Utilizando o sistema

- Dois ambientes de operação e interação

- Ambiente Gráfico (GUI)
- Ambiente texto (CLI)

- Autenticação
- getty e kdm e gdm

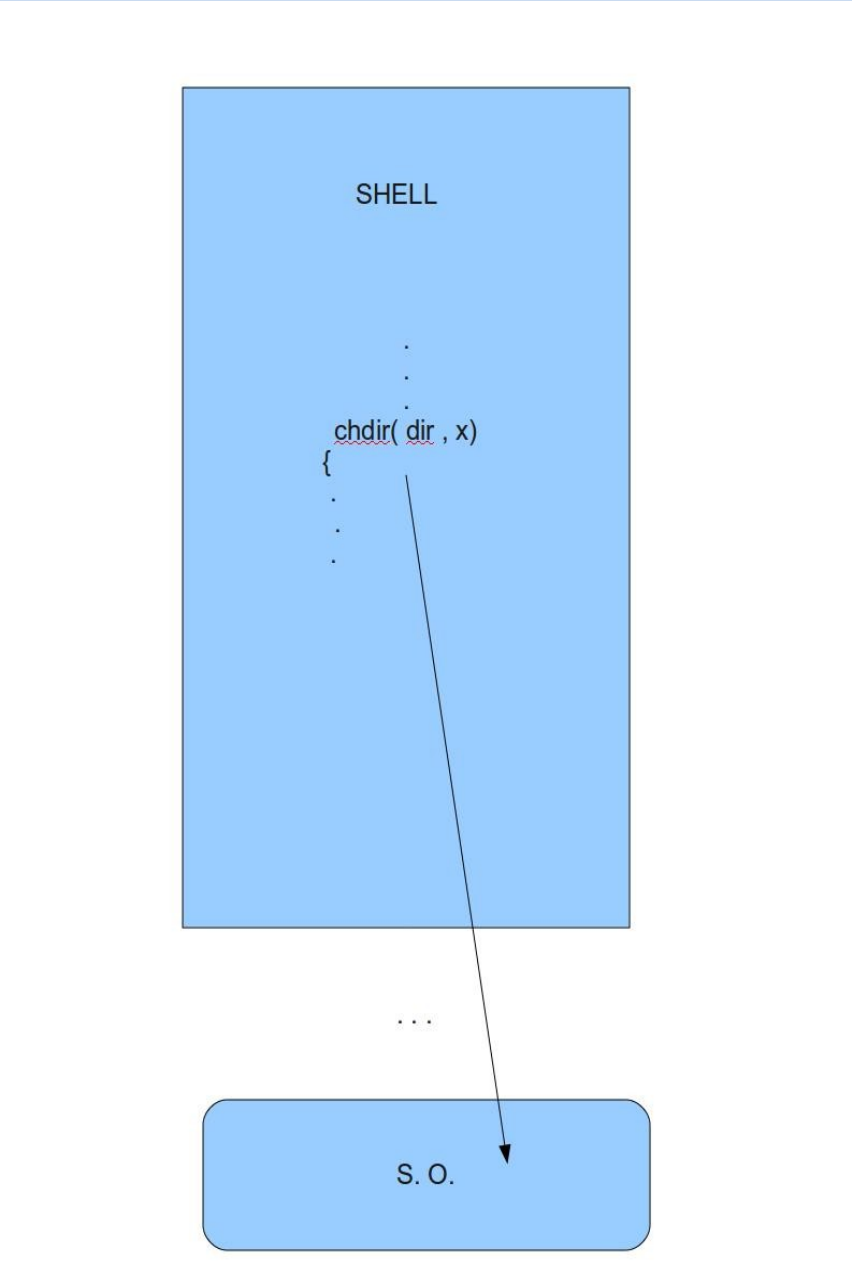
Interpretador de comandos

- Shell é a camada de software que fornece uma interface entre o usuário e o sistema operacional
  - O Shell é capaz de:
    - Executar ou substituir comandos
    - Redirecionar E/S
    - Pipes
  - Possui:
    - Ambiente com configuração de variáveis
    - Interpretador e linguagem de programação

