Tópicos Especiais em Fundamentos da Computação
Shell Scripting

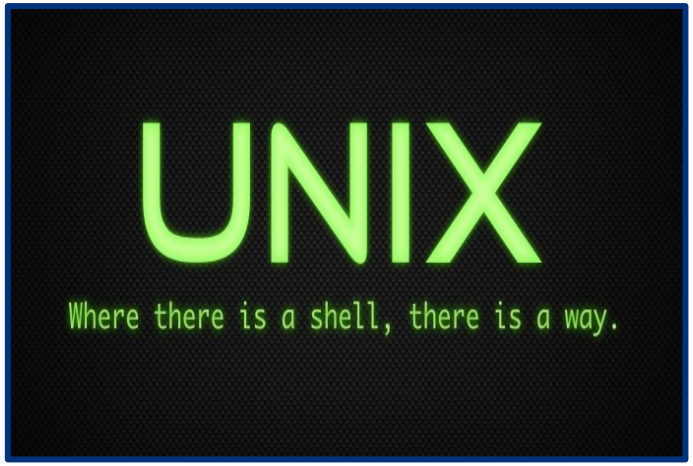
Apresentação da Disciplina

Andrei Rimsa Álvares
andrei@cefetmg.br

Sumário

- Sobre UNIX
- Sobre Shell
- Shell Scripting
- Um Shell Script Simples
- Informações, Ementa, Avaliações, Regras
- Bibliografia

SOBRE UNIX



História do UNIX

- UNIX foi criado em 1969 nos laboratórios da Bell, na época uma divisão de P&D da AT&T

- Foi inspirado no sistema operacional MULTICS (*Multiplexed Information and Computer Services*), projeto anterior com o MIT e a GE para criação de um sistema operacional para mainframe da GE

- A primeira versão foi batizada de UNICS (*Uniplexed Information and Computing Service*) que foi evoluída para os sistemas UNIX encontrados comumente atualmente

UNIX não é mais acrônimo de nada, mas uma derivação de UNICS

Dica: mais informações sobre a história do UNIX: https://youtu.be/HADp3emVABg

Vantagens

- Como UNIX foi desenvolvido em um ambiente de pesquisa, não havia pressão comercial para liberar um produto finalizado
  - O sistema era desenvolvido pelos próprios usuários
  - Os pesquisadores eram livres para experimentar e modificar programas conforme necessidade; como eram pequenos e os usuários eram desenvolvedores, isso não era um problema
  - Os pesquisadores desenvolveram o sistema para seu próprio uso e de seus colegas (todos cientistas da computação), o que levou a uma abordagem de design *non-sense*; programas faziam o que se pedia, sem "falação" e "perguntação" se você tem certeza do que quer fazer
  - A liberdade no ambiente de desenvolvimento da Bell levou à criação de um sistema não só funcional, mas também elegante

Adoção de UNIX

- UNIX foi gradualmente portado originalmente do minicomputador PDP-7 para diferentes arquiteturas
- O código-fonte era disponibilizado por uma pequena taxa com o objetivo de aumentar sua adoção
- UNIX ganhou aceitação nas universidades entre os alunos
- A medida que esses alunos assumiam cargos corporativos de compra de softwares e equipamentos, UNIX era uma escolha natural, já que eram familiarizados com a tecnologia
- Hoje, sistemas computacionais UNIX formam a infraestrutura de muitas grandes corporações

Versões UNIX

- No começo, UNIX era distribuído na forma de código-fonte, ao invés da típica forma binária
- Isso facilitou customizações para propósitos próprios, resultando em diversas versões de UNIX (sabores diferentes)
- A vasta maioria de sistemas UNIX foram construídos sobre duas versões base

AT&T System V

BSD

Versões UNIX

- Com várias versões de UNIX disponíveis o usuário tem o poder de escolha, ou seja, pode selecionar aquela que melhor atende às suas necessidades

- Algumas versões comerciais e *open-source* famosas♠

Qual o problema de ser ter tantas versões diferentes?

♠: UNIX timeline (https://bit.ly/3hPTAQo)

Desvantagens

- A mesma liberdade teve desvantagens claras com a adoção do UNIX além do ambiente original de desenvolvimento
  - Haviam muitas inconsistências entre os programas utilitários (usar a mesma letra para significar coisas diferentes, ou letras diferentes para significar a mesma coisa)
  - Muitos programas tinham limitações, quanto ao tamanho das entradas ou no número de arquivos abertos
  - Programas não eram testados tão rigorosamente quanto deveriam, tornando-os menos confiáveis
  - A documentação, embora geralmente completa, era concisa e minimalista, o que dificultava o aprendizado do sistema

Contextualização

- Shell scripts são usados principalmente para processamento e manipulação de dados textuais (não binário)

- Isso se deve ao fato do forte interesse nesse tipo de processamento que existia no início do desenvolvimento do UNIX, mas é válido por outros motivos também
  - O primeiro sistema UNIX em produção era utilizado para processamento de texto e formatação no departamento de patentes da Bell Labs

Contextualização

- Os computadores originais que executavam UNIX (PDP-11) não eram capaz de executar grandes programas

- Para executar uma tarefa complexa era preciso dividir o problema em pedaços menores e usar pequenos programas para cada
  - Alguns tipos de tarefas comuns (extrair campos em linhas, substituição de texto) era compartilhadas entre muitos projetos, o que acabaram tornando ferramentas padrões

- Isso no final acabou se tornando uma boa coisa, a limitação computacional levou a programas menores, simples e mais focados

Contextualização

- Muitas pessoas trabalhavam semi-independentemente no UNIX, reimplementando programas de outras pessoas
- Entre diferentes versões e sem nenhum obrigação de padronização, as ferramentas foram naturalmente divergindo
- Isso acontecia com muitas ferramentas, não somente com algumas

Como resolver essa situação?

