

# BASH

## Script

Baltazar Tavares (Psycho Mantys)  
<https://psychomantys.wordpress.com>  
[https://about.me/psycho\\_mantys](https://about.me/psycho_mantys)

# Redireccionamiento

- Já vimos como redirecionar de programa para programa
- Mas como redirecionar um arquivo para a entrada padrão?
  - MeuComando < teste.txt

# Redirecionar para variável

- De vez em quando, é necessário guardar a saída do comando em uma variável
- Podemos fazer assim:
  - SAIDA="\$ ( free -m )"
  - echo "\$ ( free -m )"

# Globbering

- Globbering é o processo de expansão de nomes
- Uma alternativa nativa do bash para REGEX
- Muito útil para lidar com arquivos
- Principal diferença para shell script
- Bash utiliza globbering como padrão interno
- Caso não seja expandido, deixa como esta
- Existem 4 tipos de globbering

# Globbering: Asterisco(\*)

- Relacionado com os arquivos na pasta atual
- Significa “**qualquer coisa neste ponto**”
- Não engloba arquivos começando com ponto(.)
  - A não ser que tenha explicitamente o ponto
  - Ou que seja executado antes “**shopt -s globdot**”

# Globbering: Asterisco(\*)

- Exemplo:

```
Sem título X Sem título X Sem título X
11:35 - psycho : [ samples ] $touch .opa
11:35 - psycho : [ samples ] $ls
expansao.sh* primeiraVez.sh* redirecionar.sh* while.sh*
if.sh* read.sh* segundaVez.sh*
11:35 - psycho : [ samples ] $echo *
expansao.sh if.sh primeiraVez.sh read.sh redirecionar.sh segundaVez.sh while.sh
11:35 - psycho : [ samples ] $shopt -s dotglob
11:35 - psycho : [ samples ] $echo *
.opa expansao.sh if.sh primeiraVez.sh read.sh redirecionar.sh segundaVez.sh while.sh
11:35 - psycho : [ samples ] $echo r*
read.sh redirecionar.sh
11:36 - psycho : [ samples ] $echo r*.sh
read.sh redirecionar.sh
11:36 - psycho : [ samples ] $
```



# Globbering: interrogação(?)

- Relacionado com os arquivos na pasta atual
- Significa “**qualquer carácter neste ponto**”
- Não engloba arquivos começando com ponto(.)
  - A não ser que tenha explicitamente o ponto
  - Ou que seja executado antes “**shopt -s globdot**”

# Globbering: interrogação(?)

- Exemplo:

```
Sem título x Sem título x Sem título x
11:54 - psycho : [ temp ] $ls -A
.00.txt .04.txt .08.txt .12.txt .16.txt 00.txt 04.txt 08.txt 12.txt 16.txt
.01.txt .05.txt .09.txt .13.txt .17.txt 01.txt 05.txt 09.txt 13.txt 17.txt
.02.txt .06.txt .10.txt .14.txt .18.txt 02.txt 06.txt 10.txt 14.txt 18.txt
.03.txt .07.txt .11.txt .15.txt .19.txt 03.txt 07.txt 11.txt 15.txt 19.txt
11:54 - psycho : [ temp ] $echo ?1.txt
01.txt 11.txt
11:56 - psycho : [ temp ] $shopt -s dotglob
11:56 - psycho : [ temp ] $echo ?1.txt
01.txt 11.txt
11:56 - psycho : [ temp ] $echo ??1.txt
.01.txt .11.txt
11:56 - psycho : [ temp ] $echo *1*
.01.txt .10.txt .11.txt .12.txt .13.txt .14.txt .15.txt .16.txt .17.txt .18.txt .19.txt
01.txt 10.txt 11.txt 12.txt 13.txt 14.txt 15.txt 16.txt 17.txt 18.txt 19.txt
11:56 - psycho : [ temp ] $
```

- Relacionado com os arquivos na pasta atual
- Significa “**qualquer carácter desta lista neste ponto**”
- Pode se negar uma lista de caracteres usando o “^”
- Pode se fazer um intervalo usando o carácter “-”
- Não engloba arquivos começando com ponto(.)
  - A não ser que tenha explicitamente o ponto
  - Ou que seja executado antes “**shopt -s globdot**”

# Globbering: Colchetes( [ ] )

- Exemplo:

```
Sem título x Sem título x Sem título x
14:11 - psycho : [ temp ] $echo *
00.txt 01.txt 02.txt 03.txt 04.txt 05.txt 06.txt 07.txt 08.txt 09.txt 10.txt 11.
txt 12.txt 13.txt 14.txt 15.txt 16.txt 17.txt 18.txt 19.txt
14:11 - psycho : [ temp ] $echo ?[365].txt
03.txt 05.txt 06.txt 13.txt 15.txt 16.txt
14:11 - psycho : [ temp ] $echo [.]?[365].txt
[.]?[365].txt
14:11 - psycho : [ temp ] $shopt -s dotglob
14:12 - psycho : [ temp ] $echo [.]?[365].txt
.03.txt .05.txt .06.txt .13.txt .15.txt .16.txt
14:12 - psycho : [ temp ] $
```

- **NÃO** Relacionado com os arquivos na pasta atual
- Os chaves contém uma lista separada por virgula
- Esta lista pode ser um intervalo usando “..”

