

Esercitazione 1

la shell

Obiettivi

- Prendere confidenza con i comandi per la gestione del file system
- Utilizzare gli strumenti visti a lezione per creare tre semplici script bash

File system 1/2

1. Verificare il contenuto della propria home directory
2. Spostarsi nella directory superiore e visualizzarne il percorso
3. Verificare i diritti dei file presenti
4. Verificare se e dove è possibile creare dei file
5. Creare un file nella propria **home**, dandogli il nome del proprio username
 - come si può fare utilizzando il comando **whoami**?

File system 2/2

6. Verificare che gli altri utenti del proprio gruppo possono leggere il file ma non modificarlo
7. Creare una directory di nome `temp`, e inserire in essa un link al file creato al punto 5
8. Verificare il contenuto della propria home...è cambiato qualcosa rispetto a prima?
9. Cancellare il file creato al punto 5 e verificare il contenuto di `temp`
10. Cancellare la directory `temp`

Richiami: espansione

- Quando un comando contiene uno schema, la bash lo interpreta in due passi:
 1. **Sostituzione** dello schema con i nomi che fanno “match”
 2. **Esecuzione** del comando espanso

Es:

<pre>for i in ../* do echo \$i done</pre>	1. ../* viene espanso con la lista dei file contenuti nel direttorio padre 2. viene eseguito il comando vero e proprio, con <code>i</code> che cicla sugli elementi della lista espansa
---	---

Problemi

- Come faccio a creare un file di nome
`( :--> | ) ?`
Devo poter comunicare alla shell che i caratteri speciali vanno privati del loro significato e considerati come normali.
- Come faccio a dire che nel comando
`echo direttorio corrente: pwd`
`pwd` è esso stesso un comando da eseguire?
- ...

Quoting

<code>\</code>	Il carattere successivo non viene interpretato come carattere speciale
<code>'<testo>'</code>	<code><testo></code> viene protetto da qualsiasi tipo di espansione
<code>'<testo>'</code>	<code><testo></code> viene interpretato come un comando da eseguire
<code>"<testo>"</code>	<code><testo></code> viene protetto da espansioni, con eccezione di • <code>'</code> • <code>\$</code> • <code>\</code>

Problemi (risolti)

- Come faccio a creare un file di nome
(:--> |) ?

Che succede levando questa
barra?

> \(\ :-->\ \ | \ \) oppure

> "(:--> |)" oppure

> '(:--> |)'

- **echo direttorio corrente: 'pwd'**

Espressioni

- Come faccio a eseguire una espressione aritmetica **1 + 3**?

Uso della keyword **expr** → **expr 1 + 3**

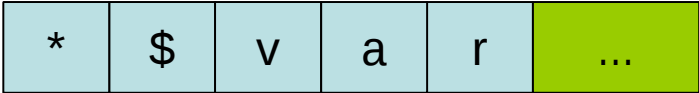
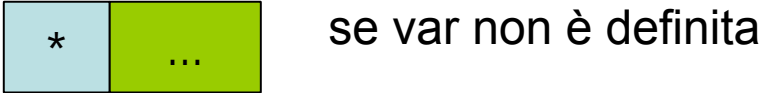
- Se dispongo di una variabile numerica var, come faccio a dire che nel comando

echo risultato: var+1

var+1 è il risultato della corrispondente espressione?

echo risultato: 'expr \$var + 1'

Altri esempi

<code>rm '\$var'</code>	Rimuove i file che fanno match con 
<code>rm "\$var"</code>	Rimuove i file che fanno match con  se var non è definita  altrimenti
<code>echo '<'pwd'>'</code>	Mostra <code><'pwd'></code>
<code>echo "<'pwd'>"</code>	Mostra <code><direttorio corrente></code>

Script

- File di testo contenente una serie di comandi
 - Ha due caratteristiche:
 - Deve essere eseguibile
 - Deve cominciare con **#!/bin/bash**
- Se non specificato, viene utilizzata la shell di **default** dell'utente (definita in /etc/passwd)
- **# è visto dalla shell come commento** ma
 - **#! è visto dal sistema operativo come magic number** che identifica uno script; il s.o. capisce così che deve invocare **/bin/bash** come interprete dello script
 - **NOTA BENE:** uno script viene **interpretato** (non esiste una fase di compilazione)!
 - L'interprete acquisisce ed esegue i comandi uno per volta

Esercizio 1

- Creare uno script che abbia la sintassi
`./ps_monitor.sh [#lines]`
- Lo script,
 - in caso di assenza dell'argomento, deve mostrare i processi di tutti gli utenti (compresi quelli senza terminale di controllo) con anche le informazioni sul nome utente e ora di inizio
 - Se viene passato come argomento un intero (**`#lines`**) deve mostrare i primi `#lines` processi ordinati in senso crescente per ora di inizio del processo
- NOTA: non tutte le righe prodotte in output da **`ps`** hanno contenuto informativo rilevante

Esercizio 2

- Creare uno script che abbia la sintassi

`./lines_counter.sh <directory> [up|down]`

- Lo script deve elencare i file contenuti nella directory con relativo numero di linee, ordinati in senso crescente (**up**) o decrescente (**down**)
- NOTA: controllare
 - Che il primo argomento sia effettivamente una directory
 - Che il secondo argomento sia effettivamente la stringa **up** o **down**

Esercizio 3

- Creare uno script che abbia la sintassi

`./backup.sh <nome file> <nome backup>`

- Se il file è una directory, lo script deve
 - creare una sottodirectory (rispetto a livello corrente) di nome:
- Se il file è un file normale, lo script deve crearne 5 copie di nome

`<nome file>*i<nome backup> i=1..5`

Suggerimenti generali

- Guardare ATTENTAMENTE il man di **bash** per la sintassi degli script
- I test, utilizzati come condizioni logiche negli script, sono un'abbreviazione del comando **test**

<pre>if [-n \$stringa] then ...</pre>	<pre>if test -n \$stringa then ...</pre>
sono equivalenti	

- Quindi, per la sintassi e le varie opzioni consultare il man di **test**

Suggerimenti sugli esercizi

- ESERCIZIO 1:
 - Guardare ATTENTAMENTE il man di **ps** e **head**
- ESERCIZIO 2:
 - Guardare ATTENTAMENTE il man di **wc** e **sort**
 - Bisogna poter esprimere una cosa del tipo: “lista di tutti i file che stanno nella directory passata come argomento”...
 - Fare molta attenzione a come si utilizza l’argomento... se la directory passata o i file in essa contengono degli spazi o altri caratteri speciali che succede?
- ESERCIZIO 3:
 - Guardare ATTENTAMENTE il man di **cp**
 - Anche qua, fare attenzione al quoting