso
echo $?
1

Obs. O comando ***test*** quer saber se eu (usuário) tenho direito de executar quequeisso (opção ***–x***).
Condição falsa. A resposta foi 1 (insucesso). Veja permissões no item 4.

test –s tavazio
echo $?
1

Obs. O comando ***test*** quer saber se o arquivo tavazio existe e se tem tamanho maior que zero (opção ***–s***).
Condição falsa. A resposta foi 1 (insucesso). O arquivo tem tamanho zero.

test –s tafechado
echo $?
0

Obs. O comando ***test*** quer saber se o arquivo tafechado existe e tem tamanho maior que zero (opção ***–s***).

Condição verdadeira. A resposta foi 0 (sucesso). O arquivo tem tamanho maior que zero.

6. ***Estudar*** o comando ***test*** com cadeia de caracteres.

   Abaixo está uma tabela com as principais opções para testes de condições de cadeias de caracteres:

Opções	**Verdadeiro ($?=0) se:**
-z cad1	o tamanho de cad1 é zero.
-n cad1	o tamanho da cadeia cad1 é diferente de zero.
cad1=cad2	as cadeias cad1 e cad2 são idênticas.
cad1	cad1 é uma cadeia não nula.

7. **Fazer** os seguintes testes no seu prompt:

   nada=
   test –z "$nada"
   echo $?
   0
   Obs. A variável nada é criada com valor nulo. O comando ***test*** quer saber se a variável nada não existe ou está vazia (opção ***–z***).
   Condição verdadeira. A resposta foi 0 (sucesso).

   test –n "$nada"
   echo $?
   1
   Obs. O comando ***test*** quer saber se a variável nada existe e não está vazia (opção ***–n***).
   Condição falta. A resposta foi 1 (insucesso).

Note que não precisamos aqui definir novamente a variável nada, porque ela foi definida no comando anterior e você pode referir-se à ela nos próximos comandos.

test $nada
echo $?
1
Obs. O comando ***test*** quer saber se a variável nada não está vazia (sem opção).
Condição falsa. A resposta foi 1 (insucesso).

nada=algo
echo $nada
algo
Obs. ***nada=algo*** atribui um valor à variável ***nada***. Mostramos o conteúdo de ***nada*** e foi listada a palavra algo.

test $nada
echo $?
0
Obs. O comando teste quer saber se a variável nada não está vazia.
Condição verdadeira. A resposta foi 0 (sucesso).

8. ***Estudar*** o comando ***test*** com inteiros.

Abaixo está uma tabela com as principais opções para testes com inteiros:

Opções	**Verdadeiro ($?=0) se:**	**Significado:**
int1 –eq int2	int1 igual a int2	Equal to
int1 –ne int2	int1 diferente de int2	Not equal to

int1 –gt int2	int1 maior que int2	Greater than
int1 –ge int2	int1 maior ou igual a int2	Greater or equal
int1 –lt int2	int1 menor que int2	less than
int1 –le int2	int1 menor ou igual a int2	less or equal

9. **Fazer** os seguintes testes no seu prompt:

typeset –i qqcoisa=10
echo $qqcoisa
10
Obs. Defini uma variável qqcoisa inteira com o valor de 10.

test "$qqcoisa" – eq 10
echo $?
0
Obs. O comando test fez um teste entre inteiros... O resultado foi sucesso (0).

test "$qqcoisa" = 10
echo $?
1
Obs. O comando test fez um teste entre cadeia de caracteres... O resultado foi insucesso (1).

Dos exemplos citados podemos inferir que, caso o objetivo do teste fosse identificar se o conteúdo da variável era exatamente igual a um valor, este teste deveria ser executado com operandos característicos de cadeias de caracteres. Por outro lado, se seu desejo fosse testar a

semelhança entre o valor e o conteúdo da variável, o operando deveria ser numérico.

10. **Criar** programa usando ***negação de condição***:

Num comando IF, se você quiser negar uma operação, basta colocar um caractere ***!*** (exclamação antes da condição).

Abaixo está um programa para testar esta condição.

10. 1 Criar script usando ***cat > teste.txt*** ou ***vi teste.txt***

Sistemas Operacionais
Estrutura de Dados
Estatística
Programação matemática
Cálculo Numérico

Feche com ***ctrl+c,*** se usou o ***cat***, ou feche o ***vi*** conforme já foi ensinado.

10.2 Criar script usando ***cat > PS21.ksh*** ou ***vi PS21.ksh***

```
#!/bin/ksh
#
# Este programa salva um arquivo no seu formato original antes de
# editá-lo pelo vi, de forma a poder recuperá-lo incólume, no caso de
# alguma coisa sair errada mo editor
#
if [ "$#" –ne 1 ]
then
        echo "Erro -> Uso: $0 <arquivo> "
fi

Arq=$1
if [ ! –f "$Arq" ]# O arquivo não existe; logo como salvá-lo?
```

```
then
        vi  $Arq
        exit 0
fi

if [ !  –w  "$Arq" ] # Será que tenho permissão de gravação?
then
        echo "Você não conseguirá sobregravar $Arq"
        exit 2
fi
cp  $Arq  $Arq~
vi  $Arq
exit  0
```

10.3 Para executar o programa, coloque permissão de acesso (***u+x***) e execute-o da seguinte forma:

./PS21.ksh teste.txt

Obs: Após executar o seu programa ***PS21.ksh***, você deve notar que o seu arquivo teste será aberto com o ***vi***. Faça alguma alteração no seu arquivo ***teste.txt*** adicionando alguma nova linha, fechando-o em seguida.
Você vai notar que ao fechar o ***vi***, o programa ***PS21.ksh*** também é terminado.
Então, com o comando ***ls –l***, veja se foi criado em seu diretório um arquivo cópia de ***teste.txt*** chamado de ***teste.txt~***. Este será a cópia original do seu arquivo.

