Repositories\Laboratorio-Sistemi-Operativi\Filters_Redirections_Pipelines.md

Filters, Redirection e Pipelines

- Il **filtro** è un programma o una subrutine che prende dati in standard input e "scrive" il risultato allo standard output. Parte più importante è che il filtro **NON** modifica il file di input.
- La **redirection** è un meccanismo che permette di cambiare il flusso di dati in ingresso e in uscita di un programma tramite l'utilizzo di > e <
- Le **pipelines** possono anche essere utilizzate insieme ai filtri. Una pipeline è una sequenza di comandi collegati tramite pipe | che permette di passare il risultato di un comando come input di un altro comando.

Filtri

head

`head` è un filtro che permette di visualizzare le prime righe di un file di testo

```
head [options] [file_name]
```

Per default sono visualizzare le prime 10 linee

head -n

```
head -n [number] [file_name]
```

Il `number` indica il numero di **righe** che si vogliono visualizzare

head -c

```
head -c [number] [file_name]
```

Il `number` indica il numero di **bytes** che si vogliono visualizzare (1 byte = 1 char)

tail

`tail` è un filtro che permette di visualizzare le ultime righe di un file di testo

```
tail [options] [file_name]
```

Le opzioni sono le stesse di head, unica cosa è che i bytes e i caratteri verranno conteggiati dalla fine verso l'inizio del file

sort

`sort` è un filtro che permette di ordinare testo e file binari (di default ordina in ordine alfabetico)

Verrà fatto il sort anche delle linee vuote, che verranno inserite a inizio file

```
sort [options] [file_name]
```

nl

`nl` è un filtro che permette di numerare le righe di un file di testo

```
nl [options] [file_name]
```

Esempio:

```
nl -s '. ' -w 10 persone.txt
```

Esegue una formattazione personalizzata

- Il comando `-w 10` indica che verrà tabulato di 10
- Il comando `-s '. '` indica che verrà inserito un punto e uno spazio tra il numero di riga e il testo

wc

`wc` è un filtro che permette di contare le righe, le parole e i caratteri di un file di testo

```
wc [options] [path]
```

Esempio:

```
giacomix@Giacomo:/mnt/c/Users/giaco/desktop$ wc test.txt
  7  21 112 test.txt
```

L'output di default è così composto:
`linee | parole | caratteri | nome del file`

Tipi di `options` possibili:

- `-c`: visualizza solo il numero di bytes
- `-m`: visualizza solo il numero di caratteri
- `-l`: visualizza solo il numero di righe
- `-w`: visualizza solo il numero di parole

cut

`cut` è un filtro che permette di selezionare parti di una riga di testo separati da un delimitatore

`cut [options] [file_name]`

Prendendo come testo il seguente file:

```
Nome Cognome Età
Mario Rossi 35
Mario Bianchi 36
Paolo Rossi 20
Pippo Verdi 44
Pluto Verdone 21
Minnie Rosa 28
```

Eseguiamo questo comando:

`cut –f <selezione "colonne"> -d <separatore> test.txt`

Vediamo cosa succede sostituendo i parametri tra <>:

```
giacomix@Giacomo:/mnt/c/Users/giaco/desktop$ cut -f 1 -d ';' test.txt
Nome Cognome Età
Mario Rossi 35
Mario Bianchi 36
Paolo Rossi 20
Pippo Verdi 44
Pluto Verdone 21
Minnie Rosa 28
```

Essendo che nel file di testo il `;` non è presente il comando restituisce tutto il file senza nessuna separazione

```
giacomix@Giacomo:/mnt/c/Users/giaco/desktop$ cut -f 1 -d ' ' test.txt
Nome
Mario
Mario
Paolo
Pippo
Pluto
Minnie
```

Essendo che nel file il testo è separato da `spazi` il comando mi restituisce tutti i caratteri per ogni riga fino al primo spazio

```
giacomix@Giacomo:/mnt/c/Users/giaco/desktop$ cut -f 1,2 -d ' ' test.txt
Nome Cognome
Mario Rossi
Mario Bianchi
Paolo Rossi
Pippo Verdi
Pluto Verdone
Minnie Rosa
```