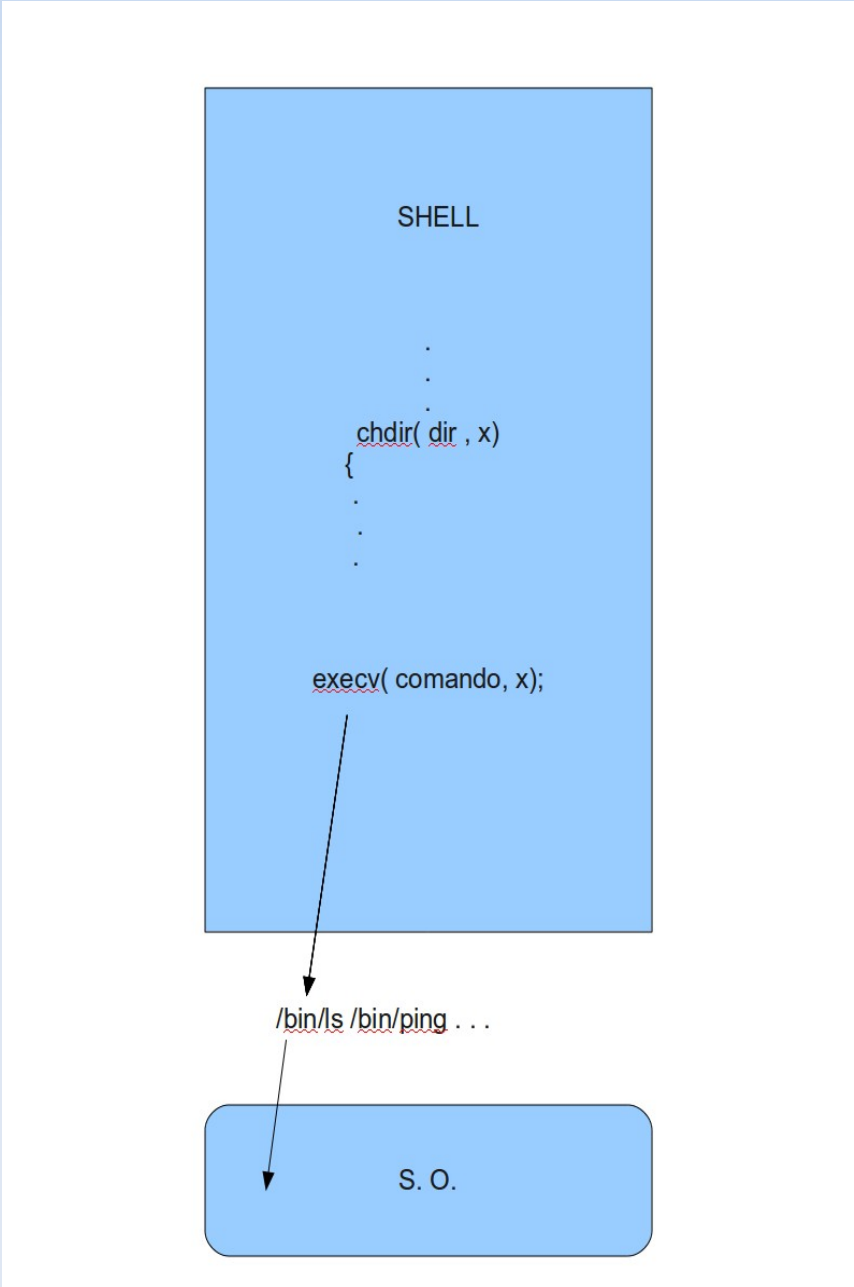
Interpretador de comandos

- Shell: Pode executar diretamente uma função e comando, substitui-los ou repassá-los ao S.O.

