Iniciar sempre só co «shebang» #!/bin/bash

Variábeis:

As variábeis en bash só poden conter caracteres alfanuméricos e deben iniciar cunha letra ou un «_»

Correcto	Incorrecto
a12	12a
_a_3	-a_3
_aa_3	_aa-3

Substitución de variábeis

${V:-predet}	$V, ou «predet» se non se establece
${V:=predet}	$V (establecese a «predet» se non foi estabelecido)
${V:?err}	$V, ou «erro» se non se estabelece

Hai unha serie de variábeis propias:

$SHELL	a «shell» que estamos a executar
$RANDOM	número aleatorio
$$	PID do proceso actual
$?	codigo de retorno da última orde
$!	PID da última orde de fondo

Probas:

Probas sobre ficherios

-d	o ficheiro é un directorio
-e	o ficheiro existe
-f	o ficheiro é un ficheiro regular
-r	tes permiso de lectura no ficheiro
-s	o ficheiro existe e non está baleiro
-w	tes permiso de escritura no ficheiro
-x	tes permiso de execución ou busca
-O	es o propietario do ficheiro
-G	O grupo do ficheiro é igual ao teu.

Probas numéricas

-nt	ficheiro máis recente
-ot	ficheiro máis antigo
-gt, -ge, -eq, -nq, -le, -lt	corresponden a: >, >=, =, !=, <= e < respectivamente

Probas con cadeas

-z	a cadea é nula, ten lonxitude cero ou non existe
-n	a cadea non é nula
=	a cadea é igual a

Probas lóxicas

&&	AND (E) lóxico
||	OR (OU) lóxico
!	NOT (NON) lóxico

Ficheiros

Redirección de ficheiros

> ficheiro	crea e sobrescribe o ficheiro
>> ficheiro	engade na fin do ficheiro
< ficheiro	lé do fichero
A | B	encadea a saída de A como entrada de B

Diferencias entre [e [[:

[[: é máis «intelixente» que [, mais o script que vaiamos a empregar xa non se podería executar sobre calquera shell. Hai que valorar se imos controlar a execución do script ou non.

```
[ -e /tmp/ficheiro ]
[ -d /tmp ]
[ -f "$tmpfile" ]
[ $var1 -ge $var2 ]
[ $var1 -ne $var2 ]
```

</> Un exemplo do diferentes que son [e [[

```
[ "q$var" = "q" ] && echo var é nulo
[[ -z $var ]] && echo var é nulo
```

Sentencias de control:

</> Se verdadeiro entón fai isto; senón fai estoutro:

```
if [[ -d /tmp ]]; then
   echo "Fai isto"
else
   echo "Fai estoutro"
fi
```

</> No caso de coincidir con algunha, executalo:

```
case $var in
    algo)
        echo "var é algo"
    ;;
    nada)
        echo "var é nada"
    ;;
    *)
        echo "var é distinto de algo e de nada"
   ;;
esac
```

</> Mentres sexa verdadeiro fai isto:

```
while [[ $count -gt 5 ]]; do
    echo "O contador pon $count"
    count=$((count+1))
done
```

</> Para todos os valores da lista fai isto:

```
for i in $(seq 1 10); do
    echo $i
done
```

Oneliners:

Execución de varias ordes nunha soa liña:

orde 1 ; orde 2	executase primeiro a orde 1 e a seguir a orde 2
orde 1 & orde 2	execútanse case en paralelo: orde 1 pasa a segundo plano e inicia o proceso orde 2
orde 1 && orde 2	executase a orde 2 só se a orde 1 sae sen ningún erro
orde 1 || orde 2	executase a orde 2 só se a orde 1 sae con algún erro