Essendo che nel file il testo è separato da `spazi` il comando mi restituisce tutti i caratteri per ogni riga fino al secondo spazio

uniq

`uniq` è un filtro che permette di rimuovere le righe duplicate di un file di testo

Attenzione: le righe duplicate devono essere consecutive, cioè non separate da spazi bianchi

```
uniq [file_name]
```

tac

`tac` è un filtro che permette di visualizzare il contenuto di un file di testo in ordine inverso

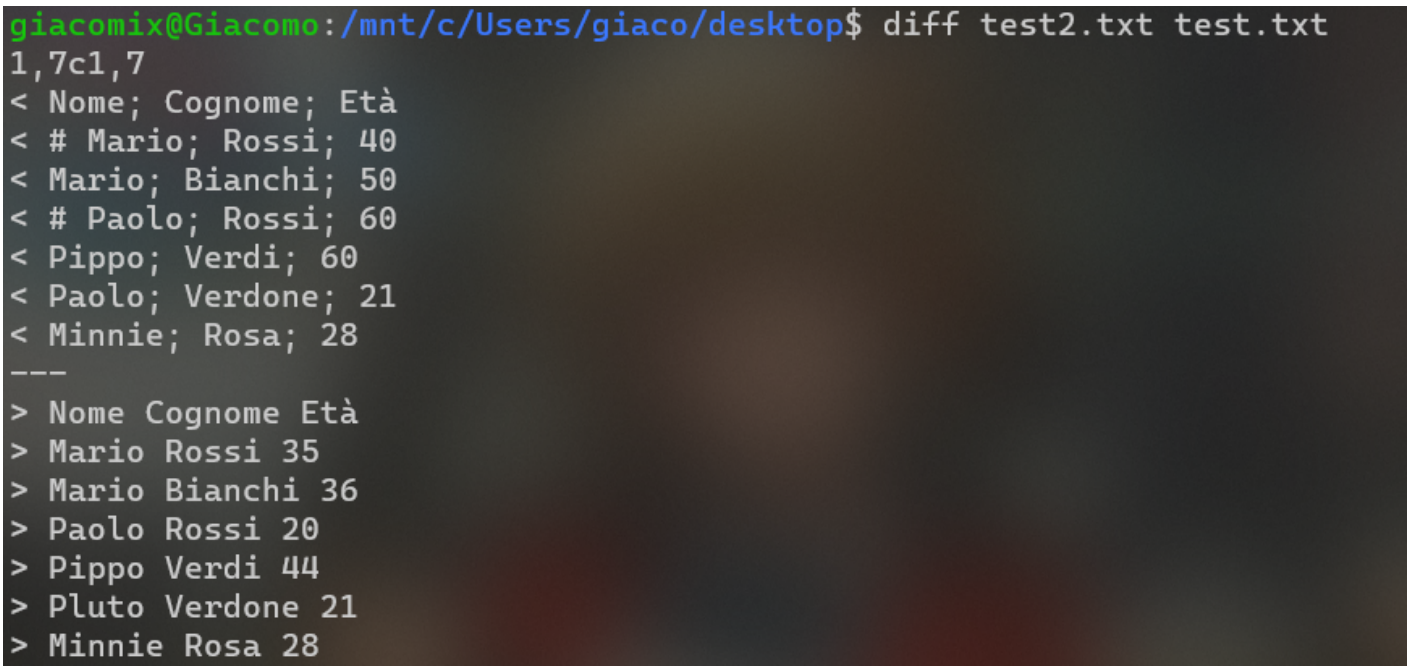
```
tac [file_name]
```

```
giacomix@Giacomo:/mnt/c/Users/giaco/desktop$ tac test.txt
Minnie Rosa 28
Pluto Verdone 21
Pippo Verdi 44
Paolo Rossi 20
Mario Bianchi 36
Mario Rossi 35
Nome Cognome Età
```

diff

Con `diff` è possibile confrontare due file di testo e visualizzare le differenze linea per linea

