Sistemi Operativi

AAF - Secondo anno - 3CFU

A.A. 2024/2025

Corso di Laurea in Matematica

Bash: primi comandi

Annalisa Massini

Dipartimento di Informatica
Sapienza Università di Roma

Argomenti trattati

1. Bash script
   - Primi passi su shell scripting
2. Generalità sui comandi
   - Informazioni sui comandi

Bash script

Primi passi su shell scripting

Cominciare

- La bash permette di avere un vero e proprio linguaggio di programmazione i cui comandi base sono i comandi della shell (non solo quelli visti finora) più gli assegnamenti e i controllori di flusso
- È possibile prendere decisioni (ad esempio, con l'`if`) ed eseguire cicli (ad esempio, con il `for` e il `while`), così alcuni comandi potrebbero non essere eseguiti o eseguiti più volte
- Se ci sono errori di sintassi:
  - la parte che precede l'errore viene sempre eseguita
  - la parte che segue l'errore potrebbe essere eseguita o no, a seconda della gravità dell'errore

Cominciare

- Tutto quello che si scrive sulla bash interattiva può essere messo in uno script; per separare i comandi, si può usare l'andata a capo al posto del ;
- Il viceversa non è vero, perché le andate a capo possono essere solo negli script, infatti, nella shell interattiva, premere invio vuol dire *esegui il comando*

Esempi

- Scrivere uno script che esegue il comando echo nei seguenti modi:

```
# script di prova su echo
echo "Hello World"
echo "Hello * World" # commento 1
echo "Hello \"*\" World"
echo Hello * World
echo "Hello " World
echo Hello " " World # commento 2
echo 'Hello ' World
echo `Hello World`
echo `Hello` World
```

- usando # si possono mettere commenti
- si può eseguire con bash e il nome del file

Shell Programming

Informazioni sui comandi

Informazioni sui comandi

- Ogni comando genera un processo
- Il processo generato, quando termina, restituisce un **return code** o **exit code** alla bash
  - Return code uguale a 0 indica *tutto ok*
  - Se il return code assume un valore tra 1 e 255 vuol dire che c'è stato un *errore*
  - ci possono essere molte cause di errore (non più di 255...)
  - per vedere il codice dell'errore: `echo $?`
  - se un comando non è riconosciuto, viene restituito 127
  - se un comando è costituito da una sequenza di comandi separati da `;` allora l'exit code è quello dell'ultimo comando eseguito
  - se un comando viene eseguito in background, l'exit code è 0; per prendere il suo vero exit code, occorre usare il comando builtin `wait`

Esercizi

- **Esercizio:** capire se l'exit code di `ls` è 0 o diverso da 0 nei seguenti casi:
  - nessun argomento
  - un file esistente come argomento
  - un file esistente ma non accessibile in lettura come argomento
  - una directory esistente ma non accessibile in lettura come argomento
  - un file non esistente come argomento
  - un file esistente e uno non esistente come argomento

Esercizi

- **Esercizio:** capire se l'exit code di `find` è 0 o diverso da 0 nei seguenti casi:
  - le opzioni non sono corrette (ad esempio: `-name file1 file2`)
  - non trova nessun file
  - trova 1 file
  - trova più di un file

Informazioni sui comandi

- La bash permette l'**esecuzione condizionale** dei comandi
- Gli operatori && e || rappresentano i connettori logici *and* e *or*, rispettivamente, e possono essere utilizzati per sfruttare il codice di ritorno restituito dai comandi
- Un comando viene eseguito solo se una data condizione è vera
- Il modo più semplice per farlo è condizionare un comando all'esecuzione di un comando precedente usando l'exit code
- Se l'esecuzione del comando precedente è:
  - *corretta* allora l'*exit code è 0*
  - *sbagliata* allora l'*exit code è diverso da 0*

Informazioni sui comandi

- Tenendo conto del significato dei connettori logici *and* e *or*, l'interprete esegue i comandi di una lista interpretandoli da sinistra verso destra
- Si ha il seguente comportamento:
  - `comando1 && comando2`: `comando2` viene eseguito solo se `comando1` è corretto, in caso contrario l'esecuzione viene interrotta
  - `comando1 || comando2`: se `comando1` restituisce il valore vero, l'elaborazione del comando composto viene interrotta perché è già possibile stabilire il valore restituito dall'intero comando, quindi `comando2` viene eseguito solo se `comando1` è sbagliato

Informazioni sui comandi

- Si possono concatenare più di 2 comandi, sia con && che || e anche usare le parentesi
- Ad esempio, con il comando composto

  `(comando1 && comando2) || comando3`

  se `comando1` restituisce il valore vero allora viene eseguito anche `comando2`,

  altrimenti, se `comando1` restituisce il valore falso, viene eseguito `comando3`