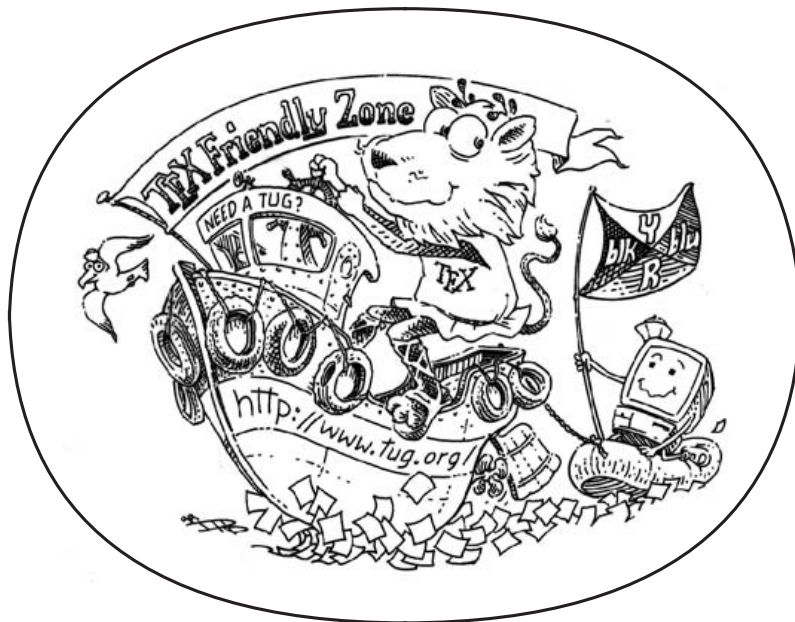


Lorenzo Pantieri & Tommaso Gordini

\LaTeX per l'impaziente



Un'introduzione all'arte di scrivere con \LaTeX

Lorenzo Pantieri · Tommaso Gordini

L^AT_EX per l'impaziente

Copyright © 2008-2017

I nomi commerciali, i loghi e i marchi registrati menzionati nella guida appartengono ai rispettivi proprietari, i pacchetti e le relative documentazioni ai rispettivi autori.

Contatti

✉ lorenzo.pantieri@gmail.com · Scrivi a Lorenzo Pantieri

✉ tommaso.gordini@gmail.com · Scrivi a Tommaso Gordini

Indice

Introduzione	v
1 Storia e filosofia	1
1.1 Storia	1
1.2 Filosofia	1
2 Installare e aggiornare	3
2.1 Ferri del mestiere	3
2.2 Windows	5
2.3 Mac	5
2.4 Linux	6
3 Basi	7
3.1 Per cominciare	7
3.2 Codifiche e lingue	9
3.3 File con cui si ha a che fare	11
3.4 File sorgente	12
3.5 Classi di documento	18
3.6 Gestire la pagina	20
3.7 Strutturare il documento	21
3.8 Stili di pagina	24
3.9 Indice generale, titoli e profondità	25
3.10 Riferimenti incrociati	26
3.11 Collegamenti ipertestuali e indirizzi elettronici	27
3.12 Pacchetti	28
3.13 Unità di misura tipografiche	30
3.14 Documenti di grandi dimensioni	30
4 Testo	33
4.1 Struttura del testo	33
4.2 Comporre i capoversi	33
4.3 Caratteri particolari e simboli	37
4.4 Modificare stile e corpo del font	39
4.5 Titoli e frontespizi	40
4.6 Note a margine e a piè di pagina	42
4.7 Evidenziare le parole	42
4.8 Ambienti testuali	43

5	Matematica	47
5.1	Formule in linea e in display	47
5.2	Nozioni introduttive	49
5.3	Operatori	55
5.4	Parentesi	56
5.5	Vettori e matrici	57
5.6	Spezzare formule lunghe	59
5.7	Raggruppare più formule	60
5.8	Modificare stile e corpo del font	61
5.9	Enunciati e dimostrazioni	62
6	Tabelle e figure	65
6.1	Strumenti fondamentali	65
6.2	Oggetti in testo e fuori testo	66
6.3	Tabelle	70
6.4	Figure	77
7	Bibliografia	81
8	Indice analitico	83
9	Personalizzazioni	85
9.1	Comandi e ambienti personali	85
9.2	Specialità	86
10	Revisione finale	87
10.1	Problemi orizzontali	87
10.2	Problemi verticali	88
	Bibliografia	89

Introduzione

\LaTeX è un programma di composizione tipografica liberamente disponibile, indicato soprattutto per scrivere documenti scientifici con la più alta qualità. Lo scopo di questo lavoro, rivolto sia a chi muove i primi passi in \LaTeX sia a quanti già lo conoscono, è di offrire ai suoi utenti italiani le conoscenze essenziali per poterlo usare con successo.

Queste note non scandagliano troppo i vari argomenti: dei pacchetti citati, infatti, si analizzano soltanto le impostazioni più importanti e se ne suggerisce l'uso, indirizzando alla relativa documentazione chi voglia approfondirne la conoscenza.

L'esposizione degli argomenti è articolata come segue.

Il primo capitolo traccia una breve storia di \LaTeX , indicandone idee di fondo e peculiarità.

Il secondo capitolo spiega come installare e aggiornare \LaTeX sul proprio calcolatore.

Il terzo capitolo presenta alcune nozioni fondamentali indispensabili per comprendere il funzionamento del programma: la sua lettura, perciò, è propedeutica a quella del resto della guida.

Il quarto capitolo descrive gli strumenti per trattare il testo.

Il quinto capitolo esplora uno dei principali punti di forza di \LaTeX : la composizione di formule matematiche.

Il sesto capitolo presenta i concetti e gli strumenti essenziali per comporre le tabelle, includere le figure in un documento e gestire la collocazione degli oggetti mobili sulla pagina.

Il settimo capitolo presenta gli strumenti per realizzare e gestire la bibliografia.

L'ottavo capitolo illustra le nozioni essenziali per generare l'indice analitico.

Il nono capitolo espone alcuni suggerimenti per fare in modo che \LaTeX produca risultati diversi da quelli predefiniti.

Il decimo capitolo dà alcuni suggerimenti per migliorare l'impaginazione del documento.

Questo *non* è un manuale su \LaTeX , ma piuttosto un tentativo di riordinare in forma scritta appunti accumulatisi nel tempo, via via che divenivamo abituali utenti di questo programma. In qualità di semplici appassionati non abbiamo nulla da insegnare; d'altra parte abbiamo studiato \LaTeX e l'abbiamo usato intensamente, acquisendo una certa esperienza che ci piacerebbe condividere con altri utenti.

Ecco lo spirito che ci ha guidati in questo lavoro: speriamo che possiate usare \LaTeX con il nostro stesso piacere.

Capitolo 1

Storia e filosofia

Questo capitolo presenta una breve storia di $\text{T}_{\text{E}}\text{X}$ e $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$, e ne indica idee di fondo e peculiarità.

1.1 Storia

1.1.1 $\text{T}_{\text{E}}\text{X}$

$\text{T}_{\text{E}}\text{X}$ è un programma di composizione tipografica realizzato da Donald Ervin Knuth, liberamente disponibile. Nel 1977 Knuth cominciò a scrivere un “motore” di tipocomposizione per esplorare le potenzialità degli strumenti di stampa digitale che allora cominciavano a muovere i primi passi. Il programma ha visto la luce nel 1982 ed è stato rivisto per l’ultima volta nel 2014. Il suo numero di versione converge a π (ora è 3, 141 592 65).

Knuth ha nascosto un trabocchetto nel nome del programma: $\text{T}_{\text{E}}\text{X}$, infatti, si pronuncia *tèch* (aspirando il *ch* finale) e non com’è scritto, perché è una parola greca scritta in greco maiuscolo (in lettere minuscole si scriverebbe $\tau\epsilon\chi$). È una radice comune non solo al greco $\tau\acute{\epsilon}\chi\eta$ (pron. *tèchne*, “arte”), ma viva ancora oggi in *tecnica* e *politecnico*. Questa etimologia illumina la scelta di Knuth: $\text{T}_{\text{E}}\text{X}$ è il nome perfetto per un programma “allo stato dell’arte”.

1.1.2 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$

$\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ ($\text{L}^{\text{A}}(\text{mport})\text{T}_{\text{E}}\text{X}$) è un programma di composizione tipografica realizzato da Leslie Lamport e liberamente disponibile, che usa $\text{T}_{\text{E}}\text{X}$ come motore di tipocomposizione. La prima versione pubblica di $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ risale al 1985, l’ultimo aggiornamento è del 2016.

$\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ *non* è $\text{T}_{\text{E}}\text{X}$. Per dare l’idea della differenza tra i due programmi, si potrebbe paragonare $\text{T}_{\text{E}}\text{X}$ a un corpo, e $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ al più popolare degli “abiti” che nel corso degli anni gli sono stati confezionati addosso per avvicinarlo al pubblico con sembianze amichevoli.

1.2 Filosofia

1.2.1 Composizione sincrona e asincrona

La caratteristica che più differenzia $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ dagli altri elaboratori di testo è il fatto che l’introduzione del testo e la sua composizione avvengono *in tempi diversi*.

Per modificare un documento scritto con un comune *word processor* (come Microsoft Word), l’utente agisce direttamente sul testo già composto come gli appare sul monitor, e ogni sua azione si traduce in una variazione *immediata* di quel testo. Perciò questo tipo di composizione è detto “composizione sincrona”. Per essere *davvero* sincrono, però, il programma deve puntare sulla rapidità della presentazione, ciò che impedisce di ottenere una

composizione perfetta. Anche se oggi la qualità dei programmi di videoscrittura migliora a ogni nuova versione, il compromesso tra velocità e qualità esiste sempre.

La “composizione asincrona”, invece, consiste nell’introdurre il testo in un editor concentrandosi unicamente su struttura logica e contenuto del documento, per darlo “in pasto” a un compositore (L^AT_EX, in questo caso) che lo impagina solo *successivamente*. L’utente può modificare il proprio lavoro anche dopo la composizione, ma sempre tenendo a mente che L^AT_EX non si limita ad aggiustarlo *nel punto* in cui è stato modificato e basta, ma riorganizza *l’intero capoverso* (e, dunque, l’intera pagina) nel migliore dei modi.

Questo secondo tipo di composizione è migliore del primo, perché la velocità di visualizzazione passa in secondo piano a tutto vantaggio della qualità: L^AT_EX elabora il testo sempre *nel suo complesso*, e per questo motivo può fare le scelte d’impaginazione migliori.

1.2.2 Istruzioni di marcatura

L’idea di Lamport era forte. E L^AT_EX centra l’obiettivo: l’utente può (quasi *deve*, si direbbe) astrarsi dai dettagli estetici che con un altro elaboratore di testo sarebbe costretto a introdurre a mano, per indirizzare le proprie energie sul *contenuto* del proprio lavoro.

L^AT_EX pretende dall’utente considerazioni sul *cosa*: «il mio documento sarà composto da un certo numero di capitoli, ciascuno diviso in paragrafi numerati, avrà indice generale e analitico, delle figure e qualche tabella». Al *come* pensa L^AT_EX, e lo fa molto bene. Per esempio, uno stesso file sorgente può generare in teoria documenti radicalmente diversi soltanto cambiandone la classe. (S’è detto *in teoria* perché, nella pratica, qualche aggiustamento manuale si rende di fatto *sempre* necessario.)

Un file da comporre con L^AT_EX è scritto in una “lingua” particolare costituita da *marcatori* (o *etichette logiche*, *mark-up* in inglese), ovvero istruzioni che il programma deve eseguire. Il file prodotto con l’editor, insomma, è un codice scritto in una sorta di linguaggio di programmazione, dato che contiene sia il vero e proprio testo del documento, sia i comandi che ordinano al programma di comporre quello che gli si dà in pasto.

Per dare l’idea di come appare un documento da comporre con L^AT_EX, si riportano alcune righe di *codice sorgente* (o più semplicemente *sorgente*), da scrivere nell’editor.

```
Due matrici  $n \times n$  complesse  $A$  e  $B$  si dicono \emph{simili} se esiste una
matrice  $n \times n$  invertibile  $T$  tale che
\begin{equation}
B=T^{-1}AT
\end{equation}
```

Il sorgente viene composto da L^AT_EX che, attraverso T_EX, produce il documento tipocomposto (*typeset*). Se il risultato non soddisfa, non si può modificare direttamente il documento a schermo, ma bisogna correggere il sorgente e poi ricomporlo.

L’esempio seguente riporta a sinistra lo stesso sorgente appena visto e a destra il risultato della composizione.

```
Due matrici  $n \times n$  complesse
 $A$  e  $B$  si dicono \emph{simili}
se esiste una matrice  $n \times n$ 
invertibile  $T$  tale che
\begin{equation}
B=T^{-1}AT
\end{equation}
```

Due matrici $n \times n$ complesse A e B si dicono
simili se esiste una matrice $n \times n$ invertibile
 T tale che

$$B = T^{-1}AT \quad (1.1)$$

Nei prossimi capitoli si spiegheranno tutte le istruzioni usate nell’esempio. Tuttavia, anche con pochi rudimenti di inglese si capisce facilmente quello che il linguaggio di marcatura ha specificato.

Capitolo 2

Installare e aggiornare

Questo capitolo spiega come procurarsi tutto l'occorrente per usare \LaTeX , come installarlo nel proprio calcolatore e come aggiornarne la distribuzione (che in questa guida si considererà *sempre* nell'ultima versione disponibile). I programmi descritti sono gratuiti e, dove non altrimenti indicato, disponibili per tutti e tre i sistemi operativi considerati in questo capitolo (Windows, Mac e Linux).

2.1 Ferri del mestiere

Per creare un documento con \LaTeX sono indispensabili almeno tre cose:

- un editor di testi con cui scrivere il file sorgente;
- il programma \LaTeX , che lo elabora e produce il documento tipocomposto;
- un programma per visualizzare il documento finito.

Si eviti da subito l'errore, molto frequente all'inizio, di confondere \LaTeX (un "motore" del tutto invisibile all'utente) con l'editor (ciò che effettivamente appare sullo schermo): i due programmi sono completamente *indipendenti*, tanto che in teoria si può usare un editor qualunque, da quelli più elementari già presenti nel proprio computer a quelli più complessi capaci di gestire numerosi linguaggi di programmazione.

In questa guida con \LaTeX s'intende *sempre* il programma `pdflatex`, e con le espressioni *documento composto* e *documento finito* un file in formato PDF.

2.1.1 Editor e visualizzatore

Anche se oggi molti software di videoscrittura tradizionali prevedono estensioni per trasformare il testo immesso in codice \LaTeX , si consiglia di usare senz'altro un editor *dedicato*, cioè destinato esclusivamente a \LaTeX e ideato per facilitarne il più possibile l'interazione con l'utente: i vantaggi sono molti e sostanziali. I più diffusi tra essi, inoltre, comprendono anche un visualizzatore di PDF, eliminando così i possibili problemi derivanti dall'usarne uno esterno.

Gli editor che gestiscono codice \LaTeX sono moltissimi, come si può scoprire con una veloce ricerca in Rete. Quelli consigliati in questa guida sono:

- `TEXworks` (<http://www.tug.org/texworks/>), multiplatforma;
- `TEXShop` (<http://www.uoregon.edu/~koch/texshop/>), specifico per Mac.

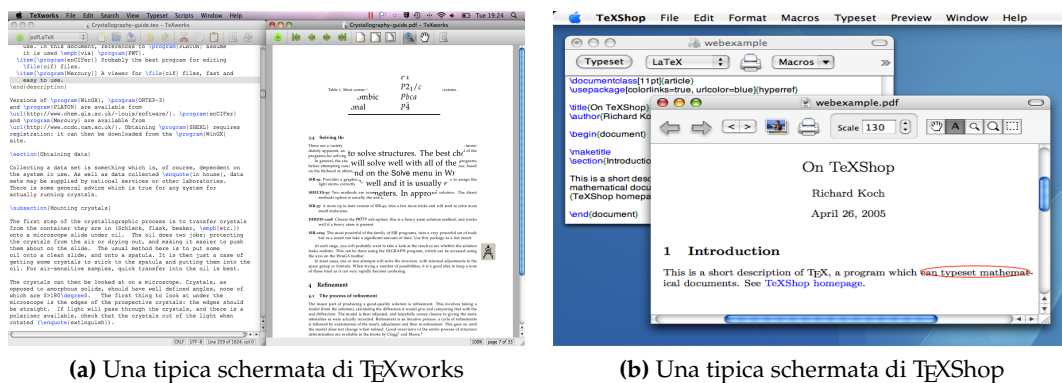
(a) Una tipica schermata di T_EXworks(b) Una tipica schermata di T_EXShop

Figura 2.1: Editor al lavoro

Questi ambienti di lavoro, adatti sia ai principianti che a utenti esperti, fanno parte di T_EX Live (si veda il paragrafo successivo), e perciò sono *già installati* sulle macchine (tranne che su Linux).

L'interfaccia grafica di T_EXworks e T_EXShop è molto semplice: se ne può vedere un esempio nella figura 2.1. I programmi sono compatibili con la tecnologia SyncT_EX: sorgente e anteprima cioè, sono *sincronizzati*, e si può “saltare” da un punto dell'uno al corrispondente punto dell'altro (e viceversa).

Per funzionare correttamente, ogni editor deve essere configurato: per esempio, bisogna impostarne la codifica (si veda il paragrafo 3.2.1) e il motore di composizione, e attivare il controllo ortografico della lingua del documento. Per farlo si consiglia di leggere attentamente la documentazione del proprio editor.

Un pregio di T_EXworks e T_EXShop è la capacità di interpretare i *commenti speciali*, cioè “righe magiche” che configurano l'editor in modo automatico, fissando una volta per tutte le impostazioni più importanti del documento. In questo modo si evitano all'utente i problemi descritti nel paragrafo 3.2.1, e si possono gestire facilmente anche progetti suddivisi in più file come una tesi o un libro. Per maggiori dettagli, si veda [Beccari e Gordini, 2016].

2.1.2 T_EX Live

L^AT_EX è uno, ma prende forma in differenti versioni (non così tante) che si chiamano *distribuzioni*. Una distribuzione è una raccolta di file e altro software (il programma L^AT_EX vero e proprio, uno o più editor dedicati e di solito altri programmi accessori) autosufficiente per produrre un documento finito. Si può installare direttamente da Internet, dal disco rigido dopo averla scaricata oppure da DVD.

Il luogo di riferimento nel Web da cui scaricare il materiale ufficiale su L^AT_EX è CTAN (*Comprehensive T_EX Archive Network*, “rete di archivi completi per T_EX”, <http://www.ctan.org/>), una rete di server dislocati in tutto il mondo, uguali tra loro e ciascuno contenente una copia integrale del sito originale (si chiamano anche *mirror*, “specchio”): ci si può così servire dal mirror più vicino, evitando di sovraccaricare la Rete e abbreviando i tempi dell'operazione.

In questa guida si consiglia di installare T_EX Live (<http://www.tug.org/texlive/>): è una distribuzione affidabile, mantenuta da decine di sviluppatori e aggiornata annualmente. Gli utenti di Windows possono contare sull'alternativa di MiK_T_EX, che gira solo su quel sistema operativo. Qualunque delle due si scelga, si abbia cura di farne un'installazione *completa*: in caso contrario, potrebbero presentarsi alcuni fastidiosi problemi (si veda la sezione 3.2.1).

Una distribuzione si modifica nel tempo, perché quasi ogni giorno molti dei *pacchetti* che la compongono (si veda il paragrafo 3.12) vengono perfezionati, altri di nuovi vengono aggiunti agli archivi in Rete e altri ancora rimossi. Quelli già presenti riescono a soddisfare praticamente tutte le esigenze di scrittura, ma per mantenere la propria distribuzione sempre efficiente bisogna *aggiornarla* alle ultime versioni dei pacchetti e *installare* quelli nuovi continuamente creati e messi a disposizione.

Nei paragrafi seguenti si spiegherà:

- come installare T_EX Live sui diversi sistemi operativi;
- come usare il programma `tlmgr` (T_EX Live ManaGeR) per aggiornarla e aggiungervi automaticamente i nuovi pacchetti; si raccomanda di farlo senz'altro subito dopo aver installato la distribuzione e poi con una certa regolarità (una volta alla settimana potrebbe andare bene).

2.2 Windows

Installare

La procedura descritta di seguito permette di installare T_EX Live sul proprio calcolatore.

1. Si scarichi il file `texlive2016.iso` da <http://tug.org/texlive/Images/> e lo si decomprima.
2. Si apra la cartella risultante, si esegua *come amministratore* il file `install-tl` al suo interno e si attenda finché non compare l'omonima finestra.
3. Si seguano le istruzioni sullo schermo senza cambiare nulla e si attenda la fine dell'operazione.

Aggiornare

L'interfaccia grafica di `tlmgr` su Windows è T_EX Live Manager.

1. Si avvii il programma (Start → Programmi → TeX Live 2016 → TeX Live Manager).
2. Si prema il pulsante **Aggiorna installati** e si attenda la fine dell'aggiornamento.

2.3 Mac

Installare

Su Mac si raccomanda di installare T_EX Live tramite MacT_EX (<http://www.tug.org/mactex/>), un installer che provvede a tutto il necessario.

1. Si scarichi il file `MacTeX.pkg` da <http://www.tug.org/mactex/>.
2. Lo si esegua e se ne seguano le istruzioni fino a completare l'installazione.

Aggiornare

L'interfaccia grafica di `tlmgr` su Mac è T_EX Live Utility.

1. Si avvii il programma (Applicazioni → TeX → TeX Live Utility).
2. Si scelga la voce **Update All Packages** dal menù **Actions** e si attenda la fine dell'aggiornamento.

