

## 1. Linha de comando

Esta secção foca alguns aspectos de utilização corrente da linha de comando.

### 1.1 - *Expansão. "Wildcards"*

Um dos mecanismos mais usados da shell é a possibilidade de "apanhar" nomes de ficheiros usando o "\*" ou, mais raramente, o "?".

De forma simples, o significado do "\*" é "qualquer coisa". Mais exactamente, ao ler um nome com \* a interpretação é: no lugar o \* pode figurar qualquer sequência de caracteres, seja vazio ("nada"), um caractere qualquer, dois, etc.

A utilização mais vulgar do \* é no ls. Por exemplo,

```
ls -l *.c
```

lista os nomes de ficheiro cuja nome adira ao padrão "qualquer coisa seguido de .c" (ou seja, os ficheiros com extensão .c). Mas pode ser usado em muitas outras circunstâncias como forma de apanhar um grupo de ficheiros. Por exemplo o comando

```
zip programas.zip *.c
```

cria um ficheiro zip com todos os .c existentes na directória corrente.

## 1.2 - Escapes

O \* e o ? são caracteres especiais para a shell. Como estes há outros, por exemplo \$, & ou >. Se tivermos a ideia não muito feliz de dar nomes a ficheiros contendo caracteres deste tipo podemos ter alguns problemas.

Por exemplo, supondo que tínhamos a ideia não muito feliz de chamar a um ficheiro

```
***star***.c
```

a utilização simples do nome do ficheiro em comandos, por exemplo

```
vi ***star***.c  
gcc ***star***.c -o ***star***
```

seria ambígua. Nesta situação teríamos que enquadrar o nome do ficheiro entre "":

```
vi '***star***.c'  
gcc '***star***.c' -o '***star***'
```

O mesmo aliás se usarmos nomes de ficheiros com espaços. Por exemplo se tivermos a ideia também não muito boa de chamar a um ficheiro

```
exemplo 1.c
```

a utilização simples do nome do ficheiro, por exemplo:

```
vi exemplo 1.c
```

é também ambígua e resolve-se com

```
vi 'exemplo 1.c'
```

Resta acrescentar que `/dev/null` é um ficheiro muito especial: representa um "poço sem fundo" (ou buraco negro) para onde se pode redirecionar qualquer output, que é descartado pelo Unix (e Linux);

Existe também o canal "standard de entrada" (`stdin`) que normalmente está associado ao teclado.

Nos exemplos seguintes, 1) é criado um ficheiro `x`; 2) o ficheiro `x` é usado como canal de entrada para o comando `cat`; 3) copia-se o conteúdo do ficheiro `x`, para um novo ficheiro `y`.

```
echo "teste" > x  
cat < x  
cat < x > y      ou      cat > y < x
```

Finalmente o `|` (designado "pipe") permite encadear comandos fazendo com que o standard output de um comando seja encaminhado para input do programa seguinte.

O exemplo seguinte mostra o encadeamento de 3 comandos: o primeiro mostra o conteúdo do ficheiro `/etc/passwd`; o segundo filtra esse conteúdo mostrando apenas as linhas que contêm a palavra `root`; o terceiro substitui ocorrências da palavra `"root"` pela palavra `"raiz"`.

```
cat /etc/passwd | grep "root" | sed 's/root/raiz/'
```

### 1.3 - Redireccionamento

A maioria dos comandos mandam o resultado para o ecrã. Por exemplo, ao fazer:

```
ls -l
```

o resultado do comando, a lista ficheiros existentes no directório corrente, vai para o ecrã. Se fizermos a seguinte variante

```
ls -l > lista.txt
```

nada aparece no ecrã; o resultado do comando vai, sim, para o ficheiro teste.txt. Dizemos que o resultado do comando é redireccionado para o ficheiro lista.txt.

Este é um exemplo de um mecanismo geral que a shell implementa e que permite redireccionar os canais "standard" de entrada e saída de um programa.

Normalmente a saída de um programa vai para o canal "standard de saída" (stdout) ou, em caso de erro, para o canal "standard de erros" (stderr). Um e outro podem ser redireccionados separadamente.

A forma básica deste mecanismo é o normal redireccionamento da saída. Ex:

```
cat /etc/passwd > x      #saída para o ficheiro x  
cat /etc/passwd >> x     #saída acrescenta-se ao ficheiro x
```

No exemplo seguinte, a saída normal é redireccionada para /dev/null; e a de erros continua no ecrã. Se x não existir, a mensagem de erro vai para o ecrã;

```
cat x 1> /dev/null      # ou apenas: cat x > /dev/null
```

No exemplo seguinte, a saída de erros é redireccionada para /dev/null. Se o ficheiro existir, aparece no ecrã; caso contrário, não aparece nada no ecrã;

```
cat x 2> /dev/null
```

O exemplo seguinte, redirecciona a saída standard para /dev/null e a saída de erros para o mesmo sítio;

```
cat x > /dev/null 2>&1
```

É claro que a PATH só é relevante se o comando for dado com um nome simples. Se ao dar o comando indicar o nome do ficheiro explícita a bash não precisa procurar. Por exemplo, se der um comando:

/bin/ls

a bash tenta executar o programa que se encontre no ficheiro indicado.

Acontece também que, por defeito, o directório corrente não está incluído na lista de directórios da PATH. Isto significa que pode ter um programa mesmo ali à mão no directório corrente e a bash não o encontra - pela simples razão de que não o procura lá.

### Exemplo:

```
cc teste.c -o teste          compila um programa, criando um programa teste  
teste                         a bash não localiza o programa no directório corrente  
teste: not found
```

Há várias formas de resolver isto. Uma é indicar o nome do programa explicitamente.

`./programa` explicitar a localização do ficheiro (comando) a executar

Outra forma é resolver o problema de vez, adicionando o directório corrente à lista de directórios onde a bash procura comandos:

`PATH=$PATH:.` juntar mais um elemento (o . ponto) à lista de directórios

## 1.4 - Variáveis de ambiente

Parte das variáveis de ambiente suportam a configuração de alguns aspectos do funcionamento da própria shell. Por exemplo a variável PS1 configura a prompt que a shell coloca no ecrã para pedir comandos.

Exemplo: o comando

```
PS1="diga> "
```

altera a prompt para uma coisa do género:

```
diga >
```

Uma das variáveis mais importantes da shell é a PATH. Esta variável contém uma lista de nomes de directórios (separados por :). Quando é dado um comando a bash procura esse comando em cada um dos directórios dessa lista.

Mais especificamente, há dois tipos de comandos, internos e externos. Internos são os que a própria bash executa; por exemplo cd, pwd ou exit são comandos internos.

Quando damos um comando externo, estamos na prática, a pedir à bash para executar um outro programa. A bash responde a este pedido em dois passos:

1. tenta localizar o programa; não conseguindo dá um erro;
2. conseguindo, faz um fork() executando no processo filho o programa pedido;

Por exemplo, ao dar o comando ls estamos a pedir à bash para executar um programa chamado ls (sem dizer em que directório esse programa se encontra). A bash vai então procurar um ficheiro chamado ls em cada um dos directórios listados na variável PATH.

Por exemplo, imagine que a variável PATH tem este conteúdo:

```
/home/programas:/etc/aquinaoesta:/bin:/home/maisprogramas
```

ao dar o comando ls a bash vai procurar:

/home/programas/ls	não encontra
/etc/aquinaoesta/ls	não encontra
/bin/ls	encontra ! executa este programa (ou pelo menos tenta)
/home/maisprogramas/ls	aqui nem chega a procurar porque encontrou antes

## 1.5 - subshell; exit

A shell é o programa interativo que aceita e promove a execução dos nossos comandos. Há vários programas diferentes para este efeito (várias shell). A que vamos utilizar é a **bash**. Outras hipóteses são **sh**, **csh**, **tcsh**, **zsh**. Todos estes programas se encontram, normalmente, no directório **/bin**.  
Exemplo: experimente o comando

```
ls -l /bin/*sh*
```

Normalmente a shell é lançada, automaticamente, quando fazemos o *login* num terminal ou quando abrimos uma nova janela de comandos num ambiente gráfico; e termina quando damos o comando **exit**.

Exemplo: experimente o comando **ps**, para ver os processos em curso no terminal/janela de comandos. Um deles será a shell (muito provavelmente o único além do próprio comando **ps** em curso no momento).

Podemos, entretanto, lançar novos processos shell executando o comando respectivo.  
Exemplo: experimente e interprete a seguinte sequência

/bin/bash	lançar uma nova shell
ps	deverá haver pelo menos dois processos shell em curso
exit	
exit	oops...

## 2.1 - *scripts*

Os comandos para a shell, que normalmente são executados interactivamente, podem também ser mandados executar através de um ficheiro.

Exemplo: escreva o seguinte conjunto de comandos num ficheiro testel

```
#!/bin/bash
echo -n "Data: "
date "+%d de %B de %Y"
echo "LISTA DE UTILIZADORES"
who
```

Para executar estes comandos podemos agora mandar "executar o ficheiro".

Normalmente será necessário, primeiro, atribuir a permissão x ao ficheiro. Trata-se de um ficheiro de texto e por isso, à partida, não é normal que tenha a permissão x. Podemos atribuí-la com

```
chmod +x testel
```

ou, por exemplo,

```
chmod 755 testel
```

Feito isso podemos então mandar executar o ficheiro com

```
./testel
```

ou simplesmente

```
testel
```

se o directório corrente estiver na variável \$PATH.

O efeito será idêntico ao que seria obtido pela execução dos comandos, um por um, na linha de comando.

## 2.2 - Variáveis

### 2.2.1 - Criação

Um script simples pode ser uma lista de comandos. Na realidade a shell inclui uma série de outros mecanismos que permitem configurar uma espécie de linguagem de programação.

O primeiro desses mecanismos de programação é a possibilidade de criação de variáveis.

Para definir uma nova variável utiliza-se um comando com a sintaxe:

```
nome_da_variavel=valor
```

Exemplo: Experimente a seguinte sequência de comandos:

```
x>Hello  
set
```

O primeiro cria uma variável da shell com nome x e conteúdo Hello; o segundo comando mostra a lista de variável, onde já deve aparece a recém-criada variável x.

Para ver o valor de uma variável pode-se usar o comando echo. Na sua forma mais simples este comando, echo, permite escrever um texto para o ecrã. Por exemplo:

```
echo Hello World
```

escreve no ecrã, o texto Hello World.

No texto a escrever pode-se incluir uma variável da shell. Para mencionar a variável escreve-se o nome antecedido de \$. Por exemplo:

```
echo $USER
```

```
echo Eu sou, portanto, o utilizador $USER
```

```
echo Eu sou o utilizador $USER e estou a executar a shell $SHELL
```

Exemplo: experimente e interprete a seguinte sequência de comandos:

x=ABC

echo \$x

echo x=\$x

echo x dá x e \\$x dá \$x

x=123

echo x foi modificado para \$x

## 2.2.2 - Manipulação de variáveis

As variáveis podem-se alterar (podem-se redefinir com outro valor). Por exemplo, experimente e interprete a seguinte sequência:

```
x=123  
echo $x  
x=456$x  
echo $x  
x=$x456  
echo $x
```

O valor a atribuir a uma variável pode ser colocado entre " ou ' (que, como habitualmente não ficarão a fazer parte do conteúdo). Este mecanismo é útil para incluir na variável caracteres que possam ter significado especial. Exemplo: o comando

```
x=      ABC
```

não funciona para incluir espaços na variável e aliás dá erro. Pode fazer:

```
x="      ABC"
```

As variáveis da shell representam sequências de texto simples. Em geral o conteúdo de uma variável não é interpretado pela shell (ou seja, é usado literalmente).  
Exemplo - experimente e interprete a seguinte sequência:

```
x=1  
y=2  
z=$x+$y  
echo $z
```

Uma variável pode ser destruída com o comando unset. A utilização de uma variável não definida não provoca qualquer erro - origina, simplesmente, uma cadeia de texto vazia.  
Exemplo - experimente e interprete a seguinte sequência:

```
x=ABC
```

```
set
```

```
echo Valor de x = $x
```

```
unset x
```

```
set
```

```
echo Valor de x = $x
```

x deve aparecer na lista de variáveis

x já não deve aparecer na lista de variáveis

## 2.2.3 - Operações aritméticas

Muitas vezes é necessário fazer contas nos scripts em shell, tal como em qualquer linguagem de programação. O comando `$(( ... ))` permite obter o resultado de expressões numéricas;  
Exemplo:

```
x=5  
echo $(( $x+1 ))
```

## 2.2.4 - Variáveis de ambiente

Identicamente, o script herda as variáveis de ambiente do processo original. Estas variáveis podem ser usadas e alteradas, apenas com efeitos durante a execução.

Considere a seguinte sequência:

```
export x=1  
y=2  
bsh_1c  
echo "x=$x; y=$y; PATH=$PATH"
```

Elabore um script bsh\_1c que mostre que:

- Pode usar e alterar as variáveis x e PATH. E a variável y ?
- Estas alterações não têm efeito no processo original;

As variáveis da shell podem-se coninar a um processo ou ser transmitidas aos processos descendentes. A estas últimas chamamos variáveis de ambiente.

Para criar uma variável de ambiente utiliza-se o comando export, com a sintaxe:

```
export nome-da-variavel=valor
```

Por exemplo, o comando

```
export v="Teste"
```

Cria uma variável de ambiente v.

## Exemplo - experimente e interprete a seguinte sequência

```
x=ABC  
export y=ABC  
set  
/bin/bash  
set  
exit  
set
```

x e y figuram na lista de variáveis da shell em que foram criadas  
abrir uma nova shell  
apenas y figura na nova shell (processo filho) entretanto criada  
voltando à shell original...

As variáveis de ambiente são transmitidas aos processos descendentes por cópia. O processo pode modificar a variável recebida, mas essa modificação não se reflecte na variável existente no processo original.

Exemplo - experimente e interprete a seguinte sequência

```
export y=ABC  
/bin/bash          abrir uma nova shell  
echo $y  
y=123  
echo $y  
exit              voltando à shell original  
echo $y
```

## 2.3 - Argumentos

Um dos mecanismos fundamentais para a construção de scripts são os argumentos passados na linha de comando. Estes argumentos são recebidos no script em variáveis especiais, designadas por:

\$1    \$2    \$3    \$4    ...

exemplo: construa o seguinte script (bsh\_1d)

```
#!/bin/bash
echo "Primeiro argumento: $1"
echo "Segundo argumento: $2"
echo "Terceiro argumento: $3"
echo "Sétimo argumento: ${7}"
echo "Décimo segundo argumento: ${12}"
```

Experimente com:

```
bsh_1d a b c
bsh_1d 1 2 3 4 5 6 7 8 9 10 11 12 13
```

↳ Construa um script para verificar o significado das variáveis especiais \$0, \$# e \$\*

↳ Construa um script para verificar o significado das variáveis especiais \$0, \$# e \$\*

A construção dos argumentos é feita após a substituição dos símbolos especiais pela shell;  
Verifique o valor dos argumentos nas seguintes situações:

```
bsh_1d ~
bsh_1d *
bsh_1d O meu nome de utilizador é $USER
bsh_1d "O meu nome de utilizador é $USER"
bsh_1d $USER $NADA          # $NADA não está definido
bsh_1d '$USER' $USER
bsh_1d '$USER $NADA $HOME'
bsh_1d "$USER $NADA $HOME"
bsh_1d O $NADA NAO EXISTE
bsh_1d O "$NADA" PODE EXISTIR
```

## 2.4 - *read*

Outro dos mecanismos importantes para construir um script é o comando `read`, que permite ler interactivamente um valor para uma variável (uma espécie de `gets` da BASH);

```
#!/bin/bash
echo "Diga qualquer coisa: "
read x
echo "Disse: $x"
```

## 2.5.1 - if

Além de alinhar comandos em sequência, a shell permite a utilização dos mecanismos de controlo habituais numa linguagem de programação, designadamente if's e ciclos.