Shell executando funções embutidas



Shell executando repassando ao executável externo



Como criar um Shell ????

```
while(1)
    {
        printf("\033[32mMEU PROTOTIPO:>\033[5m\033[0m");
        memset(cmdStr, 0x00, sizeof(cmdStr));
        fgets(cmdStr, sizeof(cmdStr), stdin);
        if( strncmp("quit", cmdStr, 4) == 0 )
        {   break;  }
         else if ( strncmp("autor", cmdStr, 5) == 0 )
        {  printf("Rodrigo Zuolo Carvalho - rodrigoz@fatecsp.br\n");  }
         else if ( strncmp("cd", cmdStr, 2) == 0 )
        { mudadir(cmdStr); }
        else
        { system(cmdStr); }
     }
    return 0;
```

Interpretador de comandos

- Principais shells em Linux:
  - sh – bourne shell. Original do Unix e não possui tantos recursos ou facilidades
  - csh – inspirado em um ambiente compatível com a linguagem C. Promoveu a popularização de comandos de linguagem C nos shells (introduziu o alias).
  - ksh – Korn Shell, um dos mais populares Unix Shells
  - bash – Bourne-Again ShellDefinitivamente o mais popular e com mais recursos dos Shells em Linux (bash2).

Interpretador de comandos

- Bourne again Shell – Recursos

  - Navegação de histórico : ctrl + r, acima e abaixo
  - Atalho para clear : ctrl + l
  - Atalho para exit : ctrl + d
  - Atalho para end line : ctrl + e
  - Atalho para home line : ctrl + a
  - Alias : #> alias lista='ls -al --color'

Interpretador de comandos

- Bash
  - Variáveis de ambiente: parâmetros que definem o funcionamento e valores para os processos e recursos do sistema
  - Variáveis podem ser locais ou globais (exportadas)
  - Processos filhos herdam valores (variáveis) dos pais
  - Variáveis podem ser definidas a qualquer momento da execução de um shell
  - Tipagem fraca de variáveis

Interpretador de comandos

- Obtendo valores:

#> printenv

#> echo $variavel

#> set

- Atribuindo

#> variavel = valor

#> variavel=`ls -la`

Exportando (deixa de ter o contexto/escopo local)

#> export $variavel (ERRADO)

#> export variavel

ATIVIDADE

Execute o seguinte comando

#> export PATH=' '

Agora, execute

#> ls -la

O comando funciona ? Por que não?

Corrija o estado da sessão sem ter realizar o reinício da mesma.

Arquivos de inicialização e configuração

- /etc/profile : perfil padrão e comum a todos usuários
- /etc/profile.d : alternativa usada em algumas distros
- /etc/bash.bashrc : inicialização do shell padrão
- ~/.bashrc : inicialização pessoal do shell
- ~/.profile : configuração personalizada de perfil
- ~/.bash_history : histórico de comandos passados ao shell
- ~/.bash_logout : instruções ao se fechar uma sessão

ATIVIDADE

Faça com que, toda nova sessão iniciada tenha como padrão o reconhecimento do seguinte comando:

#> quereisoueu

Onde, o mesmo deverá exibir na saída o usuário logado na sesão corrente. Ou seja, o *username* detentor do shell em questão.

Interpretador de comandos

- Programação e scripts

Maioria dos shells possui interpretador de comandos que permite a execução de comandos de repetição e comparação.

Utilização para processos em lote e programas simples para administração ou manipulação do sistema

Linguagem interpretada, semelhante ao ambiente C, com poucos recursos de depuração e fraca amarração

Interpretador de comandos

- Linhas interpretadas uma por vez.
- Linha inicial contendo o interpretador é responsável em processar as linhas no arquivo contidas

#! /bin/bash

- Execução dos arquivos de script diretamente da linha de comando

#> source script.sh

#> /bin/bash script.sh

#> ./script.sh

ATIVIDADE

Execute os comandos

#> read x

#> echo $x

Sabendo disso, crie um script que quando executado, solicitará o nome do usuário e em seguida vai cumprimentar esse usuário.

Interpretador de comandos

- Exemplo:

```
#!/bin/sh

lista=`ls *.log`

for file in $lista
do
        echo "copiando $file"
        cp $file $file.novo
done

echo "Script done. "
```

Interpretador de comandos

Usando Funções : a declaração precisa antecipar sua chamada

```
#!/bin/bash
fun ()
{ # A somewhat more complex function.
  i=0
  REPEATS=30
  echo
  echo "And now the fun really begins."
  echo
  sleep 1    # Hey, wait a second!
  while [ $i -lt $REPEATS ]
  do
    echo "---------FUNCTIONS---------->"
    echo "<------------ARE------------"
    echo "<------------FUN------------>"
    echo
    let "i+=1"
  done
}

funky
fun
exit 0
```