2.4 Linux

A causa di alcuni limiti imposti dagli sviluppatori, non si può installare e aggiornare T_EX Live sulle distribuzioni Linux con la semplicità degli altri sistemi operativi. Le istruzioni descritte nella guida di [Gregorio, 2013] però, permettono di farlo con estrema tranquillità. La procedura lì descritta, cui si rimanda, funziona su una distribuzione di Ubuntu correttamente installata e, con qualche modifica, anche su Fedora (e simili) e OpenSuSe (e simili).

Capitolo 3

Basi

3.1 Per cominciare

In questo paragrafo si mostrano le semplici fasi per realizzare un documento con \LaTeX , dalla scrittura del sorgente alla stampa.

Scrivere il codice

Si crei una cartella nella quale mettere *tutti* i file del documento. Dopo di che, con l’editor scelto si apra un nuovo file e si scriva il seguente codice (tutte le istruzioni verranno spiegate in questo e nel successivo capitolo):

```
\documentclass[a4paper]{article}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[italian]{babel}

\begin{document}
Ecco il mio primo documento con \LaTeX.
\end{document}
```

Infine si registri il file come `primo.tex` (`.tex` è l’estensione dei sorgenti \LaTeX).

Comporre

Si componga il codice premendo l’apposito pulsante. Via via che \LaTeX elabora il sorgente, ne rendiconta dettagliatamente la composizione in un file apposito con estensione `.log` (è il *log*, in gergo) conservato in una cartella del sistema, e contemporaneamente ne mostra sullo schermo una versione ridotta, più o meno come la seguente (per brevità se ne sono omesse alcune parti:

```
This is pdfTeX, Version 3.14159265-2.6-1.40.17 (TeX Live 2016)
(./primo.tex
LaTeX2e <2016/03/31> patch level 3
Babel <3.9r> and hyphenation patterns for 83 language(s) loaded
(/usr/local/texlive/2016/texmf-dist/tex/latex/base/article.cls
Document Class: article 2014/09/29 v1.4h Standard LaTeX document class
(/usr/local/texlive/2016/texmf-dist/tex/latex/base/fontenc.sty
(/usr/local/texlive/2016/texmf-dist/tex/latex/base/tlenc.def))
(/usr/local/texlive/2016/texmf-dist/tex/latex/base/inputenc.sty
(/usr/local/texlive/2016/texmf-dist/tex/latex/base/utf8.def
(/usr/local/texlive/2016/texmf-dist/tex/latex/base/tlenc.dfu)))
(/usr/local/texlive/2016/texmf-dist/tex/generic/babel/babel.sty
```

```
(/usr/local/texlive/2016/texmf-dist/tex/generic/babel/italian.ldf))
Output written on primo.pdf (1 page, 15596 bytes).
SyncTeX written on primo.synctex.gz.
Transcript written on primo.log.
```

Si noti che premere il pulsante di composizione dell'editor equivale a eseguire dalla riga di comando l'istruzione

```
pdflatex primo.tex
```

L'interfaccia a riga di comando (che su Windows si chiama *Prompt dei comandi*, su Mac *Terminale* e su Linux *Terminale* o *Console*) si avvia con modalità proprie di ogni sistema operativo.

In questo modo, si è consapevoli del fatto che è \LaTeX (e *non* l'editor) il programma che elabora il codice `.tex`, che emette i messaggi e che produce il PDF, lavorando "dietro le quinte". La composizione non è dunque semplicemente "il tempo che bisogna attendere per vedere il PDF", ma il processo durante il quale il programma comprende le intenzioni dell'utente (esprese con i comandi), e le trasforma in un file (tipo)grafico.

Errori e warning

Per quanto scrupolosi si possa essere, di tanto in tanto qualche errore s'infiltra nel sorgente. Quando \LaTeX s'imbatte in qualcosa che non capisce o che non può fare, arresta la composizione in corso e mostra quello che non va in un messaggio, che può essere di due tipi:

- un messaggio di vero e proprio *errore* (nomi di comandi scritti scorrettamente, parentesi dimenticate, caratteri speciali usati per sbaglio, per esempio): la composizione si arresta e il programma rimane in attesa di istruzioni da parte dell'utente;
- un messaggio di *avviso* (o *warning*, in gergo), meno grave del primo (righe troppo lunghe o troppo corte, pagine riempite troppo o troppo poco, riferimenti incrociati irrisolti, per esempio): il programma semplicemente lo notifica ma porta a termine la composizione.

Gli errori vanno corretti *obbligatoriamente*. In caso contrario, se sono pochi e non gravi si potrà comunque completare la composizione, anche se in genere con un risultato diverso da quello atteso; se invece sono gravi (ne basta anche uno solo), la composizione potrebbe bloccarsi del tutto. Ai warning, semplici commenti su questioni di secondaria importanza, ci si potrà dedicare a lavoro ultimato.

Un esempio chiarirà le idee. Il nome di un comando scritto scorrettamente provoca l'arresto della composizione e la comparsa di un messaggio di questa forma (uguale per tutti i messaggi d'errore di questo tipo):

```
! Undefined control sequence.
1.6 Ecco il mio primo documento con \latex
?
```

Quando trova un errore, il programma segnala all'utente:

- la natura del messaggio (un errore, in questo caso) e il suo emittente (\TeX , in questo caso: un errore emesso da \LaTeX comincia sempre con `! LaTeX Error:`),
- la natura dell'errore (in questo caso, `Undefined control sequence`, "sequenza di controllo non definita", ovvero "comando sconosciuto");

- la riga (*line*) esatta del codice sorgente in cui si trova l'errore (1.6);
- il testo contenuto nella riga appena segnalata (o parte di esso), alla fine del quale si trova l'errore in questione: se il testo continuasse, la riga incriminata verrebbe spezzata in corrispondenza del punto problematico e continuata subito sotto.

Come comportarsi? A essere precisi, un errore come quello appena mostrato può avere due cause: o è sbagliato il nome del comando (quello giusto è `\LaTeX` e non `\latex`: attenzione alle maiuscole!); oppure, sebbene scritto correttamente, si sta usando un comando definito da un pacchetto non caricato. Essendo giusta la prima ipotesi, basta correggere il codice, registrare il file e ricomporre (oppure, con alcuni editor, ricomporre direttamente).

A fronte di errori ricorrenti e apparentemente inspiegabili, infine, un metodo spesso efficace è quello di eliminare i file ausiliari generati dalla composizione e ricomporre il documento.

Visualizzare e stampare

Se si usa un editor dedicato e il sorgente non contiene (più) errori, la composizione andrà a buon fine e il visualizzatore di PDF si attiverà automaticamente mostrando il documento finito, che si potrà poi stampare con le modalità proprie del programma.

3.2 Codifiche e lingue

3.2.1 Le codifiche di \LaTeX

\LaTeX è un programma nato per scrivere documenti *in inglese* ad alto contenuto matematico, e a questo scopo è stato originariamente corredato di una dotazione minima di caratteri (lettere, numeri e pochi altri segni) del tutto sufficiente, tant'è vero che il codice

```
\documentclass{<...>}
\begin{document}
...
\end{document}
```

permette di scrivere senza problemi un documento in quella lingua. La corretta scrittura di parole straniere, tuttavia, potrebbe richiedere di caricare i pacchetti descritti in questo paragrafo anche in un documento completamente in inglese.

Com'è noto, l'inglese si scrive senza accenti o caratteri particolari, contrariamente alla vasta maggioranza delle altre lingue che usano l'alfabeto latino. Gli unici caratteri "complicati" dell'italiano sono le vocali accentate, presenti (non proprio tutte) anche su una tastiera italiana. Affinché \LaTeX le "accetti" se inserite nell'editor con i tasti appositi, restituendole adeguatamente nel documento composto e sillabando correttamente le parole che le contengono, *immediatamente dopo* la dichiarazione di classe vanno caricati altri due pacchetti nell'ordine seguente:

```
\documentclass[<...>]{<...>}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
```

dove:

- `fontenc` (*font encoding*, "codifica dei font") si occupa dei font che si vedranno nel documento composto e fornisce a \LaTeX i caratteri particolari di una certa lingua già disegnati. Si noti che per funzionare al meglio il pacchetto richiede una distribuzione *completa* di \LaTeX , altrimenti i caratteri potrebbero apparire "sgranati" sullo schermo (ma non a stampa).

- T1 è la codifica dei font usati per scrivere in italiano e in molte altre lingue occidentali: per lingue o alfabeti particolari si usano altre codifiche. In documento multilingue l'*ultima* codifica dichiarata è quella della lingua principale: [T2A,T1] per un documento in italiano e russo, per esempio.
- inputenc (*input encoding*, “codifica di input”) serve a L^AT_EX per interpretare correttamente i caratteri immessi nell’editor.
- utf8 è la *codifica di input*, in gergo, che permette di scrivere nell’editor i segni di numerosi alfabeti direttamente dalla tastiera, evitando di dover caricare ogni volta la codifica adatta alla lingua del documento. Se il proprio editor non supporta *pienamente* utf8, si usi latin1.

Non ci sono alternative: senza i due pacchetti appena descritti le vocali accentate non vengono visualizzate nel documento composto. Caricandoli, inoltre, il vecchio metodo di comporre *a mano* i caratteri con l’accento e fonte di non pochi problemi di sillabazione nelle parole che li contenevano, mostrato nell’esempio seguente,

```
Basta! Non se ne pu\‘o pi\‘u!
Perch\‘e non c\‘e piet\‘a
per chi deve scrivere cos\‘i?
```

Basta! Non se ne può più! Perché non c’è
pietà per chi deve scrivere così?

non serve più, e il codice è decisamente più pulito:

```
Però: caricando il pacchetto
giusto la vita diventerà così
facile che c’è molto più gusto!
```

Però: caricando il pacchetto giusto la vita
diventerà così facile che c’è molto più gusto!

3.2.2 Problemi con le codifiche

Un ulteriore aspetto da tenere bene a mente, spesso fonte di grattacapi, è il rapporto tra la codifica con cui è scritto un sorgente e quella con cui è impostato l’editor in uso, che *devono* coincidere, altrimenti aprendo un file sconosciuto si potrebbero vedere caratteri bizzarri sullo schermo. Se ciò si verificasse, *non* si componga né si registri il file, ma lo si chiuda e si seguano le procedure descritte in [Beccari e Gordini, 2016], a cui si rimanda per individuare e risolvere i principali problemi in questo senso. Comporre il file, infatti, potrebbe danneggiarlo più o meno seriamente. In ogni caso, si legga la documentazione del proprio editor: potrebbe rivelarsi molto utile per evitare di questi pasticci.

3.2.3 L^AT_EX e le lingue

Il terzo e ultimo pacchetto da caricare *sempre* è babel, che agisce su parole fisse (cioè le voci generate automaticamente dai comandi raccolti nella tabella 9.1), date, convenzioni tipografiche e sulla scelta delle regole di sillabazione, che per italiano e principali lingue europee sono già comprese in T_EX Live. Come opzioni prende una o più lingue e per un documento in italiano lo si carica nel modo seguente:

```
\usepackage[⟨...⟩,italian]{babel}
```

dove l’*ultima* dichiarata è la lingua principale del documento. Inoltre, per ciascuna lingua definisce nuovi comandi per semplificare l’immissione dei caratteri particolari nazionali, come si può vedere nella documentazione del pacchetto o in [Gregorio, 2010].

Si dà, infine, il tipico inizio di un sorgente per un documento in italiano con la corretta sequenza dei pacchetti da caricare:

Tabella 3.1: Principali file ausiliari di \LaTeX

Prodotti da	Estensione	Descrizione
Utente	.jpg, .pdf, .png .tex	Formati grafici per \LaTeX File sorgente
Classi, pacchetti e stili	.cls .sty	Classe di documento Pacchetto
Composizione	.aux .lof .log .lot .toc	Trasporta informazioni generiche Indice delle figure Rendiconta l'ultima composizione Indice delle tabelle Indice generale
Pacchetti e programmi	.idx .ind .out	Voci dell'indice analitico Prodotto di MakeIndex Segnalibri ipertestuali
Output	.pdf	Prodotto di $\text{PDF}\text{\LaTeX}$

```
\documentclass[...]{...}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[italian]{babel}
```

Il pacchetto babel definisce alcuni comandi molto utili per trattare correttamente ciascuna lingua in un documento multilingue. Supponendo di dover scrivere un documento in italiano con alcune parti in inglese, babel andrà caricato così:

```
\usepackage[english,italian]{babel}
```

Per singole parole o brevi frasi in lingua straniera è disponibile il comando

```
\foreignlanguage{<lingua>}{<testo>}
```

Per porzioni di testo in lingua più consistenti è disponibile l'ambiente

```
\begin{otherlanguage*}{<lingua>}
...
\end{otherlanguage*}
```

che però non traduce parole fisse e date. Se invece se ne avesse bisogno, si usi l'analogo ambiente otherlanguage.

3.3 File con cui si ha a che fare

Guardando nella cartella di lavoro dopo la prima composizione, si noterà che oltre al file .tex iniziale e agli eventuali file esterni come le immagini ce ne sono altri: sono i *file ausiliari* prodotti dalla composizione, e questo è un altro aspetto per cui \LaTeX è molto diverso dai programmi tradizionali. Il loro numero dipende dalla complessità del documento, ma è importante sapere che *non* vanno toccati: \LaTeX li crea e se ne serve automaticamente.

La tabella 3.1 ne raccoglie e descrive i principali.

3.4 File sorgente

Un sorgente di \LaTeX è un file di puro testo che contiene sia il testo vero e proprio del documento sia i comandi che istruiscono \LaTeX su come trattarlo.

3.4.1 Comandi e ambienti

Un comune programma di videoscrittura a composizione sincrona e \LaTeX presentano per certi versi un funzionamento simile: entrambi ricevono dall'utente sia il testo sia le istruzioni per impostarne l'aspetto. La differenza sostanziale, però, è che il primo le nasconde, proponendo all'utente menù da cui scegliere comandi "preconfezionati" che mostrano immediatamente i propri effetti; \LaTeX , invece, richiedendo di scrivere esplicitamente i comandi, mantiene queste istruzioni in superficie nei modi spiegati di seguito.

Comandi e dichiarazioni

Un *comando* è un'istruzione che ordina a \LaTeX di trattare in un certo modo una porzione più o meno ampia di testo. Si possono classificare i comandi di \LaTeX in base a forma e funzione.

Per quanto riguarda la *forma* si distinguono tre tipi di comando, a seconda che siano costituiti da:

- Un solo carattere non alfabetico. Questi comandi sono quattro in tutto: spazio `\`, tilde `~`, circonflesso `^` e trattino basso `_`.
- Una barra rovescia `\` seguita da un solo carattere non alfabetico (cioè non compreso fra A-Z o a-z). Un comando di questo tipo termina al primo carattere non alfabetico e uno o più spazi (che contano per un solo spazio) immediatamente dopo *non* vengono ignorati. Alcune tra le combinazioni più usate sono: `\{`, `\}`, `\%`, `\$`, `_`, `\&`, `\#`, `\~`.
- Una barra rovescia `\` seguita da una sequenza di caratteri alfabetici. Un comando di questo tipo termina al primo carattere non alfabetico e uno o più spazi immediatamente dopo vengono ignorati. Si noti che questi comandi distinguono maiuscole e minuscole. Alcuni esempi: `\LaTeX`, `\emph`, `\documentclass`.

I comandi del terzo tipo che "producono testo" (come `\LaTeX`, `\TeX`, `\Ars`, `\dots`, `\today` e pochissimi altri) richiedono di essere "terminati", pena una spaziatura errata dopo di sé. La cosa migliore è usare la barra rovescia o un gruppo vuoto `\{ \}`, come mostrano gli esempi seguenti:

```
\Ars è la rivista del Gruppo
Utilizzatori Italiani di
\TeX e \LaTeX. \[1ex]
\Ars\ è la rivista del Gruppo
Utilizzatori Italiani di
\TeX{} e \LaTeX.
```

```
ArsTeXnicaè la rivista del Gruppo Utilizzato-
ri Italiani di TExe L\ATeX.
```

```
ArsTeXnica è la rivista del Gruppo Utilizzato-
ri Italiani di TEx e L\ATeX.
```

La scrittura *sempre* corretta di questi comandi è una di quelle contenute nel secondo esempio. Si noti che un qualunque segno di punteggiatura immediatamente dopo il comando elimina la necessità dello spazio esplicito.

Per quanto riguarda la *funzione* si distinguono due tipi di comando, a seconda della porzione di testo su cui hanno effetto:

Tabella 3.2: Caratteri speciali di \LaTeX

Carattere	Funzione	Codice
<code>\</code>	Comincia un comando	<code>\textbackslash</code>
<code>{ }</code>	Delimitano un gruppo	<code>\{ \}</code>
<code>\$</code>	Delimita la matematica in linea	<code>\\$</code>
<code>^</code>	Esponente matematico	<code>\^{}</code>
<code>_</code>	Pedice matematico	<code>_</code>
<code>&</code>	Separa le celle in una tabella	<code>\&</code>
<code>#</code>	Numero dell'argomento	<code>\#</code>
<code>~</code>	Spazio indivisibile	<code>\~{}</code>
<code>%</code>	Commento	<code>\%</code>

- Comandi come `\textit{<testo>}` ordinano a \LaTeX di trattare in un certo modo solo il `<testo>` scritto tra le parentesi graffe.
- Comandi come `\itshape`, detti *dichiarazioni*, ordinano a \LaTeX di trattare in un certo modo *tutto* il testo successivo al punto in cui vengono dati.

In altre parole, una dichiarazione è un comando che imposta uno o più aspetti generali della composizione, e può essere data:

- nel preambolo, e allora ha effetto sull'intero documento e si annulla soltanto con un'altra dichiarazione;
- nel corpo del documento, e allora va data in un *gruppo* (cioè una porzione di testo racchiusa di solito da parentesi graffe o comandi di inizio e fine ambiente).

Sono esempi di dichiarazione: `\small`, `\linespread`, `\appendix`.

Nell'esempio seguente si vedono all'opera alcuni comandi visti fin qui:

Data odierna: <code>\today</code> . <code>\</code>	Data odierna: 23 gennaio 2017.
Sarò lì in <code>\emph{dieci}</code> minuti. <code>\</code>	Sarò lì in <i>dieci</i> minuti.
Questo <code>\itshape</code> testo è in corsivo.	Questo <i>testo</i> è in corsivo.

Si noti che il comando `\today` produce la data in cui si compone il documento secondo le convenzioni della lingua in uso.

La giustapposizione degli elementi di un comando prende il nome di *sintassi* del comando. Ciò che va tra parentesi graffe si chiama *argomento obbligatorio*, mentre ciò che va tra parentesi quadre si chiama *argomento facoltativo*. Se gli elementi da scrivere nello stesso gruppo di parentesi sono più d'uno, vanno separati con una virgola *senza ulteriori spazi*.

Ambienti

Un *ambiente* è una porzione di codice delimitata da un comando d'apertura e uno di chiusura, che \LaTeX tratta in un certo modo. La sintassi di un ambiente generico è:

```
\begin{<ambiente>}[<...>]{<...>}
...
\end{<ambiente>}
```

dove:

- `<ambiente>` è il nome dell'ambiente;

Tabella 3.3: Scorciatoie da tastiera (italiana) per alcuni caratteri frequenti

Carattere	Windows	Mac	Linux
‘	Alt + 96	⌥ 9	Alt Gr + ’
{	Alt + 123	⌥ ⇧ [Alt Gr + 7
	Alt Gr + Maiusc + [Alt Gr + Maiusc + [
}	Alt + 125	⌥ ⇧]	Alt Gr + 8
	Alt Gr + Maiusc +]		Alt Gr + Maiusc +]
~	Alt + 126	⌥ 5	Alt Gr + ì

- se presenti, argomenti facoltativi e obbligatori si scrivono dopo il solo comando d’apertura;
- l’ambiente va separato dal resto del testo con una riga bianca prima e dopo se ciò che contiene *non* appartiene al flusso del discorso (una figura, per esempio); non va separato in caso contrario.

L^AT_EX permette di annidare gli ambienti, purché l’ordine di chiamata venga rispettato:

```
\begin{ambiente}
...
\begin{Ambiente}
...
\end{Ambiente}
...
\end{ambiente}
```

3.4.2 Caratteri speciali

L^AT_EX interpreta in modo particolare alcuni caratteri molto richiesti nella scrittura del codice: sono i cosiddetti *caratteri speciali*, così chiamati perché non possono essere stampati normalmente se non come mostra la tabella 3.2. La loro alta frequenza, però, si scontra con il fatto che su nessuna tastiera, a parte quella inglese internazionale, ci sono tutti, e perciò la loro scrittura richiede combinazioni di tasti o codici numerici particolari.

Si noti che:

- i caratteri { ‘ ~ } mancano sulla tastiera italiana: la tabella 3.3 indica le scorciatoie da prendere in questi casi (su Windows, il codice relativo va digitato *sul tastierino numerico*);
- si distingue con attenzione ‘ (virgoletta alta aperta, accento grave) da ’ (apostrofo, virgoletta alta chiusa, accento acuto);
- il comando `\textbackslash` *non* sostituisce la sequenza `\\`, come potrebbe sembrare: sono infatti due comandi distinti con distinte funzionalità (verranno considerati nei prossimi capitoli).