Padronização

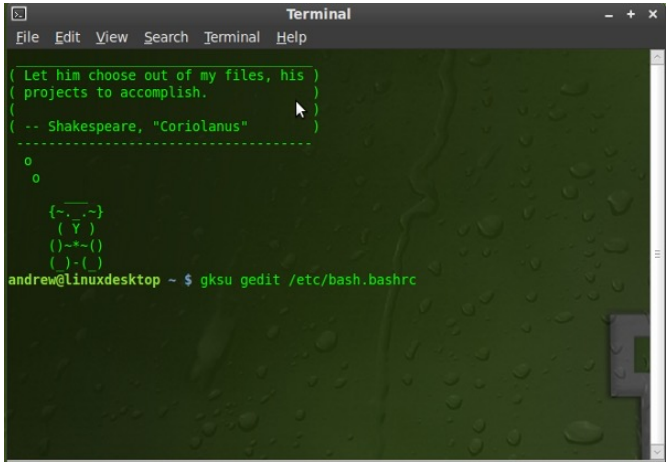
- Eventualmente, era preciso de um conjunto de ferramentas e opções padronizadas
- O resultado foi a padronização POSIX (*Portable Operating System Interface*), que atualmente engloba a biblioteca de desenvolvimento C, linguagem shell e programas utilitários com suas opções
- Atualmente, a maioria dos sistemas UNIX (comerciais e *open-source*) são todos POSIX-compatíveis
- Isso torna o aprendizado mais fácil e permite a escrita de scripts portáveis para diversas plataformas

SOBRE SHELL

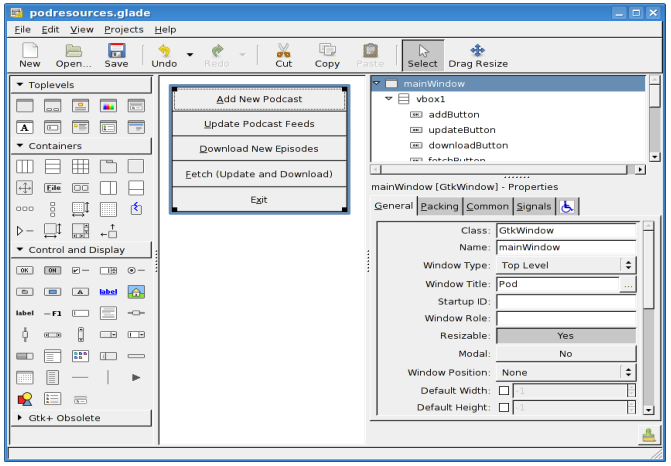
Shell Scripting

Linha de Comando

- Existem vários tipos de interface com o usuário, em que os exemplos mais clássicos são as duas a seguir



Interface baseadas em caracteres textuais



Interface gráfica

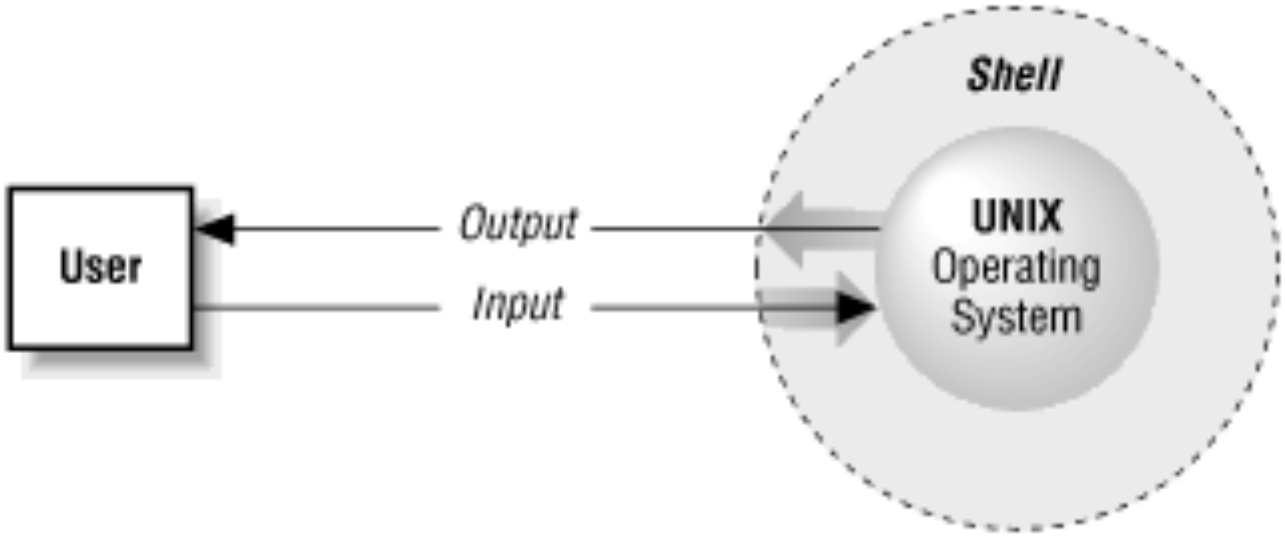
Qual é mais fácil de operar?

Interface em Linha de Comando

- A interface em linha de comando sempre foi criticada por sua complexidade e alta curva de aprendizado
- Contudo, é indiscutível que é uma das mais flexíveis e poderosas interfaces de usuários
- Além disso, normalmente são altamente programáveis

O que é Shell?