Na sua forma básica o if permite escolher, para execução, um de dois grupos de comandos alternativos. A forma geral é:

```
if comando
then
    lista-comandos-1
else
    lista-comandos-2
fi
```

Exemplo:

```
if pwd
then
    echo "o pwd funciona"
else
    echo "dificilmente alguém verá isto"
fi
```

↳faça um script bsh\_2d que:

- recebe um argumento
- faz chmod 700 do argumento
- se correu bem diz: "Comando executado com sucesso";
- caso contrário diz: "Comando falhou";

↳faça um script bsh\_2d que:

- recebe um argumento
- faz chmod 700 do argumento
- se correu bem diz: "Comando executado com sucesso";
- caso contrário diz: "Comando falhou";

O comando `test` é especialmente vocacionado para a construção de condições de `if`. Trata-se de um comando que, na prática, serve para verificar uma condição produzindo um resultado adequado à utilização na estrutura do `if`:

Exemplo:

```
echo -n "Username: "
read $x
if test $x = $USER                                # ou if [ $x = $USER ]
then
    echo "Acertou."
else
    echo "Falhou."
fi
```

\*2.5.1.3\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



◀ 2.5.3.3\_shell ✎ 2.5.3.2\_shell ✎ 2.5.3.1\_shell ✎ 2.5.2.2\_shell ✎ 2.5.2.1\_shell ✎ 2.5.1.7\_shell ✎ 2.5.1.6\_shell ✎ 2.5.1.5\_shell ✎ 2.5.1.4\_shell ✎ \*2.5.1.3\_shell ✎ ▶

```
#!/bin/bash
```

```
echo -n "Username: "
```

**read** x

```
if [ $x = $USER ]; then  
    echo "Acertou."
```

**else**

```
echo "Falhou."
```

fi

atafs@caixamaqica: ~/Dropbox/Sistemas Operativos/Aulas9a12

Arquivo Editar Ver Procurar Consola Separadores Ajuda

tafs@caixamaqica: ~/Dropbox/Sistemas Operativos... ✘ atafs@caixamaqica: ~/Dropbox/Sistemas Operativos... ✘

tafs@caixamagica:~/Dropbox/Sistemas Operativos/Aulas9a12\$ ./2.5.1.3 shell

sername:

tafs

certou.

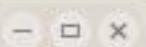
tafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$

Esta forma do comando test permite verificar se duas "strings" são iguais. Se forem, o resultado da execução do comando será "sucesso" (representado por 0). Caso contrário, será "insucesso" (representado pelo valor 1);

```
test "abc" = "xyz"      #ou [ "abc" = "xyz" ]
echo $?
[ "abc" = "xyz" ]       # ou test "abc" = "xyz"
echo $?
[ "abc" != "xyz" ]      # ou test "abc" != "xyz"
echo $?
```

O comando tem duas sintaxes alternativas: utilizar o comando test ou colocar os argumentos entre [ ]; a partir daqui vamos sempre usar esta última; mas não se esqueça que [...] representa um comando test !

2.5.1.4\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit



Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



2.5.3.3\_shell 2.5.3.2\_shell 2.5.3.1\_shell 2.5.2.2\_shell 2.5.2.1\_shell 2.5.1.7\_shell 2.5.1.6\_shell 2.5.1.5\_shell 2.5.1.4\_shell 2.5.1.3\_shell

**#!/bin/bash**

```
test "abc" = "xyz"          #ou [ "abc" = "xyz" ]
echo $?

[ "abc" = "xyz" ]          # ou test "abc" = "xyz"
echo $?

[ "abc" != "xyz" ]          # ou test "abc" != "xyz"
echo $?
echo "Qui 03 M"
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12



Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12\$

```
drwxrwxr-x 2 atafs atafs 4096 2012-04-26 21:45 dúvidas a esclarecer
drwxrwxr-x 2 atafs atafs 4096 2012-04-20 20:27 feito na aula
-rwxrwxr-x 1 atafs atafs 320 2012-05-02 20:45 shell_1.sh~
-rw-rw-r-- 1 atafs atafs 341 2012-05-02 20:33 shell_2.sh~
-rw-rw-r-- 1 atafs atafs 222 2012-05-02 21:02 shell_5.sh~
-rwxr-xr-x 1 atafs atafs 789 2012-05-02 22:30 shell_7.sh~
-rw-rw-r-- 1 atafs atafs 410 2012-05-02 21:33 shell_8.sh~
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ chmod 755 2.5.1.4_shell
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./2.5.1.4_shell
```

1

1

0

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./2.5.1.4\_shell

1

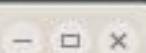
1

0

Qui 03 Mai 2012 12:07:11 WEST

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$

2.5.1.5\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit



Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

Abrir ▾ Guardar Desfazer

2.5.4.2\_shell ✎ 2.5.4.1\_shell ✎ 2.5.3.3\_shell ✎ 2.5.3.2\_shell ✎ 2.5.3.1\_shell ✎ 2.5.2.2\_shell ✎ 2.5.2.1\_shell ✎ 2.5.1.7\_shell ✎ 2.5.1.6\_shell ✎ 2.5.1.5\_shell ✎

**#!/bin/bash**

```
echo -n "Nome: "; read x
if [ -z "$x" ]; then
    echo "(vazio)"
else
    if [ $x = $USER ]; then
        echo "Acertou."
    else
        echo "Falhou."
    fi
fi
```

```
#As operações -z e -n permitem verificar, respectivamente,
#vazia (tamanho > 0)
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./2.5.1.5\_shell

Nome: atafs

Acertou.

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./2.5.1.5\_shell

Nome: sadfdsaf

Falhou.

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12\$

2.5.1.6\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

**#!/bin/bash**

```
echo -n "Nome: "; read x
if [ -z "$x" ]; then
    echo "(vazio)"
elif [ $x = $USER ]; then
    echo "Acertou"
else
    echo "Falhou."
fi

# exemplo anterior usa dois ifs encadeados.
# Em alternativa pode ser usada a sintaxe if-elif-else-fi i
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... ./2.5.1.6\_shell

Nome: atafs

Acertou

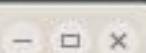
atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./2.5.1.6\_shell

Nome: sdfsd

Falhou.

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12\$

2.5.1.7\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit



Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



2.5.4.4\_shell ✘ 2.5.4.3\_shell ✘ 2.5.4.2\_shell ✘ 2.5.4.1\_shell ✘ 2.5.3.3\_shell ✘ 2.5.3.2\_shell ✘ 2.5.3.1\_shell ✘ 2.5.2.2\_shell ✘ 2.5.2.1\_shell ✘ 2.5.1.7\_shell ✘

**#!/bin/bash**

```
# -lt : less than
# -gt : greater than
#(para uma lista completa dos operadores veja o manual do comando test)
# Faça outras versões do mesmo script usando as opções -eq e -gt;
```

```
#O mesmo comando permite comparar números inteiros; alguma
#script seguinte:
```

```
x=700
n
while [ -z $n ] || [ $n -ne $x ]; do
    echo -n "diga um número: "; read n
    if [ ! -z $n ]; then
        if [ $n -lt $x ] ; then
            echo "Abaixo"
        elif [ $n -gt $x ] ; then
            echo "Acima"
        else
            echo "É esse mesmo!!"
        fi
    fi
done
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

```
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos... ✘ ataf... ✘ ataf... ✘
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./2.5.1.7_shell
./2.5.1.7_shell: linha 14: n: comando não reconhecido
diga um número: 23
Abaixo
diga um número: 567567
Acima
diga um número: 700
É esse mesmo!!
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

2.5.2.1\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

- □ ×

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

**#!/bin/bash**

#2.5.2 - Teste com ficheiros  
#Um outro grupo importante de opções do comando test serve para testar condições sobre ficheiros;  
#Por exemplo, o seguinte script verifica se um dado ficheiro existe:

```
echo -n "Insira o nome do ficheiro: "
read y
if [ -f $y ]; then
    echo ""
    echo "$y existe"
    echo ""
else
    echo "$y não existe"
fi
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... × atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... ×

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./2.5.2.1\_shell

Insira o nome do ficheiro: teste.sh

teste.sh existe

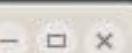
atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./2.5.2.1\_shell

Insira o nome do ficheiro: sdfdsf.sh

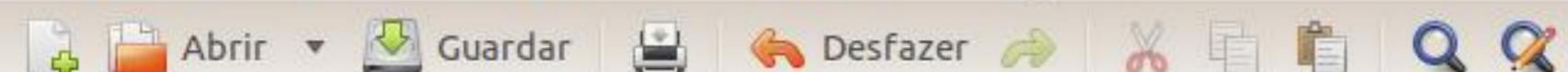
sdfdsf.sh não existe

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ █

2.5.3.2\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit



Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

**#!/bin/bash****#A construção do padrão admite mecanismos que são mais ou menos familiares:****#| - alternativas (or)****##\* - qualquer cadeia de caracteres (0 ou mais)****#? - um caracter qualquer****#0 padrão \* captura todas as sequências, conseguindo-se assim uma forma de obter um "default";**

echo -n "Insira um Club: "; read x

case \$x in  
benfica|sporting) echo "Lisboa";;  
porto|boavista) echo "Porto";;  
\*) echo "Outras cidades";;

esac

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... ./2.5.3.2\_shell

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./2.5.3.2\_shell

Insira um Club: porto

Porto

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./2.5.3.2\_shell

Insira um Club: benfica

Lisboa

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./2.5.3.2\_shell

Insira um Club: asdasda

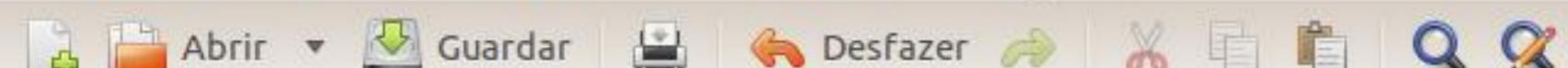
Outras cidades

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$

2.5.3.3\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

x

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



3.1.2\_shell 3.1.1\_shell 2.5.6\_shell 2.5.5\_shell 2.5.4.4\_shell 2.5.4.3\_shell 2.5.4.2\_shell 2.5.4.1\_shell 2.5.3.3\_shell 2.5.2.2\_shell

**#!/bin/bash****#Um outro exemplo... o que faz ?**

```
echo "Insira um ficheiro: "; read x
case $x in
  *.txt) echo "ficheiro de texto";;
  *.C|*.c|*.h) echo "c/c++";;
  *) echo "outros ficheiros";;
esac
```

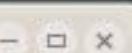
atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... x atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... x

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./2.5.3.3_shell
Insira um ficheiro:
teste.txt
ficheiro de texto
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./2.5.3.3_shell
Insira um ficheiro:
teste.C
c/c++
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

2.5.4.1\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit



Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

Abrir Guardar Desfazer

3.2\_shell 3.1.2\_shell 3.1.1\_shell 2.5.6\_shell 2.5.5\_shell 2.5.4.4\_shell 2.5.4.3\_shell 2.5.4.2\_shell 2.5.4.1\_shell 2.5.2.2\_shell

**#!/bin/bash**

```
#2.5.4 - For  
#O comando for permite repetir um conjunto de comandos, para cada um dos elementos de uma lista.  
#A sua sintaxe é
```

```
#for variavel in lista  
#do  
#lista-de-comandos  
#done
```

```
for i in "a b c"  
do  
echo "i= $i"  
done
```



atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... ./2.5.4.1\_shell

```
i= a b c  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./2.5.4.1_shell  
i= a  
i= b  
i= c  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./2.5.4.1_shell  
i= a b c  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ 
```



.goutputstream-D1Y3DW deleted ".goutputstream-D1Y...

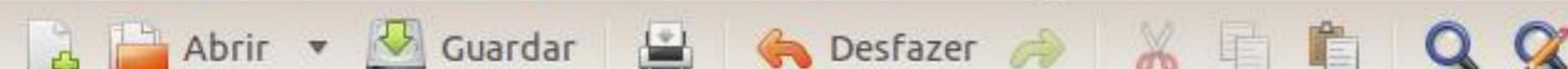
Largura do Tabulador: 8

Ln 14, Col 17

INS

2.5.4.2\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

**#!/bin/bash**

#Muitas vezes o for é utilizado para percorrer a lista de ficheiros de um directório. Para formação dessa lista aplicam-se os mecanismos habituais de expansão da shell:

```
for i in *.sh
do
    echo "entrada: $i"
done
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

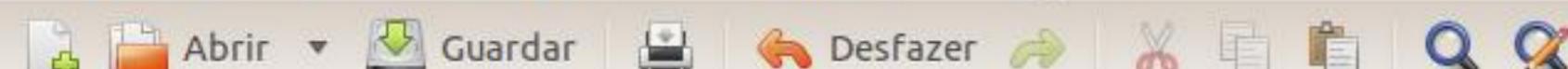
atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... ./2.5.4.2\_shell

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./2.5.4.2_shell
entrada: aula_shell_1.sh
entrada: aula_shell_2.sh
entrada: aula_shell_3.sh
entrada: aula_shell_4.sh
entrada: aula_shell_5.sh
entrada: aula_shell_6.sh
entrada: aula_shell_7.sh
entrada: aula_shell_8.sh
entrada: aula_shell_9.sh
entrada: net_shell_procurar_ficheiros_em_directorias.sh
entrada: teste.sh
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

2.5.4.3\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

- □ ×

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

**#!/bin/bash**

#Exemplo: listar ficheiros (excluindo os directórios)