11. **Testar** as condições ***and***:

Criar com **cat > *PS22a.ksh*** ou ***vi PS22a.ksh***

```
#!/bin/bash
#
# Verifica o sexo digitado de uma pessoa
#
sexo=$1
printf "”
prinft “Digitado sexo = $sexo \n”
printf "”
if [ “$sexo” =  1 ]
then
        homens=`expr $homens + 1`
else
        mulheres=`expr $mulheres + 1`
fi
printf “O total de homens é $homens \n”
printf “O total de mulheres é $mulheres \n”
```

Execute com ***./PS22a.ksh 4***

O 4 na frente do código do programa diz que você deseja o código 4 para sexo.
Obs.: Qualquer coisa que for digitada diferente de 1 será considerado mulher.
Para testar as duas condições 1 e 2 ao mesmo tempo usamos o operador ***–a***.

Será verdadeiro somente se ***todas*** as condições testadas forem verdadeiras, caso contrário, será falso.

Veja o exemplo abaixo:

Criar com **cat > *PS22b.ksh*** ou ***vi PS22b.ksh***

```
#!/bin/bash
#
# Verifica o sexo digitado de uma pessoa
#
sexo=$1
printf "”
printf “Digitado sexo = $sexo”
printf "”
if [ “$sexo” != 1 –a “$sexo” != 2 ]
then
        echo “Sexo invalido”
else
        echo “Sexo válido”
fi
```

Execute com ***./PS22b.ksh 4***

12. **Testar** as condições ***or***:

O operador lógico ***or*** (ou), representado no ambiente do ***if*** por ***–o***, serve para testar duas ou mais condições, dando resultado verdadeiro se ***pelo menos*** uma dentre as condições testadas forem verdadeiras.
Observe que a crítica de sexo escrita no item anterior, também poderia ter sido feita da forma a seguir:

Veja o exemplo abaixo:

Criar com **cat > *PS23a.ksh*** ou ***vi PS23a.ksh***

```
#!/bin/bash
#
# Verifica o sexo digitado de uma pessoa
#
sexo=$1
printf "”
printf “Digitado sexo = $sexo”
printf "”
if [ “$sexo” -lt 1 –o “$sexo” -gt 2 ]
then
        echo “Sexo invalido”
else
        echo “Sexo válido”
fi
```

Execute com ***./PS23a.ksh 4***

Observações muito importantes:

Tenha sempre em mente que ***não ou*** vale ***e***, e da mesma forma ***não e*** vale ***ou***.

Qual é a forma correta de fazer a pergunta: se sexo não igual a 1 e sexo não igual a 2 ou se sexo não igual a 1 e sexo não igual a 2?

O operador ***–a*** tem precedência sobre o operador ***–o*** desta forma, se quisermos priorizar a execução do ***or*** em detrimento ao and, devemos priorizar a expressão do ***or*** com o uso de parêntesis.

Se existisse um comando if construído da seguinte forma:

If [$sexo –eq 1 –o $sexo –eq 2 –a $nome = João –o $nome = Maria]

A primeira expressão a ser resolvida pelo interpretador de comandos do Shell seria:

If [($sexo –eq 1 –o $sexo –eq 2) –a ($nome = João –o $nome = Maria)]

Veja o exemplo abaixo:

Criar com **cat >** ***PS24.ksh*** ou ***vi PS24.ksh***

```
#!/bin/bash
#
# Verifica o sexo digitado de uma pessoa e
# Verifica um nome digitado
#
sexo=$1
nome=$2
printf "”
printf “Digitado sexo = $sexo”
printf “Digitado nome = $nome”
printf "”
If [ ( $sexo –eq 1 –o $sexo –eq 2) –a ($nome = João –o $nome = Maria) ]
then
        echo “Expressão está toda correta”
else
        echo “Algo está invalido”
fi
```

Faça os seguintes testes:

Execute com ***./PS24.ksh 1 João***
Aqui está certo o código do sexo e o nome de João.

Execute com ***./PS24.ksh 1 Maria***
Aqui está certo o código do sexo e o nome de Maria.

Execute com ***./PS24.ksh 2 João***
Aqui está certo o código do sexo e o nome de João.

Execute com ***./PS24.ksh 2 Maria***
Aqui está certo o código do sexo e o nome de Maria.

Execute com ***./PS24.ksh 4 João***
Aqui está errado o código do sexo e certo o nome de João.

Execute com ***./PS24.ksh 4 Maria***
Aqui está errado o código do sexo e certo o nome de Maria.

Execute com ***./PS24.ksh 4 Pedro***
Aqui estão ambos errados: o código do sexo e o nome.

"É verdadeiramente velho o homem que para de aprender, não importa se ele tenha 20 ou 80 anos."

Henry Ford.