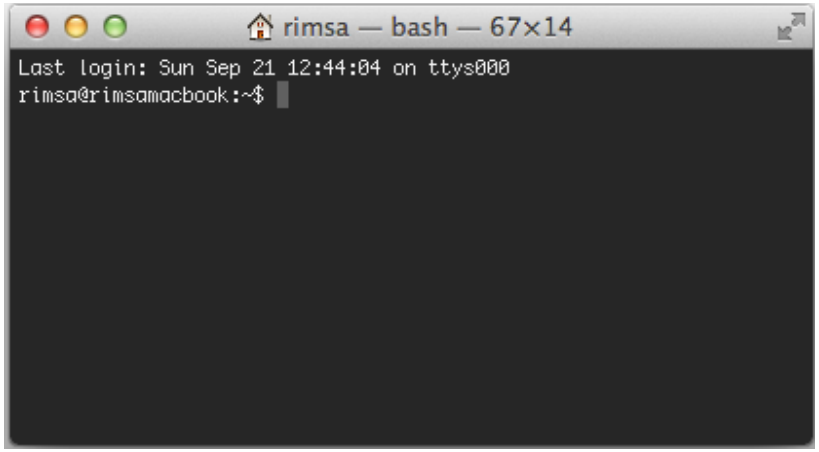
- O objetivo do shell é traduzir os comandos dados por um usuário em instruções para o sistema operacional



Acessando o Shell

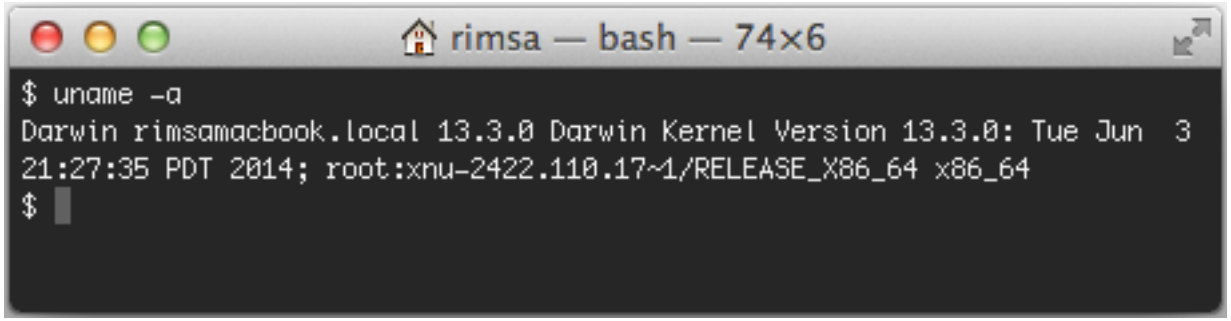
- O shell pode ser acessado de várias maneiras
  - Quando você loga por um terminal (control+alt+F1 no Linux)
  - Por um emulador de terminal pela interface gráfica
  - Por uma sessão de rede (como telnet, ssh)
  - Entre outros

- Exemplo



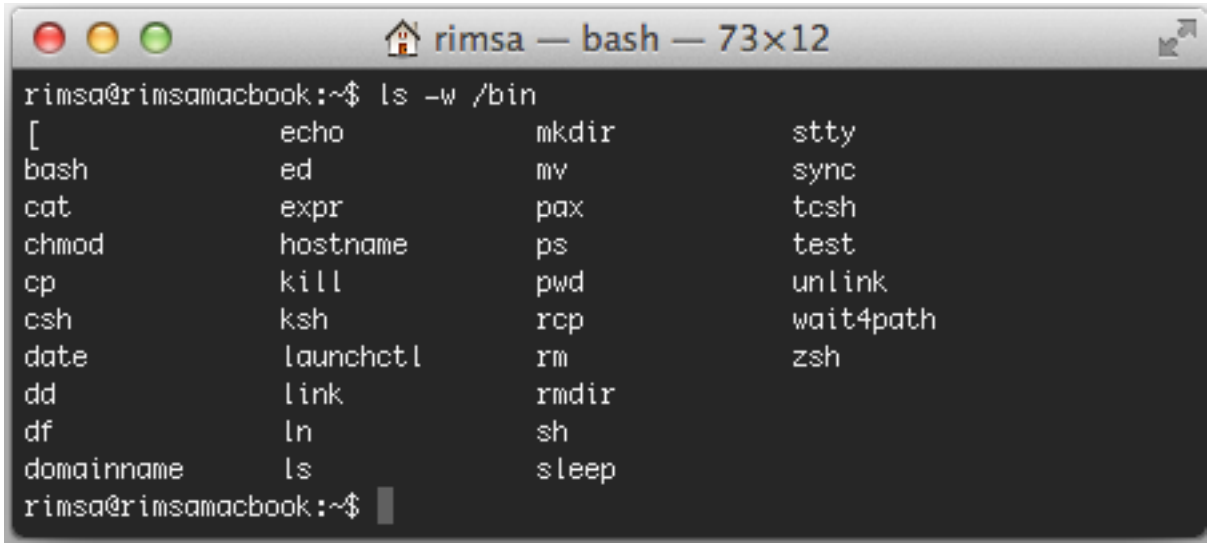
Comandos no Shell

- Após acessar o shell, ele mostrar um prompt que indica que está pronto para receber comandos
- Em resposta ao prompt, você pode dar os comandos que desejar
- O shell então executa o comando e após terminar mostra um novo prompt



Comandos Disponíveis

- Quais comandos estão disponíveis para uso?



Mas é só isso?

Comandos Disponíveis

É preciso conhecer todos eles?