Sera expandido cada item da lista fazendo uma combinação com uma string que as chaves esteja inserida.

# Globbering: Chaves( {} )

- Exemplo:

```
Sem título X Sem título X
20:04 - psycho : [ temp ] $echo professor{a,}.txt
professora.txt professor.txt
20:05 - psycho : [ temp ] $echo b{a,e,i,o,u}
ba be bi bo bu
20:08 - psycho : [ temp ] $echo b{a..u}
ba bb bc bd be bf bg bh bi bj bk bl bm bn bo bp bq br bs bt
bu
20:09 - psycho : [ temp ] $echo {0..9}
0 1 2 3 4 5 6 7 8 9
20:09 - psycho : [ temp ] $echo {0..9}{0..9}
00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19
20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39
40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59
60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79
80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99
20:09 - psycho : [ temp ] $
```

- Abra um terminal.
- Escreva um programa que:
  - Dentro da pasta atual, crie arquivos com o nome `000.txt` à `999.txt`, e pule os entre `499.txt` e `600.txt`.
  - Depois apague os `423.txt` à `474.txt`.

# Resultados:

**BASH**

```
Sem título X Sem título X
20:45 - psycho : [ temp ] $cat ../glob_ex.sh
#!/bin/bash

for file in {{0..4},{6..9}}{0..9}{0..9} ; do
    echo > "${file}.txt"
done
for file in 4{3..6}{0..9} 42{3..9} 47{0..4} ; do
    rm "${file}.txt"
done
20:46 - psycho : [ temp ] $ls
20:46 - psycho : [ temp ] $../glob_ex.sh
20:46 - psycho : [ temp ] $ls
Display all 848 possibilities? (y or n)
20:46 - psycho : [ temp ] $ls 42
420.txt 421.txt 422.txt
20:46 - psycho : [ temp ] $ls 42
```



# Arquivos

- Tudo é um arquivo
- A chave para programar em shell é saber manipular arquivos
- Cada arquivo aberto tem um numero associado
- Em sistemas menos padronizados, eles se chamam de “handles”.
- Vamos falar sobre alguns arquivos “Especiais”

- São os primeiros arquivos abertos no programa
- Sempre existem pelo menos 3 arquivos padrões:
  - `stdin` ou entrada padrão ou arquivo numero 0
  - `stdout` ou saída padrão ou arquivo numero 1
  - `stderr` ou saída de erro padrão ou arquivo nº 2

# Programas Uteis

- Sempre é bom ter ferramentas já feitas a mão
- Padrão **Posix**
- Programas bons com boas interfaces para shell
- Bastante portátil
- São comandos definidos no shell são padrão

# A Família Do grep

- Comando bastante útil, usado e conhecido.
- Tem alguns irmãos: **egrep** e **fgrep**
- Útil para localizar cadeias de caracteres em textos
- Bastante utilizado para filtrar texto.
- Suporte a **REGEX** estendidas

- Procura por um padrão passado, sem **REGEX**
- Você deve indicar o que sera procurado e os arquivos:
  - fgrep padrão [arquivos]
- Caso os arquivos sejam omitidos, sera usado a entrada padrão.

- Abra um terminal.
- Escreva um programa que:
  - Leia o nome de um usuário
  - Imprima na tela as informações do usuário que estão no arquivo `/etc/passwd`.
  - Imprima na tela as informações do usuário que estão no arquivo `/etc/group`.



# Resultados:

```
Sem título X Sem título X
10:14 - psycho : [ mini-new ] $cat samples/fgrep.sh
#!/bin/bash
echo "Digite o nome de um usuario:"
read nome

fgrep "${nome}" /etc/{passwd,group}
10:14 - psycho : [ mini-new ] $./samples/fgrep.sh
Digite o nome de um usuario:
psycho
/etc/passwd:psycho:x:1000:100:,,,:/home/psycho:/bin/bash
/etc/group:floppy:x:11:root,psycho
/etc/group:audio:x:17:root,psycho
/etc/group:video:x:18:root,psycho
/etc/group:cdrom:x:19:root,psycho
/etc/group:plugdev:x:83:root,psycho
/etc/group:power:x:84:root,psycho
/etc/group:netdev:x:86:root,psycho
10:14 - psycho : [ mini-new ] $
```

# Opções do grep

- Procura recursivamente no diretório:
  - fgrep -R **psycho** /etc
- Para não ser case-sensitive:
  - fgrep -iR **PSYCHO** /etc
- Para inverter busca, mostrando as linhas que não tem:
  - fgrep -v **psycho** /etc/passwd

