I Metacaratteri della Shell Unix

La shell Unix riconosce alcuni caratteri speciali, chiamati **metacaratteri**, che possono comparire nei comandi.

Quando l'utente invia un comando, la shell lo scandisce alla ricerca di eventuali metacaratteri, che processa in modo speciale.
Una volta processati tutti i metacaratteri, viene eseguito il comando.

Esempio:

```
user> ls *.java

 Albero.java          div.java       ProvaAlbero.java
 AreaTriangolo.java   EasyIn.java    ProvaAlbero1.java
 AreaTriangolo1.java  IntQueue.java
```

Il **metacarattere** `*` all'interno di un pathname è un'**abbreviazione** per un nome di file. Il pathname `*.java` viene espanso dalla shell con tutti i nomi di file che terminano con l'estensione `.java`. Il comando `ls` fornisce quindi la lista di tutti e soli i file con tale estensione.

Abbreviazione del Pahtname

I seguenti metacaratteri, chiamati **wildcard** sono usati per **abbreviare** il nome di un file in un pathname:

Metacarattere	**Significato**
*	stringa di 0 o più caratteri
?	singolo carattere
[ ]	singolo carattere tra quelli elencati
{ }	stringa tra quelle elencate

Esempi:

```
user> cp /JAVA/Area*.java /JAVA_backup
```

copia tutti i files il cui nome inizia con la stringa `Area` e termina con l'estensione `.java` nella directory `JAVA_backup`.

```
user> ls /dev/tty?
/dev/ttya /dev/ttyb
```

... esempi

```
user> ls /dev/tty?[234]
/dev/ttyp2 /dev/ttyp4 /dev/ttyq3 /dev/ttyr2 /dev/ttyr4
/dev/ttyp3 /dev/ttyq2 /dev/ttyq4 /dev/ttyr3

user> ls /dev/tty?[2-4]
/dev/ttyp2 /dev/ttyp4 /dev/ttyq3 /dev/ttyr2 /dev/ttyr4
/dev/ttyp3 /dev/ttyq2 /dev/ttyq4 /dev/ttyr3

user> mkdir /user/studenti/rossi/{bin,doc,lib}
```

crea le directory `bin`, `doc`, `lib` .

Il "quoting"

Il meccanismo del **quoting** è utilizzato per inibire l'effetto dei metacaratteri. I metacaratteri a cui è applicato il quoting perdono il loro significato speciale e la shell li tratta come caratteri ordinari.
Ci sono tre meccanismi di quoting:

- il metacarattere di **escape** \ inibisce l'effetto speciale del metacarattere che lo segue:

```
user> cp file file\?
user> ls file*
file     file?
```

- tutti i metacaratteri presenti in una stringa racchiusa tra **singoli apici** perdono l'effetto speciale:

```
user> cat 'file*?'
...
```

- i metacaratteri per l'abbreviazione del pathname presenti in una stringa racchiusa tra **doppi apici** perdono l'effetto speciale (ma non tutti i metacaratteri della shell):

```
user> cat "file*?"
...
```

Ridirezione dell'I/O

Di default i comandi Unix prendono l'input da **tastiera** (**standard input**) e mandano l'**output** ed eventuali **messaggi di errore** su video (**standard output, error**).

L'input/output in Unix può essere **rediretto** da/verso **file**, utilizzando opportuni metacaratteri:

Metacarattere	**Significato**
>	ridirezione dell'output
>>	ridirezione dell'output (append)
<	ridirezione dell'input
<<	ridirezione dell'input dalla linea di comando ("here document")
2>	ridirezione dei messaggi di errore (bash Linux)

Esempi:

```
user> ls LABSO > temp
user> more temp
lezione1.aux lezione1.log lezione1.tex lezione2.dvi lezione2.tex
lezione1.dvi lezione1.ps lezione2.aux lezione2.log lezione2.tex
```

... esempi

```
user> echo ciao a tutti >file     # ridirezione dell'output
user> more file
ciao a tutti

user> echo ciao a tutti >>file      # ridirezione dell'output (append)
user> more file
ciao a tutti
ciao a tutti
```

Il comando `wc` (**word counter**) fornisce numero di linee, parole, caratteri di un file:

```
user> wc <progetto.txt
21 42 77

user> wc <<delim      # here document
?  queste linee formano il contenuto
?  del testo
?  delim
2    7    44

user> man -s2 passwd      # ridirezione dei messaggi di errore
No entry for passwd in section(s) 2 of the manual.
user> man -s2 passwd 2>temp
```

Pipe

Il metacarattere | (**pipe**) serve per comporre n comandi "in cascata" in modo che l'output di ciascuno sia fornito in input al successivo. L'output dell'ultimo comando e' l'output della pipeline.

La sequenza di comandi

```
user> ls /usr/bin > temp
user> wc -w temp
459
```

ha lo stesso effetto della pipeline:

```
user> ls /usr/bin | wc -w
459
```

I comandi `ls` e `wc` sono eseguiti in parallelo: l'output di `ls` è letto da `wc` mano a mano che viene prodotto.

Per mandare in stampa la lista dei files in `/usr/bin`:

```
user> ls /usr/bin | lpr
```

Per visualizzare l'output di `ls` pagina per pagina

```
user> ls | more
```

Esercizi

- Scrivete un unico comando (pipeline) per
  - copiare il contenuto della directory `dir1` nella directory `dir2`;
  - fornire il numero di file (e directory) a cui avete accesso, contenuti ricorsivamente nella directory studenti (si può utilizzare `ls -R`? e con il comando `find`?);
  - fornire la lista dei file della home directory il cui nome è una stringa di 3 caratteri seguita da un numero.

- Qual è la differenza tra i seguenti comandi?

```
ls
ls | cat
ls | more
```

- Quale effetto producono i seguenti comandi?
  - `uniq < file`, dove `file` è il nome di un file;
  - `who | wc -l` ;
  - `ps -e | wc -l` .