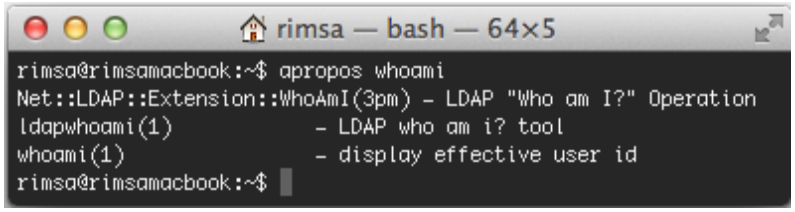
- Quais comandos estão disponíveis para uso?

rimsa — bash — 185×48

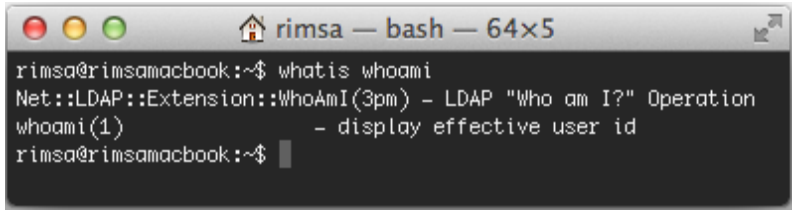
```
rimsa@rimsamacbook:~$ ls -w /usr/bin
2to3                            db_printlog                     idle2.7                         mklocale                        ptardiff                        svnversion
2to3-                           db_recover                      idlj                            mktemp                          ptardiff5.12                    sw_vers
2to3-2.7                        db_stat                         imptrace                        mmroff                          ptardiff5.16                    symbols
2to32.6                         db_upgrade                      indent                          mnthome                         ptargrep                        symbolscache
BuildStrings                    db_verify                       indxbib                         moc                             ptargrep5.16                    syscallbypid.d
CpMac                           dbicadmin                       info                            moc-4.8                         pubsub                          syscallbyproc.d
DeRez                           dbicadmin5.12                   infocmp                         moose-outdated                  pwhich                          syscallbysysc.d
GetFileInfo                     dbicadmin5.16                   infokey                         moose-outdated5.12              pwhich5.16                      sysdiagnose
MergePef                        dbilogstrip                     infotocap                       moose-outdated5.16              pwpolicy                        syslog
MvMac                           dbilogstrip5.12                 install                         more                            pydoc                           tab2space
R                               dbilogstrip5.16                 install-info                    mp2bug                          pydoc2.5                        tabs
RSA_SecurID_getpasswd           dbiprof                         install_name_tool               msgs                            pydoc2.6                        tail
ResMerger                       dbiprof5.12                     instmodsh                       nano                            pydoc2.7                        talk
Rez                             dbiprof5.16                     instmodsh5.12                   nasm                            python                          tar
RezDet                          dbiproxy                        instmodsh5.16                   native2ascii                    python-config                   tbl
RezWack                         dbiproxy5.12                    instruments                     nbdst                           python2.5                       tccutil
Rscript                         dbiproxy5.16                    iofile.d                        nc                              python2.5-config                tclsh
SetFile                         dc                              iofileb.d                       ncal                            python2.6                       tclsh8.4
SplitForks                      defaults                        iopattern                       ncdestroy                       python2.6-config                tclsh8.5
UnRezWack                       desdp                           iopending                       ncurses5.4-config               python2.7                       tee
VBoxAutostart                   diff                            iosnoop                         ndisasm                         python2.7-config                telnet
VBoxBalloonCtrl                 diff3                           iotop                           neqn                            pythonw                         testrb
VBoxHeadless                    diffstat                        ip2cc                           net-server                      pythonw2.5                      texi2dvi
VBoxManage                      dig                             ip2cc5.12                       net-server5.16                  pythonw2.6                      texi2pdf
VBoxVRDP                        dirname                         ip2cc5.16                       net-snmp-cert                   pythonw2.7                      texindex
VirtualBox                      diskhits                        ipcount                         net-snmp-config                 qc2movie                        textutil
a2p                             dispqlen.d                      ipcount5.12                     net-snmp-create-v3-user         qcollectiongenerator            tfmtodit
a2p5.12                         ditto                           ipcount5.16                     nettop                          qcollectiongenerator-4.8        tftp
a2p5.16                         dns-sd                          ipcrm                           newaliases                      qdbus                           tic
abgx360                         dnsctl                          ipcs                            newgrp                          qdbus-4.8                       tidy
addftinfo                       dprofpp                         ipmitool                        newproc.d                       qdbuscpp2xml                    tiff2icns
afconvert                       dprofpp5.12                     ippfind                         nfsstat                         qdbuscpp2xml-4.8                tiffutil
afhash                          drutil                          ipptool                         nice                            qdbusxml2cpp                    time
afida                           dscacheutil                     iprofiler                       nl                              qdbusxml2cpp-4.8                timer_analyser.d
afinfo                          dscl                            iptab                           nm                              qdoc3                           timerfires
afmtodit                        dserr                           iptab5.12                       nmedit                          qdoc3-4.8                       tkcon
afplay                          dsexport                        iptab5.16                       nohup                           qhelpgenerator                  tkmib
agentxtrap                      dsimport                        irb                             notifyutil                      qhelpgenerator-4.8              tkpp
agvtool                         dsmemberutil                    jar                             nroff                           qlmanage                        tkpp5.12
alias                           dsymutil                        jarsigner                       nslookup                        qmake                           tkpp5.16
allmemory                       dtruss                          java                            nsupdate                        qmake-4.8                       tmdiagnose
applesingle                     du                              javac                           ntp-keygen                      qt3to4                          tmutil
appletviewer                    dwarfdump                       javadoc                         ntpq                            qt3to4-4.8                      tnameserv
apply                           easy_install                    javah                           od                              qtdefaults                      toe
apr-1-config                    easy_install-2.5                javap                           odutil                          quota                           top
apropos                         easy_install-2.6                javaws                          open                            rails                           tops
apt                             easy_install-2.7                jcmd                            opendiff                        rake                            topsyscall
```