```
for i in *.sh; do
    if [ ! -d $i ] ; then
        ls -l $i
    fi
done
```



atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... × atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... ×

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./2.5.4.3_shell
-rwxr-xr-x 1 atafs atafs 377 2012-05-03 13:22 aula_shell_1.sh
-rwxr-xr-x 1 atafs atafs 167 2012-05-03 13:25 aula_shell_2.sh
-rwxr-xr-x 1 atafs atafs 354 2012-05-03 13:14 aula_shell_3.sh
-rwxr-xr-x 1 atafs atafs 321 2012-05-03 13:28 aula_shell_4.sh
-rwxr-xr-x 1 atafs atafs 285 2012-05-03 13:58 aula_shell_5.sh
-rwxr-xr-x 1 atafs atafs 372 2012-05-03 14:23 aula_shell_6.sh
-rwxr-xr-x 1 atafs atafs 813 2012-05-03 14:30 aula_shell_7.sh
-rwxr-xr-x 1 atafs atafs 426 2012-05-03 14:31 aula_shell_8.sh
-rwxr-xr-x 1 atafs atafs 441 2012-05-03 15:55 aula_shell_9.sh
./2.5.4.3_shell: linha 8: [: demasiados argumentos
-rwxr-xr-x 1 atafs atafs 190 2012-05-03 14:11 teste.sh
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

2.5.4.4\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



3.5.1\_shell 3.4\_shell 3.3\_shell 3.2\_shell 3.1.2\_shell 3.1.1\_shell 2.5.6\_shell 2.5.5\_shell 2.5.4.4\_shell 9.9.9\_shell\_Duvidas

**#!/bin/bash**

```
#Como vimos anteriormente, as variáveis especiais $1, $2, ... representam os argumentos do
#comando. Estas variáveis são adequadas para aceder aos argumentos um a um. Para aceder aos
#argumentos num ciclo são úteis as seguintes outras variáveis especiais:
#$* - todos os argumentos
#$# - o número de argumentos
```

```
echo "foram dados $# argumentos, que são: "
for a in $* ; do
    echo "-> $a"
done
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

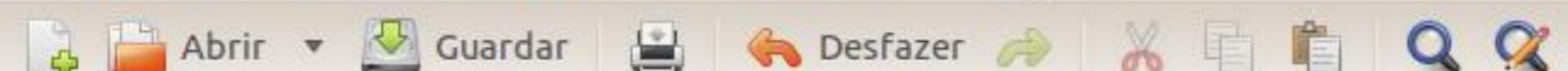
Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... ./2.5.4.4\_shell 3 6 8

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./2.5.4.4_shell 3 6 8
foram dados 3 argumentos, que são:
-> 3
-> 6
-> 8
```

2.5.5 shell (~Dropbox/Sistemas Operativos/Aulas9a12) - qedi

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



◀ 3.5.2\_shell ✎ 3.5.1\_shell ✎ 3.4\_shell ✎ 3.3\_shell ✎ 3.2\_shell ✎ 3.1.2\_shell ✎ 3.1.1\_shell ✎ 2.5.6\_shell ✎ 2.5.1\_shell ✎

```
#!/bin/bash
```

#0 ciclo while é um ciclo de condição: repete a execução de um bloco de comandos enquanto se verificar uma dada condição de controlo (ou até ela deixar de se verificar);

```
#while [ condição ]
```

#do

## #lista-de-comando

#done

```
#Altere o exemplo anterior para fazer um pequeno jogo de  
#deve ajudar dizendo se o valor certo é para "Cima" ou pa
```

v=700

x≡6

```
while [ $x != 700 ]
```

do

```
echo -n "Adivinhe o numero: "; read x
```

done

done

```
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12
Ficheiro Editar Ver Procurar Consola Separadores Ajuda
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos... ✘ atafsa@caixamagica: ~/Dropbox/Sistemas_Operativos... ✘
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./2.5.5_shell
Adivinhe o numero: 23
Adivinhe o numero: 35646
Adivinhe o numero: 700
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ █
```

2.5.6\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



3.5.3\_shell 3.5.2\_shell 3.5.1\_shell 3.4\_shell 3.3\_shell 3.2\_shell 3.1.2\_shell 3.1.1\_shell 2.5.6\_shell 9.9.9\_shell\_Duvidas

```
#!/bin/bash
```

```
#2.5.6 - Substituição  
#Os delimitadores `` substituem um comando (escrito entre os dois delimitadores) pelo seu output;  
  
#Este mecanismo é um precioso auxiliar para scripts. Por exemplo, pode ser utilizado para o backup  
#de um ficheiro com a data actual.
```

```
x=`date`  
echo ""  
echo $x  
echo ""
```

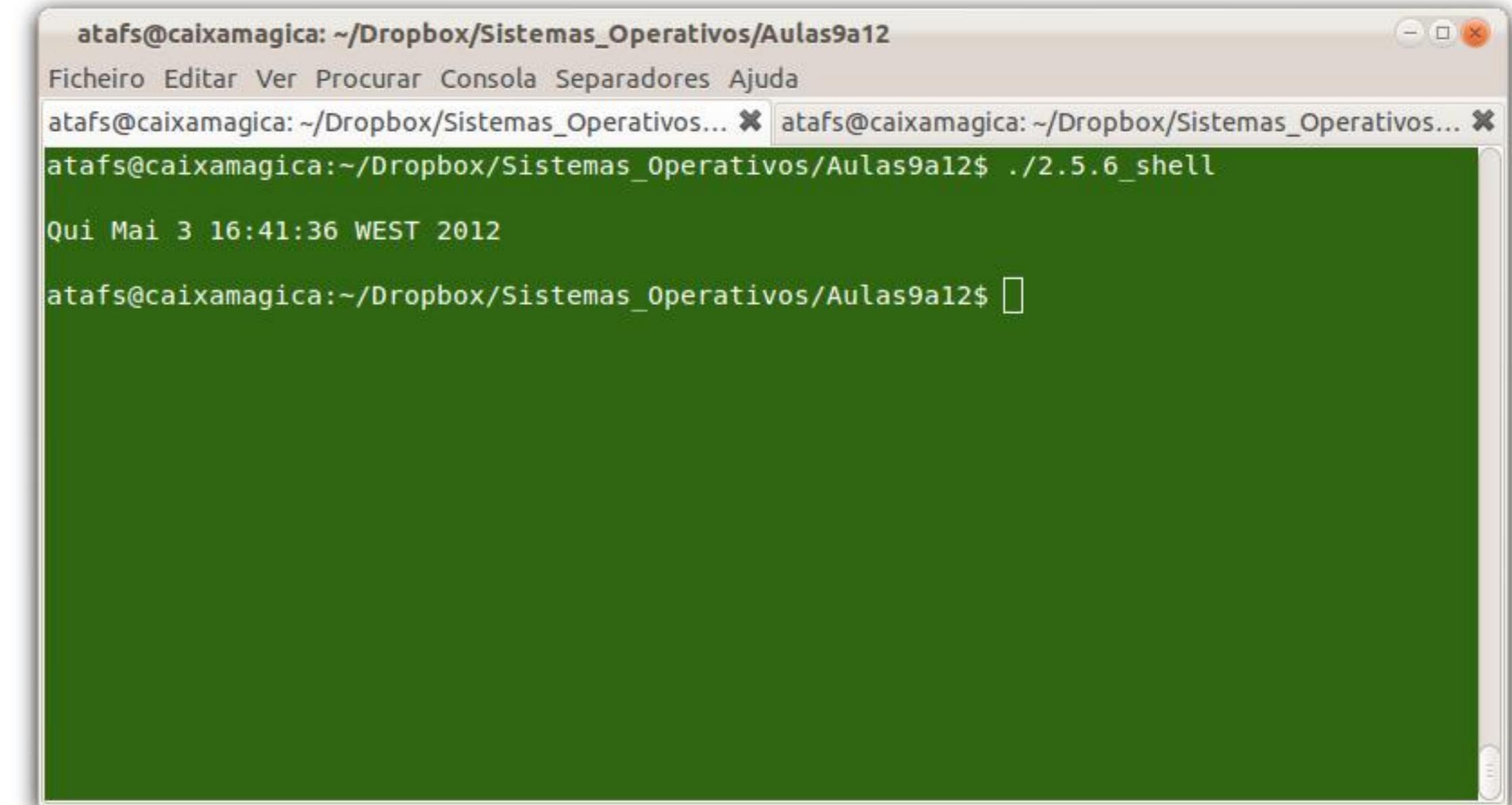
atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... ./2.5.6\_shell

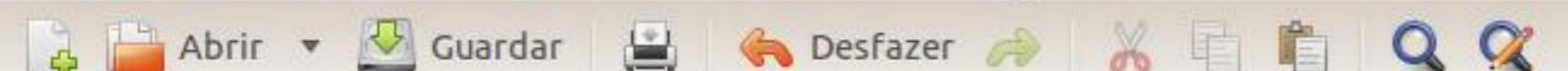
Qui Mai 3 16:41:36 WEST 2012

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$



3.1.1 shell (~Dropbox/Sistemas Operativos/Aulas9a12) - qedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



◀ 3.6.1\_shell ✎ 3.5.3\_shell ✎ 3.5.2\_shell ✎ 3.5.1\_shell ✎ 3.4\_shell ✎ 3.3\_shell ✎ 3.2\_shell ✎ 3.1.2\_shell ✎ 3.1.1\_shell ✎ 9.9.9\_shell\_Duvidas ✎ ▶

```
#!/bin/bash
```

### #3. Comandos

#Alguns comandos de apoio a scripts

#3.1 - expr

# o comando expr (/bin/expr) contempla também algumas operações com strings: em particular é

#a forma mais fácil de fazer uma operação importante que

#Exemplo: o script seguinte lê uma string e indica em que

```
echo -n "String: "
```

**read**

```
echo `expr index $s a`
```

```
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12
Ficheiro Editar Ver Procurar Consola Separadores Ajuda
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos... ✘ atafs@caixamagica: ~/Dropbox/Sistemas_Operativos... ✘
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./3.1.1_shell
String: americ
1
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./3.1.1_shell
String: tomas
4
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./3.1.1_shell
String: agrela
1
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./3.1.1_shell
String: 54149
0
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ █
```

3.1.2\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



3.6.2\_shell 3.6.1\_shell 3.5.3\_shell 3.5.2\_shell 3.5.1\_shell 3.4\_shell 3.3\_shell 3.2\_shell 3.1.2\_shell 9.9.9\_shell\_Duvidas

```
#!/bin/bash
```

```
#Na realidade o comando expr serve para fazer o cálculo de expressões simples, quer em texto quer
#em números inteiros, sendo portanto, também, um alternativa ou complemento ao cálculo numérico
#com $(( ));
```

#Ex: o seguinte script lê uma string, representando o nome completo de uma pessoa, e mostra apenas
#o primeiro nome próprio:

```
#Mais exercícios:
```

```
#exercício (faça todos os cálculos com expr);
#- altere para mostrar o primeiro e último nome;
#- altere para mostrar o último apelido e depois todos os
```

```
echo -n "String: "; read s

n=`expr index "$s" " "
if [ $n -gt 0 ] ; then
    m=`expr $n - 1`
fi
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... ./3.1.2\_shell

String: Américo Tomás

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$

sh ▾ Largura do Tabulador: 8 ▾ Ln 25, Col 1 INS

\*3.2\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

```
#!/bin/bash
```

```
#3.2 - grep
```

```
#0 comando grep permite procurar uma string num ficheiro. Na forma normal o comando mostra as  
#linhas do ficheiro que contêm esse padrão.  
#Com a opção -c mostra o número de linhas nas  
#mesmas condições;
```

```
# Mais Exercícios:
```

```
# Faça um script que indique o número de ficheiros .c , .h  
# Faça um script que receba um argumento e indique cada um  
# aparece e quantas vezes;  
# Faça um script que indique os ficheiros .h pendurados (n  
# Faça um script que indique, para cada ficheiro .h, a lis
```

```
grep $USER /etc/passwd  
grep -c $USER /etc/passwd
```



atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

```
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos... ✘ atafs@caixamagica: ~/Dropbox/Sistemas_Operativos... ✘  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./3.2_shell  
atafs:x:1000:1000:Atafs,,,:/home/atafs:/bin/bash  
1  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

3.3\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.1.2\_shell 4.1.1\_shell 3.6.2\_shell 3.6.1\_shell 3.5.3\_shell 3.5.2\_shell 3.5.1\_shell 3.4\_shell 3.3\_shell 9.9.9\_shell\_Duvidas

```
#!/bin/bash
```

```
#3.3 - basename
```

#Há um conjunto de comandos que, embora possam ser usados interactivamente, são especialmente importantes no quadro da programação com a shell; é o caso, por exemplo, do comando basename;

#o comando extrai o nome base de um nome completo de ficheiro, retirando:

```
#- o caminho até ao directório;  
#- a parte final (se for igual ao segundo argumento)
```

```
basename /etc/passwd      #passwd  
basename teste.c .c       #teste  
basename /etc/rc.1 .1       #rc  
basename teste.c .x       #teste.c
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

```
atafs@caixamagica:~/Dropbox/... ./3.3_shell  
passwd  
teste  
rc  
teste.c  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```



3.4\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

**#!/bin/bash****#3.4 - wc**

#0 comando wc dá indicações sobre a dimensão de um ficheiro: tipicamente o número de linhas,  
#palavras e caracteres; a opção -l dá apenas o número de linhas;

**#Mais exercícios:**

#faça um script que indique o número de linhas de código de todos os ficheiros .c  
#ficheiros .h e .c);

**#exemplo:**

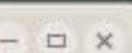
```
cat /etc/passwd | wc  
cat /etc/passwd | wc -l
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

```
atafs@caixamagica: ~/Dropbox/... ✘ ataf... ✘ ataf... ✘ ataf... ✘  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./3.4_shell  
      33      55    1614  
33  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

3.5.1\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit



Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

Abrir ▾ Guardar Desfazer

4.2\_shell X 4.1.3\_shell X 4.1.2\_shell X 4.1.1\_shell X 3.6.2\_shell X 3.6.1\_shell X 3.5.3\_shell X 3.5.2\_shell X 3.5.1\_shell X 9.9.9\_shell\_Duvidas X

**#!/bin/bash****#3.5 - sed**

#Alguns comandos do Unix são especialmente importantes quando se trata de scripts; Um desses comandos é o sed, que introduz também a noção de expressões regulares;  
#Essencialmente, o sed lê uma entrada e produz uma saída transformada;  
#A transformação a fazer é indicada num comando dado como argumento sed;  
#Há duas transformações importantes:  
#- Seleccionar parte das linhas do ficheiro original;  
#- Substituir parte do texto do ficheiro original;

#o seguinte comando selecciona as linhas do ficheiro /etc/

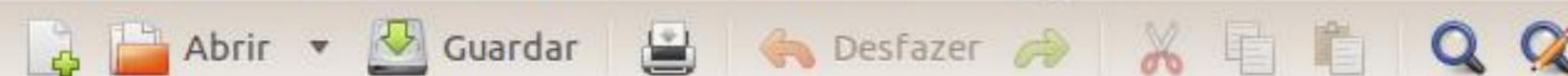
sed -n '/root/p' /etc/passwd



atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12  
Ficheiro Editar Ver Procurar Consola Separadores Ajuda  
atafs@caixamagica: ~/Dropbox/... X ataf... X ataf... X  
atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./3.5.1\_shell  
root:x:0:root:/root:/bin/bash  
atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$

3.5.2\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



3.5.2\_shell

**#!/bin/bash**

#A opção -n faz com que o sed reproduza apenas as linhas em que isso é pedido explicitamente;  
#nesta caso aparecem apenas as linhas seleccionadas pelo comando que tem a forma:

**#/expressão-regular/p**

#o efeito deste comando é escrever (o "p" é de print) as linhas que aderem à expressão-regular

#indicada; neste caso aderem à expressão regular as linhas

#Numa expressão-regular podem-se usar símbolos com signifi

#flexibilidade na localização de sequências de texto; essa

# \* Qualquer sequência de caracteres;

# ? - Um carácter qualquer

# ^ - Início da linha

# \$ - Fim da linha

# [ ] - um dos caracteres indicados;

# \x - escape de um caractere especial; ex \\$ para r

#Listar apenas as directorias

**ls -l | sed -n '/^d/p'**

#Listar os ficheiros executáveis

**ls -l | sed -n '/^??x??x??x/p'**

#Os [ ] denotam um carácter, dentro dos indicados na list

#que seja a ou b ou c;

#Na lista podem figurar intervalos; ex: [a-z] denota uma l

#O símbolo ^ antes da lista denota todos os caracteres men

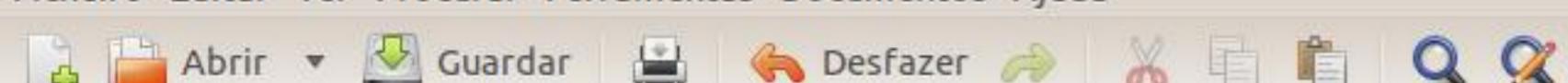
#ex: [^abc] qualquer carácter que não seja o a ou b ou c;

#ex: [^a-b] qualquer carácter que não seja uma letra minú