3.4.3 Struttura del file sorgente

L^AT_EX si aspetta di trovare il sorgente da elaborare strutturato in un certo modo. Elementi fondamentali sono almeno una dichiarazione di classe

```
\documentclass{<...>}
```

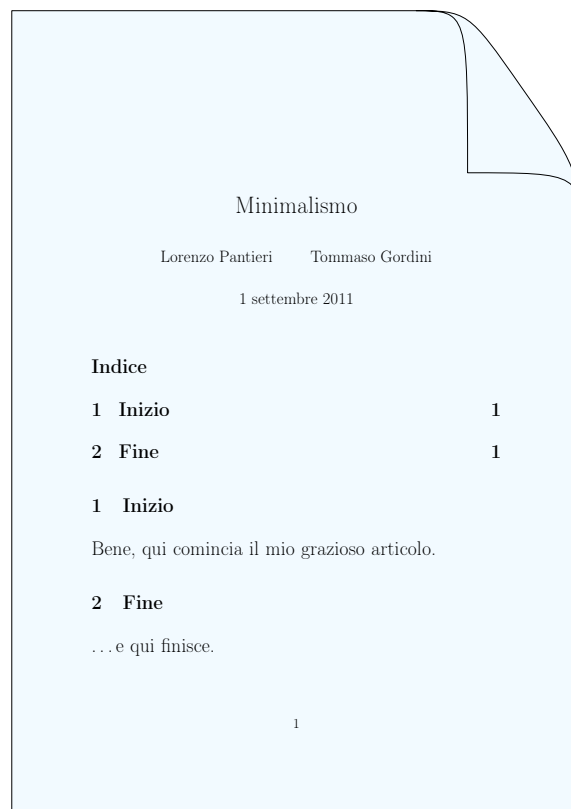


Figura 3.1: Documento elementare composto con \LaTeX

e le dichiarazioni d’inizio e fine documento:

```
\begin{document}
...
\end{document}
```

Tutte le istruzioni scritte tra `\documentclass` e `\begin{document}` *inclusi* costituiscono il *preambolo del documento* (o semplicemente *preambolo*) e comprendono:

- il caricamento di pacchetti che estendono le capacità di \LaTeX ;
- le definizioni di comandi e ambienti personalizzati, che si consiglia di organizzare come indicato nel paragrafo 3.14;
- le opzioni generali del documento.

Si noti che un sorgente \LaTeX richiede *un solo* preambolo.

Fra `\begin{document}` e `\end{document}` va scritto il *corpo del documento*, cioè il vero e proprio testo che \LaTeX elaborerà e mostrerà nel documento finito. La figura 3.1 mostra il risultato della composizione del codice seguente (si possono intercalare o meno righe bianche per evidenziarne la struttura e facilitarne l’individuazione delle parti):

```
\documentclass[a4paper]{article}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[italian]{babel}

\begin{document}

\author{Lorenzo Pantieri \and Tommaso Gordini}
```

```

\title{Minimalismo}
\maketitle

\tableofcontents

\section{Inizio}
Bene, qui comincia il nostro grazioso articolo\dots

\section{Fine}
\dots e qui finisce.

\end{document}

```

Dove:

- `\begin{document}` segnala l’inizio del documento;
- `\author` e `\title` (che si possono dare anche prima del comando precedente) ne specificano rispettivamente nome dell’autore e titolo;
- `\and` si spiega da sé;
- `\maketitle` produce il contenuto dei due comandi precedenti, *dopo* i quali deve essere dato;
- `\tableofcontents` produce l’indice generale dopo *due* composizioni;
- `\section{<titolo>}` produce un titolo di sezione (un paragrafo, in questo caso);
- `\dots` produce i puntini di sospensione ... ;
- `\end{document}` segnala la fine del documento.

L^AT_EX ignora tutto ciò che si trova dopo `\end{document}`: questo spazio, perciò, potrebbe essere un buon posto per appuntare un promemoria sul documento in lavorazione.

3.4.4 Spazi e righe vuote

Il modo in cui L^AT_EX tratta spazi, tabulazioni e righe vuote nel sorgente è particolarissimo, e decisamente diverso da quello di tutti i comuni elaboratori di testo. Infatti:

- una tabulazione è considerata come uno spazio;
- più spazi consecutivi sono considerati come un solo spazio;
- spazi o tabulazioni all’inizio di una riga vengono ignorati;
- una sola interruzione di riga è trattata come uno spazio;
- una riga vuota tra due righe di testo separa due capoversi;
- più righe vuote consecutive sono trattate come una sola riga vuota.

L’esempio seguente mostra all’opera i casi appena descritti:

Non ha alcuna importanza se si mettono uno o tanti spazi dopo una parola.

E neppure se si mettono tanti spazi all’inizio di una riga o se la s’interrompe.

Le cose cambiano se si saltano una o più righe, perché in questo modo si comincia un nuovo capoverso.

Non ha alcuna importanza se si mettono uno o tanti spazi dopo una parola. E neppure se si mettono tanti spazi all’inizio di una riga o se la s’interrompe.

Le cose cambiano se si saltano una o più righe, perché in questo modo si comincia un nuovo capoverso.

3.4.5 Commenti

Quando \LaTeX incontra un carattere di percento % (tranne che nella forma \backslash\%) elaborando un sorgente, ignora il resto della riga, l’interruzione di riga, e tutti gli spazi bianchi all’inizio della riga successiva. Questo carattere è utile, dunque, per appendere una nota o un promemoria (che non verrà stampato) proprio lì dove serve.

Talvolta però va usato per spezzare parole troppo lunghe o per dividere righe in cui non sono permessi spazi bianchi o interruzioni (come nel caso di comandi troppo lunghi per stare in una sola riga dell’editor). Eccolo all’opera:

```
Ecco un % semplice,
% ma istruttivo
esempio: Supercal%
          ifragilist%
          ichespiralidoso.
```

Ecco un esempio: Supercalifragilistichespíralidoso.

3.4.6 Sorgenti ordinati

Spesso gli utenti di \LaTeX sottovalutano l’importanza di un sorgente “pulito” (con rientri, incolonnamenti, eccetera) e commentato. *Non* è indispensabile, ma si consiglia di farlo ugualmente: l’ordine ne facilita la gestione, specie se a uno stesso progetto lavorano più persone, e rende più facile individuare eventuali errori.

Durante la stesura si raccomanda di usare i commenti, di suddividere con chiarezza il documento e di aiutarsi eventualmente con rientri, incolonnamenti, a capo e righe vuote supplementari. Ulteriori consigli in questo senso verranno forniti nelle prossime pagine al momento opportuno.

Si realizzano ora le indicazioni precedenti in un sorgente “ben scritto”:

Esempio di articolo composto con \LaTeX

```
% Un articolo scritto con LaTeX
\documentclass[a4paper,11pt]{article}
\usepackage[T1]{fontenc}           % codifica dei font
\usepackage[utf8]{inputenc}       % lettere accentate da tastiera
\usepackage[italian]{babel}       % lingua del documento
\usepackage{lipsum}               % genera testo fittizio
\usepackage{url}                  % per scrivere gli indirizzi Internet

\begin{document}

\author{Lorenzo Pantieri \and Tommaso Gordini}
\title{Il titolo}
```

```

\maketitle

\begin{abstract}
\lipsum[1]
\end{abstract}

\tableofcontents

\section{Un paragrafo}
\lipsum[1]

\subsection{Un sottoparagrafo}
\lipsum[1]

\section{Un paragrafo}
\label{sec:esempio}
\lipsum[1]

% Bibliografia
\begin{thebibliography}{9}
\bibitem{pantieri:arte}
Pantieri, Lorenzo e Tommaso Gordini (2017),
\emph{L'arte di scrivere con \LaTeX},
\url{http://www.lorenzopantieri.net/LaTeX_files/ArteLaTeX.pdf}.
\end{thebibliography}

\end{document}

```

3.5 Classi di documento

La prima informazione che \LaTeX si aspetta di trovare nel sorgente è il *tipo di documento* che si desidera realizzare (la *classe*, in gergo), da specificare di norma come prima cosa con il comando

```
\documentclass[<opzioni>]{<classe>}
```

dove:

- *<opzioni>* sono le impostazioni generali del documento;
- *<classe>* è la classe di documento scelta.

Di seguito si elencano le principali classi di documento *standard* (cioè definite dal programma):

`article` per scrivere articoli;

`report` per scrivere relazioni o tesi suddivise in capitoli;

`book` per scrivere libri;

`letter` per scrivere lettere.

Esistono numerose altre classi non standard per i documenti più diversi. Tra quelle più diffuse ci sono memoir (che permette molta libertà nel personalizzare il documento), toptesi e suftesi (per tesi di laurea e dottorato), beamer (per presentazioni).

Tabella 3.4: Opzioni più comuni delle classi standard di \LaTeX . I simboli ●, ◐ e ○ indicano rispettivamente che l'opzione è predefinita, applicabile (anche se non predefinita), non applicabile.

Opzione	book	report	article
10pt	●	●	●
letterpaper	●	●	●
oneside	◐	●	●
twoside	●	◐	◐
openany	◐	●	○
openright	●	◐	○
titlepage	●	●	◐
final	●	●	●

Le *opzioni* date a `\documentclass` si dicono anche *globali*, perché agiscono sull'intero documento. Di seguito si descrivono quelle più comuni per le tre classi `article`, `report` e `book` (la classe `letter` meriterebbe una trattazione a parte, perciò non viene considerata). Si noti che alcune di esse hanno un'applicabilità limitata, come mostra la tabella 3.4.

`10pt`, `11pt`, `12pt` Impostano la dimensione del font principale del documento. Omettendo l'opzione, il valore predefinito è `10pt`.

`a4paper`, `a5paper`, ... Definiscono le dimensioni del foglio, che per impostazione predefinita è nel formato `letterpaper`. Le altre opzioni possibili sono `executivepaper`, `legalpaper` e `b5paper`.

`oneside`, `twoside` Specificano se verrà composto un documento a singola o doppia facciata rispettivamente. Per impostazione predefinita, le classi `article` e `report` sono a singola facciata e la classe `book` è a doppia facciata.

`openany`, `openright` L'opzione `openany`, predefinita nella classe `report`, fa cominciare un capitolo nella successiva pagina a disposizione; l'opzione `openright`, predefinita nella classe `book`, lo fa cominciare sempre in una pagina destra. Entrambe non sono disponibili nella classe `article`, che non ammette la suddivisione in capitoli.

`twocolumn` Dà a \LaTeX le istruzioni per comporre l'intero documento su due colonne (si veda anche il paragrafo 9.2).

`titlepage`, `notitlepage` Specificano se dopo il titolo del documento debba avere inizio una nuova pagina (come accade con `report` e `book`) o no (come accade con `article`) rispettivamente.

`fleqn` Allinea le formule a sinistra rispetto a un margine rientrato.

`leqno` Mette la numerazione delle formule a sinistra anziché a destra.

`draft`, `final` L'opzione `draft` evidenzia le righe composte in modo non ottimale con un rettangolino nero ■ accanto, facilitandone l'individuazione sulla pagina. Ciò non accade con `final`. Si noti che entrambe influenzano il comportamento degli altri pacchetti caricati o addirittura li disabilitano del tutto (si veda il paragrafo 10.1 per risolvere il problema).

Un tipico sorgente, allora, potrebbe cominciare con la riga

```
\documentclass[a4paper,11pt,twoside]{article}
```

che ordina a \LaTeX di impaginare il documento come un articolo, su carta di formato A4, con un carattere di 11 punti e impostato per la stampa fronte/retro.

3.6 Gestire la pagina

3.6.1 Il tormentone dei margini

I margini della pagina in tipografia rivestono funzioni importantissime, prima fra tutte quella di delimitare in modo chiaro il testo. Il lettore così potrà individuarlo agevolmente sulla pagina e appoggiare i pollici su uno spazio sufficientemente confortevole per maneggiare comodamente il documento. È perciò che nei documenti impostati per la stampa in fronte/retro i margini esterni sono più ampi di quelli interni, che appaiono duplicati perché adiacenti.

La maggior parte degli utenti europei, che stampa su carta in formato A4, ritiene troppo ampi i margini predefiniti da \LaTeX nelle classi standard, e che di conseguenza la pagina non sia sufficientemente riempita. Prima di buttarsi nella frenesia dell' "allarghiamo un po' questa strettissima pagina", però, è doveroso riflettere.

I margini di \LaTeX derivano da convenzioni tipografiche ampiamente verificate e accettate, e mettono l'utente nelle vantaggiose condizioni di potersene servire per ottenere risultati professionali *già alla prima composizione* e senza doverci mettere le mani. Modificarli, perciò, significherebbe dover studiare un (bel) po' di tipografia prima di raggiungere risultati accettabili.

L'esperienza dimostra che leggere diventa tanto più difficile quanto più numerosi sono i caratteri in una singola riga di testo: l'occhio è costretto a compiere movimenti più ampi e si affatica presto (perciò quotidiani e riviste sono stampati su più colonne). Robert Bringhurst ha codificato quest'esperienza nella sua celebre "regola", che considera ottimale il numero di circa 66 caratteri per riga (spazi inclusi), indipendentemente dal font usato. Se si considera che per riempire meglio la pagina \LaTeX usa già *in partenza* una riga più lunga del limite stabilito da Bringhurst, è chiaro che modifiche in questo senso vanno evitate il più possibile.

In alcune circostanze, tuttavia, può essere desiderabile o necessaria una maggiore copertura della pagina: tra i numerosi pacchetti scritti a questo scopo se ne consigliano due.

Il pacchetto LayAureo (se ne veda la documentazione in italiano) definisce un layout di pagina pronto per l'uso, permettendo di impostare facilmente anche lo spazio per la rilegatura con la chiave `binding=<dimensione>`. Il pacchetto agisce semplicemente caricandolo e non è personalizzabile: dunque, o piace o non piace.

Se servissero proporzioni di pagina ancora diverse (perché la propria facoltà impone un modello di tesi particolare, per esempio) da applicare anche a una sola pagina, può risolvere il problema il pacchetto `geometry`, completamente configurabile.

Si immagini di dover comporre un documento in formato A4 con margini superiore e inferiore di 3 cm, sinistro e destro (che nella stampa in fronte/retro diventano interno ed esterno) di 3,5 cm e di voler destinare alla rilegatura uno spazio di 5 mm. Il codice da scrivere nel preambolo è il seguente:

```
\usepackage{geometry}
\geometry{a4paper,top=3cm,bottom=3cm,left=3.5cm,right=3.5cm,%
         heightrounded,bindingoffset=5mm}
```

Tra le opzioni del pacchetto che come il precedente, si noti bene, agisce anche solo caricandolo, si consiglia sempre anche `heightrounded`, che modifica ulteriormente di poco le dimensioni della gabbia del testo per farle contenere un numero intero di righe.

Si eviti *assolutamente*, invece, di toccare comandi interni di L^AT_EX come `\textwidth`, `\oddsidemargin`, eccetera, perché la loro azione non tiene in nessun conto le proporzioni di pagina [Fairbairns, 2014].

3.6.2 Interlinea e riempimento della pagina

Interlinea

Non pochi editori, relatori e regolamenti di facoltà impongono di impaginare pubblicazioni e tesi in modo non professionale, a partire dalla questione dell'*interlinea*. L'interlinea standard di L^AT_EX garantisce un risultato tipografico ottimale e non andrebbe modificata senza una ragione precisa. Per farlo si consiglia il pacchetto `setspace`, che modifica lo *scartamento* (o *avanzamento di riga*), ovvero lo spazio fra le righe di base di due righe adiacenti; il pacchetto definisce tre scartamenti *globali* da impostare *nel preambolo* come segue:

- `\singlespacing` (scartamento 1);
- `\onehalfspacing` (scartamento 1,5);
- `\doublespacing` (scartamento 2).

Si può modificare l'interlinea soltanto in alcune parti del documento con gli ambienti `singlespace`, `onehalfspace` e `doublespace`, da usare nel modo consueto.

Se, infine, ne servisse una ancora diversa, si può dare *nel preambolo* il comando standard

```
\linespread{<fattore di scala>}
```

che moltiplica lo scartamento per il *<fattore di scala>*.

3.7 Strutturare il documento

La tabella 3.5 raccoglie le principali istruzioni che producono una sezione nel documento e ne descrive il comportamento nelle classi standard. Non si considera la classe `letter` poiché non prevede alcun tipo di sezionamento.

3.7.1 Sezionare il corpo del testo e modificarne la numerazione

Per suddividere un documento in sezioni basta dare *nel corpo del testo* i comandi elencati nella prima parte della tabella, a proposito dei quali si noti quanto segue.

- I nomi inglesi dei comandi corrispondono alle unità di suddivisione del testo in vigore nei Paesi anglosassoni, e alcuni di essi non hanno corrispondenti italiani.
- Il comando `\section` produce una sezione equivalente al nostro *paragrafo*, mentre `\paragraph` ("capoverso", in inglese) una sezione non numerata non equivalente né al nostro *paragrafo* né al nostro *capoverso* (che ammette al massimo un titoletto o una breve indicazione). Le stesse considerazioni valgono anche per i rimanenti comandi.
- Il comando `\part` non influenza la numerazione dei capitoli.
- Si consiglia di evitare suddivisioni così fini come quelle permesse dagli ultimi tre comandi e di assicurarsi che ogni sezione contenga *almeno due* sezioni di livello immediatamente inferiore: in caso contrario, l'unico comando di sezionamento presente diventa superfluo.

Tabella 3.5: Istruzioni considerate in questa guida che producono una sezione nel documento e loro comportamento nelle classi standard. I simboli indicano che la sezione possiede la caratteristica sempre (●), mai (○), solo nella classe article (◐), solo nelle classi article e report (◑). Tra parentesi i pacchetti richiesti.

A che cosa serve	Istruzione	La sezione prodotta ha				Sezione
		numero	titolo	testatine	posto nell'indice	
Corpo del testo	\part	●	●	○	●	Parte
	\chapter	●	●	●	●	Capitolo
	\section	●	●	●	●	Paragrafo
	\subsection	●	●	○	●	Sottoparagrafo
	\subsubsection	◐	●	○	◐	Sotto-sottoparagrafo
	\paragraph	○	●	○	○	Sezione di livello ancora più basso
Indici	\subparagraph	○	●	○	○	Sezione al più basso livello possibile
	\tableofcontents	○	●	●	○	Indice generale
	\listoffigures	○	●	●	○	Elenco delle figure
	\listoftables	○	●	●	○	Elenco delle tabelle
Bibliografia	\printindex	○	●	●	○	Indice analitico (makeidx)
	\thebibliography	○	●	●	○	Bibliografia manuale
Varie	\printbibliography	○	●	●	○	Bibliografia automatica (biblatex)
	abstract	○	◐	○	○	Sommario

Tabella 3.6: Struttura generale di un libro o una tesi. Le voci in corsivo sono obbligatorie, quelle in tondo sono facoltative, quelle asteriscate *non* devono comparire nell'indice generale.

Supersezionamento	Sezione
Materiale iniziale	<i>Frontespizio</i>
	Colophon*
	Dedica*
	Sommario*
	<i>Indice generale*</i>
	Elenco delle figure*
	Elenco delle tabelle*
	Altri elenchi*
	Prefazione
	Ringraziamenti*
	Introduzione non numerata
Materiale principale	Introduzione numerata
	<i>Capitoli</i>
	Una o più appendici numerate
Materiale finale	Una o più appendici non numerate
	Glossario
	<i>Bibliografia</i>
	Indice analitico

3.7.2 Altri sezionamenti

Le altre istruzioni mostrate nella tabella 3.5 producono le corrispondenti sezioni descritte nei prossimi capitoli. Si noti che l'ambiente `abstract`, destinato a ospitare il sommario (o riassunto) del lavoro, è ammesso solo nelle classi `article` e `report` perché di solito nei libri è sostituito dall'introduzione.

3.7.3 Materiale iniziale, principale e finale

Oltre ai comandi appena illustrati, la sola classe `book` prevede tre dichiarazioni che agiscono al più alto livello possibile e costituiscono una specie di "supersezionamento". Vanno date *sempre* nel corpo del documento e si comportano come segue:

- `\frontmatter` ("materiale iniziale") non numera le sezioni e numera le pagine con numeri romani minuscoli (i, ii, iii, eccetera);
- `\mainmatter` ("materiale principale") numera le sezioni e le pagine con numeri arabi (la numerazione della pagine riprende da 1);
- `\backmatter` ("materiale finale") non numera le sezioni e continua la numerazione araba delle pagine dal materiale principale.

3.7.4 Appendici

Per produrre le appendici basta dare la dichiarazione `\appendix`, che cambia i numeri dei capitoli (o dei paragrafi, se la classe impostata è `article`) in lettere.

3.7.5 Struttura generale di un libro o una tesi

La tabella 3.6 mostra una *possibile* successione dei componenti di una pubblicazione di una certa consistenza, come un libro o una tesi di laurea o dottorato.

Si noti che:

- l'introduzione va senz'altro nel materiale principale se è a propria volta divisa in sezioni, mentre va in quello iniziale se è breve e contiene solo una sintetica esposizione dell'argomento;
- le appendici vanno valutate caso per caso in base a numero e importanza: se è una sola e poco importante può andare nel materiale finale; se invece è funzionale al corpo principale del documento va in quello principale.

Si ricordi, infine, che una sezione concepita come un capitolo non va *mai* messa nel materiale finale.

3.8 Stili di pagina

Lo *stile di pagina* è l'organizzazione del contenuto di testatina e piede scelta per il documento, e va indicato nell'argomento `\style` del comando

```
\pagestyle{\style}
```

L^AT_EX prevede tre stili di pagina predefiniti e uno personalizzabile, descritti di seguito.

plain Mette i numeri di pagina nel piede, lasciando vuota la testatina. È lo stile predefinito nelle classi `article` e `report`.

empty Lascia testatina e piede vuoti.

headings Lascia il piede vuoto e compone le testatine come segue: il numero di pagina è sempre posto nel margine esterno, seguito dal titolo del capitolo corrente nella testatina di sinistra e preceduto dal titolo del paragrafo corrente in quella di destra. È lo stile predefinito nella classe `book` e agisce nello stesso modo nelle classi `report` e `article` impostate con `twoside`, con la differenza che nella seconda le testatine riportano i titoli di paragrafo e sottoparagrafo correnti, rispettivamente. Se invece s'imposta la classe con `oneside`, la testatina riporta soltanto il titolo della suddivisione maggiore.

myheadings È simile a `headings` nel risultato e va usato quando non si vuole che le testatine dipendano dai titoli delle sezioni (capitolo e paragrafo) correnti. L'utente deve specificarne il contenuto a ogni nuovo capitolo (o paragrafo, se la classe è `article`), dando `\markboth` per comporre entrambe oppure `\markright` per comporre soltanto quella di destra.