Obtendo Ajuda

- Obter o propósito do comando

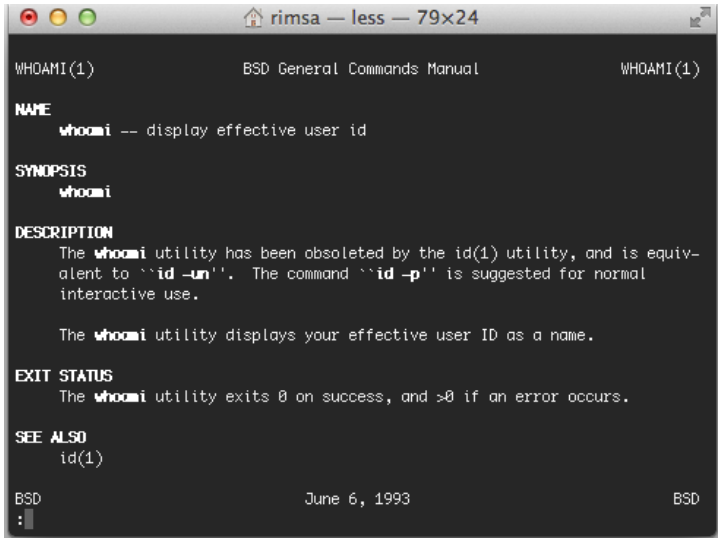


– **apropos** *comando*



– **whatis** *comando*

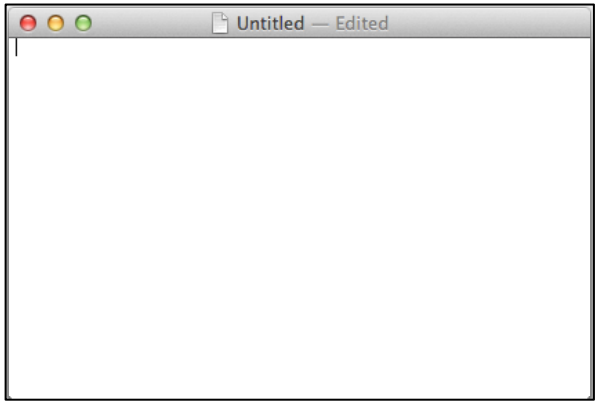
- Obtendo o manual do comando
  - **man** *comando*

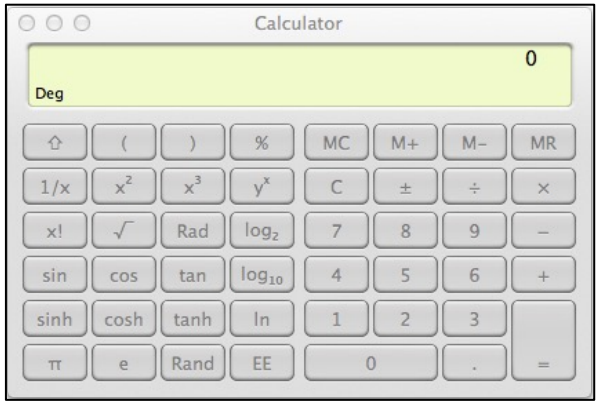


SHELL SCRIPTING

Shell Scripting

- Quando se quer realizar alguma tarefa em um computador, é melhor utilizar uma ferramenta que é apropriada para a situação
  - Você não irá usar editor de texto para fazer o balanço de uma empresa





  - Nem vai usar uma calculadora para escrever uma proposta de negócio

Shell Scripting

- Shell scripts são usados frequentemente para tarefas administrativas do sistema
- Podem ser usados combinando programas já existentes para realizar uma tarefa específica
- Depois de descobrir como fazer, pode-se agrupar os comandos em um arquivo formando uma "receita" (*script*)
- Assim, outros usuários podem fazer uso desse *script* como um programa fechado, sem saber como funciona internamente

Linguagens de Script vs Linguagens Compiladas

- A maioria dos programas de médio e grande porte são escritos em linguagens compilada (C, Pascal, C++, Java, C#); traduzidos do código-fonte para código objeto que é executado pelo computador
  - A vantagem é que são eficientes
  - A desvantagem é que trabalham no baixo nível (como copiar todos os arquivos de um diretório para outro em C++?)

- Linguagens de scripts normalmente são interpretadas; seu código é lido, traduzido e então executado
  - A vantagem é que trabalham em um nível de abstração mais alto
  - A desvantagem é que são mais ineficientes que linguagens compiladas

Por quê então usar shell scripts?

Por Quê Usar Shell Scripts?

- O *trade-off* vale a pena
  - pode ser que demore uma hora para codificar um script simples que demoraria algumas horas em outras linguagens (como C/C++)
  - e, mesmo mais lento, o script vai executar rápido o suficiente de forma que o desempenho não será relevante

Por Quê Usar Shell Scripts?

- Como o *shell* é universal entre sistemas UNIX e a linguagem é padronizada pelo *POSIX*, *shell scripts* podem ser escritos uma vez (se tomadas as devidas precauções) e usados numa variedade de sistemas diferentes