```
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12
atafs@caixamagica: ~/Dropbox/... ✘ atafsa@caixamagica: ~/Dropbox/S... ✘ atafsa@caixamagica: ~/Dropbox/S... ✘
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./3.5.2_shell
drwxrwxr-x 2 atafs atafs 4096 2012-04-26 21:45 dúvidas a esclarecer
drwxrwxr-x 2 atafs atafs 4096 2012-04-20 20:27 feito na aula
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

3.5.3.2\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



3.5.3.2\_shell 4.8.2\_shell 4.8.1\_shell 4.7.2\_shell 4.7.1\_shell 4.6.3\_shell 4.6.2\_shell 4.6.1\_shell 4.5.4\_shell 4.5.3\_shell 4.5.2\_shell

**#!/bin/bash**

#0 comando seguinte exemplifica

#sed 's/a/Aluno/' /etc/passwd

#Neste caso o comando para o s  
#(como é evidente, o sed não a  
#Este comando sed tem a forma

#s/expressão-1/expressão-2/

#e tem como efeito substituir  
#(s/expressão-1/expressão-2/g

#Exemplo: eliminar a palavra "do" e substituir todos os "a" por "AA"

echo "maria **do** carmo joaquina" | sed 's/do //g; s/a/AA/g'

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

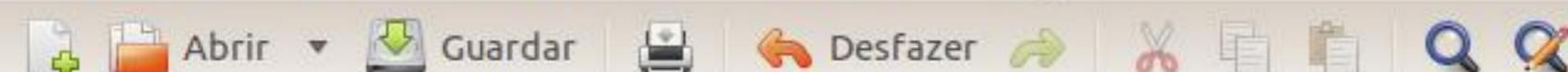
Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

./3.5.3.2\_shell: linha 22: s/a/A/g': Ficheiro ou directória inexistente  
atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ echo "Maria do Ca  
rmos Joaquina" | sed 's/do //g; s/a/A/g'  
sed: -e expressão #1, carácter 1: comando desconhecido: 'o'  
bash: s/a/A/g': Ficheiro ou directória inexistente  
atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ echo "Maria do Carmos Joaquina" | sed 's/do //g; s/a/A/g'  
sed: -e expressão #1, carácter 1: comando desconhecido: 'o'  
bash: s/a/A/g': Ficheiro ou directória inexistente  
atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$

3.6.1.1\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.3.5\_shell 4.3.4\_shell 4.3.3\_shell 4.3.2\_shell 4.3.1\_shell 4.2\_shell 4.1.3\_shell 4.1.2\_shell 4.1.1\_shell 3.6.2\_shell 3.6.1.1\_shell

```
#!/bin/bash
```

```
#3.6 - awk
```

```
#0 awk é um comando muito ampliador de script. Ficheiro Editar Ver Procurar Consola Separadores Ajuda  
#scripts de manipulação de texto. O exemplo seguinte mostra como usar o awk.
```

```
#ls -l | awk '{print $1 $3 ;'
```

```
#a exemplo do sed, também o awk pode ser usado para processar o resultado de outros comandos.  
#o comando é indicado entre { e }.  
#Neste caso a instrução é o print.
```

```
#o que importa aqui é o facto de o awk, ao processar cada linha, dividir os campos (separados por  
#espaços) em variáveis $1 $2 etc; assim, ao escrever estas variáveis, estamos a escrever as colunas  
#1 e 3 do resultado do ls -l;  
#O exemplo anterior imprime todas as linhas; uma variante é um comando da forma:
```

```
#/expressão-regular/ { comandos }
```

```
#que aplica o bloco de comandos apenas à linhas seleccionadas pela expressão-regular. Exemplo:  
#Mostra o mesmo conteúdo mas apenas para os directórios;
```

```
ls -l | awk '/^d/ { print $1 $3 }'
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ chmod 755 4.6.1.1\_shell

chmod: impossível aceder a «4.6.1.1\_shell»: Ficheiro ou directoria inexistente

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ chmod 755 4.6.1.1\_shell

chmod: impossível aceder a «4.6.1.1\_shell»: Ficheiro ou directoria inexistente

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ chmod 755 3.6.1.1\_shell

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./3.6.1.1\_shell

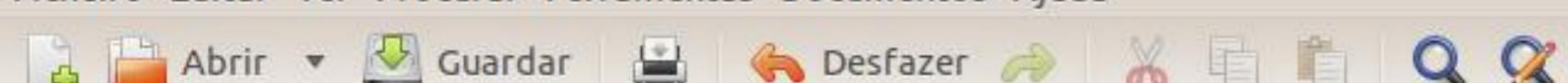
drwxrwxr-xatafs

drwxrwxr-xatafs

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12\$

\*3.6.1\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.3.5\_shell 4.3.4\_shell 4.3.3\_shell 4.3.2\_shell 4.3.1\_shell 4.2\_shell 4.1.3\_shell 4.1.2\_shell 4.1.1\_shell 3.6.2\_shell \*3.6.1\_shell

**#!/bin/bash****#3.6 - awk**

#0 awk é um comando muito ampli Ficheiro Editar Ver Procurar Consola Separadores Ajuda

#scripts de manipulação de tex atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

#0 exemplo seguinte mostra ape atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

-rwxrwxr-xatafs  
-rw-rw-r--atafs  
-rw-rw-r--atafs  
-rwxr-xr-xatafs  
-rw-rw-r--atafs  
-rwxr-xr-xatafs  
-rwxr-xr-xatafsls -l | awk '{print \$1 \$3 ; }'  
  
#a exemplo do sed, também o aw atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

#o comando é indicado entre { atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

#Neste caso a instrução é o pr

./3.6.1\_shell: linha 28: -l: comando não reconhecido atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

#o que importa aqui é o facto de o awk, ao processar cada linha, dividir os campos (separados por

#espaços) em variáveis \$1 \$2 etc; assim, ao escrever estas variáveis, estamos a escrever as colunas

#1 e 3 do resultado do ls -l;

#0 exemplo anterior imprime todas as linhas; uma variante é um comando da forma:

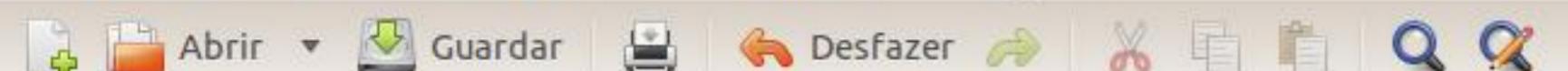
**#expressão-regular/ { comandos }****#que aplica o bloco de comandos apenas à linhas seleccionadas pela expressão-regular. Exemplo:****#Mostra o mesmo conteúdo mas apenas para os directórios;**

# ls -l | awk '/^d/ { print \$1 \$3 }'



3.6.2.1\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.3.6\_shell 4.3.5\_shell 4.3.4\_shell 4.3.3\_shell 4.3.2\_shell 4.3.1\_shell 4.2\_shell 4.1.3\_shell 4.1.2\_shell 4.1.1\_shell 3.6.2.1\_shell

#Trata de forma diferente directórios e ficheiros comuns;  
#É ainda possível associar um bloco de comandos a uma situação especial, denotada pela marca  
#BEGIN que ocorre antes da leitura de entrada; isto é útil para fixar alguns parâmetros importantes  
#como o FS que indica o carácter de separação dos campos:  
#Exemplo:

```
cat /etc/passwd | awk 'BEGIN { FS=":" ; } { print $1 $5; }'
```

#Mostra o nome e descrição  
#Exemplo:

#Considere o ficheiro user  
#josesilva:sporting:1212-1  
#cebola:x:2333-2222-232322  
#tintim:stromp:2323-2323-2  
#jaquim:ola:2232-1111-1111  
#pedro:benfica:3333-3333-3  
  
#O comando seguinte extrai

```
#cat users.txt| awk -F':'
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 bash: ./3.6.2.1\_shell: Permissão negada

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ chmod 755 3.6.2.1\_shell

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./3.6.2.1\_shell

rootroot

daemondaemon

binbin

syssys

syncsync

gamesgames

manman

lplp

mailmail



3.6.2\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.3.6\_shell 4.3.5\_shell 4.3.4\_shell 4.3.3\_shell 4.3.2\_shell 4.3.1\_shell 4.2\_shell 4.1.3\_shell 4.1.2\_shell 4.1.1\_shell 3.6.2\_shell

```
#!/bin/bash
```

#É possível associar um bloco de comandos a diferentes condições de selecção. Por exemplo:

```
ls -l | awk '
/^d/ { print $1 $3; }
/^-/ { print ficheiro $2; }'
```

#Trata de forma diferente  
#É ainda possível associar  
#BEGIN que ocorre antes da  
#como o FS que indica o ca  
#Exemplo:

```
#cat /etc/passwd | awk 'BE
1
1
1
1
drwxrwxr-xatafs
1
1'
```

#Mostra o nome e descrição  
#Exemplo:

```
#Considere o ficheiro user
#josesilva:sporting:1212-1
#cebola:x:2333-2222-232322
#tintim:stromp:2323-2323-2
#jaquim:ola:2232-1111-11111111
#pedro:benfica:3333-3333-3333333
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

```
1
1
1
1
drwxrwxr-xatafs
1
1

atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

4.1.1.1\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.4.1\_shell 4.3.6\_shell 4.3.5\_shell 4.3.4\_shell 4.3.3\_shell 4.3.2\_shell 4.3.1\_shell 4.2\_shell 4.1.3\_shell 4.1.2\_shell 4.1.1.1\_shell

#include &lt;stdio.h&gt;

```
int main( int argc, char *argv[] ) {
    int i;
    for ( i = 1; i < argc; i++ ) {
        printf ("%d: %s\n", i, argv[i] );
    }
}
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.1.1.1_shell
./4.1.1.1_shell: linha 3: syntax error near unexpected token `(
./4.1.1.1_shell: linha 3: `int main( int argc, char *argv[] ) {
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

#!/bin/bash

## #4.1 - Expansão

#Questão muito importante: quando fazemos um comando como o ls -l \*.c ou zip programas.c \*.c  
#quem é que transforma o \*.c numa lista de nomes de ficheiros, a shell ou os comandos. Resposta: a  
#shell. Por exemplo, ao fazermos

#ls -l \*.sh

#a shell vai invocar o comando ls passando-lhe como argumentos todos os nomes de ficheiro que

4.1.1.2\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.4.1\_shell 4.3.6\_shell 4.3.5\_shell 4.3.4\_shell 4.3.3\_shell 4.3.2\_shell 4.3.1\_shell 4.2\_shell 4.1.3\_shell 4.1.2\_shell 4.1.1.2\_shell

```
#quem é que transforma o *.c numa lista de nomes de ficheiros, a shell ou os comandos. Resposta: a
#shell. Por exemplo, ao fazermos
```

```
#ls -l *.sh
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./4.1.1.2\_shell

./4.1.1.2\_shell: linha 46: ./eco: Ficheiro ou directória inexistente

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$

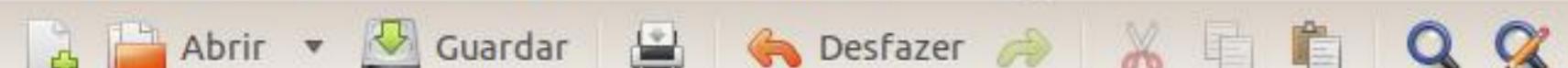
```
#Experimente, por exemplo:
```

```
./eco a b c
```

```
./eco *.c
```

4.1.1.3\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.4.1\_shell 4.3.6\_shell 4.3.5\_shell 4.3.4\_shell 4.3.3\_shell 4.3.2\_shell 4.3.1\_shell 4.2\_shell 4.1.3\_shell 4.1.2\_shell 4.1.1.3\_shell

```
#quem é que transforma o *.c numa lista de nomes de ficheiros, a shell ou os comandos. Resposta: a
#shell. Por exemplo, ao fazermos
```

```
#ls -l *.sh
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 ✘ atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 ✘

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.1.1.3_shell
./4.1.1.3_shell: linha 47: ./eco: Ficheiro ou directorio inexistente
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

```
#Experimente, por exemplo:
./eco a b c
./eco *.sh
```

Actividades >\_Consola Seg 22:04 Atafs

4.1.1\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

Abrir Guardar Desfazer

4.4.1\_shell 4.3.6\_shell 4.3.5\_shell 4.3.4\_shell 4.3.3\_shell 4.3.2\_shell 4.3.1\_shell 4.2\_shell 4.1.3\_shell 4.1.2\_shell 4.1.1\_shell

```
#!/bin/bash

#4.1 - Expansão

#Questão muito importante: quando fazemos um comando como o ls -l *.c ou zip programas.c *.c
#quem é que transforma o *.c numa lista de nomes de ficheiros, a shell ou os comandos. Resposta: a
#shell. Por exemplo, ao fazermos
```



```
ls -l *.sh
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./4.1.1\_shell

-rwxr-xr-x 1 atafs atafs 190 2012-05-03 14:11 teste.sh

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$

```
// echo
#include <stdio.h>

int main( int argc, char
#     int i;
#     for ( i = 1; i <
#         printf ("%d: %s\n", i, argv[i] );
#
# }
```