Si può cambiare lo stile della pagina *corrente* con il comando

```
\thispagestyle{\style}
```

Gestire le testatine

Nelle classi standard le testatine vengono prodotte dai comandi mostrati nella tabella 3.5. Presenza o meno sulla pagina, contenuto e stile si possono regolare facilmente, ma *si raccomanda di non abusare* di queste possibilità e di attenersi alle scelte tipografiche della classe in uso. Si ricorda che le dichiarazioni di sezionamento descritte nel paragrafo 3.7.3

non influenzano il comportamento delle testatine, che saranno dunque sempre presenti anche nelle sezioni non numerate prodotte da `\frontmatter`.

Per eliminare le testatine da una sezione che *l'utente* non ha numerato, invece, basta impostare per quella sezione lo stile di pagina `plain` (si veda il paragrafo 3.8) e ripristinare poi lo stile generale come segue:

```
\chapter*{Prefazione}
\pagestyle{plain}
...

\chapter{Introduzione}
\pagestyle{headings}
...
```

Si noti bene che se ci si dimentica di farlo, la sezione non numerata porterà le testatine prodotte dall'ultimo comando "utile" in questo senso.

Se comunque le si volessero, vanno inserite a mano con `\markboth`

```
\markboth{\MakeUppercase{\langle testatina di sinistra \rangle}}{\MakeUppercase{\langle testatina di destra \rangle}}
```

che si usa come segue:

```
\chapter*{Prefazione}
\markboth{\MakeUppercase{Prefazione}}{\MakeUppercase{Prefazione}}
```

dove `\MakeUppercase` produce il proprio argomento in tutte maiuscole (come nelle testatine standard della classe `book`).

Eliminare testatine inutili e aggiungere pagine bianche

Per eliminare testatine e piedi *comunque* presenti nelle pagine bianche alla fine di un capitolo usando le classi standard con l'opzione `openright` (scelta consigliata), basta caricare il pacchetto `emptypage`.

3.9 Indice generale, titoli e profondità

3.9.1 Indice generale, miniindici e indici abbreviati

Il comando

```
\tableofcontents
```

produce *nel punto in cui viene dato* la sezione contenente l'indice generale con relativi titolo e testatina. Si noti che per ottenerlo nel documento finito servono *due* composizioni successive.

3.9.2 Gestire i titoli

Titoli non numerati né indicizzati

Di tutti i comandi di sezionamento elencati nella tabella 3.5 esiste anche una *variante asterisco* formata dal nome del comando con un asterisco alla fine, che genera titoli *non numerati* e che nemmeno andranno a finire nell'indice generale, di qualunque livello essi siano. Il titolo precedente, per esempio, si è ottenuto con il comando

```
\subsubsection*{Titoli non numerati né indicizzati}
```

Per mandare nell'indice anche i titoli che normalmente non ci finirebbero, *subito dopo* il relativo comando di sezionamento basta dare

```
\addcontentsline{⟨indice⟩}{⟨livello⟩}{⟨titolo⟩}
```

dove:

- *⟨indice⟩* è il tipo di indice in cui far comparire la voce in questione (normalmente si sceglie *toc*, *lot* o *lof* per l'indice generale, delle tabelle e delle figure rispettivamente);
- *⟨livello⟩* è il nome del livello di sezionamento in questione (si noti che *paragraph* e *subparagraph* non sono ammessi);
- *⟨titolo⟩* è il titolo di sezione che finirà nell'indice.

Applicato al titolo di questa sezione, il codice sarebbe:

```
\subsection*{Titoli non numerati né indicizzati}
%\phantomsection
\addcontentsline{toc}{subsection}{Titoli non numerati né indicizzati}
```

Si noti che se *hyperref* è caricato, *subito prima* di *\addcontentsline* va dato anche il comando *\phantomsection* per evitare possibili errori nei collegamenti ipertestuali e nei segnalibri del documento finito (in tal caso, si decommenti la riga corrispondente).

Titoli alternativi nell'indice generale

Nell'indice generale finiscono i titoli scritti nell'argomento dei comandi di sezionamento. Se, però, un titolo è troppo lungo per starci agevolmente (si noti che un titolo non dovrebbe *mai* andare a capo) o si hanno particolari esigenze, lo si può sostituire con un titolo alternativo più breve, da inserire nell'argomento facoltativo degli stessi comandi:

```
\chapter[Leggilo! È emozionante!]{Questo è un titolo lunghissimo e molto noioso}
```

Si noti che il titolo breve comparirà anche nelle testatine, se previste dalla classe di documento in uso e che, ovviamente, non si può usare se il comando è asteriscato.

3.10 Riferimenti incrociati

Nei documenti si trovano spesso riferimenti incrociati a sezioni, figure, tabelle, teoremi e altri elementi. Per realizzarli si usano i comandi standard

```
\label{⟨etichetta⟩}
\ref{⟨etichetta⟩}
\pageref{⟨etichetta⟩}
```

dove:

- *\label* assegna agli elementi contrassegnati un' *⟨etichetta⟩* arbitraria e *univoca*, che si consiglia di scrivere sempre nella forma *⟨abbreviazione⟩:⟨parola chiave⟩* (dove la prima è un'abbreviazione dell'elemento in questione, come *tab* per una tabella, e la seconda una stringa identificativa) e avendo cura di evitare i caratteri accentati (scrivendo in francese, inoltre, si sconsiglia di mettere i due punti nelle etichette);
- *\ref* produce il numero dell'elemento messo in *⟨etichetta⟩*;
- *\pageref* produce il numero di pagina in cui l'elemento compare.

Per produrre i riferimenti incrociati nel documento sono necessarie *due* composizioni successive, altrimenti al loro posto si vedranno altrettanti ??.

Per esempio, se s'identifica questo paragrafo con

```
\section{Riferimenti incrociati}
\label{sec:rif-inc}
```

poi ci si può riferire a esso con

```
Ecco un riferimento a questo
paragrafo: ‘‘si veda il
paragrafo~\ref{sec:rif-inc}’’.
```

Ecco un riferimento a questo paragrafo: “si veda il paragrafo 3.10”.

Si noti che è una buona abitudine unire il riferimento alla parola precedente con uno *spazio indivisibile*: garantisce in genere risultati tipografici ottimali.

Se il documento contiene molti riferimenti incrociati potrebbe essere utile controllare la correttezza delle etichette o averle sempre sotto controllo: il pacchetto `showkeys` le visualizza nel margine della pagina.

3.11 Collegamenti ipertestuali e al Web

Collegamenti ipertestuali e al Web

Il pacchetto `hyperref`, che di regola va caricato *per ultimo*, crea i collegamenti ipertestuali all'interno del documento, rendendo cliccabili i riferimenti incrociati visti nella sezione precedente, quelli a voci bibliografiche, a indirizzi Internet e molto altro. Se ne possono specificare le opzioni come di consueto oppure, se numerose, si può usare il comando `\hypersetup`:

```
\usepackage{hyperref}
\hypersetup{<chiave>=<valore>,<...>}
```

Il pacchetto permette di gestire collegamenti e segnalibri in modo molto fine: se ne veda la documentazione.

Per impostazione predefinita, `hyperref` circonda il collegamento con un riquadro colorato che *non* viene stampato. Si può avere il testo del collegamento colorato (con colori predefiniti o a piacere) scrivendo:

```
\usepackage[colorlinks]{hyperref}
```

Questo è utile per un documento da leggere a schermo o da stampare a colori, ma si ricordi che la stampa in bianco e nero restituisce i colori come sfumature di grigio, a volte poco leggibili.

Si possono avere tutti i collegamenti in nero e senza riquadri scrivendo semplicemente

```
\hypersetup{hidelinks}
```

Oltre ai collegamenti ipertestuali per i riferimenti incrociati, `hyperref` permette anche di realizzare collegamenti al Web con il comando `\href`:

```
\href{<indirizzo Internet>}{<testo del collegamento>}
```

Scrivendo

```
Visita il sito del
\href{http://www.guitex.org/}{\GuIT}.
```

Visita il sito del $\mathbb{G}\mathbb{U}\mathbb{I}\mathbb{T}$.

basta cliccare sul logo $\mathbb{G}\mathbb{U}\mathbb{I}\mathbb{T}$ per accedere al sito omonimo.

Indirizzi Internet e di posta elettronica

Il pacchetto `url` (caricato automaticamente da `hyperref`) definisce il comando `\url`, utile per scrivere un indirizzo Internet:

<code>\url{http://www.guitex.org/}</code>	<code>http://www.guitex.org/</code>
---	-------------------------------------

Per i collegamenti a un indirizzo di posta elettronica conviene definire nel preambolo un apposito comando `\mail` (si veda il paragrafo 9.1),

```
\newcommand{\mail}[1]{\href{mailto:#1}{\texttt{#1}}}
```

da usare come segue:

<code>\mail{lorenzo.pantieri@gmail.com}</code>	<code>lorenzo.pantieri@gmail.com</code>
--	---

3.12 Pacchetti

Scrivendo un documento, prima o poi ci s’imbatte in problemi che \LaTeX non riesce a risolvere da solo. Il suo linguaggio standard, per esempio, non gestisce l’inclusione delle immagini, né sillaba documenti in lingue diverse dall’inglese, né ancora permette di modificare facilmente i margini di pagina. Per aggirare “ostacoli” di questo tipo si sfrutta la struttura modulare del programma, che estende le proprie capacità di base tramite moduli aggiuntivi chiamati *pacchetti*.

3.12.1 Caratteristiche

Che cosa sono?

Fondamentalmente, un pacchetto è un file “di stile” (con estensione `.sty`) scritto in linguaggio \LaTeX , contenente istruzioni che permettono di svolgere alcune operazioni.

Come sapere se servono?

In genere, se per ottenere il risultato sperato si deve faticare troppo, probabilmente qualcuno che si è già trovato nella stessa situazione ha provveduto a creare un pacchetto per semplificare il lavoro.

\TeX Live non comprende *tutti* i pacchetti presenti su CTAN. Infatti, componendo un sorgente può capitare che \LaTeX produca un messaggio di errore del tipo

```
Can't find file steroid.sty
```

Ciò significa che è stato caricato un pacchetto (`steroid`, nell’esempio considerato) non presente nella distribuzione. Si risolve il problema seguendo le istruzioni contenute nel paragrafo 2.1.2.

Viceversa, può accadere di usare un comando definito da un pacchetto che ci si è dimenticati di caricare: si otterrà un messaggio di errore di comando sconosciuto, che purtroppo non aiuta a indovinare il pacchetto che serve.

E se serve un pacchetto che non c’è in \TeX Live?

Per usare un pacchetto che *non può* esserci nella distribuzione (perché non ne è prevista l’inclusione in \TeX Live, o è un pacchetto personale o coperto da una licenza particolare, o è una versione sperimentale che si vuole comunque provare, o ancora perché l’aggiornamento o la pubblicazione cadono nel periodo di “congelamento”) la strada più semplice è copiarne i file nella cartella di lavoro.

Come scovare il pacchetto che fa al proprio caso?

Questo è l'unico aspetto del lavoro con \LaTeX in cui gusto, abilità e fortuna la fanno da padroni: cercando su \TeX Catalogue (<http://texcatalogue.ctan.org/>) si trovano preziosi riferimenti e soluzioni per risolvere moltissimi problemi.

3.12.2 Caricamento e precauzioni

Come caricarli?

I pacchetti si caricano *nel preambolo* con il comando

```
\usepackage[\langle opzioni \rangle]{\langle pacchetto \rangle}
```

dove:

- *\langle opzioni \rangle* è una voce o un elenco di voci separate con la virgola costituite da un solo elemento o un'espressione del tipo *\langle chiave \rangle = \langle valore \rangle* che specificano le impostazioni del pacchetto;
- *\langle pacchetto \rangle* è il nome del pacchetto, che va scritto *sempre* in tutte minuscole (LayAureo si scrive così, ma si carica come `layaureo`, per intenderci).

Si noti che con uno stesso comando `\usepackage` si possono caricare più pacchetti *senza opzioni*, separandone i nomi con la virgola.

Quali precauzioni prendere?

Non si possono caricare i pacchetti in un ordine casuale, anche se ciò è permesso entro certi limiti. Sequenze di caricamento ben precise, come si è già visto per `fontenc`, `inputenc` e `babel` sono richieste anche per altri pacchetti, ma non è questa la sede per elencarle tutte. I messaggi d'errore notificati dal programma in tal senso sono chiari, di solito, ma le precauzioni da prendere non sono mai troppe. Di seguito si danno alcuni consigli che dovrebbero limitare i problemi.

- \LaTeX richiede di caricare (direttamente o indirettamente) i pacchetti *solo nel preambolo e una volta sola*, con *tutte* le opzioni che servono.
- Molti pacchetti ne caricano automaticamente degli altri: lo si può scoprire leggendo la documentazione. Non sapendolo e ricaricando un pacchetto, si ottiene un errore.
- Talvolta il caricamento dei pacchetti sottostà a vincoli precisi: alcuni vanno caricati prima di altri e viceversa, pena un errore. La documentazione del pacchetto indicato nell'errore potrebbe contenere informazioni utili: di solito basta modificare l'ordine di caricamento o eliminare la doppia chiamata.

3.12.3 Usarli al meglio: la documentazione

Chi scrive o aggiorna un pacchetto per \LaTeX ne scrive anche, e quasi sempre in inglese, la *documentazione*, che spesso si compone di due parti distinte:

- il manuale d'uso, che dichiara lo scopo del pacchetto e ne descrive i comandi;
- il codice che costituisce il pacchetto, destinato a chi voglia eventualmente svilupparlo (nel caso di pacchetti molto corposi, il codice costituisce un file a sé).

Tabella 3.7: Principali unità di misura tipografiche riconosciute da L^AT_EX (*m-width* generalmente approssima la larghezza della *M* nel font in uso)

Unità	Codice	Valore
centimetro	cm	
millimetro	mm	
punto tipografico	pt	0,3514 mm
<i>x-height</i>	ex	Circa uguale all'altezza della <i>x</i> nel font in uso
<i>m-width</i>	em	Circa uguale al corpo del font in uso

I pacchetti contenuti in ogni distribuzione di L^AT_EX sono già corredati della relativa documentazione (quasi sempre un PDF omonimo), facilmente raggiungibile con il programma `texdoc`, integrato in T_EX Live. Il programma si lancia dalla riga di comando o con le scorciatoie che di solito ogni editor definisce a questo scopo e prevede numerose opzioni di ricerca: eseguendo

```
texdoc <nome del pacchetto>
```

da una posizione qualunque sul proprio computer, in un attimo si apre il relativo manuale.

3.13 Unità di misura tipografiche

Nei prossimi capitoli spesso si useranno istruzioni che richiedono di esprimere una lunghezza in una qualche *unità di misura tipografica*. Dal momento che alcune di esse poco hanno a che fare con quelle più conosciute del sistema metrico decimale, nella tabella 3.7 si mostrano quelle effettivamente usate in questa guida.

Esistono inoltre i comandi di spaziatura `\quad` (o *quadrato*) e `\qquad` (o *quadrato*), che producono rispettivamente uno spazio di 1 e 2 em. Si noti che in un ambiente puramente testuale il loro uso è di regola fortemente sconsigliato e va limitato a casi particolarissimi.

3.14 Documenti di grandi dimensioni

Per scrivere senza sorprese un documento di grandi dimensioni come un libro o una tesi è importantissimo organizzarne razionalmente il materiale. Prendendo a esempio questa guida (ma i suggerimenti valgono anche per una tesi di laurea o un altro documento), si sono messi *tutti* i suoi file in una cartella `artelatex`, strutturata in sottocartelle come segue:

- La sottocartella `inizio`, con il materiale iniziale suddiviso nei corrispondenti file come `ringraziamenti.tex`, `introduzione.tex`, eccetera.
- La sottocartella `capitoli`, con il materiale principale suddiviso nei corrispondenti file come `basi.tex`, `testo.tex`, `tabellefigure.tex`, eccetera.
- La sottocartella `fine`, con il materiale finale suddiviso nei corrispondenti file come `acronimi.tex`, `sitiinternet.tex`, eccetera.
- La cartella `immagini`, con tutte le immagini incluse nella guida. Se sono molte, le si potrebbe distribuire in ulteriori sottocartelle corrispondenti ai diversi capitoli. Immaginando di chiamarle grafici e foto, basta scrivere nel file di impostazioni (si veda poco più sotto)

```
\graphicspath{{grafici/},{foto/}}
```

La cartella `artelatex` *deve* contenere anche altri due file:

- il file *principale* del documento, `artelatex.tex`, cioè quello che contiene dichiarazione di classe, preambolo, impostazioni generali e ambiente `document`;

Se il documento non è troppo corposo, invece, se ne possono mettere tutti i file in una sola cartella e l'indicazione del percorso non serve più.

Infine, si sono raccolte definizioni di comandi e ambienti personali e impostazioni generali del documento in un pacchetto `impostazioni-arte.sty` (si scrive con l'editor in uso, si registra con estensione `.sty`, *non* richiede il preambolo e *non* va composto), caricato nel preambolo come un normale pacchetto immediatamente prima dell'inizio del documento. Questi piccoli accorgimenti "puliscono" il file principale semplificando notevolmente il proprio lavoro.

Nell'ambiente `document` si caricano i file `.tex` in cui si è suddiviso il documento, scrivendo il nome del file *senza l'estensione* nell'argomento del comando `\input`, indicandone l'eventuale percorso, come nell'esempio seguente:

```
\begin{document}

...
\input{inizio/ringraziamenti}
\input{inizio/introduzione}
...
\input{capitoli/basi}
\input{capitoli/installare}
\input{capitoli/testo}
...
\input{fine/acronimi}
...

\end{document}
```

Questo accorgimento evidenzia molto chiaramente la struttura del documento e snellisce il file principale. In pratica, `\input` costruisce il documento "attaccando" semplicemente uno dopo l'altro i vari file. Si noti che:

- Questi ultimi *non* devono contenere alcun preambolo, ma solo comando di sezionamento e contenuto della sezione.
- `\input` permette l'annidamento (cioè ammette altri `\input` nel proprio argomento).
- Si abbia cura che il percorso dei file inclusi (anche quelli con `\graphicspath`) sia *relativo* alla cartella dove si trova il file principale (un percorso assoluto verrebbe addirittura *rifiutato* da L^AT_EX per motivi di sicurezza) e *non* contenga spazi.
- Per evitare problemi che poi sarebbe difficile risolvere, si raccomanda di nominare i file con *una sola* parola alfanumerica senza maiuscole, punti, spazi intermedi e caratteri particolari. Se fosse davvero necessario separare i due membri del nome del file, a parte `prima.tex` o `parte.prima.tex`, per esempio, si preferisca `parte-prima.tex`.

Il metodo appena spiegato è particolarmente utile per includere nel documento elementi come tabelle o grafici particolarmente complessi e possibile fonte di errori difficilmente individuabili se composti direttamente nel sorgente. Lo si può usare con profitto anche per comporre una sezione del documento alla volta (commentando quelle che non servono) con notevole risparmio di tempo.

Capitolo 4

Testo

4.1 Struttura del testo

Lo scopo principale di chi scrive un testo è comunicare idee e conoscenze al lettore, che le comprenderà tanto più quanto meglio sono strutturate, e ne apprezzerà tanto più la struttura quanto più la forma tipografica del documento rispecchia la costruzione logica del suo contenuto.

Le suddivisioni elencate nella tabella 4.1 sono fondamentali per comprendere l'articolazione di un testo scritto, e vengono chiamate in generale *sezioni*.

A questo proposito si noti che:

- il sezionamento dei documenti è compito dell'utente, perché \LaTeX non lo fa automaticamente;
- è estremamente importante scandire il testo in capoversi, per chi scrive e per chi legge: le informazioni sono meglio articolate e più facilmente memorizzabili.

4.2 Comporre i capoversi

Spesso si sottovaluta l'importanza di scrivere un testo ben strutturato, e usando \LaTeX altrettanto frequentemente si comincia un nuovo capoverso senza nemmeno rendersene conto.

È molto facile commettere quest'ultimo errore se il testo contiene formule matematiche. Infatti l'abitudine, diffusa, di lasciare una riga vuota tra la fine di una formula e la prosecuzione del testo si sconta nel documento finito con altrettanti nuovi capoversi, anche laddove il flusso del discorso non li richiederebbe affatto.

Cominciare un nuovo capoverso

Con l'ovvia eccezione del primo capoverso di una sezione, per cominciare un nuovo capoverso con \LaTeX si hanno due possibilità:

- si lascia una riga vuota nel sorgente (di solito si fa così);
- si dà il comando `\par`.

In ogni caso, non lo si faccia *mai* con `\\` (qualche esempio di questa guida lo ha richiesto per motivi di spazio).

Osservando gli esempi che seguono, si cerchi di capire perché a volte c'è la riga bianca e altre no. (Se non si comprendono ancora tutti i comandi, si leggano interamente questo capitolo e i primi paragrafi del capitolo 5, e poi si ritorni su questo punto.)

