

# Bash: history list (I)

L'**history list** è un tool fornito dalla shell bash che consente di evitare all'utente di digitare più volte gli stessi comandi:

- bash memorizza nell'history list gli ultimi **500 comandi** inseriti dall'utente;
- l'history list viene memorizzata nel file `.bash_history` nell'home directory dell'utente al momento del logout (e riletta al momento del login);
- il comando `history` consente di visualizzare la lista dei comandi:

```
$ history | tail -5
 511 pwd
 512 ls -al
 513 cd /etc
 514 more passwd
 515 history | tail -5
```

- ogni riga prodotta dal comando `history` è detta **evento** ed è preceduta dal **numero dell'evento**.

## Bash: history list (II)

Conoscendo il numero dell'evento corrispondente al comando che vogliamo ripetere, possiamo eseguirlo, usando il metacarattere !:

```
$ !515
```

```
history | tail -5
```

```
512 ls -al
```

```
513 cd /etc
```

```
514 more passwd
```

```
515 history | tail -5
```

```
516 history | tail -5
```

Se l'evento che vogliamo ripetere è l'ultimo della lista è sufficiente usare !!:

```
$ !!
```

```
history | tail -5
```

```
513 cd /etc
```

```
514 more passwd
```

```
515 history | tail -5
```

```
516 history | tail -5
```

```
517 history | tail -5
```

## Bash: history list (III)

Oltre a riferirsi agli eventi tramite i loro numeri, è possibile eseguire delle ricerche testuali per individuare quello a cui siamo interessati:

```
$ !ls
```

```
ls -al
```

```
total 491
```

```
drwxr-xr-x 16 root root 0 Oct 15 21:35 .
```

```
drwxr-xr-x 16 root root 0 Oct 15 21:35 ..
```

```
-rw-r--r-- 1 root root 87515 Jul 10 04:28 Mutttrc
```

```
drwxr-xr-x 2 root root 0 Oct 15 21:27 WindowMaker
```

```
...
```

In questo modo la shell comincia a cercare a partire dall'ultimo evento, procedendo a ritroso, nell'history list un comando che inizi con `ls`.

Racchiudendo con due caratteri `?` la stringa da ricercare (e.g. `$ !?ls?`), la shell controllerà che quest'ultima appaia in un punto qualsiasi del comando (non necessariamente all'inizio).

## Bash: history list (IV)

Talvolta può capitare di voler ripetere un comando eseguito precedentemente dopo aver operato qualche modifica:

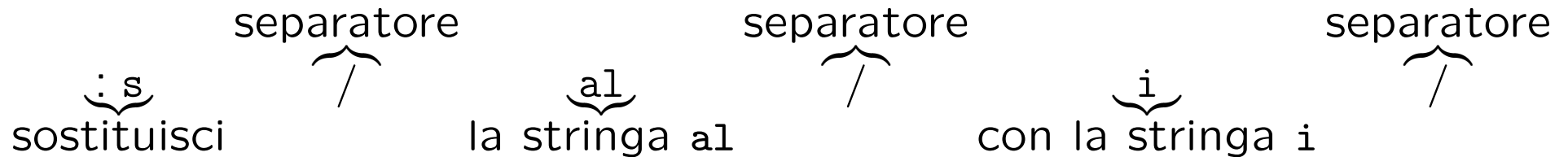
```
$ !ls:s/a1/i/
```

```
ls -i
```

```
1067566292 Mutttrc          123123 mib.txt
 204714 WindowMaker       123127 mime.conf
```

```
...
```

In questo modo la shell, dopo aver trovato l'evento cercato (`ls -a1`), sostituisce la stringa `a1` con `i` (`:s/a1/i/`), producendo il comando `ls -i`.

  
:s      separatore      a1      separatore      i      separatore  
sostituisci      la stringa a1      con la stringa i

# Bash: command line editing

La shell bash mette a disposizione dell'utente dei semplici **comandi di editing** per facilitare la ripetizione degli eventi:

- utilizzando i **tasti cursore**:
  - con la **freccia verso l'alto** si scorre l'history list a ritroso (un passo alla volta) facendo apparire al prompt il comando corrispondente all'evento;
  - analogamente con la **freccia verso il basso** si scorre l'history list nella direzione degli eventi più recenti.
  - le frecce sinistra e destra consentono di spostare il cursore sulla linea di comando verso il punto che si vuole editare;
- le combinazioni di tasti **Ctrl-A** e **Ctrl-E** spostano il cursore, rispettivamente all'inizio ed alla fine della linea di comando;
- il tasto **Backspace** consente di cancellare il carattere alla sinistra del cursore;
- il tasto invio (**enter**) esegue il comando.

# Bash: command completion (I)

Una caratteristica molto utile della shell bash è la sua abilità di **tentare di completare** ciò che stiamo digitando al prompt dei comandi (nel seguito <Tab> indica la pressione del tasto Tab).

```
$ pass<Tab>
```

La pressione del tasto <Tab> fa in modo che la shell, sapendo che vogliamo impartire un comando, cerchi quelli che iniziano con la stringa `pass`. Siccome l'unica scelta possibile è data da `passwd`, questo sarà il comando che ritroveremo automaticamente nel prompt.

