 Instituto Federal de Garopaba

Relatório de desenvolvimento de projeto: Shell Script

Garopaba-SC

2018

Equipe: Diogo Felipe, Igor, Fabio e Charles

1.Introdução

1.1. Este trabalho orientado pelo Docente e Coordenador Luiz Antônio Schalata Pacheco atuado no curso de Informática, tem como objetivo criar um programa que atenda a uma série de instruções, e caso não atenda a essas instruções, o programa deve gerar uma saída alternativa para o usuário. O Programa tem o seguinte problema: Você recebe uma lista com 10 nomes de pessoas. Faça um Shell Script que para cada pessoa crie um usuário no sistema. O login do usuário deve obedecer seguinte ordem de preferência:

a) O primeiro nome do usuário

b) O sobrenome do usuário

c) O primeiro nome dos usuários seguido das iniciais de seus outros nomes

d) As iniciais dos primeiros nomes do usuário seguido do seu sobrenome

EXEMPLO:

Luiz Antônio Schalata Pacheco-luiz

b) O sobrenome do usuário

c) O primeiro nome dos usuários seguido das iniciais de seus outros nomes

d) As iniciais dos primeiros nomes do usuário seguido do seu sobrenome

EXEMPLO:

Luiz Antônio Schalata Pacheco-luiz

Luiz José Pedro Pacheco-Pacheco

Luiz Marcos Pacheco-luizmp

Luiz Mauro Pacheco-lmpacheco

Todo o programa foi criado com a linguagem Shell Script. Foram feitas várias etapas até a conclusão, desde a procura pelos comandos que iam ser usados no Script até a Prototipagem e montagem do Programa.

2. Shell Script

2.1. O que é?

Shell Script é uma ferramenta de automação. Com um arquivo de texto executável o usuário ou sistema é capaz de executar uma sequência de operações, instruções e testes.

As possibilidades com o shell script vão até onde a imaginação do programador alcançar.

Podemos tanto fazer um shell para criar usuários em massa em um sistema, quanto criar um shell para atualizar de forma automática, o sistema de um servidor.

Fonte: https://www.vivaolinux.com.br/artigo/O-que-e-Shell-Script

2.2. Variáveis e parâmetros

Na programação, uma variável é, um objeto capaz de reter e representar o valor de um valor ou expressão, e que, “existe” apenas durante o tempo de execução.

Quando nos referimos a variável, do ponto de vista da programação de computadores, estamos tratando de uma região da memória previamente identificada cuja finalidade é armazenar os dados ou informações de um programa por um determinado espaço do tempo.

Sendo assim algumas das variáveis especiais que podemos usar no shell script são:

$LOGNAME

É o nome do usuário que está acessando o computador atualmente no sistema.

$HOSTNAME

É o nome da máquina em que está sendo executado o programa.

$0

É o nome do script que está sendo executado.

$1 ao $9

Contém os argumentos na ordem em que foram passados.

$#

Contém o número de argumentos que foram passados sem contar o nome do script em $0.

$\*

São todos os argumentos em uma única linha.

Fonte: https://www.vivaolinux.com.br/dica/Shell-script-Variaveis-especiais

2.3. Estrutura IF

Em toda linguagem de programação existe alguma estrutura para tomada de decisões. Essa estrutura em geral é feita através do comando IF. O comando IF analisa se uma expressão é verdadeira e então executa um bloco de código, a sua sintaxe de execução é:

if [CONDICAO];

then

  AÇÕES

Fi

2.3.1. Else

Caso a expressão contida no comando IF for considerada falsa um outro bloco pode ser executado o Else, a sua sintaxe de execução é:

if [ CONDICAO ];

   then

       AÇÕES\_1

  else

      AÇÕES\_2

Fi

2.4. Estrutura Case

O comando case é muito parecido com o if e alguns outros comandos mas o case usa uma sintaxe mais compacta, exemplo da sintaxe do comando:

case VARIAVEL in

(Caso 1)

Ações\_1

Esac

Fonte: http://blog.evaldojunior.com.br/aulas/blog/shell%20script/2011/05/08/shell-script-parte-2-controle-de-fluxo.html

2.5. Laços de repetição

É uma estrutura como o if e o case mas diferente deles em vez de executar o bloco de comando apenas 1 vez o laço de repetição executa o bloco de comando quantas vezes forem necessárias até que a condição descrita no comando seja atendida.