Tabella 4.1: Lunghezza orientativa delle sezioni di un testo scritto

Sezione	Lunghezza orientativa
Parte	Imprecisabile
Capitolo	Da una decina a un centinaio di pagine
Paragrafo	Da mezza a una decina di pagine
Sottoparagrafo	Da poche righe a un paio di pagine
Capoverso	Da una a una ventina di righe
Enunciato	Da una parola a una decina di righe

```
\dots quando Einstein propose
l'equazione
\begin{equation}
E = mc^2
\end{equation}
che è la più nota e la meno compresa
formula della Fisica.
```

...quando Einstein propose l'equazione

$$E = mc^2 \quad (4.1)$$

che è la più nota e la meno compresa formula della Fisica.

```
\dots che, rispetto ai precedenti,
ha alcuni vantaggi.
```

...che, rispetto ai precedenti, ha alcuni vantaggi.

```
La formula
\begin{equation}
D=F-R
\end{equation}
definisce un modello molto diverso
di transistor
```

La formula

$$D = F - R \quad (4.2)$$

definisce un modello molto diverso di transistor

```
\dots da cui segue la legge di
Kirchhoff sulle correnti:
```

...da cui segue la legge di Kirchhoff sulle correnti:

```
\begin{equation}
\sum_{k=1}^n I_k = 0
\end{equation}
```

$$\sum_{k=1}^n I_k = 0 \quad (4.3)$$

```
La legge di Kirchhoff sulle tensioni
può essere ricavata\dots
```

La legge di Kirchhoff sulle tensioni può essere ricavata...

Capoversi ben composti

Un documento “ben composto” si riconosce da alcuni elementi: il testo è giustificato, le parole sono adeguatamente spaziate tra loro e sillabate a fine riga se proprio non ci stanno, i capoversi presentano la prima riga rientrata per facilitare la lettura. Di solito tutto questo si ottiene dando a mano le rispettive impostazioni; con \LaTeX , invece, non occorre nemmeno pensarci, perché il programma:

- giustifica il testo per impostazione predefinita;
- rientra automaticamente la prima riga di ogni capoverso tranne il primo (se per qualche motivo non si volesse il rientro, basta cominciare la riga interessata con `\noindent`);
- numera automaticamente le pagine del documento;

- non aggiunge spazio supplementare tra un capoverso e l'altro tranne quando non ha abbastanza materiale per riempire perfettamente la pagina.

Talvolta, invece, questo spazio supplementare potrebbe servire. Lo si può inserire con i seguenti comandi:

- `\bigskip`, `\medskip` e `\smallskip`, avendo cura di lasciare una riga bianca *prima*, inseriscono uno spazio verticale rispettivamente “grande”, “medio” e “piccolo” la cui ampiezza è in funzione del font utilizzato.
- `\vspace{⟨lunghezza⟩}` inserisce uno spazio verticale pari a `⟨lunghezza⟩` (che va perso se dopo la composizione viene a trovarsi all'inizio di una pagina: per mantenerlo basta usare la forma `\vspace*`.)

La tipografia anglosassone (predefinita in \LaTeX) non prevede il rientro della prima riga del primo capoverso di una sezione. Per ottenerlo, secondo una consuetudine spesso seguita in Italia, basta semplicemente caricare il pacchetto `indentfirst` nel modo consueto.

Interrompere una riga senza cominciare un nuovo capoverso

In casi particolari può essere necessario interrompere una riga. Per farlo si usano i comandi `\\` o `\newline`, e se ne incomincia una nuova *senza iniziare un nuovo capoverso* (e senza rientro, dunque, come qui).

Si può inserire uno spazio aggiuntivo tra due linee dello stesso capoverso con il comando `\\[⟨lunghezza⟩]`, in cui `⟨lunghezza⟩` può essere espressa in una qualunque delle unità di misura tipografiche accettate da \LaTeX , avendo cura di usare *il punto* come separatore decimale (si veda la tabella 3.7).

Dividere le parole a fine riga

In generale, \LaTeX cerca di interrompere le righe sempre nel miglior punto possibile. Se, però, non riesce a farlo neppure secondo i propri severi criteri, le lascia fuoriuscire dal margine destro e avverte l'utente con un messaggio di *Overfull hbox*. Non sempre è facile individuarle: nel capitolo 10 si spiega come fare.

L'algoritmo di sillabazione di \LaTeX funziona correttamente con quasi tutte le parole, ma in particolari circostanze si potrebbe volere una divisione diversa da quella automatica. Con nomi propri o tecnicismi come *nitroidrossilamminico* o *macroistruzione*, per esempio, a volte si richiede la sillabazione etimologica anziché quella che \LaTeX esegue di default: *nitro-idrossil-amminico* invece di *ni-troi-dros-si-lam-mi-ni-co* e *ma-cro-i-stru-zio-ne* invece di *ma-croi-stru-zio-ne*.

In questi casi basta scrivere le parole nell'argomento di `\hyphenation` (nel preambolo) già *sillabate*, separandole con uno spazio ed evitando caratteri speciali e simboli:

```
\hyphenation{nitro-idrossil-amminico ma-cro-istru-zio-ne}
```

Il comando precedente funziona anche al contrario. Una scrittura come la seguente

```
\hyphenation{nitro-idrossil-amminico FORTRAN}
```

sillaba *nitroidrossilamminico* e *Nitroidrossilamminico* come suggerito nell'argomento, ma non *FORTRAN*, *Fortran* e *fortran*. Si può usare `\hyphenation` per forzare qualunque cesura si desideri: se si vuole spezzare la parola *melograno* soltanto tra *melo* e *grano*, si scrive:

```
\hyphenation{melo-grano}
```

Se la parola in questione compare nel documento una sola volta, si può suggerirne la sillabazione direttamente nel testo. Il comando `\-` spezza la parola nel punto (o nei punti) in cui viene dato, e *in quel punto soltanto*.

Nel 1896 venne scoperto l'acido
nitro\ -idrossil\ -amminico.

Nel 1896 venne scoperto l'acido nitro-
idrossilamminico.

Si noti che anche gli interventi sulla sillabazione, come tutti quelli operati “a mano” sul documento, dovrebbero essere effettuati durante la revisione finale immediatamente precedente la stampa. La prima cura per un *Overfull hbox*, per esempio, dovrebbe consistere *sempre* nel riformulare l'enunciato piuttosto che nell'imporre una sillabazione particolare.

Il comando

`\mbox{⟨testo⟩}`

serve per mantenere unita una parola *senza* usare `\hyphenation`. Va usato all'occorrenza, magari perché in un certo punto del documento non va bene che la parola sia spezzata, ma altrove sì:

Entro quest'anno avrò imparato bene
il Fortran. `\[1ex]`
Entro quest'anno avrò imparato bene
il `\mbox{Fortran}`.

Entro quest'anno avrò imparato bene il For-
tran.
Entro quest'anno avrò imparato bene il
Fortran.

Spazi interparola e punti fermi

Per giustificare i capoversi \LaTeX inserisce spazi interparola variabili e migliora la leggibilità del testo separando gli enunciati con uno spazio leggermente più ampio di quello inserito da un comune elaboratore di testo. Il programma interpreta diversamente il punto:

- un punto (fermo, interrogativo o esclamativo) dopo una *minuscola* indica la fine di un enunciato, e dopo di esso \LaTeX inserisce uno spazio supplementare;
- un punto dopo una *maiuscola* indica la fine di un'abbreviazione, e dopo di esso ci sarà uno spazio normale.

Le eccezioni alle regole generali appena esposte vanno specificate esplicitamente. I casi sono tre:

- immediatamente *dopo* un punto di fine abbreviazione dentro un enunciato (tranne se l'abbreviazione ne è l'ultima parola), si usa `_`;
- immediatamente *prima* di un punto di fine enunciato che segue una maiuscola (che per \LaTeX indica *comunque* un'abbreviazione) si usa `\@`;
- per tenere unite espressioni che non si vogliono o non possono *mai* essere spezzate da un fine riga si usa lo *spazio indivisibile* prodotto dalla tilde `~`.

L'esempio seguente mostra `\@` all'opera:

CEE. Poi CE. Ora UE. `_`
CEE\@. Poi CE\@. Ora UE\@.

CEE. Poi CE. Ora UE.
CEE. Poi CE. Ora UE.

La spaziatura corretta è quella prodotta dalla seconda scrittura.

Si osservi come agisce la tilde nei due esempi seguenti:

Tabella 4.2: Virgolette, tratti e puntini di sospensione

	Segno	Codice	Risultato
Virgolette	semplici alte	' '	' '
	doppie alte	' ' ' '	“ ”
		“ ”	“ ”
	doppie basse	<< >>	« »
		« »	« »
Tratti	trattino	-	-
	tratto	--	—
	lineetta	---	—
	meno	\$-\$	—
Puntini		\dots	...

Avevo studiato a fondo il manuale
del prof. Beccari. `\\[1ex]`
Avevo studiato a fondo il manuale
del prof.~Beccari.

Avevo studiato a fondo il manuale del prof.
Beccari.

Avevo studiato a fondo il manuale del
prof. Beccari.

Questi concetti sono spiegati nel
paragrafo `\ref{sec:par}`. `\\[1ex]`
Questi concetti sono spiegati nel
paragrafo~`\ref{sec:par}`.

Questi concetti sono spiegati nel paragrafo
4.2.

Questi concetti sono spiegati nel paragra-
fo 4.2.

Come si può notare, la seconda scrittura di ciascuna coppia, che è quella corretta, evita che le righe finiscano o comincino nel modo sbagliato.

4.3 Caratteri particolari e simboli

4.3.1 Virgolette, tratti e puntini di sospensione

Virgolette

In tipografia si usano comunemente tre tipi di virgolette: gli ‘apici’, le “virgolette inglesi” e le «virgolette caporali». La tabella 4.2 mostra come ottenerle.

Gli esempi seguenti le mostrano all’opera:

Ora è essere chiaro il concetto
di ‘composizione asincrona’.

Ora è essere chiaro il concetto di ‘composi-
zione asincrona’.

La Delta di Dirac
è una “funzione impropria”.

La Delta di Dirac è una “funzione impropria”.

<<Se stai attento, capisci tutto.>>

«Se stai attento, capisci tutto.»

Trattini, tratti e lineette

La tipografia distingue quattro tipi di tratto: tre (*trattino*, *tratto* e *lineetta*) corrispondono a un numero crescente di trattini consecutivi, mentre il quarto è il segno matematico *meno*. La tabella 4.2 mostra come ottenerli, e gli esempi seguenti ne illustrano alcuni possibili usi:

Tabella 4.3: Accenti e caratteri particolari

\‘o	ò	\u{o}	ö	\~o	õ	\.o	ó
\’o	ó	\t{oo}	ôo	\r{o}	õ	\d{o}	o
\~o	ô	\"o	ö	\c{o}	ç	\=o	ō
\v{o}	ö	\H{o}	ő	\k{o}	ø	\b{o}	ü
\OE	Œ	\AE	Æ	\AA	Å	\O	Ø
\oe	œ	\ae	æ	\aa	å	\o	ø
\L	Ł	\DH	Ð	\DJ	Đ	\TH	Þ
\l	ł	\dh	ð	\dj	đ	\th	þ

Stratford-on-Avon, e-mail \\
p.~13-67, 1921-28 \\
Ottica~--~Schema generale \\
---~Eccomi~--- disse. \\
\$0\$, \$1\$ e \$-1\$

Stratford-on-Avon, e-mail
p. 13-67, 1921-28
Ottica – Schema generale
— Eccomi — disse.
0, 1 e −1

Puntini di sospensione

Se inseriti battendo tre punti consecutivi, i *puntini di sospensione* potrebbero compromettere la spaziatura tra le parole o la corretta interruzione di riga. \LaTeX risolve il problema definendo il comando `\dots`, che li produce correttamente spaziati e li tiene uniti *in ogni caso*:

Non così... ma così: \\
Londra, Parigi\dots{} Berlino.

Non così... ma così:
Londra, Parigi... Berlino.

4.3.2 Accenti, caratteri particolari, apici e pedici

Accenti e caratteri particolari

\LaTeX permette di usare accenti e caratteri particolari di molte lingue (nella tabella 4.3 sono esemplificati per la lettera *o*, ma funzionano anche per tutte le altre lettere), come si può vedere nell’esempio seguente:

Na\~{i}f, Stra\~{s}se, !‘Se\~{n}orita!,
Sm\~{o} rrebr\~{o} d, z\~{l} oty

Naïf, Straße, ¡Señorita!, Smørrebrød, złoty

Il simbolo ufficiale dell’euro (€), da mettere sempre *dopo* l’eventuale numero, si ottiene con il comando `\euro` del pacchetto `eurosym`.

Apici e pedici

L’opzione `italian` di `babel` definisce una coppia di comandi che producono il proprio argomento in tondo *anche in modo matematico*. In modo testuale, inoltre, mantengono anche lo stile corrente, qualunque sia:

- `\ap{<testo>}` produce un apice come nelle abbreviazioni oggi in disuso *sig.^{ra}* o *f.^{lli}* (l’alternativa è `<testo>`);
- `\ped{<testo>}` produce un pedice, utile per qualche sostanza chimica come la vitamina B₁₂, per esempio.

Tabella 4.4: Comandi per modificare lo stile del font

Comando	Dichiarazione	Stile
<code>\emph</code>	<code>\em</code>	<i>Evidenziato</i>
<code>\textit</code>	<code>\itshape</code>	<i>Corsivo</i>
<code>\textsc</code>	<code>\scshape</code>	MAIUSCOLETTO
<code>\textbf</code>	<code>\bfseries</code>	Nero
<code>\textsl</code>	<code>\slshape</code>	<i>Inclinato</i>
<code>\textrm</code>	<code>\rmfamily</code>	Tondo
<code>\textsf</code>	<code>\sffamily</code>	Senza grazie
<code>\texttt</code>	<code>\ttfamily</code>	Macchina per scrivere

Per l'elenco completo di *tutti* i simboli e i caratteri speciali di \LaTeX (diverse migliaia), si veda [Pakin, 2015].

4.4 Modificare stile e corpo del font

4.4.1 Modificare lo stile

I comandi elencati nella tabella 4.4 modificano lo *stile* del proprio argomento (e solo di quello), lasciando invariato il testo successivo:

La parola che segue è in
`\textit{corsivo}`.
 Il resto del testo è normale.

La parola che segue è in *corsivo*. Il resto del testo è normale.

I comandi si possono combinare, ma la combinazione richiesta potrebbe non esserci nel font in uso.

A ciascun comando corrisponde una dichiarazione che si comporta come spiegato nel paragrafo 3.4.1. Anche le dichiarazioni si possono combinare:

L'espressione che segue `{\itshape è in {\bfseries nero corsivo}}`.

L'espressione che segue è *in nero corsivo*.

ma si consiglia di usarle, se proprio necessarie, per porzioni di testo consistenti e non per singole parole come qui.

4.4.2 Modificare il corpo

L'effettivo corpo del font in un documento dipende da tre fattori:

- la classe di documento scelta;
- l'opzione di corpo (eventualmente) assegnata alla classe;
- le (eventuali) dichiarazioni per modificare il corpo del font date nel testo.

Le dichiarazioni elencate nella tabella 4.5 modificano il *corpo* del font. Anche il contenuto della tabella risente dei fattori appena elencati: in particolare, `\normalsize` è il corpo del testo principale di questa guida.

Lettere `{\Large grandi}` e
`{\scriptsize piccole}`.

Lettere **grandi** e piccole.

Tabella 4.5: Dichiarazioni per modificare il corpo del font

Dichiarazione	Risultato
<code>\tiny</code>	Esempio
<code>\scriptsize</code>	Esempio
<code>\footnotesize</code>	Esempio
<code>\small</code>	Esempio
<code>\normalsize</code>	Esempio
<code>\large</code>	Esempio
<code>\Large</code>	Esempio
<code>\LARGE</code>	Esempio
<code>\huge</code>	Esempio
<code>\Huge</code>	Esempio

Si noti che le dichiarazioni appena viste *modificano anche l'interlinea* del capoverso interessato, com'è giusto che sia, ma solo se esso termina entro il loro raggio d'azione. Nei due esempi seguenti, `\par` produce effetti differenti a seconda di dove lo si dà.

```
{\large Socrate: «Platone mentirà
nella frase seguente».\par}
```

Socrate: «Platone mentirà nella frase seguente».

```
{\large Platone: «Socrate ha detto il
vero nella frase precedente».\par}
```

Platone: «Socrate ha detto il vero nella frase precedente».

Come si può osservare nel primo esempio, fuori dal gruppo `\par` non funziona più, con un risultato finale poco gradevole. La scrittura corretta pertanto è la seconda.

Questo viaggio tra stili e dimensioni si conclude con un simpatico consiglio, che starà all'utente seguire o meno:

Ricorda! *Quanti Più corpi e stili scegli di usare in un documento, tanto più LEGGIBILE e bello diventa.*

4.5 Titoli e frontespizi

Titoli standard

Il comando

```
\maketitle
```

dato *dopo* `\begin{document}` produce il “titolo” del documento, un blocco di informazioni definite dai comandi

```
\title{<titolo>}
\author{<autore>}
\date{<data>}
```

Il loro funzionamento si spiega da sé, ma si osservi quanto segue:

- un `<titolo>` troppo lungo per stare su una sola riga si spezza con `\\` (ma lo si eviti il più possibile);



Figura 4.1: Esempio d'uso di frontespizio

- i vari $\langle autore \rangle$ di un documento scritto a più mani si separano con `\and`;
- \LaTeX stampa la $\langle data \rangle$ della composizione anche se `\date` non viene dato, mentre la omette lasciandone vuoto l'argomento (`\date{}`).

Per inserire ringraziamenti veloci si usa il comando

```
\thanks{\langleringraziamenti\rangle}
```

che nelle classi standard produce il proprio argomento come una nota al piede con un simbolo a esponente. Lo si può dare *dentro* l'argomento di uno qualunque dei tre comandi appena esaminati. Ecco un esempio:

```
\author{Lorenzo Pantieri \and Tommaso Gordini%
        \thanks{Ringraziamo i membri del \GuIT.}}
```

Nelle classi \mathcal{AMS} `\thanks` va invece dato in una riga a sé e *fuori* dai comandi.

Frontespizio

Il titolo generato dal comando `\maketitle`, si deve riconoscere, è piuttosto spartano, anche se si può accettare in articoli e relazioni. Si consiglia di comporre il frontespizio di una tesi di laurea o di dottorato con il pacchetto `frontespizio` (si veda la figura 4.1). Il pacchetto, personalizzabile, permette di inserire tutti i dati necessari, prevede opzioni per usare i diversi stili di carattere e inserire loghi universitari e immagini in filigrana. Se ne veda la ricca documentazione (in italiano).

Infine, se nessuna delle soluzioni precedenti va bene, si può comporre un frontespizio personalizzato con l'ambiente `titlepage` (da aprire *subito dopo* `\begin{document}`) all'interno del quale si è completamente padroni dell'impaginazione.

4.6 Note a margine e a piè di pagina

In linea generale si usino le note *con grande moderazione*: specie quelle al piede, infatti, interrompono la lettura e possono creare seri problemi d’impaginazione. Si tenga presente che una nota *deve* potersi omettere leggendo: se il suo contenuto si rivela essenziale alla comprensione del discorso, evidentemente va tolto dalla nota e messo nel corpo del testo.

Note a margine

Una nota
a margine

Una nota di questo tipo si ottiene molto semplicemente con il comando

```
\marginpar{<testo della nota a margine>}
```

Nei documenti impostati per la stampa in fronte/retro le note vengono stampate nel margine destro nelle pagine dispari e nel margine sinistro in quelle pari. Nei documenti solo fronte, invece, saranno sempre nel margine destro.

Note a piè di pagina

Il comando

```
\footnote{<testo della nota a piè di pagina>}
```

produce una nota in fondo alla pagina corrente con un riferimento nel testo costituito da un numero a esponente. Le note al piede dovrebbero essere messe, per quanto possibile, alla fine del relativo capoverso *dopo* il punto fermo.¹

Le note a piè di pagina sono
l’emblema della meticolosità.%
\footnote{Eccone un esempio.}

Le note a piè di pagina sono l’emblema della
meticolosità.^a

^aEccone un esempio.

Si tenga presente che:

- la loro numerazione riprende a ogni `\chapter` o `\section`;
- se sono poche o pochissime, anziché il riferimento numerico predefinito se ne consiglia uno simbolico: basta scrivere nel preambolo

```
\renewcommand{\thefootnote}{\fnsymbol{footnote}}
```

4.7 Evidenziare le parole

Scrivendo a macchina, le parole importanti si evidenziano con una sottolineatura; in tipografia, invece, le parole si evidenziano mettendole *in corsivo*. Le possibilità sono due.

Per evidenziare una parola o una porzione di testo *indipendentemente dal contesto* in cui si trovano, \LaTeX definisce il comando standard

```
\emph{<testo>}
```

che si vede all’opera nell’esempio seguente:

```
\emph{All’interno di un testo già  
evidenziato, \LaTeX{} evidenzia  
con lo \emph{stile tondo}.}
```

*All’interno di un testo già evidenziato, \LaTeX
evidenzia con lo stile tondo.*

¹Così.

Esiste anche un altro comando standard, `\textit`, che produce il proprio argomento in corsivo *in ogni caso*. Per cogliere la differenza logica tra corsivo ed evidenziato, si osservino le due scritture:

<code>\emph{Tra \textit{un} minuto.} \\\</code>	<i>Tra un minuto.</i>
<code>\textit{Tra \emph{un} minuto.}</code>	<i>Tra un minuto.</i>

4.8 Ambienti testuali

4.8.1 Elenchi puntati, numerati e descrizioni

In un documento gli elenchi sono molto importanti. Infatti:

- fanno “respirare” il testo;
- ne migliorano la leggibilità;
- permettono di strutturare i pensieri.

