Programmazione 2
e Laboratorio di Programmazione

Corso di Laurea in

Informatica

Università degli Studi di Napoli "Parthenope"

Anno Accademico 2023-2024

Prof. Luigi Catuogno

1

Descrizione del Corso

Libro di testo

[FdP]

H. M. Deitel, P. J. Deitel

C++ Fondamenti di programmazione

II ed. (2014) Maggioli Editore (Apogeo Education)
ISBN: 978-88-387-8571-9

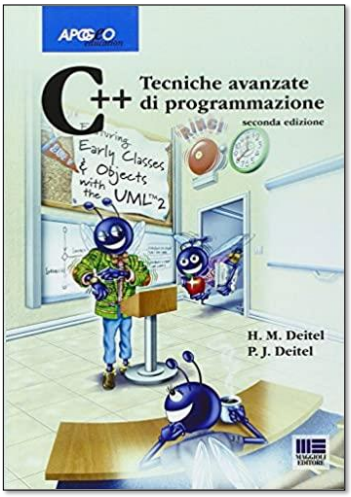
2

Descrizione del Corso

Libro di testo H. M. Deitel, P. J. Deitel

[TAP] C++ Tecniche avanzate di programmazione

II ed. (2011) Maggioli Editore (Apogeo Education)
ISBN: 978-88-387-8572-6



3

Descrizione del Corso

Altre risorse

[RFBash] Chet Ramey, Brian Fox
Bash Reference Manual
V5.2 - GNU software Foundation (2022)

4

Primi passi in Linux

Interazione con la linea di commando e l'ambiente della Shell

5

Redirezione dell'I/O e PIPE

6

Redirezione dell'I/O sui file

Impartiamo il seguente comando:

```
utente@linuxbox:~$ ls -l /etc
```

L'output è l'elenco dei file contenuti in quella directory, ed è piuttosto lungo. Possiamo salvare l'output del comando in un file per esaminarlo con calma.

```
utente@linuxbox:~$ ls -l /etc > ls-etc.txt
utente@linuxbox:~$ ls
ls-etc.txt
utente@linuxbox:~$ more ls-etc.txt
```

Il comando **`more`** visualizza il file indicato sullo schermo *a pagine. Una versione più recente è il comando* **`less`** *(sic).*

7

Redirezione dell'I/O sui file

Il carattere speciale **>** prescrive la *redirezione dell'output* verso il file indicato a seguire.

```
utente@linuxbox:~$ ls -l /etc > ls-etc.txt
```

- Se il file non esiste, viene creato
- Se il file esiste, il suo contenuto precedente è sostituito dal nuovo

```
utente@linuxbox:~$ ls -l . > ls-etc.txt
utente@linuxbox:~$ more ls-etc.txt
```

Il file adesso conterrà il contenuto della vostra home directory

```
utente@linuxbox:~$ rm ls-etc.txt
utente@linuxbox:~$
```

Il comando **`rm`** è utilizzato per rimuovere un file di cui si specifica il pathname. La rimozione è IRREVERSIBILE

8

Redirezione dell'I/O sui file

La sequenza **>>** prescrive la redirezione dell'output *in coda (append)* al file indicato a seguire.

```
utente@linuxbox:~$ cat /etc/hostname >informazioni
utente@linuxbox:~$ cat /etc/issue >> informazioni
```

- Se il file non esiste, viene creato
- Se il file esiste, il contenuto più recente è aggiunto in coda a quello precedente

Il comando **`cat`** visualizza sullo schermo il contenuto del (dei) file indicati sulla linea di comando. Se i file sono più d'uno, i contenuti sono visualizzati in sequenza.
A differenza di **`more`**, **`cat`** non è un *pager*

9

Redirezione dell'I/O sui file

Creaiamo un nuovo file di testo che contiene, per esempio, un elenco di nomi.

```
utente@linuxbox:~$ nano elenco.txt
```

Linux fornisce un ricco insieme di *filtri,* programmi che effettuano una elaborazione dell'input e ne producono il risultato in output. Ad esempio: **`sort`**

```
utente@linuxbox:~$ sort < elenco.txt
Abate
Abbate
Basile
Caputo
Esposito
utente@linuxbox:~$
```

Altri filtri storicamente presenti nei sistemi UNIX sono: **`shuf`**, **`uniq`**, **`tr`**

10

Redirezione dell'I/O sui file

Il carattere speciale **<** prescrive la *redirezione dell' input* dal file indicato a seguire. E' possibile combinare redirezioni nei due sensi.

```
utente@linuxbox:~$ sort < elenco > elenco_ordinato
```

E' opportuno notare che le redirezioni interessano i *canali di I/O su console* dei processi. Questi sono noti come *standard Input (**stdin**)* e *standard Output (**stdout**)* e sono unici (per ciascun processo). Pertanto non è possibile indicare più redirezioni sulla stessa linea di comando.