4.1.2\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.4.2\_shell 4.4.1\_shell 4.3.6\_shell 4.3.5\_shell 4.3.4\_shell 4.3.3\_shell 4.3.2\_shell 4.3.1\_shell 4.2\_shell 4.1.3\_shell 4.1.2\_shell

```
#!/bin/bash
```

```
#MAIS AVANÇADO
#Muitas vezes é preciso impedir a shell de seguir o procedimento normal de expansão de caracteres.
#Por exemplo, para escrever um $ no ecrã é preciso indicar à shell para não interpretar o $ como
#iniciador do nome de uma variável. Diz-se então que estamos a fazer o escape (do significado
#habitual) do carácter.
```

```
#Uma das maneiras de escapar
```

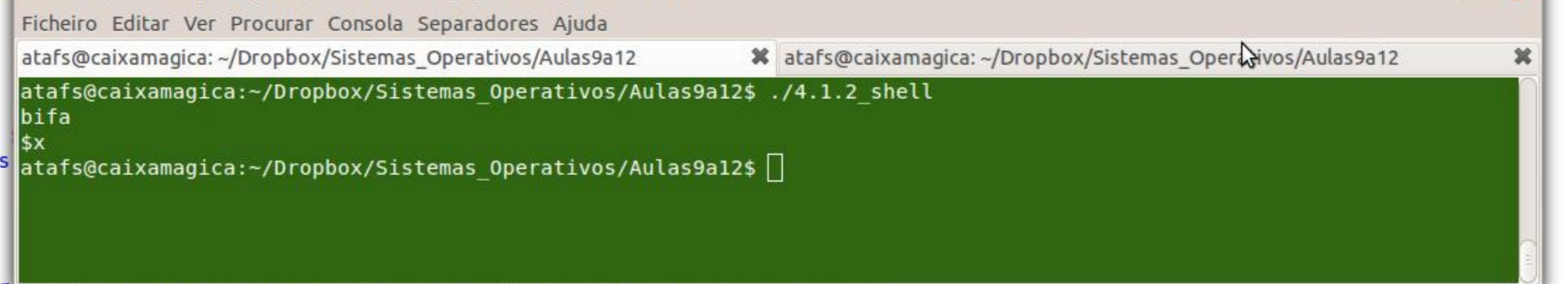
```
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12
```

```
Ficheiro Editar Ver Procurar Consola Separadores Ajuda
```

```
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12
```

```
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12
```

```
x=bifa
echo $x
# o
echo '$x'
# os
```



```
#O mecanismo de expansão deriva da presença de caracteres com significado especial para a shell.
#No caso anterior o significado especial é dado pelo $. As ' ' retiram o significado especial ao $ e por
#isso $x fica a ser só, apenas e literalmente $x.
#(Em rigor, as ' ' são necessárias apenas para lidar com o $. Ou seja, poderíamos obter o efeito
#desejado apenas com echo '$x'. Mas '$x' também funciona e é muito mais claro.)
```

```
#{São também caracteres especiais da shell os seguintes:
```

```
# * ? [ ] ' " \ $ ; & ( ) | ^ < > { }
#que normalmente, para serem assumidos de forma literal, devem também ser escapados).
```

4.1.3\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.4.3\_shell 4.4.2\_shell 4.4.1\_shell 4.3.6\_shell 4.3.5\_shell 4.3.4\_shell 4.3.3\_shell 4.3.2\_shell 4.3.1\_shell 4.2\_shell 4.1.3\_shell

```
#!/bin/bash
```

```
#Há 3 elementos sintáticos que impedem a expansão: as " ", as ' ' e a \.
```

```
#As aspas são as mais fracas: escapam apenas alguns caracteres. Não escapam, por exemplo, o $ e,
```

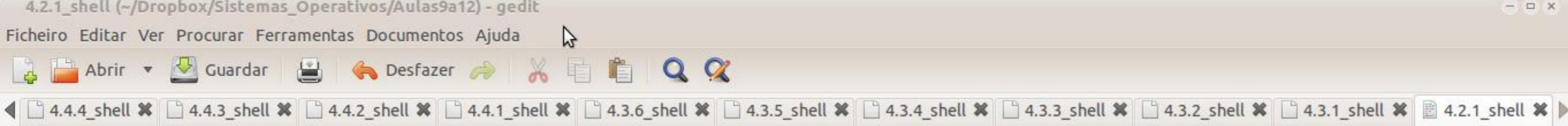
```
#por isso, as variáveis colocadas entre aspas continuam a ser expandidas.
```

```
#Por exemplo:
```

```
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12
Ficheiro Editar Ver Procurar Consola Separadores Ajuda
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.1.3_shell
*** /home/atafs ***
$HOME
$HOME
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

```
echo '$HOME'
echo \$HOME
#(para escrever \ usa-se \\).
```





#A variável especial \$? representa o resultado do último comando executado. Veja o valor de \$?  
#após o comando

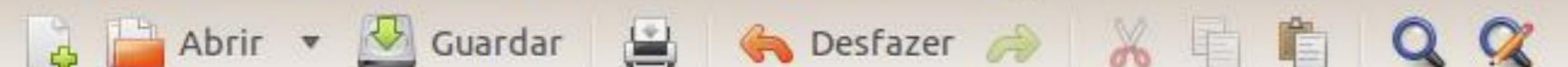
```
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12
Ficheiro Editar Ver Procurar Consola Separadores Ajuda
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12  atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.2.1_shell
chmod: impossível aceder a «x»: Ficheiro ou directória inexistente
1
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ 
```

#{veja para o caso de sucesso  
# qual o problema com este s  
#!/bin/bash

```
#chmod 700 x
#if $?
#then
#    echo "Ok"
#else
#    echo "Erro"
#fi
```

4.2.2\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.4.4\_shell 4.4.3\_shell 4.4.2\_shell 4.4.1\_shell 4.3.6\_shell 4.3.5\_shell 4.3.4\_shell 4.3.3\_shell 4.3.2\_shell 4.3.1\_shell 4.2.2\_shell

```
#chmod 700 x  
#echo $?
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 x atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 x

```
#(veja para o caso de sucess  
# qual o problema com este s  
#!/bin/bash  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.2.2_shell  
./4.2.2_shell: linha 59: 0: comando não reconhecido  
Erro  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

chmod 700 users.txt

if \$?  
then  
 echo "Ok"  
else  
 echo "Erro"  
fi

4.2\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.4.4\_shell 4.4.3\_shell 4.4.2\_shell 4.4.1\_shell 4.3.6\_shell 4.3.5\_shell 4.3.4\_shell 4.3.3\_shell 4.3.2\_shell 4.3.1\_shell 4.2\_shell

```
#!/bin/bash
```

```
#4.2- if / test / review  
#É importante perceber a subtileza da "condição" do if. Numa linguagem de programação  
#clássica a decisão do if é baseada numa condição. Na shell a decisão é baseada no  
#resultado da execução de um comando.  
#Exemplo: considere o seguinte comando
```

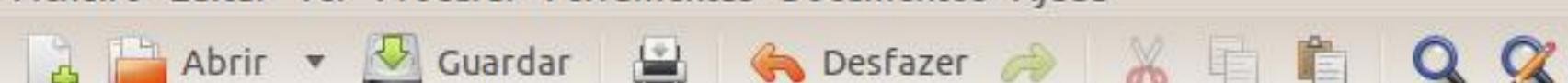
```
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12  
Ficheiro Editar Ver Procurar Consola Separadores Ajuda  
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12 ✘ atafsa@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.2_shell  
Ficheiro:  
users.txt  
Comando executado!  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.2_shell  
Ficheiro:  
sadsa.c  
chmod: impossível aceder a «sadsa.c»: Ficheiro ou directória inexistente  
Como deve ter percebido, não correu bem.  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

```
echo "Ficheiro:"  
read x  
if chmod 700 $x  
then  
    echo "Comando executado!"  
else  
    echo "Como deve ter percebido, não correu bem."  
fi
```



4.3.1.1\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.5.1\_shell 4.4.4\_shell 4.4.3\_shell 4.4.2\_shell 4.4.1\_shell 4.3.6\_shell 4.3.5\_shell 4.3.4\_shell 4.3.3\_shell 4.3.2\_shell 4.3.1.1\_shell

**#!/bin/bash**

#4.3 - exit; encadeamento de comandos;

#Interactivamente o comando

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./4.3.1.1\_shell

Seg Mai 7 22:36:00 WEST 2012

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$

#num script tem o efeito

#0 exit pode ser utilizada

**#!/bin/bash****date****exit**

# o script termina aqui

# este comando não chega a ser executado



4.3.1\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.5.1\_shell 4.4.4\_shell 4.4.3\_shell 4.4.2\_shell 4.4.1\_shell 4.3.6\_shell 4.3.5\_shell 4.3.4\_shell 4.3.3\_shell 4.3.2\_shell 4.3.1\_shell

**#!/bin/bash**

#4.3 - exit; encadeamento de comandos;

#Interactivamente o comando exit serve para terminar uma shell; ex:

**/bin/bash****atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12**  
# in  
# te

exit

#num script tem o efeito cor  
#0 exit pode ser utilizado p  
#!/bin/bash#date  
#exit  
#echo "Bye."# o script termina aqui  
# este comando não chega a ser executado

```
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12
Ficheiro Editar Ver Procurar Consola Separadores Ajuda
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.3.1_shell
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ exit
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```



Actividades >\_Consola Seg 22:42 Atafs

4.3.2\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

Abrir Guardar Desfazer

4.5.2\_shell 4.5.1\_shell 4.4.4\_shell 4.4.3\_shell 4.4.2\_shell 4.4.1\_shell 4.3.6\_shell 4.3.5\_shell 4.3.4\_shell 4.3.3\_shell 4.3.2\_shell

```
#!/bin/bash

#Exemplo: o seguinte script compila um programa em C, criando um executável com o mesmo nome;
#o nome do ficheiro é dado num argumento para o script; o primeiro passo é justamente verificar o
#argumento:
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

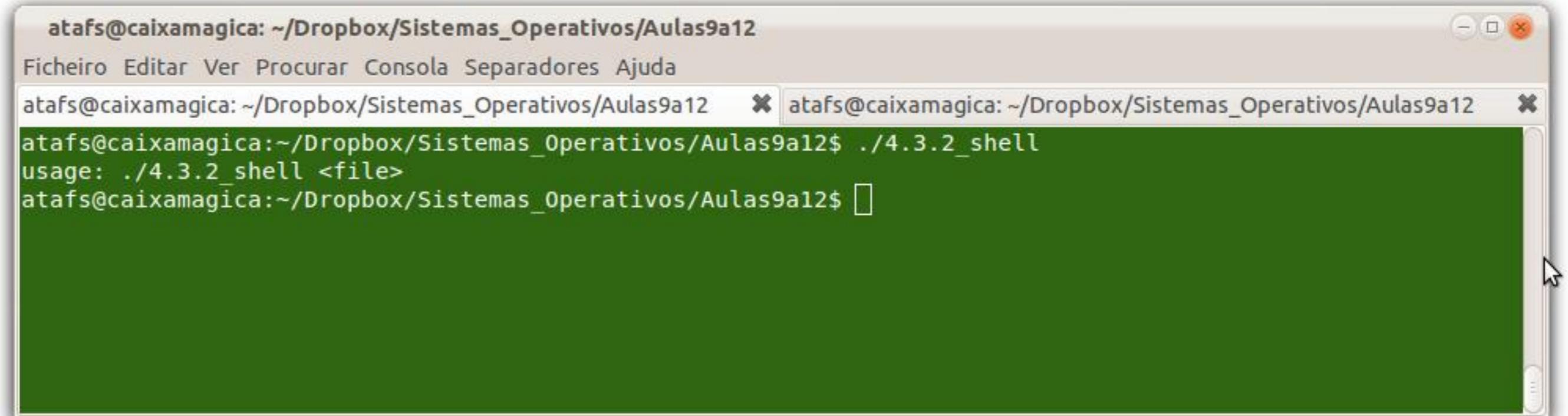
Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./4.3.2\_shell

usage: ./4.3.2\_shell <file>

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$



```
#Além da função de instrução de controlo, o exit tem outro efeito importante que é estabelecer o
#resultado – o exit status do comando;
```

4.3.3.1\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.5.3\_shell 4.5.2\_shell 4.5.1\_shell 4.4.4\_shell 4.4.3\_shell 4.4.2\_shell 4.4.1\_shell 4.3.6\_shell 4.3.5\_shell 4.3.4\_shell 4.3.3.1\_shell

**#!/bin/bash**

#Exemplo: considere o seguinte script com o nome trivial

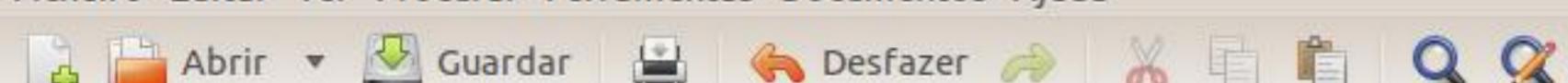
```
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12
Ficheiro Editar Ver Procurar Consola Separadores Ajuda
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12  atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.3.3.1_shell
./4.3.3.1_shell: linha 22: trivial: comando não reconhecido
127
./4.3.3.1_shell: linha 24: syntax error near unexpected token `;'
./4.3.3.1_shell: linha 24: `if [ trivial 0 ] ; then ; echo "funciona."; fi'
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ 
```

```
trivial 2
echo $? # dá ?
if [ trivial 0 ] ; then ; echo "funciona."; fi
```

#0 símbolo \$? representa o exit status do último comando a ser executado

4.3.3\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.5.3\_shell 4.5.2\_shell 4.5.1\_shell 4.4.4\_shell 4.4.3\_shell 4.4.2\_shell 4.4.1\_shell 4.3.6\_shell 4.3.5\_shell 4.3.4\_shell 4.3.3\_shell

```
#!/bin/bash
```

#Exemplo: considere o seguinte script com o nome trivial

```
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12
Ficheiro Editar Ver Procurar Consola Separadores Ajuda
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12  atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.3.3_shell
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ 
```

#verifique o efeito da seguinte sequênc

```
trivial 2
echo $? # dá ?
if [ trivial 0 ] ; then ; echo "funciona." ; fi
```

