

Le espressioni regolari e GREP

Linux ha a disposizione una serie di programmi in grado di compiere ricerche su files di testo. Le ricerche si fanno tramite le *espressioni regolari* (o pattern), che consentono di trovare una stringa o un insieme di stringhe nel file che soddisfano determinate regole, dettate appunto dal pattern.

Le espressioni regolari sono largamente utilizzate dalla shell e da comandi e strumenti (ad esempio editor) in Unix. Uno dei comandi che li utilizza direttamente è il comando `grep` che consente di fare ricerche all'interno di file.

La sintassi del comando `grep` è:

```
grep pattern nomefile
```

un pattern viene cercato all'interno del file `nomefile` . Si consiglia di racchiudere sempre il pattern tra apici per evitare che la shell lo interpreti.

Ecco un semplice esempio:

```
grep 'Mario' rubrica.txt
```

cerca la parola `Mario` all'interno del file `rubrica.txt` visualizzando le linee del file che la contengono.

Ovviamente `'Mario'` è il pattern (in questo caso, banalmente, una stringa).

Usando i metacaratteri (`+` , `*` , `.` , ecc.) è possibile fare ricerche più complesse.

Ecco una tabella dei principali metacaratteri usati per creare espressioni regolari:

<code>^</code>	vero all'inizio della riga
<code>\$</code>	vero alla fine della riga
<code>.</code>	vero per ogni singolo carattere tranne newline
<code>[str]</code>	vero per ogni singolo carattere in <code>str</code>
<code>[^str]</code>	vero per ogni singolo carattere non appartenente a <code>str</code>
<code>[a-b]</code>	vero per ogni carattere compreso tra <code>a</code> e <code>b</code>
<code>*</code>	vero per zero o più ripetizioni del carattere precedente
<code>+</code>	vero per una o più ripetizioni del carattere precedente (non supportato da <code>sed</code>)

esistono molti altri metacaratteri, ma non tutti i programmi li supportano.

È utile ricordare che, come tutti gli altri comandi di Unix, `grep` è un filtro, quindi si può scrivere:

```
cat rubrica.txt | grep 'Mario'
```

Esempi d'uso di metacaratteri:

```
grep '^A' file.txt
```

 cerca in un file di testo tutte le righe che iniziano per `A`

verificare che `^` risulta verificato solo se il carattere `A` si trova all'inizio di una riga (si può immaginare `^` come un carattere di inizio riga).

Analogamente

```
grep 'A$' file.txt
```

 cerca in un file di testo tutte le righe che finiscono per `A`.

```
grep '^tavolo$' file.txt
```

cerca le righe che contengono la parola tavolo come unica parola della riga

```
grep '[pcm]anna' file.txt
```

cerca le righe che contengono le **stringhe** panna , canna , manna, quindi anche le parole mannaia, spanna, ... ecc.

Se occorre, invece, cercare solo le tre **parole** panna , canna , manna, occorre utilizzare il comando

```
grep -e '^[pcm]nna$' -e '^[pcm]nna_' -e '_[pcm]nna$' -e  
'_[pcm]nna_' file.txt
```

(il carattere _ è stato usato per evidenziare l'esistenza di uno spazio, deve essere ovviamente sostituito con un carattere 'spazio').

L'opzione -e consente di effettuare elaborazioni multiple.

```
grep '^.....are' file.txt
```

cerca le stringhe iniziali di otto caratteri che finiscono per are :

```
grep '..$' file.txt
```

cerca le righe che contengono esattamente due caratteri.

```
grep '[aeiou]..' file.txt ( Da estendere con l'opzione -e )
```

cerca le parole di tre caratteri che iniziano con una vocale

```
grep '[0-9]$' file.txt
```

cerca tutte le righe che finiscono con una cifra.

```
grep -e '_[0-9]+$' -e '^[0-9]+$'
```

cerca tutte le righe che finiscono con un numero.

(N.B. nell'esempio il carattere _ indica uno spazio, ed il metacarattere + risulta verificato per **uno o più** caratteri compresi tra 0 e 9)

```
grep -e '^[^0-9]+$' -e '^^[^0-9]+$'
```

cerca tutte le righe che **non** finiscono con un numero.

```
ls -l | grep 'd.w..w..w.'
```

visualizza i nomi delle directory con permesso di scrittura per tutti gli utenti.