11

Le *pieline*

La *pipe* è uno strumento di comunicazione tra processi (*Inter-Process communication o IPC*)

- Un processo può inviare dati a un alto *come se stesse scrivendo in un file;*
- L'altro processo riceve i dati *come se li stesse leggendo da un file;*
- Nella shell, è possibile costruire *«catene di comandi»* in cui ciascuno riceve il suo input dal precedente e invia il suo output al successivo
- Il carattere speciale che indica la pipe è **|** *(barra verticale)*

12

Le *pieline: alcuni esempi*

L'output di **`ls`** diventa l'input di **`more`**

```
$ ls -l /etc | more
```

L'output di **`cat`** diventa l'input di **`sort`** il cui output diventa l'input di **`more`**

```
$ cat /etc/passwd | sort | more
```

L'output di **`cat`** diventa l'input di **`sort`** il cui output diventa l'input di **`uniq`**. L'output di **`uniq`** è redirezionato nel file **`elenco3`**.

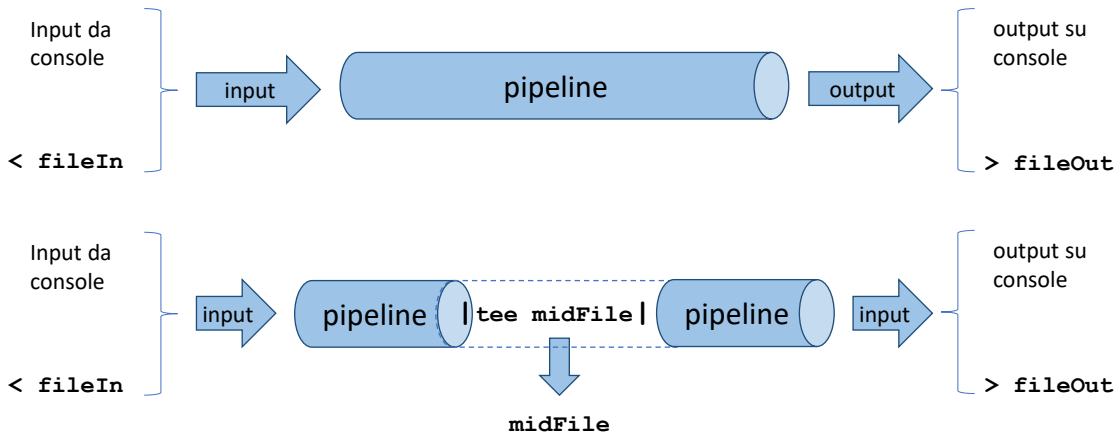
```
$ cat elenco1 elenco2 | sort | uniq > elenco3
```

L'input di **`shuf`** viene dal file **`elenco3`**, l'output è inviato a **`more`**.

```
$ shuf < elenco3 | more
```

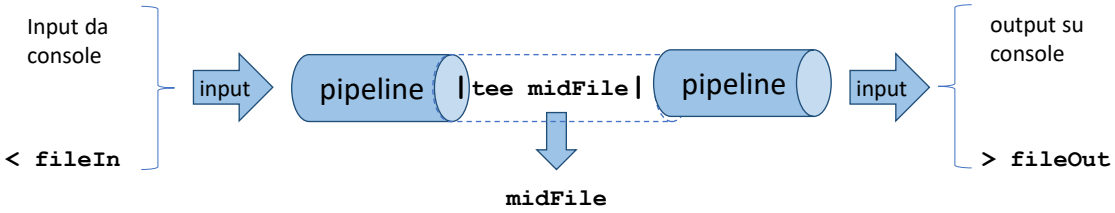
13

Le *pieline: il filtro* **`tee`**



14

Le *pieline: il filtro* **tee**



L'output di **cat** diventa l'input di **sort** il cui output diventa l'input di **tee. tee** copia il suo input nel suo output (e lo manda a **uniq**) ma ne fa anche una copia nel file **elenco2.5**

```
$ cat elenco1 elenco2 | sort | tee elenco2.5 |uniq > elenco3
```

15

Esercizio 3

Consultare il manuale in linea dei seguenti comandi e creare dei semplici esempi di utilizzo:

- **tee**
- **head**
- **tail**
- **grep**

16

Mappe

Guida allo studio e approfondimenti

17

Mappe

[RFBash] Il testo costituisce una guida completa (e abbastanza chiara) all'uso dell'interprete di comandi, dalle funzionalità base a quelle più avanzate.

Le pipeline sono introdotte nella sezione 3.2.3

Le redirezioni dell'I/O sono trattate nella sezione 3.6

18