L’elenco precedente è stato ottenuto con l’ambiente `itemize` come segue:

<p>Gli elenchi:</p> <pre>\begin{itemize} \item fanno “respirare” il testo; \item ne migliorano la leggibilità; \item permettono di strutturare i pensieri. \end{itemize}</pre>	<p>Gli elenchi:</p> <ul style="list-style-type: none"> • fanno “respirare” il testo; • ne migliorano la leggibilità; • permettono di strutturare i pensieri.
--	---

Il comando `\item` mette un pallino nero prima di ogni elemento dell’elenco.

L’ambiente `enumerate` si usa come `itemize`, ma qui a ogni elemento `\item` premette un numero puntato:

<p>Ecco un elenco numerato:</p> <pre>\begin{enumerate} \item Mane; \item Tekel; \item Fares. \end{enumerate}</pre>	<p>Ecco un elenco numerato:</p> <ol style="list-style-type: none"> 1. Mane; 2. Tekel; 3. Fares.
--	--

Si noti che è opportuno usare una lista numerata se in seguito ci si deve riferire a un suo elemento particolare (anche assegnandogli un’etichetta) o se per esempio si devono elencare le fasi di un procedimento. Altrimenti è più opportuna una lista puntata.

L’ambiente `description` si usa per le *descrizioni*, elenchi in cui il segno distintivo è una parola o un’espressione che si deve descrivere o spiegare, da scrivere nell’argomento facoltativo (*in questo caso*, però, obbligatorio) di `\item`:

<p>E ora una descrizione:</p> <pre>\begin{description} \item[itemize] Per gli elenchi puntati. \item[enumerate] Per gli elenchi numerati. \item[description] Per gli elenchi in cui ogni elemento comincia con un testo a piacere. \end{description}</pre>	<p>E ora una descrizione:</p> <p>itemize Per gli elenchi puntati.</p> <p>enumerate Per gli elenchi numerati.</p> <p>description Per gli elenchi in cui ogni elemento comincia con un testo a piacere.</p>
--	--

Come si può osservare, \LaTeX evidenzia automaticamente l'argomento del comando secondo le impostazioni generali della classe di documento in uso.

\LaTeX permette di annidare anche gli elenchi (si consiglia di non annidare mai più di una lista dentro l'altra, però), come mostra l'esempio seguente:

```
Gli elenchi:
\begin{itemize}
\item sono facili da usare;
\item rendono più chiaro il testo:
  \begin{itemize}
    \item articolandolo;
    \item facilitandone la lettura;
  \end{itemize}
\item permettono di strutturare
      i pensieri.
\end{itemize}
```

Gli elenchi:

- sono facili da usare;
- rendono più chiaro il testo:
 - articolandolo;
 - facilitandone la lettura;
- permettono di strutturare i pensieri.

Si noti che \LaTeX cambia automaticamente il contrassegno negli elenchi annidati, che si possono individuare più facilmente nel sorgente rientrandoli leggermente.

Di seguito si riportano alcune convenzioni tipografiche comunemente seguite nella composizione delle liste (le stesse osservate in questa guida):

- ogni voce di un elenco *semplice* (i cui elementi sono costituiti da un solo enunciato) comincia con l'iniziale minuscola e termina con il punto e virgola tranne l'ultima, seguita dal punto fermo;
- ogni voce di un elenco *complesso* (in cui *almeno uno* degli elementi sia composto da più di un enunciato) comincia con l'iniziale maiuscola (anche dopo il segno di due punti) e termina con il punto fermo.

Non bisogna per forza uniformare *tutti* gli elenchi di un documento a criteri stabiliti a priori: l'importante è essere coerenti volta per volta.

4.8.2 Allineare e centrare i capoversi

\LaTeX definisce tre ambienti standard per allineare un capoverso a sinistra:

```
\begin{flushleft}
Questo testo è allineato a \
sinistra. \LaTeX{} non cerca di
creare righe di uguale lunghezza.
\end{flushleft}
```

Questo testo è allineato a sinistra. \LaTeX non cerca di creare righe di uguale lunghezza.

a destra:

```
\begin{flushright}
Questo testo è allineato a \
destra. \LaTeX{} non cerca di
creare righe di uguale lunghezza.
\end{flushright}
```

Questo testo è allineato a destra. \LaTeX non cerca di creare righe di uguale lunghezza.

o per centrarlo sulla pagina:

```
\begin{center}
Al centro \
dell'universo.
\end{center}
```

Al centro
dell'universo.

Come si può osservare, il testo va a capo automaticamente, a meno di un'interruzione esplicita con `\`.

4.8.3 Citazioni

Esistono due modi per scrivere le citazioni con \LaTeX : “in linea” e “in display”.

Citazioni in linea

Una citazione “in linea” è un testo tra virgolette appartenente al flusso del discorso, come quando si cita il motto kantiano «il cielo stellato sopra di me, la legge morale dentro di me».

Una citazione in linea
è un’espressione appartenente
al flusso del discorso:
«il cielo stellato sopra di me,
la legge morale dentro di me».

Una citazione in linea è un’espressione appartenente al flusso del discorso: «il cielo stellato sopra di me, la legge morale dentro di me».

Citazioni di questo tipo sono generalmente brevi.

Citazioni in display

Una citazione “in display” è un testo che va composto entro margini più ampi di quelli correnti e separandolo dal contesto con adeguati spazi bianchi, in modo da metterlo “in mostra” e bene in risalto sulla pagina.

I due ambienti standard definiti da \LaTeX allo scopo, `quote` e `quotation`, non sono del tutto soddisfacenti perché, per esempio, non riducono automaticamente il corpo del testo citato come richiedono le buone tradizioni tipografiche. Per ottenere citazioni in display ben composte si consiglia il pacchetto `quoting`, da impostare come segue nel preambolo se si prevede di mantenere lo stesso stile in tutte le citazioni del documento (scelta consigliata):

```
\usepackage{quoting}
\quotingsetup{font=small}
```

Il pacchetto definisce l’omonimo ambiente `quoting` da usare così:

Una citazione in display è un testo che \LaTeX compone su linee a sé:
`\begin{quoting}`
Il cielo stellato sopra di me,
la legge morale dentro di me.
`\end{quoting}`
La citazione è centrata e separata dal resto del testo.

Una citazione in display è un testo che \LaTeX compone su linee a sé:

Il cielo stellato sopra di me, la legge morale dentro di me.

La citazione è centrata e separata dal resto del testo.

4.8.4 Codici e algoritmi

Talvolta capita di dover scrivere parole o frammenti di testo in modo *verbatim* (“alla lettera”), cioè *non* interpretando spazi, caratteri speciali, rientri, a capo, simboli e comandi, che potrebbero avere un’importanza particolare e devono rimanere tali. Questa modalità è utile per riportare esempi di codici informatici e linguaggi di programmazione, come si è fatto in questa guida tutte le volte che si è mostrata la sintassi dei comandi e degli ambienti di \LaTeX .

Per scrivere un frammento di testo *verbatim in linea* e che non debba andare a capo, \LaTeX definisce il comando standard

```
\verb! <testo verbatim> !
```

Il carattere `!` è solo uno dei possibili *caratteri delimitatori*, cioè con la sola funzione di indicare inizio e fine del *testo verbatim*: a questo scopo si può usare un carattere qualunque, *tranne* `*`, purché non compaia tra i caratteri da riprodurre. Se ne consiglia uno tra `! ? | @`. Si noti che \LaTeX non sillaba il *testo verbatim*: se troppo lungo, infatti, sporgerà nel margine destro.

Per scrivere testo verbatim *in display e su più righe*, invece, c'è l'ambiente standard `verbatim`, da usare come di consueto.

Sia `\verb` sia `verbatim` prevedono una variante asterisco che riproduce lo spazio in modo visibile con il carattere `_`, come si può osservare negli esempi seguenti.

Il logo “ \LaTeX ” si ottiene
con il comando `\verb!\LaTeX!`.

Il logo “ \LaTeX ” si ottiene con il comando
`\LaTeX`.

```
\begin{verbatim}
Nell'ambiente verbatim
i comandi di \LaTeX,
gli
a capo, gli      spazi,
    i rientri e i
caratteri speciali (\{}%$_&\#^~)
non vengono interpretati.
\end{verbatim}
```

Nell'ambiente verbatim
i comandi di `\LaTeX`,
gli
a capo, gli spazi,
 i rientri e i
caratteri speciali (`\{}%$_&\#^~`)
non vengono interpretati.

```
\begin{verbatim*}
Il comando \verb*
e l'ambiente verbatim*
mostrano gli spazi così.
\end{verbatim*}
```

Il `_` comando `_` `\verb*`
e `_` l'ambiente `_` `verbatim*`
mostrano `_` gli `_` spazi `_` così.

Questi strumenti generalmente riescono a soddisfare le esigenze più comuni, ma non si possono personalizzare in alcun modo (con colori e sfondi particolari, riquadri, possibilità di definire ambienti e linguaggi personali, per esempio). Si rimanda chi ne avesse bisogno al pacchetto `listings`.

Capitolo 5

Matematica

Questo capitolo esplora uno dei principali punti di forza di \LaTeX , anche se ne intacca solamente la superficie: la composizione di formule matematiche. Si spiegheranno strumenti sufficienti per la *maggior parte* delle esigenze, ma potrebbe darsi che *quella particolare* necessità non trovi risposta in queste pagine. Se così fosse, molto probabilmente la soluzione sta in una delle funzioni del pacchetto `amsmath` (che non potrà essere descritto per intero, dati i limiti di questo lavoro) o di qualche altro pacchetto dedicato.

Di qui in avanti si danno per caricati i pacchetti `amsmath` e `amssymb`.

5.1 Formule in linea e in display

In generale, nelle formule matematiche le variabili vengono rese in *corsivo matematico*, diverso dal *corsivo ordinario*.

Con \LaTeX si può scrivere la matematica in due modi: “in linea” e “in display”.

5.1.1 Formule in linea

Una formula “in linea” è incorporata nel testo: $\lim_{n \rightarrow \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$, per esempio. Come si può osservare, \LaTeX fa *il possibile* per comprimerla e modificare meno che può l’interlinea nel capoverso che la contiene. Non ci si preoccupi se con questa modalità di scrittura elementi che di solito si trovano sopra o sotto un simbolo gli compaiono accanto: è tipograficamente corretto. L’esempio seguente mostra come si scrive una formula di questo tipo:

Una formula in linea è incorporata nel testo:
`\lim_{n \to \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}`.
`\LaTeX` modifica il meno possibile l’interlinea del capoverso.

Una formula in linea è incorporata nel testo: $\lim_{n \rightarrow \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$. \LaTeX modifica il meno possibile l’interlinea del capoverso.

Le formule in linea si scrivono tra dollari `$...$` (oppure, ma meno spesso, tra i comandi `\(...\)`) e si consiglia di usarle solo con espressioni di altezza contenuta come le seguenti:

Ci sono voluti secoli per dimostrare che quando $n > 2$ `\emph{non}` ci sono tre interi positivi a , b , c tali che $a^n + b^n = c^n$.

Ci sono voluti secoli per dimostrare che quando $n > 2$ *non* ci sono tre interi positivi a , b , c tali che $a^n + b^n = c^n$.

5.1.2 Formule in display

Una formula “in display”, invece, è un’espressione che \LaTeX compone su linee a sé, separate dal contesto con adeguati spazi bianchi per “metterla in mostra” e farla risaltare sulla pagina. L’esempio in linea del paragrafo precedente diventa, in display:

$$\lim_{n \rightarrow \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$$

Come si può osservare, ora la formula è centrata, non compressa, e tutti i suoi elementi occupano il giusto spazio con un risultato finale di grande respiro.

L’unico modo corretto per scrivere queste formule è usare uno dei due ambienti matematici seguenti:

- `equation` per le formule numerate;
- `equation*` (di solito abbreviato in `\[...\]`) per quelle non numerate.

Degli altri modi esistenti per farlo, oggi *non devono essere più usati*:

- I dollari doppi `$$...$$`, che potrebbero compromettere la corretta spaziatura verticale delle formule o il funzionamento dell’opzione di classe `fleqn` [Fairbairns, 2014].
- Gli ambienti standard `eqnarray` e `eqnarray*` (per sistemi di formule numerate e non numerate rispettivamente), perché prima e dopo = inseriscono più spazio del dovuto. È un difetto conservato in \LaTeX 2_ε per mantenerne la compatibilità con le vecchie versioni del programma.

Si considerino gli esempi seguenti:

Una formula in display è composta su linee a sé stanti:

```
\[
\lim_{n \to \infty}
\sum_{k=1}^n \frac{1}{k^2} =
\frac{\pi^2}{6}
\]
```

Una formula in display è composta su linee a sé stanti:

$$\lim_{n \rightarrow \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$$

Se f è continua e

```
\[
F(x)=\int_a^x f(t)\,dt
\]
```

allora

```
\begin{equation}
F'(x)=f(x)
\end{equation}
```

Se f è continua e

$$F(x) = \int_a^x f(t) dt$$

allora

$$F'(x) = f(x) \quad (5.1)$$

Si scrivono in display espressioni complesse e di grandi dimensioni (troppo sacrificate tra le righe di un capoverso) e formule più contenute a cui si voglia dare un risalto particolare.

I comandi `\label` ed `\eqref` permettono i riferimenti incrociati alle formule (come già visto nel paragrafo 3.10):

```
\begin{equation}
\label{eqn:eulero}
e^{i\pi}+1=0
\end{equation}
Dalla formula~\eqref{eqn:eulero}
si deduce che\dots
```

$$e^{i\pi} + 1 = 0 \quad (5.2)$$

Dalla formula (5.2) si deduce che...

5.1.3 Modo matematico e modo testuale

La modalità con cui si scrive la matematica (*modo matematico*) differisce per alcuni aspetti da quella con cui si scrive il testo (*modo testuale*). Ecco i principali.

- \LaTeX inserisce automaticamente gli spazi in base alla struttura della formula e ignora quelli che trova nel sorgente (interruzioni di riga comprese). Se serve, si possono inserire *a mano* ulteriori spazi con i comandi raccolti nella tabella 5.6.
- Nella scrittura delle formule *non sono ammesse righe vuote*.
- \LaTeX mette in corsivo matematico *tutte* le lettere che trova in una formula, considerando altrettante variabili. Per inserire in una formula in display un (breve) testo in tondo e spaziato normalmente si usa il comando `\text`, esplicitando la spaziatura prima e dopo.

I due esempi seguenti mostrano quanto si è appena descritto:

```
$x+y+z=n$ \\  
$ x + y + z = n $
```

$$x + y + z = n$$

$$x + y + z = n$$

```
\[  
z^2+1=0 \quad  
\text{per } z=\pm i\}  
\]
```

$$z^2 + 1 = 0 \quad \text{per } z = \pm i$$

Il comando `\pm` produce \pm (`\mp` produce \mp).

5.2 Nozioni introduttive

Questo paragrafo descrive i comandi più usati per scrivere le formule matematiche. (Ulteriori comandi sono raccolti nelle tabelle contenute nelle prossime sezioni.)

5.2.1 Raggruppamenti

La maggior parte dei comandi matematici agisce soltanto sul carattere immediatamente successivo. Si evita questo comportamento racchiudendo il testo interessato in un gruppo di parentesi graffe:

```
\[  
a^x+y \neq a^{x+y}  
\]
```

$$a^x + y \neq a^{x+y}$$

5.2.2 Esponenti, indici e radici

Apici e pedici si scrivono dopo i caratteri `^` e `_` rispettivamente:

```
$a_1$ \quad $x^2$ \quad  
$e^{-\alpha t}$ \quad $a^3_{ij}$
```

$$a_1 \quad x^2 \quad e^{-\alpha t} \quad a_{ij}^3$$

Il comando `\quad` produce uno spazio orizzontale di un *quadrato*.

Gli indici di secondo ordine vanno messi in un gruppo di graffe insieme a quelli di ordine superiore: una scrittura come `x_n_k` non ha senso.

Dalla successione x_n estrarre x_{n_k} .

Dalla successione x_n estrarre x_{n_k} .

Il simbolo di radice quadrata si ottiene con `\sqrt`, quello di radice n -esima con

`\sqrt[n]{...}`

L^AT_EX calcola automaticamente le dimensioni della radice:

`\[`
`\sqrt{x} \quad \sqrt{\frac{a}{b}}`
`\]`

$$\sqrt{x} \quad \sqrt{\frac{a}{b}}$$

5.2.3 Somme, prodotti e frazioni

Il simbolo di sommatoria è generato da `\sum` e quello di produttoria da `\prod`. Gli estremi si scrivono come indici.

Trova il massimo della funzione
`\[`
`f(x_1,\dots,x_n)=\prod_{k=1}^n x_k`
`\]`
 sotto la condizione
`\[`
`\sum_{k=1}^n x_k^2=1`
`\]`

Trova il massimo della funzione

$$f(x_1, \dots, x_n) = \prod_{k=1}^n x_k$$

sotto la condizione

$$\sum_{k=1}^n x_k^2 = 1$$

Una frazione, anche complessa, si ottiene semplicemente con il comando

`\frac{<numeratore>}{<denominatore>}`

Per piccole quantità di materiale frazionario, a volte la forma n/m è più gradevole sulla pagina:

`\[`
`\frac{1}{x^2+1} \quad x^{1/2}`
`\]`

$$\frac{1}{x^2+1} \quad x^{1/2}$$

5.2.4 Limiti, derivate e integrali

Il comando

`\lim_{<variabile> \to <valore>}`

produce i limiti, e `\infty` produce ∞ .

`\[`
`\lim_{x \to 0} \frac{\sin x}{x} = 1`
`\quad`
`\lim_{n \to +\infty} f_n = \delta`
`\]`

$$\lim_{x \rightarrow 0} \frac{\sin x}{x} = 1 \quad \lim_{n \rightarrow +\infty} f_n = \delta$$

Le derivate si scrivono con il carattere `'`, che produce il segno di *primo*.

`\[`
`y=x^2 \quad y'=2x \quad y''=2.`
`\]`

$$y = x^2 \quad y' = 2x \quad y'' = 2.$$

Tabella 5.1: Simboli insiemistici (\bigcup e \bigcap sono operatori)

\subset	<code>\subset</code>	\supset	<code>\supset</code>	\cup	<code>\cup</code>
\subseteq	<code>\subseteq</code>	\supseteq	<code>\supseteq</code>	\cap	<code>\cap</code>
\in	<code>\in</code>	\ni	<code>\ni</code>	\notin	<code>\notin</code>
\complement	<code>\complement</code>	\setminus	<code>\setminus</code>	\emptyset	<code>\emptyset</code>
\bigcup	<code>\bigcup</code>	\bigcap	<code>\bigcap</code>		

Il comando `\int` produce il simbolo di integrale. Gli estremi di integrazione si scrivono come indici, e un indice formato da più di una lettera o una cifra va messo tra parentesi graffe.

```
\[
\int_a^{a+T} f(x) \, dx =
\int_0^T f(x) \, dx
\]
```

$$\int_a^{a+T} f(x) \, dx = \int_0^T f(x) \, dx$$

Come si può osservare, lo spazio sottile `\,`, allontana dx da $f(x)$.

Per gli integrali multipli ci sono i comandi `\iint`, `\iiint`, `\iiiint` e `\idotsint`.

```
\[
\iint_D f(x,y) \, dx \, dy \quad \iint\limits_D f(x,y) \, dx \, dy
\iiint g \, dx \, dy \, dz
\]
```

$$\iint_D f(x,y) \, dx \, dy \quad \iiint g \, dx \, dy \, dz$$

Per gli integrali curvilinei lungo curve chiuse si usa `\oint`:

```
\[
\oint f(z) \, dz = 2\pi i
\]
```

$$\oint f(z) \, dz = 2\pi i$$

5.2.5 Insiemi numerici

I simboli degli insiemi numerici si ottengono con `\mathbb` (*blackboard bold*, “nero da lavagna”).

```
\[
x \in \mathbb{R} \quad z \in \mathbb{C}
\]
```

$$x \in \mathbb{R} \quad z \in \mathbb{C}$$

I simboli usati nell’esempio precedente sono raccolti insieme ad altri simboli insiemistici nella tabella 5.1.

Se si scrivono nel preambolo le definizioni seguenti (si veda il paragrafo 9.1)

```
\newcommand{\numberset}{\mathbb}
\newcommand{\N}{\numberset{N}}
\newcommand{\R}{\numberset{R}}
```

per avere \mathbb{N} basta scrivere `\N`, e si può cambiare notazione con un’unica modifica.

Tabella 5.2: Lettere greche

α	<code>\alpha</code>	κ	<code>\kappa</code>	ς	<code>\varsigma</code>
β	<code>\beta</code>	λ	<code>\lambda</code>	τ	<code>\tau</code>
γ	<code>\gamma</code>	Λ	<code>\Lambda</code>	υ	<code>\upsilon</code>
Γ	<code>\Gamma</code>	μ	<code>\mu</code>	Υ	<code>\Upsilon</code>
δ	<code>\delta</code>	ν	<code>\nu</code>	ϕ	<code>\phi</code>
Δ	<code>\Delta</code>	ξ	<code>\xi</code>	Φ	<code>\Phi</code>
ϵ	<code>\epsilon</code>	Ξ	<code>\Xi</code>	φ	<code>\varphi</code>
ε	<code>\varepsilon</code>	π	<code>\pi</code>	χ	<code>\chi</code>
ζ	<code>\zeta</code>	Π	<code>\Pi</code>	ψ	<code>\psi</code>
η	<code>\eta</code>	ϖ	<code>\varpi</code>	Ψ	<code>\Psi</code>
θ	<code>\theta</code>	ρ	<code>\rho</code>	ω	<code>\omega</code>
Θ	<code>\Theta</code>	ϱ	<code>\varrho</code>	Ω	<code>\Omega</code>
ϑ	<code>\vartheta</code>	σ	<code>\sigma</code>		
ι	<code>\iota</code>	Σ	<code>\Sigma</code>		

5.2.6 Lettere greche

Le lettere greche minuscole e maiuscole si ottengono con i comandi elencati nella tabella 5.2 (per l'omicron minuscolo si usa il carattere latino *o*, e per le maiuscole che non vi compaiono si usano le corrispondenti maiuscole latine, identiche a quelle greche). Per le sei di esse prefissate con `var-` si noti quanto segue:

- `\varpi` e `\varsigma` non si usano praticamente mai;
- le altre quattro vanno usate in modo esclusivo: *o* la forma principale o la sua variante (in Europa si usa generalmente la seconda).