# Opções do grep

- Contar as ocorrências de um padrão:
  - fgrep -c **psycho** /etc/group
- Como verificar se um padrão foi encontrado:
  - fgrep -qs **psycho** /etc/passwd
  - Não tem nenhum efeito!!
  - Para ser usado num “if”

```
if grep -qs “${USR}” /etc/passwd ; then
    echo “Achou!!”
then
    echo “Não achou :/”
fi
```

- Procura por um padrão passado, com **REGEX**
- Você deve indicar o que sera procurado e os arquivos:
  - `grep 'padrão' [arquivos]`
- Caso os arquivos sejam omitidos, sera usado a entrada padrão.
- Todas as opções do **fgrep** funcionam com o **grep**

- Procura por um padrão passado, com **REGEX** estendida
- Você deve indicar o que sera procurado e os arquivos:
  - `egrep 'padrão' [arquivos]`
- Caso os arquivos sejam omitidos, sera usado a entrada padrão.
- Todas as opções do **fgrep** funcionam com o **egrep**

- “Concatena” o conteúdo de vários arquivos:
  - cat [arquivos]
- Caso os arquivos sejam omitidos, será usado a entrada padrão.
- Exemplo:
  - cat
  - cat `/etc/passwd`
- Opções mais uteis:
  - cat -vet `/etc/passwd`

- Usado para extrair campos ou pedaços de fluxos
- Caso os arquivos sejam omitidos, será usado a entrada padrão.
- Exemplo:
  - `cut [opções] [arquivos]`
- Toda vez que você precisar de um campo, pode usar o `cut`

# Cut para separa campos

- Usando o cut para pegar alguns caracteres:
  - `date | cut -c 9-10 # Pega o dia de hoje`
- Usando o cut para pegar campos:
  - `date | cut -f 3 -d" "`
  - `# Pegando o 3 campo separado pelo campo " "`



# Exemplo 19

- Abra um terminal.
- Escreva um programa que:
  - Leia o nome de um usuário
  - Verifique se o usuário existe através do arquivo [/etc/passwd](#)
  - Se o usuário existir, imprima os grupos que ele pertence através do arquivo [/etc/group](#).

# Resultados Feio e Bobo:

```
Sem título X Sem título X
10:14 - psycho : [ mini-new ] $cat samples/fgrep.sh
#!/bin/bash
echo "Digite o nome de um usuario:"
read nome

fgrep "${nome}" /etc/{passwd,group}
10:14 - psycho : [ mini-new ] $./samples/fgrep.sh
Digite o nome de um usuario:
psycho
/etc/passwd:psycho:x:1000:100:,,,:/home/psycho:/bin/bash
/etc/group:floppy:x:11:root,psycho
/etc/group:audio:x:17:root,psycho
/etc/group:video:x:18:root,psycho
/etc/group:cdrom:x:19:root,psycho
/etc/group:plugdev:x:83:root,psycho
/etc/group:power:x:84:root,psycho
/etc/group:netdev:x:86:root,psycho
10:14 - psycho : [ mini-new ] $
```

```
13:46 - psycho : [ mini-new ] $cat samples/usuario.sh  
#!/bin/bash
```

```
echo 'Digite o nome do usuario:'
```

```
read nome
```

```
if fgrep -qs "${nome}:" /etc/passwd ; then  
    echo "Usuario existe!!"  
    echo "Seus grupos sao:"  
    fgrep "${nome}" /etc/group | cut -f1 -d':'
```

```
else  
    echo "usuario nao existe!"
```

```
fi
```

```
13:46 - psycho : [ mini-new ] $./samples/usuario.sh
```

```
Digite o nome do usuario:
```

```
Psycho
```

```
usuario nao existe!
```

```
13:46 - psycho : [ mini-new ] $./samples/usuario.sh
```

```
Digite o nome do usuario:
```

```
psycho
```

```
Usuario existe!!
```

```
Seus grupos sao:
```

```
floppy
```

```
audio
```

```
video
```

```
cdrom
```

```
plugdev
```

```
power
```

```
netdev
```

```
13:46 - psycho : [ mini-new ] $
```

- [www.shellscript.com.br](http://www.shellscript.com.br)
  - Site do livro muito bom sobre shell.
- <http://aurelio.net/shell/>
  - Site obrigatório de um dos maiores programadores de shell do brasil.
- <http://twiki.softwarelivre.org/TWikiBar/WebHome>
  - Site do “Pai do Shell” Brasileiro, Júlio Neves.
- <http://thobias.org/>
  - Site com muitas dicas uteis, artigos e programas.

# Links

- <http://groups.yahoo.com/group/shell-script>
  - Lista de e-mail sobre shell script
- <https://psychomantys.wordpress.com>
  - Blog pessoal.
- [https://about.me/psycho\\_mantys](https://about.me/psycho_mantys)
  - Onde me encontrar e links sobre minha pessoa
- <http://pt.wikipedia.org/>
  - Site que varias vezes me ajudou!!

Fim!

[psycho@localhost ~]# logout

<https://psychomantys.wordpress.com>  
[https://about.me/psycho\\_mantys](https://about.me/psycho_mantys)

**BASH**