Se il numero di caratteri digitati è insufficiente per la shell al fine di determinare univocamente il comando, avviene quanto segue:

- viene prodotto un suono di avvertimento al momento della pressione del tasto Tab;
- alla seconda pressione del tasto Tab la shell visualizza una lista delle possibili alternative.
- digitando ulteriori caratteri, alla successiva pressione del tasto Tab, la lunghezza della lista diminuirà fino ad individuare un unico comando.

## Bash: command completion (II)

Oltre a poter completare i comandi, la shell bash può anche completare i nomi dei file usati come argomento:

```
$ tail -2 /etc/p<Tab><Tab>
```

```
passwd printcap profile
```

```
$tail -2 /etc/pa<Tab><Tab>
```

```
bianchi:fjKppCZxEvouc:500:500:~/home/bianchi:/bin/bash
```

```
rossi:Yt1a4ffkGr02:501:500:~/home/rossi:/bin/bash
```

In questo caso alla prima doppia pressione del tasto Tab, la shell presenta tre possibili alternative; digitando una a e premendo il tasto Tab, la shell ha una quantità di informazione sufficiente per determinare in modo univoco il completamento del nome di file.

# Alias

Alias già visti:

1. . (directory corrente)
2. .. (directory madre)

Esiste anche l'alias `~user` che sta per la directory home dell'utente `user`:

```
user> cd ~user          # equivale a cd /home/user
user> cd ~user/doc     # equivale a cd /home/user/doc
```

Gli alias sono trattati dalla shell come i metacaratteri, nel senso che la shell scandisce la linea di comando impartita dall'utente processando i caratteri alias prima di eseguire i comandi.



## Il comando alias

Il comando `alias` serve per creare nuovi alias:

```
user> alias dir='ls -a'
```

```
user> dir
```

```
.  ..  .bash_history
```

```
user> alias ls='ls -l'
```

```
user> ls *.java
```

```
-rw-r--r--    1 user      users          0 Oct 16 17:24 Figura.java
-rw-r--r--    1 user      users          0 Oct 16 17:24 Quadrato.java
-rw-r--r--    1 user      users          0 Oct 16 17:24 Triangolo.java
```

Per rimuovere uno o più alias:

```
user> unalias dir ls
```

All'uscita dalla shell gli alias creati con il comando `alias` sono automaticamente rimossi.

## Metacaratteri comuni a tutte le shell (I)

Simbolo	Significato	Esempio d'uso
>	Ridirezione dell'output	<code>ls &gt;temp</code>
>>	Ridirezione dell'output (append)	<code>ls &gt;&gt;temp</code>
<	Ridirezione dell'input	<code>wc -l &lt;text</code>
<<delim	ridirezione dell'input da linea di comando (here document)	<code>wc -l &lt;&lt;delim</code>
*	Wildcard: stringa di 0 o più caratteri, ad eccezione del punto (.)	<code>ls *.c</code>
?	Wildcard: un singolo carattere, ad eccezione del punto (.)	<code>ls ?.c</code>
[...]	Wildcard: un singolo carattere tra quelli elencati	<code>ls [a-zA-Z].bak</code>
{...}	Wildcard: le stringhe specificate all'interno delle parentesi	<code>ls {prog,doc}*.txt</code>

## Metacaratteri comuni a tutte le shell (II)

Simbolo	Significato	Esempio d'uso
	Pipe	<code>ls   more</code>
;	Sequenza di comandi	<code>pwd;ls;cd</code>
	Esecuzione condizionale. Esegue un comando se il precedente fallisce.	<code>cc prog.c    echo errore</code>
&&	Esecuzione condizionale. Esegue un comando se il precedente termina con successo.	<code>cc prog.c &amp;&amp; a.out</code>
(...)	Raggruppamento di comandi	<code>(date;ls;pwd)&gt;out.txt</code>
#	Introduce un commento	<code>ls # lista di file</code>
\	Fa in modo che la shell non interpreti in modo speciale il carattere che segue.	<code>ls file.\*</code>
!	Ripetizione di comandi memorizzati nell'history list	<code>!ls</code>

# Esercizi (I)

- Ridefinire il comando `rm` in modo tale che non sia chiesta conferma prima della cancellazione dei file.
- Definire il comando `rmi` (`rm` interattivo) che chiede conferma prima di rimuovere un file.
- Sapendo che il comando `ps` serve ad elencare i processi del sistema, scrivere una pipeline che fornisca in output il numero di tutti i processi in esecuzione.
- Salvare in un file di testo l'output dell'ultimo evento contenente il comando `ls`.
- Scrivere un comando che fornisce il numero dei comandi contenuti nella history list.

## Esercizi (II)

- Scrivere un comando che fornisce i primi 15 comandi della history list.
- Quali sono i comandi Unix disponibili nel sistema che iniziano con `ls`?
- Fornire almeno due modi diversi per ottenere la lista dei file della vostra home directory il cui nome inizia con `al`.
- Qual è l'effetto dei seguenti comandi?
  - `ls -R || (echo file non accessibili > tmp)`
  - `(who | grep rossi) && cd ~rossi`
  - `(cd / ; pwd ; ls | wc -l )`