#0 símbolo \$? representa o exit status do último comando a ser executado



4.4.3\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

**#!/bin/bash**

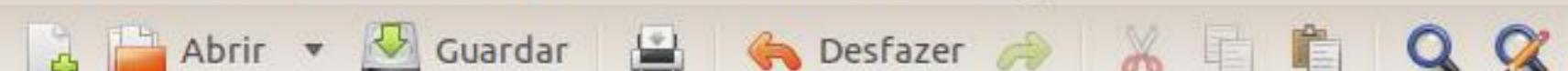
```
#As variáveis do script (as que são herdadas ou as que são criadas no próprio script) estão disponíveis  
#na função; podem ser aí alteradas tal como podem ser criadas novas variáveis; as variáveis  
#trabalhadas deste modo são todas "globais" (no sentido que o termo tem na programação clássica):  
#existem podem ser criadas, alteradas e destruídas em todo o lado;  
#Exemplo:
```

```
f () {  
    echo "$FUNCNAME : Y= $Y"  
    X=77  
    Y=88  
    echo "$FUNCNAME : X= $X"  
    echo "$FUNCNAME : Y= $Y"  
}  
  
Y=66  
echo "Y= $Y"  
f  
echo "X= $X"  
echo "Y= $Y"
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 Ficheiro Editar Ver Procurar Consola Separadores Ajuda atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 atafsa@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./4.4.3\_shell  
Y= 66  
f : Y= 66  
f : X= 77  
f : Y= 88  
X= 77  
Y= 88  
atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$

\*4.3.4\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



#!/bin/bash

#Considere a seguinte versão do exemplo b):

```
#- se não for dado um argumento o script  
#- ao contrário, se tudo correr bem, t  
# de execução (não adianta fazer exit  
# apenas estabelecer o resultado de s  
# em vez de exit 0 não seria melhor e  
  
if [ ! $# -eq 1 ] ; then  
    echo "usage: $0 <file>"  
    exit 1  
fi  
cc $1 -o `basename $1 .c`  
exit 0
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 ✘ atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 ✘

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./4.3.4\_shell users.txt  
cc: error: users.txt: Ficheiro ou directória inexistente  
cc: fatal error: no input files  
compilation terminated.  
atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$

4.3.5\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

4.6.1\_shell 4.5.4\_shell 4.5.3\_shell 4.5.2\_shell 4.5.1\_shell 4.4.4\_shell 4.4.3\_shell 4.4.2\_shell 4.4.1\_shell 4.3.6\_shell 4.3.5\_shell

```
#executa os comandos por ordem enquanto derem "sucesso" ( ou seja, termina a sequência se um  
#deles falhar)  
#A sequência:
```

```
#comando1 || comando2 || ....
```

```
#executa os comandos por ordem até um  
#enquanto falharem);  
#exemplo
```

```
trivial 0 && echo "Ok."  
trivial 1 && echo "NOP"  
trivial 0 || echo "NOP"  
trivial 1 || echo "Ok."
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

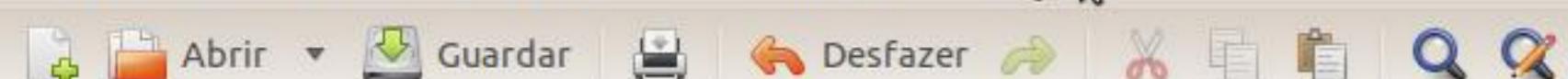
atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 × atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 ×

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.3.5_shell  
.4.3.5_shell: linha 40: trivial: comando não reconhecido  
.4.3.5_shell: linha 41: trivial: comando não reconhecido  
.4.3.5_shell: linha 42: trivial: comando não reconhecido  
NOP  
.4.3.5_shell: linha 43: trivial: comando não reconhecido  
Ok.  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```



4.3.6.1\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.6.1\_shell 4.5.4\_shell 4.5.3\_shell 4.5.2\_shell 4.5.1\_shell 4.4.4\_shell 4.4.3\_shell 4.4.2\_shell 4.4.1\_shell 4.3.6.1\_shell teste

```
#      echo "$0: invalid arguments"
#      exit 1
#fi
#cc $1 -o `basename $1 .c`
#exit $?
```

#Uma alternativa poderia ser:

```
if [ $# -eq 1 ] && [ -f $1 ] ; then
:
else
    echo "$0: invalid arguments"
    exit 1
fi
cc $1 -o `basename $1 .c`
exit $? 
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

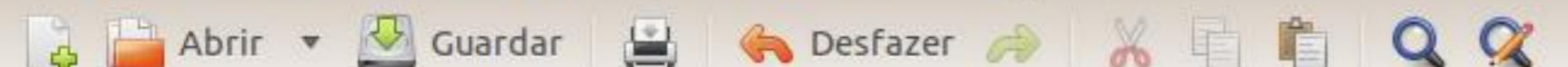
atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 ✘ atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 ✘

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.3.6.1_shell
./4.3.6.1_shell: invalid arguments
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ 
```

#Verificadas as duas condições é executado o comando : (que não faz nada) , e o script segue depois  
#do if; de contrário, termina com exit;  
#0 comando

4.3.6.2\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.6.1\_shell 4.5.4\_shell 4.5.3\_shell 4.5.2\_shell 4.5.1\_shell 4.4.4\_shell 4.4.3\_shell 4.4.2\_shell 4.4.1\_shell 4.3.6.2\_shell teste

```
#else
#      echo "$0: invalid arguments"
#      exit 1
#fi
#cc $1 -o `basename $1 .c`
#exit $?
```

```
#Verificadas as duas condições é executado o comando : (que "não faz nada") e o script segue depois
#do if; de contrário, termina com exit;
#0 comando
```

comp teste.c &amp;&amp; teste

#compile o programa e, em caso de sucesso

```
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12
Ficheiro Editar Ver Procurar Consola Separadores Ajuda
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12  ✘ atafsa@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12  ✘
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ 4.3.6.1_shell teste.c && teste
4.3.6.1_shell: comando não encontrado
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

4.3.6\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



```
#!/bin/bash
```

```
#A seguinte versão do script comp verifica se é dado um argumento e se o ficheiro correspondente  
#existe; caso uma das condições não se verifique o script termina com erro:
```

```
#!/bin/bash
```

```
if [ ! $# -eq 1 ] || [ ! -f $1 ] ; then  
    echo "$0: invalid arguments"  
    exit 1  
fi  
cc $1 -o `basename $1 .c`  
exit $? 
```

```
#Uma alternativa poderia ser:
```

```
#if [ $# -eq 1 ] && [ -f $1 ] ; then  
#    :  
#else  
#    echo "$0: invalid arguments"
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 ✘ atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 ✘

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.3.6_shell teste  
cc: error: -o: Ficheiro ou directória inexistente  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ 
```

4.4.1.1\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

```
#As funções são um elemento estruturante parecido com as funções das linguagens de programação  
#clássicas; no caso dos scripts, enquadram um conjunto de comandos que são executados quando a  
#função é invocada através do seu nome; ex:
```

```
#sintaxe:  
#exemplo()  
#    echo $FUNCNAME says hello  
#}
```

```
#echo "chamar a função..."  
#exemplo  
#echo "repete..."  
#exemplo
```

```
#As funções aceitam argumentos numa s  
#scripts; exemplo:
```

```
#!/bin/bash
```

```
say () {  
    echo "I say, " $1  
}
```

```
say hello  
say hello hello  
say "hello hello hello"
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 ✘ atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 ✘

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.4.1.1_shell  
I say, hello  
I say, hello  
I say, helohello hello  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

4.4.1\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

```
#As funções são um elemento estruturante parecido com as funções das linguagens de programação  
#clássicas; no caso dos scripts, enquadram um conjunto de comandos que são executados quando a  
#função é invocada através do seu nome; ex:
```

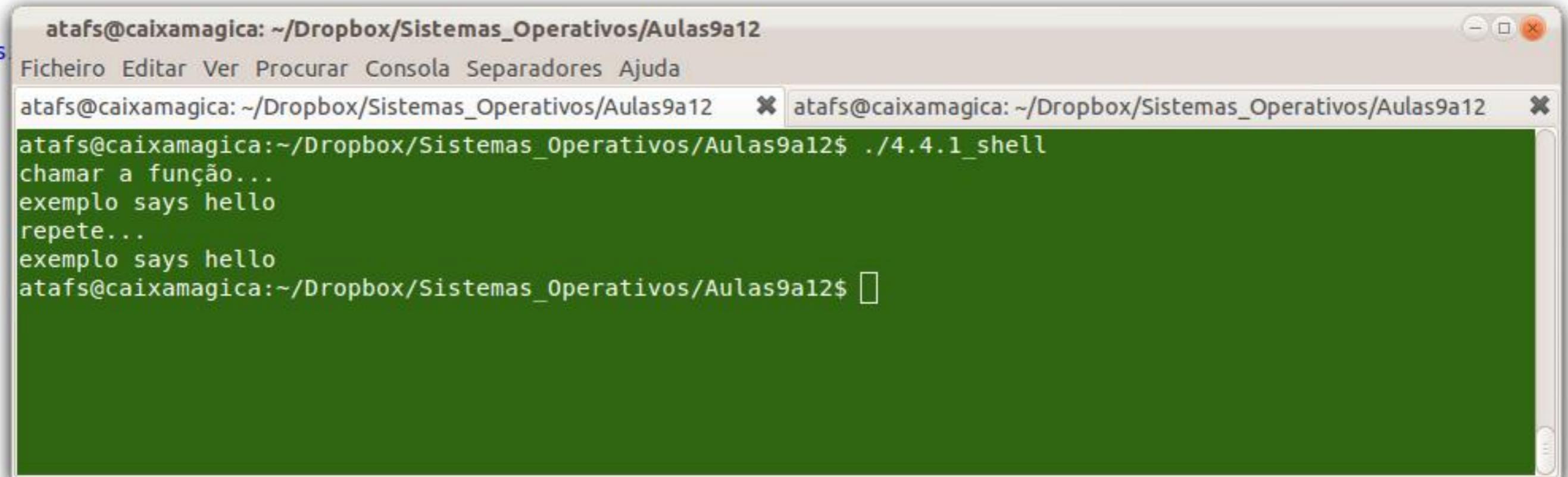
```
#sintaxe:  
exemplo() {  
    echo $FUNCNAME says hello  
}
```

```
echo "chamar a função..."  
exemplo  
echo "repete..."  
exemplo
```

```
#As funções aceitam argumentos numa s  
#scripts; exemplo:
```

```
#!/bin/bash
```

```
#say () {  
#    echo "I say, " $1  
#}  
#  
#say hello  
#say hello hello  
#say "hello hello hello"
```



```
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12  
Ficheiro Editar Ver Procurar Consola Separadores Ajuda  
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12 ✘ atafsa@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12 ✘  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.4.1_shell  
chamar a função...  
exemplo says hello  
repete...  
exemplo says hello  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

4.4.2\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

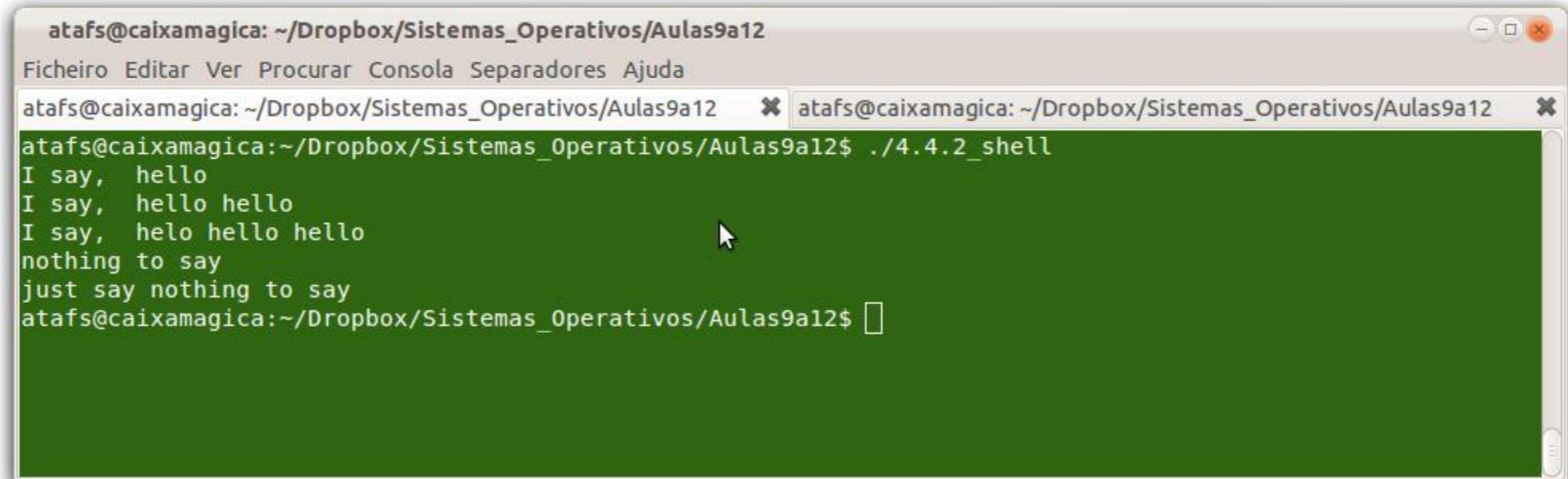
Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

```
#!/bin/bash
```

```
#0 comando return termina a função, em determinado ponto, permitindo também formar um resultado  
#de retorno; ex:  
#!/bin/bash
```

```
say () {  
    if [ $# -eq 0 ] ; then  
        echo "nothing to say"  
        return 1  
    fi  
    echo "I say, " $*  
    return 0  
}
```

```
say hello  
say hello hello  
say "heло hello hello"  
say  
echo just say `say`
```



atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12  
Ficheiro Editar Ver Procurar Consola Separadores Ajuda  
atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 ✘ atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 ✘  
atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./4.4.2\_shell  
I say, hello  
I say, hello hello  
I say, heло hello hello  
nothing to say  
just say nothing to say  
atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$

4.4.4\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.8.1\_shell 4.7.2\_shell 4.7.1\_shell 4.6.3\_shell 4.6.2\_shell 4.6.1\_shell 4.5.4\_shell 4.5.3\_shell 4.5.2\_shell 4.5.1\_shell 4.4.4\_shell

**#!/bin/bash**

#Exemplo: no seguinte script é feita uma função readline que lê uma string:

```
readline () {
    echo "readline..."
    msg=""
    if [ $# -gt 0 ] ; then
        msg="$*: "
    fi

    str=""
    while [ -z $str ] ; do
        echo -n $msg
        read str
    done
    STR=str
}

readline "Teste"
echo "Lido: " $STR
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 \* atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 \*

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.4.4_shell
readline...
Teste:Olá Américo
./4.4.4_shell: linha 17: [: Olá: binary operator expected
Lido: str
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

4.5.1\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.8.2\_shell 4.8.1\_shell 4.7.2\_shell 4.7.1\_shell 4.6.3\_shell 4.6.2\_shell 4.6.1\_shell 4.5.4\_shell 4.5.3\_shell 4.5.2\_shell 4.5.1\_shell

**#!/bin/bash****#4.5 - Arrays**

```
Nos scripts podem-se usar variáveis indexadas; ex:  
#note a utilização da sintaxe ${} para isolar o nome das variáveis;
```

```
a[0]="Hello"  
a[3]="World"  
echo "${a[0]} ${a[3]}"
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 ✘ atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 ✘

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.5.1_shell  
Hello World  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

4.5.2\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



4.8.2\_shell 4.8.1\_shell 4.7.2\_shell 4.7.1\_shell 4.6.3\_shell 4.6.2\_shell 4.6.1\_shell 4.5.4\_shell 4.5.3\_shell 4.5.2\_shell