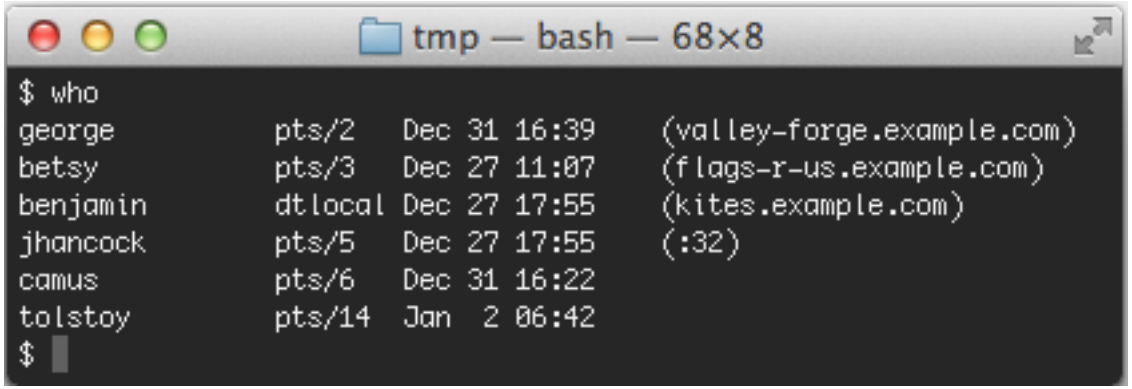
- Razões para usar *shell scripting*
  - **Simplicidade**: é uma linguagem de alto nível que se pode expressar operações complexas de forma simples e clara
  - **Portabilidade**: Ao usar somente funcionalidades padronizadas (*POSIX*), têm se grandes chances de funcionar em vários sistemas
  - **Fácil desenvolvimento**: pode-se escrever *scripts* úteis e poderosos em pouquíssimo tempo

UM SHELL SCRIPT SIMPLES

Um Shell Script Simples

- Imagine que você queira contar quantas pessoas estão atualmente *logadas* no sistema

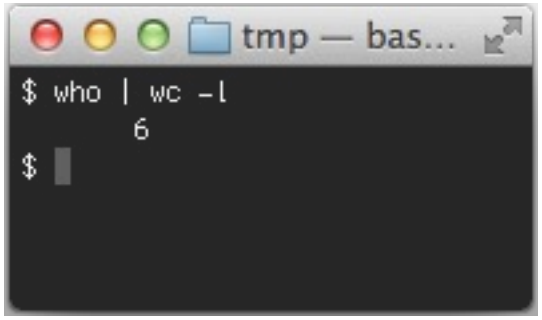
- O comando **who** pode ser usado para listar os usuários logados



  - Em servidores multiusuários esta lista pode ser tão extensa que a barra de rolagem iria correr antes de dar tempo de você contar
  - Mesmo se não for esse o caso, seria muito chato ter que contar manualmente toda vez

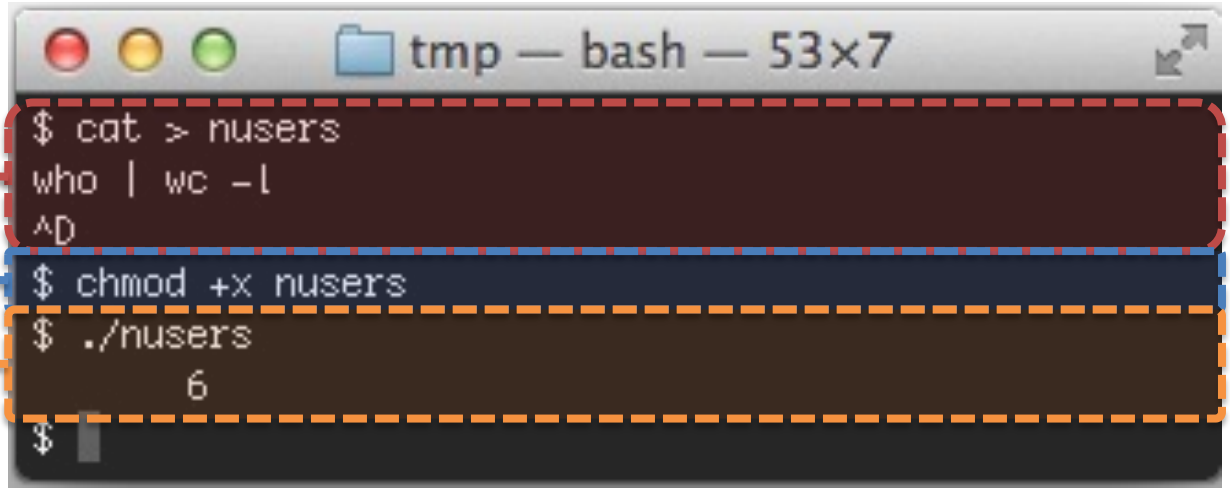
Um Shell Script Simples

- O que está faltando aqui é somente contar quantos usuários agora
- Para isso, podemos utilizar o comando **`wc`** (`word count`) que é capaz de contar linhas, palavras e caracteres; nesse caso queremos contar apenas linhas (opção `-l`)
- O símbolo `|` (pipe) cria um túnel entre dois programas: a saída do comando **`who`** virá entrada do comando **`wc`**



Automatizando o Processo

- Podemos automatizar o processo seguindo os seguintes passos
  1. Criar um arquivo contendo os comandos para realização da tarefa
  2. Torná-lo executável
  3. Executar o script



Automatizando o Processo

- Quando o *shell* tenta executar o programa, ele pede ao *kernel* para criar um novo processo e rodar o programa nesse processo
  - O *kernel* sabe como fazer isso para programas compilados, mas não para scripts; o que resultará em uma falha

- O *shell* percebe que o *kernel* não conseguiu executar porque "o formato do arquivo não é reconhecido" e então considera que então só pode ser um script e o executa

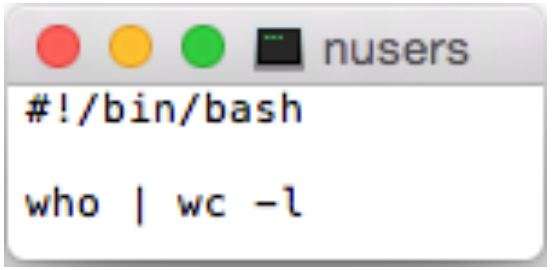
- Esse mecanismo é ótimo quando se têm apenas um *shell*, mas sistemas UNIX podem ter vários; é preciso ter uma forma melhor de dizer como executar o *script*, até com outros interpretadores