2.5.1 Estrutura While

É uma estrutura de repetição simples que executara o bloco do programa enquanto a condição descrita no while fora verdadeira, a sintaxe do while é:

while( <condição> )

do

<comandos>

done

2.5.2 Estrutura For

O For é outra estrutura de repetição bem usada mas diferente do while o bloco do programa do For é executado até a condição descrita nele fora atendida, sua sintaxe é:

for ( <condição> )

do

<comandos>

done

Fonte: https://cleitonbueno.com/shell-script-estrutura-de-repeticao-e-arrays/

3. Execução

3.1 Metodo

O metodo utilizado foi pensado da seguinte forma:

a) Separar todos nomes/sobrenomes em uma variavel.

b) Modificar as variaveis.

c) Juntar as variaveis de acordo com as condições dadas na introdução.

3.2 O que foi utilizado?

Tudo utilizado na execução foi estrutura IF e estrutura While. Alguns comandos para edição de string tais como: sed,tr,cut,grep,wc,expr,cat.

3.2.1 Sed

Geralmente, o sed recebe como entrada, através de uma canalização, os dados sobre os quais irá atuar. Aceita expressões regulares, o que lhe confere maior poder, convertendo-se em uma excelente ferramenta para administradores de sistemas.

fonte: <https://pt.wikipedia.org/wiki/Sed>

3.2.2 Tr

O tr é um pequeno utilitário Unix para excluir ou traduzir uma cadeia de caracteres em outra, lendo da entrada-padrão e escrevendo para a saída-padrão.

fonte: <https://pt.wikipedia.org/wiki/Tr_(Unix)>

3.2.3 Cut

Emite as partes selecionadas das linhas de cada arquivo na saída padrão. Necessita de argumentos obrigatórios.

fonte: Ubuntu linux - Terminal: Cut --help

3.2.4 Grep

Executa uma busca num arquivo apartir dos argumentos dados.

fonte: Ubuntu linux - Terminal: Grep --help

3.2.5 Wc

O wc é um comando usado para contar palavras e, por extensão, contar bytes e linhas.

fonte: <https://pt.wikipedia.org/wiki/Wc_(Unix)>

3.2.6 Expr

Utilizado para cálculos feitos em linha de comando. Ele pode efetuar cálculos com expressões numéricas inteiras, expressões lógicas , expressões regulares e strings.

fonte: <https://www.vivaolinux.com.br/artigo/expr-Definicao-e-uso-do-comando>

3.2.7 Cat

Exibe na tela o conteúdo de um arquivo.

fonte: Ubuntu linux - Terminal: Cat --help

3.3 Separar nomes

Utilizando os comandos cat,grep e cut é feito a separação da string. Cat para abrir o arquivo, grep para conseguir a linha com o nome, e cut para conseguir cara parte do nome e sobrenome,tudo isso como dado como resultado numa variavel. Por fim um sed para retirar todos os números da string pega.

*nome[$i]=`cat -n ListaSemA2.txt | grep -n ^ | grep ^$i`*

*sed 's/[0-9]\+//g' < Nomes$i.txt > Nome$i.txt*

*i=contador*

3.4 Modificar as variaveis

Eu modifico a variavel apenas com um cut para isolar cada primeira letra dos sobrenomes numa variavel terceira também. Sendo assim é facilitada minha parte de apenas juntas as letras com os nomes completos.