**#!/bin/bash**

```
#É raro haver interesse em usar variáveis indexadas, isoladamente, em vez de variáveis comuns.  
#Normalmente o que se pretende é usar um conjunto de posições contíguas em processamentos  
#iterativos (ou seja, o correspondente aos arrays nas linguagens de programação clássicas);  
#Exemplo: gerar 6 números aleatórios:
```

```
i=0  
while [ $i -le 5 ] ; do  
    num[i]=$RANDOM  
    echo "Número : $i ${num[i]}"  
    i=$(( $i + 1 ))  
done
```

```
#A variável $RANDOM fornece um número  
#gerado situa-se entre 0 e 2**15 (2 )
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 \* atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 \*

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.5.2_shell  
Número : 0 23988  
Número : 1 29976  
Número : 2 10508  
Número : 3 3519  
Número : 4 9496  
Número : 5 20880
```

4.5.3\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



9.9.9\_shell\_Duvidas ✎ 4.8.2\_shell ✎ 4.8.1\_shell ✎ 4.7.2\_shell ✎ 4.7.1\_shell ✎ 4.6.3\_shell ✎ 4.6.2\_shell ✎ 4.6.1\_shell ✎ 4.5.4\_shell ✎ 4.5.3\_shell ✎

**#!/bin/bash**

```
#Um while deste género pode ser escrito de maneira mais familiar com a seguinte sintaxe alternativa,  
#mais simpática para ciclos iterativos;  
#Exemplo: o seguinte script gera 6 números aleatórios entre 1 e 20:
```

```
for (( i=0; i < 5; i++ )) ; do  
    num[i]=$( ( 1 + 20 * $RANDOM / 2**15 ))
```

**done**

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 ✎ atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12 ✎

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.5.3_shell  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

4.5.4\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



9.9.9\_shell\_Duvidas ✎ 4.8.2\_shell ✎ 4.8.1\_shell ✎ 4.7.2\_shell ✎ 4.7.1\_shell ✎ 4.6.3\_shell ✎ 4.6.2\_shell ✎ 4.6.1\_shell ✎ 4.5.4\_shell ✎

**#!/bin/bash**

#Exemplo: o seguinte script gera 6 números aleatórios, entre 1 e 20, apresentando-os por ordem;

```
for (( i=0; i < 5; i++ )) ; do
    num[i]=$( 1 + 20 * $RANDOM / 2**15 )
done

for (( i=0; i < 5; i++ )) ; do
    for (( j=0; j < 5; j++ )) ; do
        if [ ${num[i]} -lt ${num[j]} ] ; then
            x=${num[i]}
            num[i]=${num[j]}
            num[j]=$x
        fi
    done
done

for (( i=0; i < 5; i++ )) ; do
    echo "Numero: $i ${num[i]}"
done
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... ✎ atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... ✎

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.5.4_shell
Numero: 0 1
Numero: 1 2
Numero: 2 6
Numero: 3 10
Numero: 4 16
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

#Exercício: faça um script que gere uma aposta do totoloto, ie, 6 números diferentes, entre 1 e 49;

4.6.1\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



9.9.9\_shell\_Duvidas ✎ 4.8.2\_shell ✎ 4.8.1\_shell ✎ 4.7.2\_shell ✎ 4.7.1\_shell ✎ 4.6.3\_shell ✎ 4.6.2\_shell ✎ 4.6.1\_shell ✎

**#!/bin/bash****#4.6 - Strings**

```
#length – obter o comprimento, ie o número de caracteres, de uma string;  
#O exemplo seguinte mostra o tamanho de uma string lida
```

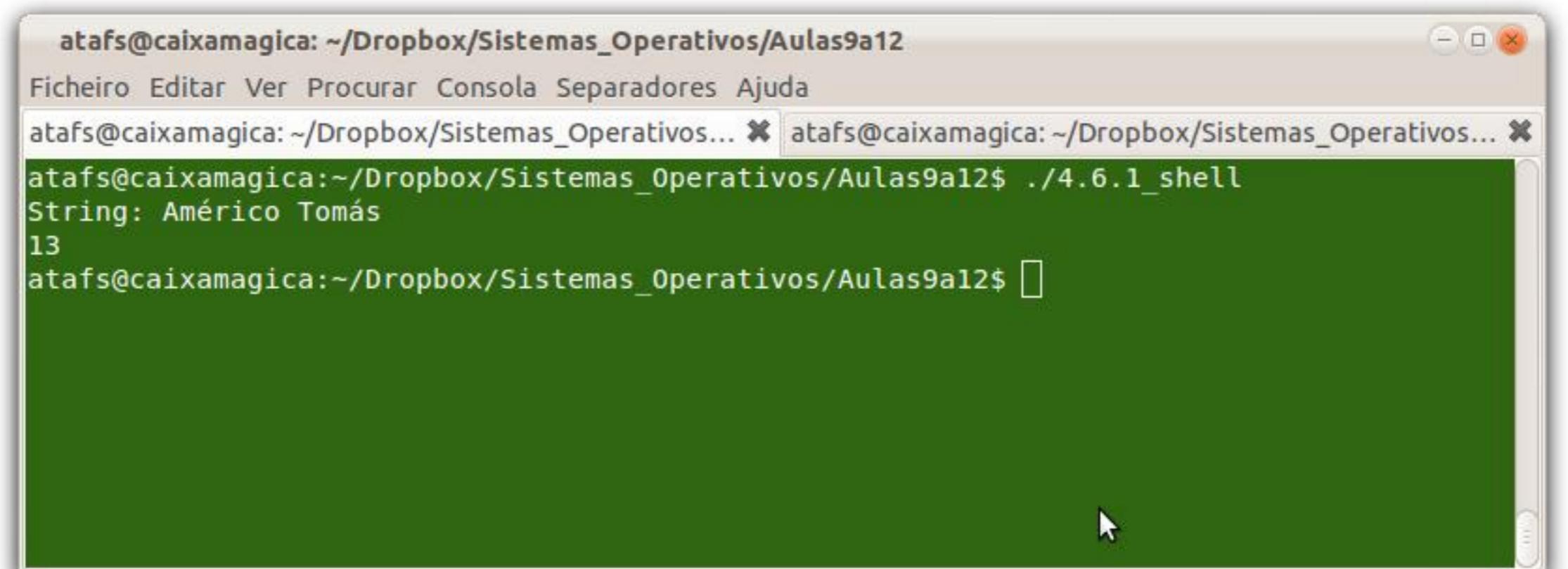
```
echo -n "String: "  
read s  
echo ${#s}
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... ✎ atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos... ✎

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.6.1_shell  
String: Américo Tomás  
13  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```



4.6.2\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



9.9.9\_shell\_Duvidas ✎ 4.8.2\_shell ✎ 4.8.1\_shell ✎ 4.7.2\_shell ✎ 4.7.1\_shell ✎ 4.6.3\_shell ✎ 4.6.2\_shell ✎

**#!/bin/bash**

```
#substring – extrair parte de uma string;  
#o exemplo seguinte lê uma string e mostra parte dos caracteres lidos:
```

```
echo -n "String: "  
read s  
echo ${s:5}  
echo ${s:5:3}  
n=${#s}  
if [ $(( n % 2 )) -eq 1 ] ; then  
    m=$(( n / 2 ))  
    echo ${s:$m:1}  
fi
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... ✎ atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos... ✎

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.6.2_shell  
String: Américo Tomás  
co Tomás  
co  
o  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

#Exercício: altere para mostrar também:  
#- o último caractere  
#- a primeira metade da string;

4.6.3\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



9.9.9\_shell\_Duvidas ✘ 4.8.2\_shell ✘ 4.8.1\_shell ✘ 4.7.2\_shell ✘ 4.7.1\_shell ✘ 4.6.3\_shell ✘

**#!/bin/bash**

```
#substituição – substituir uma parte da string  
#o exemplo seguinte substitui a letra a pela letra x na string lida;
```

```
echo -n "String: "  
read s  
s1=${s/a/x} ; echo $s1  
s1=${s//a/y} ; echo $s1
```

#experimente uma entrada com várias letr

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... ✘ atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... ✘

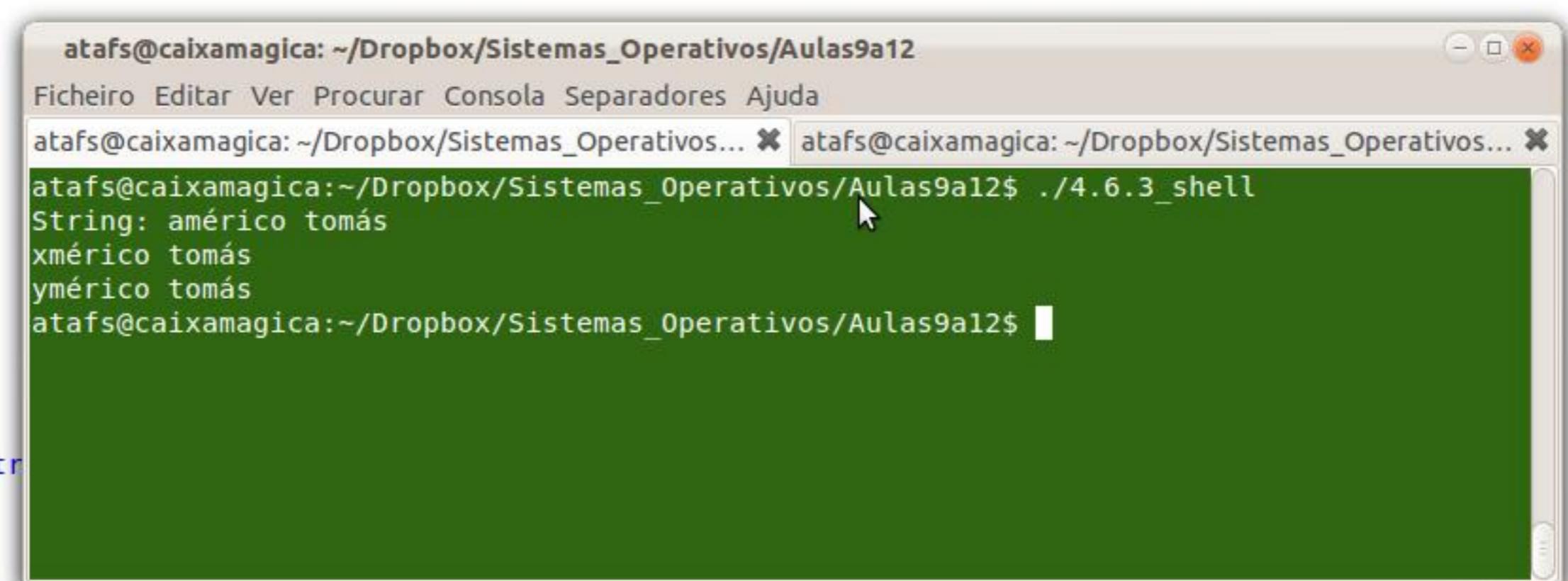
atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./4.6.3\_shell

String: américo tomás

xmérico tomás

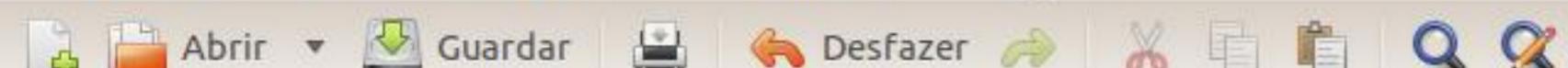
ymérico tomás

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ █



4.7.1\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



9.9.9\_shell\_Duvidas ✎ 4.8.2\_shell ✎ 4.8.1\_shell ✎ 4.7.2\_shell ✎ 4.7.1\_shell ✎

**#!/bin/bash****#4.7 - Separação de palavras****#Evidentemente que exercícios como o anterior são mais fáceis de realizar usando os  
#mecanismos da shell que naturalmente separam palavras; ex:**

```
echo -n "String: "
read s
p=""
for i in $s ; do
    if [ -z $p ] ; then
        p=$i; echo "Primeiro: $p"
    fi
done
echo "Ultimo: $i"
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... ✎ atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos... ✎

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.7.1_shell
String: amérigo tomás
Primeiro: amérigo
Último: tomás
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

4.7.2\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



9.9.9\_shell\_Duvidas ✎ 4.8.2\_shell ✎ 4.8.1\_shell ✎ 4.7.2\_shell ✎

**#!/bin/bash**

```
#Estes métodos ganham ainda maior utilidade com a possibilidade de escolher o separador de
#palavras, que pode ser indicado à shell na variável IFS; normalmente o separador é o espaço, mas
#pode-se alterar através desta variável;
#Exemplo: o seguinte script mostra a lista de directório da PATH, um por linha:
```

```
IFS=:
for d in $PATH ; do
echo $d
done
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... ✎ atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos... ✎

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./4.7.2_shell
/usr/lib/lightdm/lightdm
/usr/local/sbin
/usr/local/bin
/usr/sbin
/usr/bin
/sbin
/bin
/usr/games
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

4.8.1\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



9.9.9\_shell\_Duvidas ✎ 4.8.2\_shell ✎ 4.8.1\_shell ✎

```
#/bin/bash

#4.8.1 - Acertar na soma
#Faça um script em bash que atribua números aleatórios às variáveis x e y, faça a sua soma e peça ao
#utilizador para adivinhar o resultado. Quando o utilizador acertar, o script deverá indicar o tempo que
#foi usado para fazer a conta em segundos.
```

```
x=$(( $RANDOM / 100 ))
y=$(( $RANDOM / 100 ))
s=$((x+y))
di=$(date "+%s")
echo -n "Qual a soma de $x com $y ? "
read g
while [[ "$g" -ne $s ]]; do
    echo -n "Tente de novo: "
    read g
done
df=$(date "+%s")
t=$((df-$di))
echo "Acertou em $t segundos"
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

```
atafs@caixamagica: ~/Dropbox/Sistemas_Operativos... ✎ atafs@caixamagica: ~/Dropbox/Sistemas_Operativos... ✎
```

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./4.8.1\_shell

Qual a soma de 160 com 256 ? 160

Tente de novo: 200

Tente de novo: 23

Tente de novo: 416

Acertou em 95 segundos

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

Actividades >\_Consola Seg 23:42 Atafs

4.8.2\_shell (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

Abrir Guardar Desfazer

9.9.9\_shell\_Duvidas 4.8.2\_shell

```
#caramelo
#!/bin/bash

#readc: lê um caracter
readc() {
    echo -n "letra ( dispõe de $ntry tentativas ) : "
    read c
}

#marca : marca o caracter lido
marcac() {
    newd=""
    ok=0
    tryok=1
    for (( i=0; i < n ; i++ )) ; do
        sx=${s:i:1}
        dx=${d:i:1}
        if [ $sx = $c ] || [ $dx != "-" ]
            newd="$newd$sx"
        else
            newd="$newd-"
            ok=1
        fi
        if [ $sx = $c ] && [ $dx = "-" ] ; then
            tryok=0
        fi
    done
}
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./4.8.2\_shell

./4.8.2\_shell: linha 38: syntax error near unexpected token `else'

./4.8.2\_shell: linha 38: `else'

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$

sh ▾ Largura do Tabulador: 8 ▾ Ln 38, Col 21 INS

aula\_shell\_1.sh (~/Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

- □ ×

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

Abrir ▾ Guardar Desfazer Undo Redo Busca Pesquisa

aula\_shell\_9.sh × aula\_shell\_8.sh × aula\_shell\_7.sh × aula\_shell\_6.sh × aula\_shell\_5.sh × aula\_shell\_4.sh × aula\_shell\_2.sh × aula\_shell\_1.sh ×

**#!/bin/bash**