Shebang

- Isso é feito através de uma primeira linha especial no arquivo de script – que começa com #! (*shebang*)

- O *kernel* escaneia o resto da linha procurando um caminho para o programa capaz de interpretar o script; além de procurar uma **única** opção após o comando
  - Ex.: `#!/bin/csh -f`
  - Ex.: `#!/usr/bin/awk -f`

- Podemos reescrever nosso script especificando o uso do interpretador **`/bin/bash`**



INFORMAÇÕES, EMENTA, AVALIAÇÕES, REGRAS

Shell Scripting

Informações

- Todo o conteúdo da disciplina encontra-se no site: http://rimsa.com.br/page/classes/decom042

Tópicos Especiais em PC: Programação em Linux

DECOM.42

Ementa: Introdução à programação no sistema operacional Linux; Comandos e ferramentas de programação do Linux; Programação de scripts em terminal; O comando sed; A linguagem de programação AWK

Informações

- Dúvidas sobre a disciplina apenas pela lista de discussão:
  https://groups.google.com/d/forum/decom042
  (decom042@googlegroups.com)

Erros nos slides valem 0,25pts extras

Cuidado: somente serão considerados se notificados até antes da segunda prova

Ementa

Primeiro módulo

1) Comandos básicos
2) Processamento de texto
3) Sistema de arquivos
4) Processos
5) Entradas, saídas e redirecionamentos
6) Expansões

Segundo módulo

7) Parâmetros e variáveis
8) Estruturas de fluxo condicionais
9) Estruturas de fluxo de repetição
10) Expressões regulares
11) sed
12) AWK

Avaliações

- Prova 1: 30pts
- Prova 2: 30pts
- Prova suplementar: 30pts
- Prova especial: 100pts

- Listas de exercícios: 12pts (1pt cada)
- Projeto: 30pts

Regras

- A presença é obrigatória em 75% das aulas
  - Em todas as aulas haverá chamada
  - Não haverá abono de faltas, salvo nos casos previstos por lei

- A prova é individual e sem consulta
  - Após o início da prova, deve-se esperar no mínimo 30 minutos antes de entregar a prova
  - Colas serão penalizadas com nota zero

Regras

- Sobre as listas de exercícios
  - Fazer de forma **individual**
  - Fazer de forma **manuscrita** mais que 50% de cada lista e **digitalizar para PDF** para que ela seja considerada
  - Entregar na semana posterior ao conteúdo dado

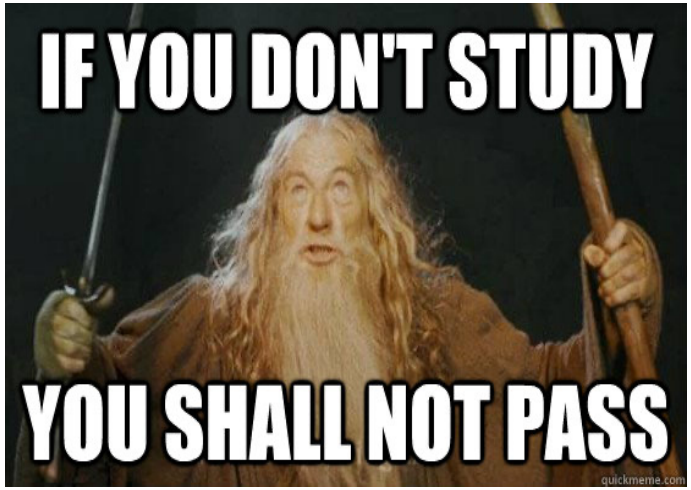
- Sobre o projeto
  - Projetos copiados, parcialmente ou integralmente, de colegas, da internet ou de qualquer outra fonte serão avaliados com nota zero