*echo "${sobrenome[$i]}" > LetraSobrenome$i.txt*

*lsobre[$i]=`cat LetraSobrenome$i.txt | cut -b 1`*

*echo(escreve o conteúdo indicado)*

O exemplo dado acima é repitido novamente mais 9 vezes para completar todos os 10 nomes da lista, assim criando as variaveis separadas agora apenas esperando a junção de acordo com as condições.

3.5 Junção de acordo com condições

Agora é apenas juntar as variaveis que ja estão prontas para junção.

Começando então a primeira condição de apenas escrever o primeiro nome como usúario.

*echo "${nome[$i]}" > Usuarios$i.txt*

*verificador[$i]=`cat Usuario$i.txt``*

Agora começa as verificações apartir dos IF, com a segunda condição.

*if [ "${verificador[$i]}" == "${verificador[$x]}" ] && [ "$x" != "$i" ]; then*

*echo "${sobrenome[$i]}" > Usuario$i.txt*

*verificador[$i]=`expr substr ${verificador[$i]} 1 10` #limita caractere ate 10*

Seguido da terceira condição, que neste caso tem uma coisa diferente, ele verifica a quantidade de palavras dadas na linha da lista (utilizando wc), e determina qual letra de qual sobrenome ele vai precisar utilizar na montagem do usúario.

*if [ "${verificador[$i]}" == "${verificador[$x]}" ] && [ "$x" != "$i" ] && [ $pass1=1 ]; then #Terceira Condição*

*if [ ${quant[$i]} == 2 ]; then*

*echo "${nome[$i]}${lsobre[$i]}" > Usuario$i.txt*

*verificador[$i]=`expr substr ${verificador[$i]} 1 10` #limita caractere ate 10*

O "if quant" é repitido 10 vezes até achar sua condição.

Por sinal temos a quarta e ultima condição, utilizando o mesmo metódo de contagem para utilizar a determinada letra de sobrenome (wc).

*if [ "${verificador[$i]}" == "${verificador[$t]}" ] && [ "$x" != "$i" ] && [ $pass2=1 ]; then #Quarta e ultima condição*

*if [ ${quant[$i]} == 2 ]; then*

*echo "${pnome[$i]}${sobrenome[$i]}" > Usuario$i.txt*

*verificador[$i]=`expr substr ${verificador[$i]} 1 10` #limita caractere ate 10*

Repitindo novamente 10 vezes o "if quant" até achar sua condição.

3.6 Arquivo final

O arquivo final é dado na junção de todos os usúarios criados nas condições que são colocados no arquivo UsuariosFinais.txt da seguinte forma:

*touch UsuariosFinais.txt*

*echo -e "Usuario 1 é ${verificador[0]}\nUsuario 2 é ${verificador[1]}\nUsuario 3 é ${verificador[2]}\nUsuario 4 é ${verificador[3]}\nUsuario 5 é ${verificador[4]}\nUsuario 6 é ${verificador[5]}\nUsuario 7 é ${verificador[6]}\nUsuario 8 é ${verificador[7]}\nUsuario 9 é ${verificador[8]}\nUsuario 10 é ${verificador[9]}\n " | tr -d " " > UsuariosFinais.txt*

Dessa forma o resultado final pode ser consultado no arquivo.

4. Conclusão

Durante a execução deste trabalho, foi usado o conteúdo que foi experimentado no laboratório, sobre a linguagem Shell Script(Estruturas while, for, etc). Também foram aprendidas diversas coisas testando os comandos no Linux

(mesmo fora das aulas), e discutindo entre os membros do grupo. Além do conhecimento das aulas e do que conseguimos ao experimentar, foi necessária a pesquisa em variados sites externos, tais como fórums,

para adquirir-se conhecimento necessário, e realizar a conclusão do projeto. Por fim, foi aprendido a real importância de algo como o Shell Script, que é várias vezes necessário para projetos extensivos que poderiam levar uma

grande quantidade de tempo caso fossem feitos manualmente.