È perciò conveniente ridefinire queste quattro varianti come caratteri normali, scrivendo nel preambolo (si veda il paragrafo 9.1):

```
\renewcommand{\epsilon}{\varepsilon}
\renewcommand{\theta}{\vartheta}
\renewcommand{\rho}{\varrho}
\renewcommand{\phi}{\varphi}
```

5.2.7 Simboli che sormontano altri simboli

Il comando

```
\overset{\langle primo argomento \rangle}{\langle secondo argomento \rangle}
```

produce il simbolo indicato nel *primo argomento* rimpicciolito e sovrapposto a quello scritto nel *secondo argomento* (di solito un simbolo di relazione binaria), che rimane delle sue dimensioni e nella posizione abituale. Il comando `\underset` fa l'opposto.

Il simbolo

```
\[
```

```
\overset{R}{\sim}
```

```
\]
```

indica un'equivalenza rispetto a R .

Il simbolo

 $\overset{R}{\sim}$

indica un'equivalenza rispetto a R .

Tabella 5.3: Accenti in modo matematico

\bar{a}	<code>\bar{a}</code>	\hat{a}	<code>\hat{a}</code>	\tilde{a}	<code>\tilde{a}</code>
\vec{a}	<code>\vec{a}</code>	\dot{a}	<code>\dot{a}</code>	\ddot{a}	<code>\ddot{a}</code>
\check{a}	<code>\check{a}</code>	\widehat{abc}	<code>\widehat{abc}</code>	\widetilde{abc}	<code>\widetilde{abc}</code>

5.2.8 Barre e accenti

Il comando `\bar` pone un trattino sul proprio argomento: il simbolo \bar{x} indica un nome di variabile distinto da x (si veda la tabella 5.3).

I comandi `\overline` e `\underline` (il secondo dei quali si usa piuttosto raramente) sopralineano e sottolineano rispettivamente tutto il proprio argomento: il simbolo \overline{x} indica un operatore applicato alla variabile x :

<code>\bar{x}</code>	\bar{x}	<code>\bar{X}</code>	\bar{X}	<code>\overline{m+n}</code>	$\overline{m+n}$
----------------------	-----------	----------------------	-----------	-----------------------------	------------------

I comandi `\vec` e `\overrightarrow` agiscono come `\bar` e `\overline`, ma producono frecce anziché barre orizzontali.

<code>\vec{x}</code>	\vec{x}	<code>\overrightarrow{AB}</code>	\overrightarrow{AB}
----------------------	-----------	----------------------------------	-----------------------

Esistono tre tipi di barra verticale, distinguibili dallo spazio richiesto prima e dopo:

- semplice `|` (ottenibile anche con `\vert`);
- delimitatore sinistro e destro (`\lvert` e `\rvert` rispettivamente);
- relazione binaria (`\mid`) per la divisibilità e il *tale che* negli insiemi.

<code>F(x) _{x=\gamma(t)}</code>	$F(x) _{x=\gamma(t)}$	<code> x </code>	$ x $
<code>\lvert x \rvert</code>	$\lvert x \rvert$	<code>\mid</code>	\mid
Se <code>p \mid n^2</code> , allora <code>p \mid n</code> .		Se <code>p \mid n^2</code> , allora <code>p \mid n</code> .	

I comandi appena esaminati prevedono forme analoghe per barre verticali doppie: `\lVert` (o `\lvert`), `\rVert`, `\rvert` e `\parallel`.

La barra laterale è prevista talvolta nel calcolo degli integrali:

<code>\int_a^b f(x) dx</code>	$\int_a^b f(x) dx$	<code>F(x) _a^b</code>	$F(x) _a^b$
-------------------------------	--------------------	------------------------	-------------

La sua altezza va regolata a mano, premettendole uno dei comandi che verranno descritti nel paragrafo 5.4.

Per aggiungere alle variabili un accento matematico, come un cappello o una tilde, si possono usare i comandi della tabella 5.3. I comandi `\widehat` e `\widetilde` producono rispettivamente simboli di cappello e tilde che sormontano tutto il proprio argomento (di tre lettere al massimo).

5.2.9 Punti, frecce e simboli logici

In matematica esistono due tipi di due punti, distinguibili dal diverso spazio richiesto prima e dopo:

- semplice `:`, spaziato come in un'operazione binaria (divisione);

Tabella 5.6: Spazi in modo matematico

Comando	Tipo di spazio
<code>\,</code>	Spazio sottile positivo
<code>\!</code>	Spazio sottile negativo
<code>\quad</code>	Spazio di un <i>quadrato</i>
<code>\qquad</code>	Spazio di un <i>quadrato</i> ne

5.2.10 Spazi in modo matematico

Può accadere, anche se di rado, che la spaziatura scelta da \LaTeX per le formule risulti insoddisfacente. Per modificarla si usano i comandi raccolti nella tabella 5.6.

5.3 Operatori

5.3.1 Caratteristiche generali

In \LaTeX , le funzioni come \sin , \cos e \log presentano le seguenti caratteristiche:

- per essere più visibili sulla pagina (in accordo con le norme ISO-UNI) vengono rese in tondo normale e non in corsivo matematico come le variabili;
- richiedono una particolare spaziatura prima e dopo, che il programma inserisce automaticamente;
- i comandi che le producono, come `\sin`, `\cos` e `\log`, sono detti *operatori*.

Li si vede all'opera negli esempi seguenti:

<code>\[</code>			
<code>\sin2x \qquad \log\log x \qquad</code>	$\sin 2x$	$\log \log x$	$\log(x + y)$
<code>\log(x+y)</code>			
<code>\]</code>			

Si osservi che:

- nella prima formula, fra \sin e 2 c'è più spazio che fra 2 e x ;
- nella seconda i tre elementi sono separati da uno spazio sottile;
- nella terza non c'è alcuno spazio tra \log e la parentesi.

L'unico modo corretto di scriverli è quello appena mostrato: omettendo la barra rovescia si otterrebbe " $\sin 2x$ ", tutto in corsivo matematico e senza alcuna spaziatura, che in matematica non significa nulla. Soltanto scrivendo gli operatori come si è appena mostrato \LaTeX si comporta nel giusto modo e assegna loro font e spazi corretti.

I seguenti sono alcuni operatori predefiniti (la tabella 5.7 ne riporta l'elenco completo):

<code>\arccos x\$, \$\exp x\$, \$\log x\$, \$\det A\$, \$\min_{x \in A} f(x)\$</code>	$\arccos x$, $\exp x$, $\log x$, $\det A$, $\min_{x \in A} f(x)$
---	--

I due comandi `\bmod` e `\pmod` riguardano la relazione di congruenza modulo m :

<code>\$a\bmod b\$ \quad \$a\equiv b \pmod{m}\$</code>	$a \bmod b$ $a \equiv b \pmod{m}$
--	-----------------------------------

Tabella 5.7: Operatori predefiniti

<code>\min</code>	<code>\max</code>	<code>\inf</code>	<code>\sup</code>	<code>\gcd</code>	<code>\arg</code>
<code>\sin</code>	<code>\cos</code>	<code>\tan</code>	<code>\cot</code>	<code>\sec</code>	<code>\csc</code>
<code>\sinh</code>	<code>\cosh</code>	<code>\tanh</code>	<code>\coth</code>	<code>\exp</code>	<code>\lim</code>
<code>\arcsin</code>	<code>\arccos</code>	<code>\arctan</code>	<code>\log</code>	<code>\lg</code>	<code>\ln</code>
<code>\liminf</code>	<code>\limsup</code>	<code>\deg</code>	<code>\det</code>	<code>\dim</code>	<code>\hom</code>
<code>\ker</code>	<code>\Pr</code>				

5.3.2 Definire nuovi operatori

Le pubblicazioni specialistiche introducono continuamente nuove funzioni che devono poter essere definite, non essendo contemplate né da \LaTeX né dai pacchetti dedicati. Risolve il problema il comando `\DeclareMathOperator`.

Per esempio, per definire la funzione matematica “sgn” che denoti il segno di un numero reale (funzione non prevista né da \LaTeX né da `amsmath`), si scrive nel preambolo

```
\DeclareMathOperator{\sgn}{sgn}
```

Nel documento, poi, si darà `\sgn` per ottenere “sgn” nel font corretto e adeguatamente spaziato su entrambi i lati.

5.4 Parentesi

\LaTeX e `amsmath` definiscono numerosi simboli per parentesi e altri delimitatori. Le parentesi tonde e quadre si scrivono con i corrispondenti caratteri da tastiera, mentre quelle graffe *anche in modo matematico* devono essere precedute da `\`. Tutti gli altri delimitatori vengono generati da comandi dedicati.

<code>\[</code> <code>{a,b,c}\ne\{a,b,c\}</code> <code>\]</code>	$a, b, c \neq \{a, b, c\}$
--	----------------------------

Talvolta bisogna aggiustarne a mano le dimensioni: lo si può fare prefissandoli con i comandi `\big`, `\Big`, `\bigg` e `\Bigg`. I comandi `\bigl` (*big left*) e `\bigr` (*big right*) ingrandiscono lievemente le parentesi:

<code>\[</code> <code>\bigl((x-y)+(x+y) \bigr)</code> <code>\]</code>	$((x - y) + (x + y))$
--	-----------------------

I comandi `\Bigl` e `\Bigr` producono parentesi ancora più grandi:

<code>\[</code> <code>\Bigl(1+\frac{1}{n}\Bigr)^n</code> <code>\]</code>	$\left(1 + \frac{1}{n}\right)^n$
--	----------------------------------

I comandi `\biggl` e `\biggr` ne generano di più grandi ancora:

<code>\[</code> <code>\biggl(\sum_n x_n^2\biggr)^{1/2}</code> <code>\]</code>	$\left(\sum_n x_n^2\right)^{1/2}$
---	-----------------------------------

Se non basta, ci sono anche `\Biggl` e `\Biggr`.

I comandi `\overbrace` e `\underbrace` creano lunghe graffe orizzontali sopra o sotto un'espressione:

```
\[
\underbrace{1+2+3}_{6} + 4
\]
```

$$\underbrace{1+2+3}_{6}+4$$

Per scrivere coefficienti binomiali si usa il comando `\binom`:

```
\[
(a+b)^n=\sum_{\substack{k \\ 0\leq k\leq n}} \binom{n}{k} a^{n-k} b^k
\]
```

$$(a+b)^n = \sum_{0 \leq k \leq n} \binom{n}{k} a^{n-k} b^k$$

Il comando

```
\substack{\langle sopra \rangle \\ \langle sotto \rangle}
```

produce un indice su più righe.

Per i sistemi di equazioni si può usare l'ambiente `cases` (si veda il paragrafo 5.7.3):

```
\[
\begin{cases}
x+y=2 \\
x-y=0
\end{cases}
\]
```

$$\begin{cases} x+y=2 \\ x-y=0 \end{cases}$$

5.5 Vettori e matrici

I vettori si scrivono di solito in tondo nero (corsivo, secondo le norme ISO-UNI) oppure in semplice corsivo matematico; talvolta, soprattutto nei testi di fisica, sono sormontati da una freccia. Nel primo caso si può usare il comando `\mathbf`; nel secondo il comando `\boldsymbol`; nel terzo il comando `\vec`. Può essere conveniente ridefinire nel preambolo quest'ultimo comando (si veda il paragrafo 9.1):

```
\renewcommand{\vec}{\boldsymbol}
```

In questo modo basta scrivere `\vec{v}` per ottenere v e si può cambiare notazione con un'unica modifica (si veda anche il paragrafo 5.8).

Le matrici si scrivono negli ambienti `pmatrix`, `bmatrix`, `Bmatrix`, `vmatrix` e `Vmatrix`, che hanno come delimitatori rispettivamente parentesi tonde, quadre (*braces*), graffe (*curly braces*), barre verticali e doppie barre verticali. Esiste anche l'ambiente `matrix` senza delimitatori.

Gli elementi della matrice vengono centrati automaticamente, e righe e colonne si scrivono come una normale tabella `tabular`, ricordando che gli spazi espliciti sono ignorati.

```
\[
\begin{pmatrix}
1 & 2 \\
3 & 4
\end{pmatrix}
\]
```

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

Tabella 5.8: Simboli misti (`\bigodot`, `\bigoplus` e `\bigotimes` sono operatori)

\vee	<code>\vee</code>	\wedge	<code>\wedge</code>	\div	<code>\div</code>
\Re	<code>\Re</code>	\Im	<code>\Im</code>	\aleph	<code>\aleph</code>
∂	<code>\partial</code>	∇	<code>\nabla</code>	\cdot	<code>\cdot</code>
\hbar	<code>\hbar</code>	\hslash	<code>\hslash</code>	\circ	<code>\circ</code>
\imath	<code>\imath</code>	\jmath	<code>\jmath</code>	\bullet	<code>\bullet</code>
ℓ	<code>\ell</code>	\wp	<code>\wp</code>	$\sqrt{}$	<code>\sqrt{}</code>
\dagger	<code>\dagger</code>	\ddagger	<code>\ddagger</code>	$*$	<code>*</code>
\vdash	<code>\vdash</code>	\dashv	<code>\dashv</code>	\angle	<code>\angle</code>
\triangleleft	<code>\triangleleft</code>	\triangleright	<code>\triangleright</code>	\triangle	<code>\triangle</code>
\square	<code>\square</code>	\blacksquare	<code>\blacksquare</code>	\diamond	<code>\diamond</code>
\odot	<code>\odot</code>	\oplus	<code>\oplus</code>	\otimes	<code>\otimes</code>
\bigodot	<code>\bigodot</code>	\bigoplus	<code>\bigoplus</code>	\bigotimes	<code>\bigotimes</code>

```
\[
\begin{bmatrix}
1 & 2 \\
3 & 4
\end{bmatrix}
\end{bmatrix}
```

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

Si possono scrivere matrici con punti ellittici, come nell'esempio seguente:

```
\[
A=
\begin{bmatrix}
x_{11} & x_{12} & \dots \\
x_{21} & x_{22} & \dots \\
\vdots & \vdots & \ddots
\end{bmatrix}
\end{bmatrix}
```

$$A = \begin{bmatrix} x_{11} & x_{12} & \dots \\ x_{21} & x_{22} & \dots \\ \vdots & \vdots & \ddots \end{bmatrix}$$

Come si può osservare, il comando `\vdots` produce tre punti ellittici verticali e `\ddots` tre in diagonale.

Il comando

```
\hdotsfor{<n>}
```

riempie di punti la riga della matrice per $\langle n \rangle$ colonne:

```
\[
\begin{bmatrix}
a_{11} & \dots & a_{1n} \\
a_{21} & \dots & a_{2n} \\
\hdotsfor{3} \\
a_{n1} & \dots & a_{nn}
\end{bmatrix}
\end{bmatrix}
```

$$\begin{bmatrix} a_{11} & \dots & a_{1n} \\ a_{21} & \dots & a_{2n} \\ \dots & \dots & \dots \\ a_{n1} & \dots & a_{nn} \end{bmatrix}$$

La tabella 5.8 raccoglie altri simboli matematici di uso comune. Per un loro elenco completo si veda [Pakin, 2015].

Tabella 5.9: Simboli di relazione

\leq	<code>\le</code>	\geq	<code>\ge</code>	\sim	<code>\sim</code>
\ll	<code>\ll</code>	\gg	<code>\gg</code>	\simeq	<code>\simeq</code>
\prec	<code>\prec</code>	\succ	<code>\succ</code>	\approx	<code>\approx</code>
\preceq	<code>\preceq</code>	\succeq	<code>\succeq</code>	\asymp	<code>\asymp</code>
\parallel	<code>\parallel</code>	\perp	<code>\perp</code>	\cong	<code>\cong</code>
\mid	<code>\mid</code>	\propto	<code>\propto</code>	\equiv	<code>\equiv</code>
\neq	<code>\ne</code>				

5.6 Spezzare formule lunghe

\LaTeX non spezza automaticamente una formula più lunga d'una riga. Solo chi l'ha scritta, infatti, ne conosce il ritmo di lettura e sa dov'è più opportuno andare a capo e se allinearne o meno le varie parti.

Per spezzare e raggruppare formule in display, il pacchetto `amsmath` definisce (fra gli altri) gli ambienti `multline`, `split`, `gather` e `align`, che si descrivono di seguito.

5.6.1 Spezzare formule senza incolonnarle

Per spezzare una formula in più righe non incolonnate si usa l'ambiente `multline`.

```
\begin{multline}
f=a+b+c \\
+d+e+g+h \\
+r+s+t
\end{multline}
```

$$\begin{aligned}
 f &= a + b + c \\
 &\quad + d + e + g + h \\
 &\quad + r + s + t \quad (5.3)
 \end{aligned}$$

Si noti che:

- la prima riga viene allineata a sinistra e l'ultima a destra;
- le rimanenti vengono centrate;
- il numero progressivo della formula viene messo nel margine destro in corrispondenza dell'ultima riga.

La variante asterisco `multline*` produce formule dello stesso tipo *non numerate*.

5.6.2 Spezzare formule incolonnandole

Per spezzare una formula in più righe incolonnate si usa l'ambiente `split`.

```
\begin{equation}
\begin{split}
a &= b+c-d \\
&= e-f \\
&= g
\end{split}
\end{equation}
```

$$\begin{aligned}
 a &= b + c - d \\
 &= e - f \\
 &= g
 \end{aligned} \quad (5.4)$$

Come si può osservare:

- il carattere `&` incolonna le righe della formula a partire dal punto in cui viene dato (di solito subito prima di un `=`);
- `split` va *necessariamente* usato dentro un altro ambiente per la matematica in display.

5.7 Raggruppare più formule

Per raggruppare più formule in display, il pacchetto `amsmath` definisce (fra gli altri) gli ambienti `gather` e `align`, descritti di seguito.

5.7.1 Raggruppare formule senza incolonnarle

L'ambiente `gather` raggruppa più formule centrandole e numerando autonomamente ciascuna su una riga a sé e, se necessario, assegnandole un'etichetta.

<code>\begin{gather}</code>		
<code>a=b+c \\</code>	$a = b + c$	(5.5)
<code>V+F-S=2</code>	$V + F - S = 2$	(5.6)
<code>\end{gather}</code>		

Si noti che si avrebbe un risultato simile scrivendo ogni formula in un ambiente `equation`, ma poi lo spazio tra di esse sarebbe esagerato. La variante asterisco `gather*` produce formule dello stesso tipo non numerate.

5.7.2 Raggruppare formule incolonnandole

L'ambiente `align` incolonna gruppi di due o più formule mettendo e numerando ciascuna su una riga a sé, come mostra l'esempio seguente:

<code>\begin{align}</code>		
<code>a &= b+c+d \\</code>	$a = b + c + d$	(5.7)
<code>e &= f \notag \\</code>	$e = f$	
<code>x-1 &= y+z</code>	$x - 1 = y + z$	(5.8)
<code>\end{align}</code>		

La variante asterisco `align*` produce formule dello stesso tipo non numerate. (Si ottiene lo stesso risultato dando `\notag` alla fine della formula interessata).

L'ambiente è utile anche per incolonnare più righe di formule autonome. In tal caso, `&` assume due significati diversi a seconda della posizione in cui si trova sulla riga:

- se occupa un posto dispari, indica a \LaTeX i punti da incolonnare;
- se occupa un posto pari, è un separatore come in `tabular`.

<code>\begin{align}</code>			
<code>a &= b & c &=d & e &=f \\</code>	$a = b$	$c = d$	$e = f$
<code>u &= v & w &=x & y &=z</code>	$u = v$	$w = x$	$y = z$
<code>\end{align}</code>			

5.7.3 Casi

L'ambiente `cases` serve per le definizioni fatte per casi. Graffa e allineamento sono automatici; il testo nella seconda colonna va nell'argomento di `\text`.

<code>\[</code>	
<code>n!=</code>	
<code>\begin{cases}</code>	
<code>1 & \text{se } \\$n=0\\$ \\</code>	$n! = \begin{cases} 1 & \text{se } n = 0 \\ n(n-1)! & \text{se } n \geq 1 \end{cases}$
<code>n(n-1)! & \text{se } \\$n\geq 1\\$</code>	
<code>\end{cases}</code>	
<code>\]</code>	

Tabella 5.10: Stili dei font matematici (`\mathscr` richiede il pacchetto `mathrsfs`)

Stile	Codice	Risultato
Tondo	<code>\mathrm{ABCdef123}</code>	ABCdef123
Corsivo	<code>\mathit{ABCdef123}</code>	<i>ABCdef123</i>
Nero	<code>\mathbf{ABCdef123}</code>	ABCdef123
Dattilografico	<code>\mathtt{ABCdef123}</code>	ABCdef123
Senza grazie	<code>\mathsf{ABCdef123}</code>	ABCdef123
Gotico	<code>\mathfrak{ABCdef123}</code>	<mathfrak{abcdef123}< math=""></mathfrak{abcdef123}<>
Nero da lavagna	<code>\mathbb{ABC}</code>	\mathbb{ABC}
Calligrafico	<code>\mathcal{ABC}</code>	\mathcal{ABC}
Manoscritto	<code>\mathscr{ABC}</code>	\mathscr{ABC}

5.8 Modificare stile e corpo del font

In modo matematico, \LaTeX armonizza stile e corpo del font con il contesto in cui le formule si trovano. A volte, però, può essere necessario modificare a mano questi due parametri: questo paragrafo spiega come farlo.