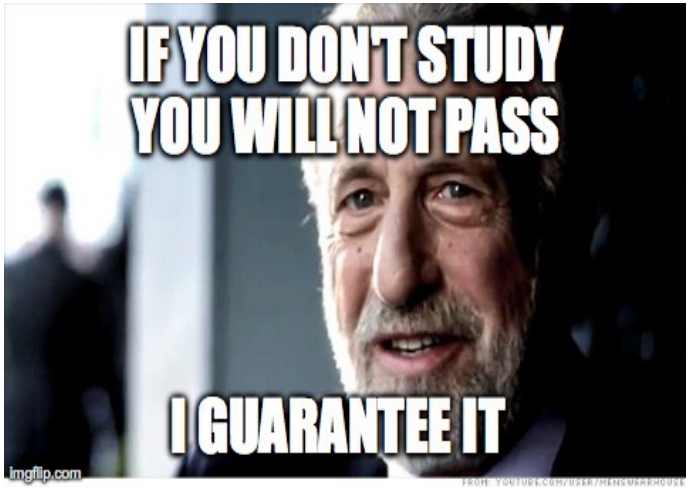
Regras

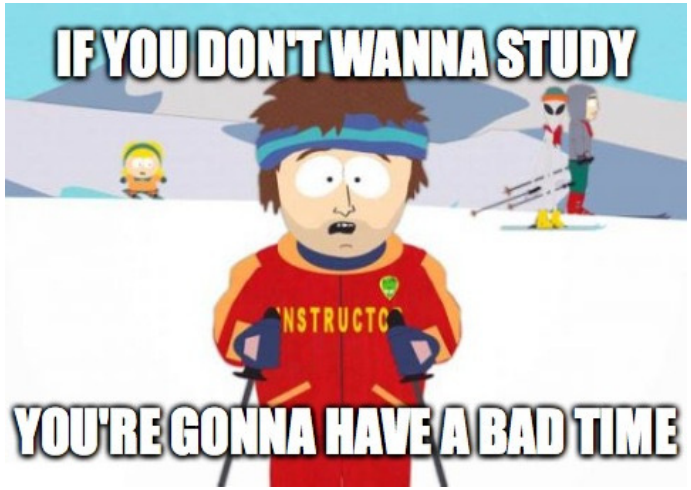
- Sobre uso de IA

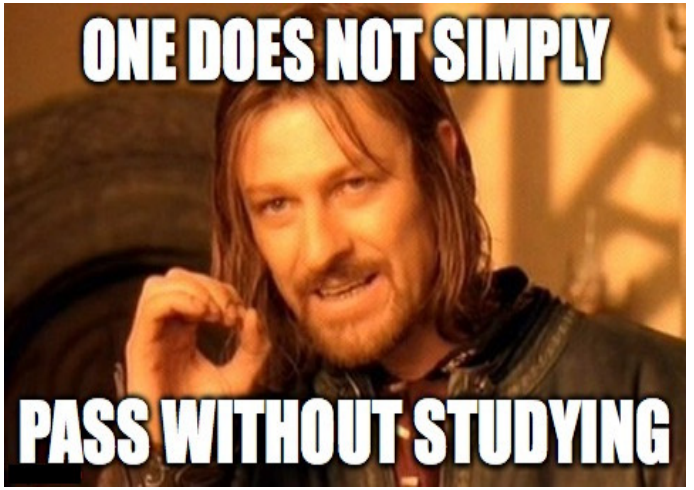
  - Não é proibido o uso de IA, mas use-a com cautela
  - Use-a a seu favor, como uma tutora para seus estudos
    - Não simplesmente copie e cole soluções dadas pela IA
    - Tente discutir como ela pode te ajudar a entender como resolver problemas

Estudem!


IF YOU DON'T STUDY
YOU SHALL NOT PASS

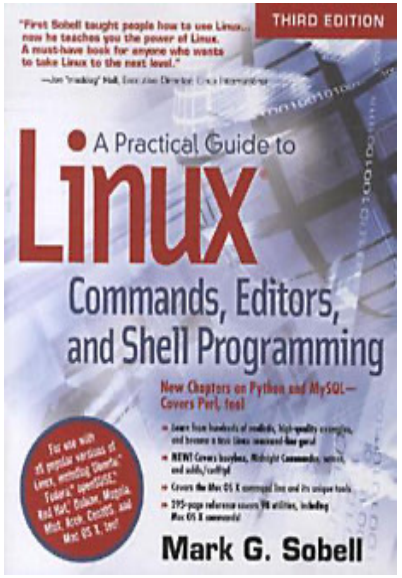

IF YOU DON'T STUDY
YOU WILL NOT PASS
I GUARANTEE IT

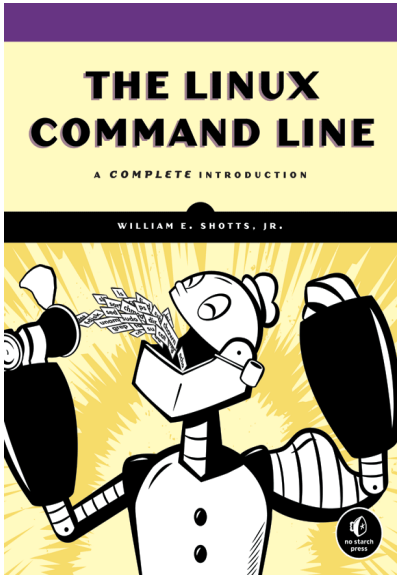

IF YOU DON'T WANNA STUDY
YOU'RE GONNA HAVE A BAD TIME


ONE DOES NOT SIMPLY
PASS WITHOUT STUDYING

BIBLIOGRAFIA

Bibliografia

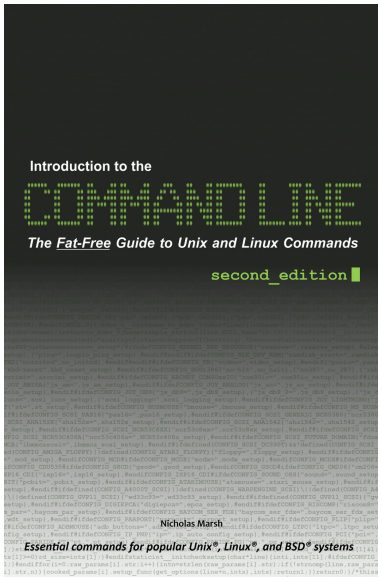




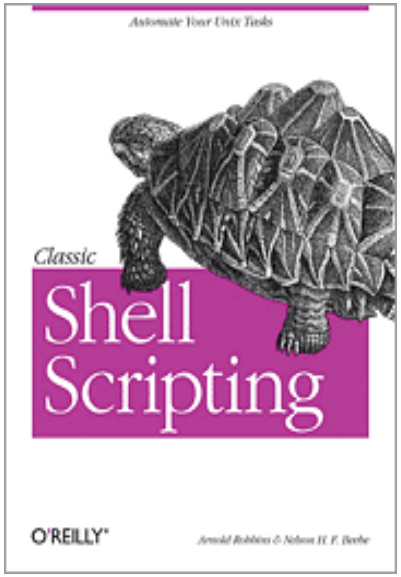
- Sobell M.
- A Practical Guide to Linux Commands, Editors, and Shell Programming
- 3ª edição, 2012

- Shotts W.
- The Linux Command Line: A complete introduction
- 2ª edição, 2012

Bibliografia



- Marsh N.
- Introduction to the Command Line
- 2ª edição, 2010



- Robbins A., Beebe N.
- Classic Shell Scripting
- 1ª edição, 2005

ISSO É TUDO, PESSOAL!