`diff [options] [file_name_1] [file_name_2]`



La prima rida deve essere letta nel seguente modo:

`righe del primo file | lettera | righe del secondo file`

La `lettera` può essere:

- `a`: riga aggiunta
- `d`: riga eliminata
- `c`: riga modificata

Inoltre:

- `<` indica le righe del primo file
- `>` indica le righe del secondo file
- `---` è il separatore fra files

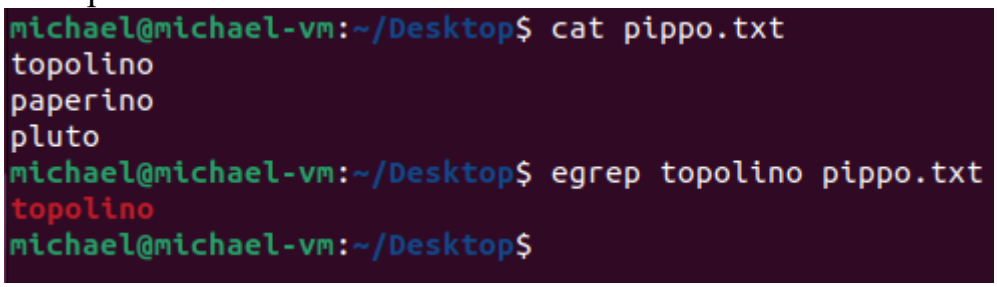
  Esempio di lettura della prima riga:
  `1,7c1,7`
  il range di righe *1* - *7* del **primo file** è stato modificato e sostituito il rispettivo range *1* - *7* del **secondo file**

egrep

`egrep [options] [pattern] [file_name]`

Il comando **grep** o **egrep** (extended grep) cerca nel file di testo una riga di testo che corrisponde ad uno o più patterns passati come parametro
Si usa perlopiù con le [Espressioni Regolari]

Esempio:



sed

Si tratta di un comando per cercare e rimpiazzare un'espressione con un'altra. Il formato è il seguente:

Esempio:

```
sed [command] [file_name]
```

NOTA: s sta per sostituisci e **g** per globalmente, senza **g** il comando sostituisce solo la ***prima istanza*** della riga di testo

- **-e** - se usato prima dell'espressione per rimpiazzare, è utile per combinarne più insieme

  Esempio:

  ```
  sed -e 's/a/A/' -e 's/b/B/' persone.txt
  ```

  Rimpiazza le lettere minuscole a e b con le loro lettere maiuscole

Ovviamente, è possibile utilizzare questo comando in combinazione con le espressioni regolari.

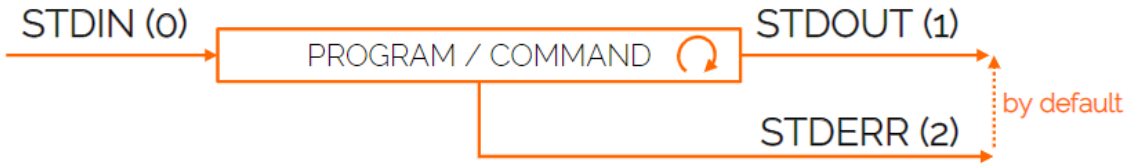
Pipeline e redirezione

I flussi di dati nella shell si servono di canali di comunicazione. Specificatamente, si parla di:

- **STDIN (0)** - è lo standard input
- **STDOUT (1)** - è lo standard output
- **STDERR (2)** - è lo standard error

Lo standard error viene reindirizzato di default nello standard output:

Esempio:



Per rdirezionare manualmente l'input e l'output si utilizzano i seguenti operatori:

- > - permette di redirezionare l'output di un programma o di un comando in un file, piuttosto che stamparlo a schermo.
  - E' possibile creare anche più redirection consecuitive

[n]<&word

Viene utilizzato per duplicare l'input dei descrittori file

Return Home