```
if [ $1 -ge $2 ];then          #test é um comando... $1 é o primeiro argumento de um script
    echo "o Argumento $1 é maior ou igual a $2"
else
    echo "O Argumento $1 não é maior que $2"
    exit 1
fi
```

```
echo "atafs> $0"
echo "Arg1: $1"
echo "Arg2: $2"
echo "Cardinal: $#"
echo "asterisco: $*"

#vai mandar para o bash
```

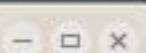
atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Oper... × ataf... × ataf... ×

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./aula_shell_1.sh 9 3
o Argumento 9 é maior ou igual a 3
atafs> ./aula_shell_1.sh
Arg1: 9
Arg2: 3
Cardinal: 2
asterisco: 9 3
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

aula\_shell\_2.sh (~/Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit



Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

Abrir Guardar Desfazer

aula\_shell\_9.sh ✘ aula\_shell\_8.sh ✘ aula\_shell\_7.sh ✘ aula\_shell\_6.sh ✘ aula\_shell\_5.sh ✘ aula\_shell\_4.sh ✘ aula\_shell\_2.sh ✘

**#!/bin/bash**

```
if pwd | ls -l  
then  
    echo "Funcionou"  
else  
    echo "Falhou completamente"  
fi
```

#vai mandar para o bash interpretar... podia interpretar com piten, etc...

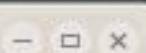
atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Oper... ✘ ataf... ✘ ataf... ✘

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./aula_shell_2.sh  
/home/atafs/Dropbox/Sistemas_Operativos/Aulas9a12  
Funcionou  
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

aula\_shell\_3.sh (~/Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit



Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

Abrir Guardar Desfazer

aula\_shell\_9.sh ✘ aula\_shell\_8.sh ✘ aula\_shell\_7.sh ✘ aula\_shell\_6.sh ✘ aula\_shell\_5.sh ✘ aula\_shell\_4.sh ✘ aula\_shell\_3.sh ✘ aula\_shell\_2.sh ✘ aula\_shell\_1.sh ✘

**#!/bin/bash**

```
echo "Linha de Comando: $0"
echo "Arg1: $1"
echo "Arg2: $2"
echo "Arg3: $3"
echo "Cardinal: $#"
echo "asterisco: $*"
```

```
if test 6 -lt $1
then
    echo "Funcionou"
else
    echo "Falhou com"
fi
#vai mandar para o bash
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Oper... ✘ ataf... ✘ ataf... ✘ ataf... ✘

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./aula\_shell\_3.sh 8

Linha de Comando: ./aula\_shell\_3.sh

Arg1: 8

Arg2:

Arg3:

Cardinal: 1

asterisco: 8

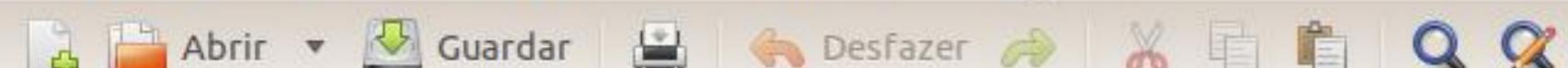
Funcionou

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$

aula\_shell\_4.sh (~/Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

- □ ×

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



aula\_shell\_9.sh ✘ aula\_shell\_8.sh ✘ aula\_shell\_7.sh ✘ aula\_shell\_6.sh ✘ aula\_shell\_5.sh ✘ aula\_shell\_4.sh ✘

**#!/bin/bash**

```
if test $# -ge 1          #test é um comando... $1 é o primeiro argumento de um script
then
    echo "Posso continuar"
else
    echo "Faltam argumentos"
    exit 1
fi

echo "Arg1: $0"
echo "Arg1: $1"
echo "Cardinal: $#"
echo "asterisco: $*"

#vai mandar para o bash
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Oper... ✘ ataf... ✘ ataf... ✘

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./aula\_shell\_4.sh 3 2 4 6
Posso continuar

Arg1: ./aula\_shell\_4.sh

Arg1: 3

Cardinal: 4

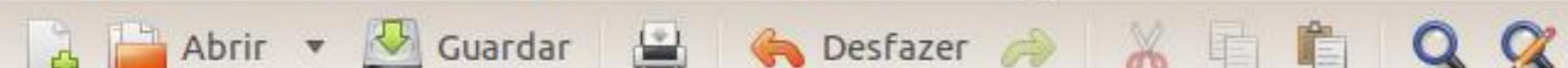
asterisco: 3 2 4 6

atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$

aula\_shell\_5.sh (~Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit



Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



aula\_shell\_9.sh ✘ aula\_shell\_8.sh ✘ aula\_shell\_7.sh ✘ aula\_shell\_6.sh ✘ aula\_shell\_5.sh ✘ teste.sh ✘

```
#!/bin/bash
```

```
for i in "maria" 1 2 3 Américo $1 *.sh; do
    echo "0 elemento: $i"
    if [ -x $i ]; then
        echo "Tem permissão de execução."
        echo ""
    else
        echo "nem Pensar"
    fi
```

done

echo "Adeus \$USER"

#vai mandar para o bash i

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Oper... ✘ ataf... ✘ ataf... ✘

```
0 elemento: aula_shell_6.sh
Tem permissão de execução.

0 elemento: aula_shell_7.sh
Tem permissão de execução.

0 elemento: aula_shell_8.sh
Tem permissão de execução.

0 elemento: aula_shell_9.sh
Tem permissão de execução.

0 elemento: teste.sh
Tem permissão de execução.

Adeus atafs
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

aula\_shell\_6.sh (~/Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

- □ ×

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda



aula\_shell\_9.sh ✘ aula\_shell\_8.sh ✘ aula\_shell\_7.sh ✘ net\_shell\_procurar ...os em directorias.sh ✘ aula\_shell\_6.sh ✘

```
#!/bin/bash

if [ $1 -ge $2 ];then          #test é um comando... $1 é o primeiro argumento de um script
    echo "Posso continuar. $1 é maior que $2"

elif [ $2 -ge $1 ];then
    echo "Já não. $1 não é maior que $2"

fi

atafs@caixamagica: ~/Dropbox/Sistemas_Operativos/Aulas9a12
Ficheiro Editar Ver Procurar Consola Separadores Ajuda
atafs@caixamagica: ~/Dropbox/Sistemas_Oper... ✘ ataf... ✘ ataf... ✘ ataf... ✘
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./aula_shell_6.sh 43 5546
Já não. 43 não é maior que 5546
Arg1: ./aula_shell_6.sh
Arg1: 43
Cardinal: 2
asterisco: 43 5546
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```

\*aula\_shell\_7.sh (~/Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

[aula\\_shell\\_9.sh](#) ✎ [aula\\_shell\\_8.sh](#) ✎ [\\*aula\\_shell\\_7.sh](#) ✎ [net\\_shell\\_procurar ...os em directorias.sh](#) ✎

```
x=123
y=-1
while [ -z "$y" ] || [ "$y" -ne $x ]; do

    echo -n "diga número: "
    read y
    if [ ! -z "$y" ]; then
        if [ "$y" -lt $x ]; then
            echo "É muito pequeno. Tente de Novo"
        elif [ "$y" -gt $x ]; then
            echo "É muito Grande. Tente de Novo"
        else
            echo ""
            echo "Parabéns!! É mesmo esse!!!"
            echo ""
        fi
    fi
done
```

#como interrogar o utilizador???
#Meu script que recebe um x=\$1

```
#while x=$123 ; do
#    #como interrogar o utilizador???
#    read "Qual o número a inserir "
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./aula\_shell\_7.sh

diga número: 12  
É muito pequeno. Tente de Novo

diga número: 345  
É muito Grande. Tente de Novo

diga número: 123  
Parabéns!! É mesmo esse!!!

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ █

aula\_shell\_8.sh (~/Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit

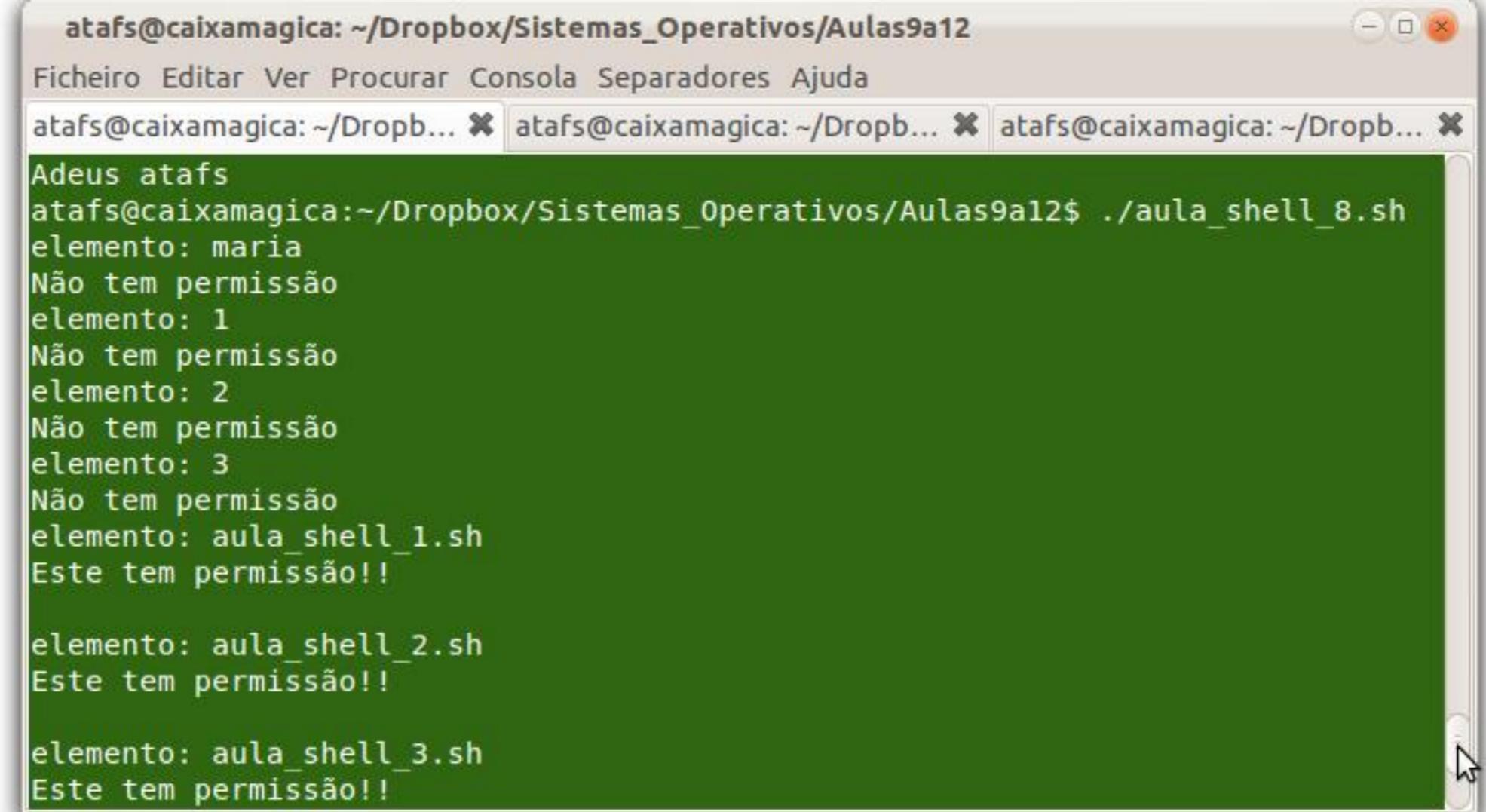
Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

Abrir Guardar Desfazer Recortar Copiar Pesquisar

aula\_shell\_9.sh ✘ aula\_shell\_8.sh ✘ aula\_shell\_7.sh ✘ net\_shell\_procurar ...os em directorias.sh ✘

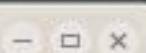
```
#!/bin/bash
#vai mandar para o bash interpretar... podia interpretar com piten, etc...
```

```
for i in "maria" 1 2 3 $1 *.sh; do
    echo "elemento: $i"
    if [ -x $i ]; then
        echo "Este tem permissão!!"
        echo ""
    else
        echo "Não tem permissão"
    fi
done
echo "Adeus $USER"
```



Adeus atafs  
atafs@caixamagica:~/Dropbox/Sistemas\_Operativos/Aulas9a12\$ ./aula\_shell\_8.sh  
elemento: maria  
Não tem permissão  
elemento: 1  
Não tem permissão  
elemento: 2  
Não tem permissão  
elemento: 3  
Não tem permissão  
elemento: aula\_shell\_1.sh  
Este tem permissão!!  
  
elemento: aula\_shell\_2.sh  
Este tem permissão!!  
  
elemento: aula\_shell\_3.sh  
Este tem permissão!!

aula\_shell\_9.sh (~/Dropbox/Sistemas\_Operativos/Aulas9a12) - gedit



Ficheiro Editar Ver Procurar Ferramentas Documentos Ajuda

Abrir ▾ Guardar Desfazer

aula\_shell\_8.sh ✘ aula\_shell\_7.sh ✘ net\_shell\_procurar ...os em directorias.sh ✘ aula\_shell\_9.sh ✘

```
#!/bin/bash

if [ -z $1 ]; then
    echo "Introduza um argumento"; read opc
    ↪

    case $opc in
        ?enfica|sporting) echo "Lisboa" ;;
        porto|boavista) echo "Porto" ;;
        ?scte) echo "O Iscte é o Maior" ;;
        *iul) echo "$opc deve pertencer ao ISCTE" ;;
        *) echo "Desconheço" ;;
    esac
fi

# faço aos if, consigo fazer coisas sempre que comece por
# sintaxe rígida... receitas sempre iguais... com meia dúz
```

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos/Aulas9a12

Ficheiro Editar Ver Procurar Consola Separadores Ajuda

atafs@caixamagica: ~/Dropbox/Sistemas\_Operativos... ✘ atafsa@caixamagica: ~/Dropbox/Sistemas\_Operativos... ✘

```
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$ ./aula_shell_9.sh
Introduza um argumento
iscte
O Iscte é o Maior
atafs@caixamagica:~/Dropbox/Sistemas_Operativos/Aulas9a12$
```