Modificare lo stile

La tabella 5.10 raccoglie i comandi per cambiare lo stile del font (che agiscono su lettere e numeri, ma *non* sui segni di operazione).

Per comporre simboli matematici in nero corsivo si consiglia il comando `\boldsymbol`.

<code>\[</code> <code>\mu, M \quad</code> <code>\boldsymbol{\mu}, \boldsymbol{M}</code> <code>\]</code>	μ, M $\boldsymbol{\mu}, \boldsymbol{M}$
--	---

Gli indici letterali vanno scritti in corsivo matematico se rappresentano quantità variabili (cioè se sono *simboli*), in tondo se rappresentano apposizioni di una grandezza fisica (cioè se sono semplice *testo*). In quest'ultimo caso si usa il comando `\textup`.

<code>\[</code> <code>V_{\textup{eff}} \quad</code> <code>\psi_{\textup{incidente}}</code> <code>\]</code>	$V_{\textup{eff}}$ $\psi_{\textup{incidente}}$
---	--

Si confrontino le due scritture seguenti:

<code>\[</code> <code>\$V_{\textup{eff}}\$ \quad \$V_{\textup{eff}}\$</code> <code>\]</code>	$V_{\textup{eff}}$ $V_{\textup{eff}}$
--	---------------------------------------

Come si può osservare, \LaTeX interpreta il pedice nel primo codice come tre variabili da moltiplicare, rendendole in corsivo matematico e spaziandole di conseguenza. Il codice corretto è il secondo.

Modificare il corpo

In modo matematico si può impostare a mano la dimensione del font con le dichiarazioni `\displaystyle`, `\textstyle`, `\scriptstyle` e `\scriptscriptstyle`. Si considerino gli esempi seguenti:

```


$$\sum_{k=1}^n z^k \quad \quad \quad \sum_{k=1}^n z^k \quad \quad \quad \sum_{k=1}^n z^k$$


```

$$\sum_{k=1}^n z^k \quad \quad \quad \sum_{k=1}^n z^k \quad \quad \quad \sum_{k=1}^n z^k$$

```

\[\[
x_G = \frac{\sum_{i=1}^n m_i x_i}{\sum_{i=1}^n m_i}
\]]

```

$$x_G = \frac{\sum_{i=1}^n m_i x_i}{\sum_{i=1}^n m_i}$$

Come si può osservare, il cambiamento del corpo influisce anche sul modo in cui vengono resi gli indici.

5.9 Enunciati e dimostrazioni

Di qui in avanti si dà per caricato *anche* il pacchetto `amsthm`.

5.9.1 Enunciati

Nella scrittura della matematica è utile poter disporre di un metodo per introdurre e numerare definizioni, teoremi e strutture simili. I tipi di enunciato non sono predefiniti, ma vanno dichiarati dall'utente, chiamato a prendere alcune decisioni globali:

- stabilire il tipo di enunciato da inserire (per esempio, definizioni e teoremi);
- assegnare un nome a ogni ambiente (per esempio, definizione e teorema)
- titolare gli enunciati (per esempio, con *Definizione* e *Teorema*).

Il comando `\newtheorem`, dato *nel preambolo*, permette di fare le relative dichiarazioni globali e prevede due forme di definizione:

```
\newtheorem{<nome dell'enunciato>}{<titolo>}[<sezione>]
```

oppure, in alternativa,

```
\newtheorem{<nome dell'enunciato>}[<numerato come>]{<titolo>}
```

dove:

- `<nome dell'enunciato>` è una parola chiave che identifica l'enunciato;
- `<titolo>` specifica il titolo dell'enunciato che comparirà nel documento;
- `<sezione>` specifica il livello di sezionamento (chapter o section, di regola) a cui collegare la numerazione dell'enunciato;
- in `<numerato come>` si scrive il nome di un enunciato dichiarato in precedenza, in modo che quello nuovo ne prosegua la numerazione.

La variante asterisco `\newtheorem*` produce enunciati non numerati.

Il testo dell'enunciato va messo nel corrispondente ambiente, e un'eventuale specificazione (fra parentesi tonde, nel documento finito) si scrive nell'argomento facoltativo immediatamente *dopo* il comando d'apertura, così:

```
\begin{⟨nome dell'enunciato⟩}[⟨eventuale specificazione⟩]
...
\end{⟨nome dell'enunciato⟩}
```

Il pacchetto `amsthm` fornisce tre stili predefiniti per gli enunciati (`plain`, `definition` e `remark`) i cui dettagli tipografici dipendono dalla classe di documento in uso, anche se in linea di massima il primo dei tre produce il proprio contenuto in corsivo, mentre gli altri due lo lasciano in tondo. Di seguito si riportano le principali categorie di enunciato, ciascuna associata al proprio stile più tipico:

`plain` Per teoremi, lemmi, corollari, proposizioni, congetture, criteri, leggi e algoritmi.

`definition` Per definizioni, condizioni, problemi ed esempi.

`remark` Per osservazioni e annotazioni.

A questo punto le nozioni teoriche dovrebbero bastare. Alcuni esempi mostreranno quanto si è appena esaminato. Scrivendo nel preambolo

```
\theoremstyle{definition}
\newtheorem{definizione}{Definizione}
```

e

```
\theoremstyle{plain}
\newtheorem{teorema}{Teorema}
```

gli ambienti `definizione` e `teorema` si usano così:

```
\begin{definizione}[di Gauss]
Si dice \emph{matematico} colui
per il quale è ovvio che

$$\int_{-\infty}^{+\infty} e^{-x^2} dx = \sqrt{\pi}.$$

\end{definizione}
\begin{teorema}
I matematici sono molto rari.
\end{teorema}
```

Definizione 1 (di Gauss). Si dice *matematico* colui per il quale è ovvio che $\int_{-\infty}^{+\infty} e^{-x^2} dx = \sqrt{\pi}$.

Teorema 1. *I matematici sono molto rari.*

```
Il seguente teorema è noto a tutti.
\begin{teorema}[di Pitagora]
In un triangolo rettangolo, la somma
dei quadrati costruiti sui cateti è
uguale al quadrato costruito
sull'ipotenusa.
\end{teorema}
La dimostrazione è lasciata
per esercizio.
```

Il seguente teorema è noto a tutti.

Teorema 2 (di Pitagora). *In un triangolo rettangolo, la somma dei quadrati costruiti sui cateti è uguale al quadrato costruito sull'ipotenusa.*

La dimostrazione è lasciata per esercizio.

Come si può osservare, \LaTeX :

- produce etichetta e numero (automatico) dell'enunciato in tondo nero e la conclude con un punto fermo;
- separa ogni enunciato dal resto del testo senza rientrarlo;
- mette le definizioni in tondo e i teoremi in corsivo.

L'esempio seguente riguarda la numerazione di tre enunciati consecutivi, il primo e il terzo dei quali sono dello stesso tipo, ma il secondo no. Scrivendo nel preambolo

```
\theoremstyle{plain}
\newtheorem{legge}{Legge}
\newtheorem{decreto}[legge]{Decreto}
```

gli ambienti `legge` e `decreto` si usano come segue:

```
\begin{legge}
\label{lex:capo}
Il capo ha ragione.
\end{legge}
\begin{decreto}[Aggiornamento
alla legge~\ref{lex:capo}]
Il capo ha \emph{sempre} ragione.
\end{decreto}
\begin{legge}
Se il capo ha torto, vedere la
legge~\ref{lex:capo}.
\end{legge}
```

Legge 1. *Il capo ha ragione.*

Decreto 2 (Aggiornamento alla legge 1). *Il capo ha sempre ragione.*

Legge 3. *Se il capo ha torto, vedere la legge 1.*

Si osservi che il numero assegnato a *Decreto* prosegue la numerazione di *Legge* anziché cominciarne una nuova, perché l'argomento facoltativo `[legge]` nella definizione del teorema assegna entrambi gli enunciati allo stesso contatore. Come di consueto, `\label` permette riferimenti incrociati anche a enunciati matematici.

Se si desidera introdurre un enunciato *Murphy*, per esempio, la cui numerazione sia collegata al paragrafo corrente, è sufficiente specificare nell'argomento facoltativo l'opzione `section`, in questo modo:

```
\newtheorem{murphy}{Murphy}[section]
```

L'ambiente `murphy` così definito si usa come al solito:

```
\begin{murphy}
Se qualcosa può andar male, lo farà.
\end{murphy}
```

Murphy 5.9.1. *Se qualcosa può andar male, lo farà.*

5.9.2 Dimostrazioni

L'ambiente `proof` permette di scrivere una dimostrazione, che nel documento finito sarà chiusa da un quadratino. Si osservi l'esempio seguente:

```
\begin{teorema}[di Euclide]
I numeri primi sono infiniti.
\end{teorema}
\begin{proof}
Per esercizio.
\end{proof}
```

Teorema 3 (di Euclide). *I numeri primi sono infiniti.*

Dimostrazione. Per esercizio. □

Per sostituire la scritta *Dimostrazione* con un'altra, per esempio *Soluzione*, basta scrivere

```
\begin{proof}[Soluzione]
```

Capitolo 6

Tabelle e figure

Tabelle e figure sono tra gli “oggetti” più problematici, perché non si possono spezzare su più pagine. In questo capitolo, che spiega come servirsene senza sorprese, si danno per caricati i pacchetti `booktabs` e `caption` per le tabelle, e `graphicx` per le figure.

6.1 Strumenti fondamentali

Per inserire tabelle e figure in un documento da comporre con \LaTeX esistono tre strumenti essenzialmente:

- l’ambiente standard `tabular`, per tabelle che contengono prevalentemente testo;
- l’ambiente standard `array`, per tabelle che contengono prevalentemente matematica;
- il comando `\includegraphics` definito dal pacchetto `graphicx`, per includere nel documento le figure quando sono file *esterni* (come tutte quelle di questa guida).

Li si vede all’opera nei tre esempi seguenti.

```
La tabella
\begin{center}
\begin{tabular}{ll}
\toprule
Alcaloide & Origine \\
\midrule
atropina & belladonna \\
morfina & papavero \\
\bottomrule
\end{tabular}
\end{center}
mostra l’origine di due alcaloidi.
```

La tabella

Alcaloide	Origine
atropina	belladonna
morfina	papavero

mostra l’origine di due alcaloidi.

```
La tabella
\[
\begin{array}{ll}
\toprule
f(x) & f'(x) \\
\midrule
x^n & nx^{n-1} \\
\sin x & \cos x \\
\bottomrule
\end{array}
\]
mostra le derivate di due funzioni.
```

La tabella

$f(x)$	$f'(x)$
x^n	nx^{n-1}
$\sin x$	$\cos x$

mostra le derivate di due funzioni.

```

La figura
\begin{center}
\includegraphics[width=
0.5\textwidth]{Rettili}
\end{center}
riproduce
l'incisione su legno
\emph{Tassellazione
del piano con rettili}
di M.~Escher.

```

La figura



riproduce l'incisione su legno *Tassellazione del piano con rettili* di M. Escher.

Si noti che:

- Tutti e tre *non* cominciano un nuovo capoverso, ma producono un'unità tipografica indivisibile che il programma tratta come se fosse un unico carattere, e che nel caso di `tabular` e `\includegraphics` va centrata rispetto alla giustezza del testo mettendola nell'ambiente `center`.
- Una tabella `array` va racchiusa a propria volta tra comandi matematici: se in testo, di solito si usano `\[...\]`, che sostituiscono l'ambiente `center` (ma se l'opzione di classe `fleqn` è attiva, la tabella non risulterà più centrata); se *mobile*, invece, si usano `$.~$.`
- Quando richiesta, si consiglia di assegnare all'oggetto una larghezza *relativa* espressa con una frazione della giustezza stabilita dalla classe in uso (`\textwidth`). Una larghezza *assoluta* causerebbe ovvi inconvenienti cambiando classe di documento o aumentando le colonne di composizione.
- L'ambiente `center` si omette anche quando si vuole in linea una figura particolarmente piccola, come mostra l'esempio seguente:

```

La mela morsicata
\includegraphics[width=
0.10\textwidth]{apple}
è il logo di Apple.

```

La mela morsicata  è il logo di Apple.

6.2 Oggetti in testo e fuori testo

In tipografia esistono due tipi di oggetto: “in testo” e “fuori testo”.

6.2.1 Tabelle e figure in testo

Osservando con attenzione gli esempi del paragrafo precedente, si possono notare le caratteristiche degli oggetti “in testo”, i quali:

- appartengono al flusso del discorso e non possono esserne scorporati senza comprometterne la comprensione;
- non prevedono didascalia (proprio perché la loro funzione è spiegata nel contesto) né riferimenti incrociati a sé stessi;
- devono essere, perciò, quanto mai chiari e intuitivi.

Apparentemente innocui, oggetti di questo tipo possono comportare in realtà problemi di impaginazione a volte irrisolvibili. S’immagini, per esempio, di essere arrivati quasi alla fine della pagina, e di dover inserire *proprio lì*, perché richiesto dal discorso, una figura alta cinque centimetri avendone però soltanto tre a disposizione: va da sé che *lì* la figura *non* ci può stare in nessun modo: se lo spazio fisico non c’è, non lo si può inventare! Questa situazione, si badi bene, non si verifica solo con L^AT_EX, ma si dà indipendentemente dal programma in uso per scrivere. Come fare?

Una prima soluzione ingenua (e da evitare) potrebbe essere quella di cominciare una nuova pagina ogni volta che un oggetto non può stare in quella corrente; in questo modo, però, le pagine interrotte rimarrebbero parzialmente bianche, con un risultato tipografico insoddisfacente.

Ecco perché gli oggetti in testo devono essere *eccezionali* (tornano utili per mettere un logo “proprio lì” e in pochissime altre circostanze) e di piccole o piccolissime dimensioni.

6.2.2 Tabelle e figure fuori testo

Si risolve il problema rendendo gli oggetti “fuori testo” (o “mobili”; in inglese *floating*, “galleggianti”) e lasciando fare a L^AT_EX che, nell’esatto ordine in cui oggetti dello stesso tipo sono definiti nel sorgente, li metterà nel punto *per lui* migliore (sulla pagina corrente se ci stanno, oppure in pagine *successive* a quella in cui finirebbero) riempiendo lo spazio rimanente con l’altro materiale a disposizione. Questa soluzione è molto vantaggiosa, perché garantisce l’ottimale riempimento della pagina tipico di L^AT_EX.

A differenza di quelli in testo, gli oggetti fuori testo:

- non appartengono al flusso del discorso e per esigenze tipografiche possono essere spostati altrove dal punto esatto in cui stanno nel sorgente;
- devono avere *obbligatoriamente* un’etichetta, un numero progressivo per gli eventuali riferimenti incrociati e una didascalia che ne descriva il contenuto.

Indipendentemente dalle dimensioni del documento e degli oggetti, perciò, si raccomanda di includerli *sempre* così (come si è fatto in tutta questa guida), vincendo quanto prima le iniziali e comprensibili perplessità derivanti dal vederli molto spesso in un punto diverso da quello in cui li si è definiti, come si spiegherà tra poco.

Nei prossimi paragrafi si spiegano gli ambienti mobili e come comportarsi durante la stesura del documento; nel paragrafo 10.2 si mostra che cosa (eventualmente) si può fare durante la revisione per risolvere collocazioni poco gradite e migliorarlo ulteriormente.

Ambienti standard per gli oggetti mobili

Per rendere mobile un oggetto basta inserirne il relativo codice nell’ambiente standard `table`

```
\begin{table}[\langlepreferenze di collocazione\rangle]
...
\end{table}
```

se è una tabella, oppure in quello `figure`

```
\begin{figure}[\langlepreferenze di collocazione\rangle]
...
\end{figure}
```

se è una figura. Come si può notare, entrambi accettano l’argomento facoltativo *⟨preferenze di collocazione⟩*, il cui funzionamento verrà spiegato nel paragrafo 6.2.2. (Se si sta componendo un documento a due colonne si possono usare le varianti asterisco dei due ambienti,

che mettono l'oggetto sull'intera pagina e non sulla singola colonna di composizione.) I due ambienti appena esaminati richiedono alcuni comandi importanti, descritti di seguito.

Il comando

```
\caption{⟨didascalia⟩}
```

produce, nell'ordine, l'intestazione *Tabella* o *Figura*, il numero progressivo dell'oggetto e la sua *⟨didascalia⟩*. Esiste anche la variante

```
\caption*{⟨didascalia⟩}
```

che produce didascalie prive di intestazione e numero, utili per esempio in tavole illustrate fuori testo o in un libro d'arte.

I due comandi

```
\listoftables  
\listoffigures
```

producono *nel punto in cui li si dà* rispettivamente la sezione contenente l'elenco delle tabelle o delle figure (esattamente come fa `\tableofcontents` per l'indice generale) con relativi titolo e testatina. Le loro voci sono le stesse didascalie degli oggetti o, se troppo lunghe, una versione ridotta da scrivere nell'argomento facoltativo di `\caption`:

```
\caption[⟨didascalia breve⟩]{⟨didascalia normale⟩}
```

Si badi bene a mettere i due indici appena visti solo se il numero degli oggetti è rilevante o è importante per il lettore poterli ritrovare agevolmente: un elenco di sole due figure, per esempio, non avrebbe alcuna utilità. Anche questi due indici richiedono *due* composizioni successive.

Il comando `\label`, da dare sempre *dopo* il corrispondente `\caption`, assegna all'oggetto un'etichetta per i riferimenti incrociati (si veda il paragrafo 3.10).

Codici tipo

Il modo migliore per introdurre un oggetto mobile nel sorgente è scriverne il relativo ambiente preceduto e seguito da una riga vuota. Ecco un esempio tipico per una tabella `tabular` (in modo del tutto analogo si compone una tabella `array`, ricordandosi di racchiuderla tra dollari) con il relativo richiamo:

```
\dots qui finisce un capoverso.
```

```
\begin{table}  
\caption{⟨...⟩}  
\label{tab:esempio}  
\centering  
\begin{tabular}{⟨...⟩}  
...  
\end{tabular}  
\end{table}
```

```
La tabella~\ref{tab:esempio} è un esempio di tabella mobile.
```

E uno per una figura:

```
\dots qui finisce un capoverso.
```

```
\begin{figure}  
\centering  
\includegraphics[width=0.5\textwidth]{...}
```

Tabella 6.1: Preferenze di collocazione per gli oggetti mobili

Preferenza	Chiede a \LaTeX di mettere l'oggetto
h	Qui (<i>here</i>), se possibile
t	In cima (<i>top</i>) alla pagina
b	In fondo (<i>bottom</i>) alla pagina
p	In una pagina di soli oggetti mobili (<i>page of floats</i>)
!	Dove vorrebbe l'utente per quanto possibile

```
\caption{\dots}
\label{fig:esempio}
\end{figure}
```

La figura~\ref{fig:esempio} è un esempio di figura mobile.

Come si può osservare:

- per centrare un oggetto mobile sulla pagina si usa `\centering`, perché l'ambiente `center` lascia tra testo e oggetto uno spazio verticale eccessivo (ma adeguato per un oggetto in testo);
- la corretta posizione della didascalia varia a seconda dell'oggetto cui è apposta: la tradizione italiana la vuole prima di una tabella e dopo una figura, e in queste posizioni *deve* essere messa anche nel sorgente.

Personalizzare la didascalia

Il pacchetto `caption` permette anche di personalizzare finemente le didascalie in *ogni* loro aspetto. Quelle di questa guida, per esempio, sono state composte aggiungendo le seguenti opzioni a quelle appena viste:

```
\captionsetup{format=hang,labelfont=bf}
```

dove:

- `format=hang` allinea (*hang*) alla prima riga quelle successive (\LaTeX centra automaticamente le didascalie che occupano una sola riga);
- `labelfont=bf` imposta l'etichetta della didascalia in caratteri neri.

Che cosa fare durante la stesura

Durante la stesura del documento, si sa, bisogna concentrarsi sul contenuto del lavoro, lasciando fare a \LaTeX tutto il resto. La gestione degli oggetti non si sottrae a questa regola, per cui in prima battuta si consiglia di inserirli tutti *senza specificare alcuna preferenza di collocazione* (si veda poco sotto): il risultato è generalmente ottimo. In linea di massima si troveranno gli oggetti “abbastanza” vicini al punto in cui stanno nel sorgente, e in particolare:

- quasi sempre in cima alla pagina (corrente o una delle successive);
- raramente in basso;
- se sono grandi (secondo i parametri di \LaTeX), in un pagina di soli oggetti mobili, sempre successiva al punto in cui li si è messi nel sorgente.

Basta sfogliare un qualunque libro ben composto per verificare che le cose stanno proprio così.

Non ci si lamenti per non vederli esattamente dove li si è inseriti nel sorgente! Non è un difetto di \LaTeX , questo, ma un vantaggio, proprio come lo è il non doversi preoccupare di numerare a mano sezioni e pagine o il non dover pensare a quanto spazio ci va tra un titolo e il testo successivo.

I risultati automatici sono *quasi sempre* migliori di quelli che si potrebbero ottenere cercando di collocare gli oggetti a mano. Quando i gusti dell'utente non coincidono con quelli del programma, tuttavia, si possono usare le *preferenze di collocazione* raccolte nella tabella 6.1, che "suggeriscono" a \LaTeX come si vorrebbero vedere gli oggetti sulle pagine del proprio documento. \LaTeX seguirà i suggerimenti nell'ordine in cui li trova (fa eccezione la preferenza *p*, come si spiega nel paragrafo 10.2). Di seguito si propongono un paio delle possibilità più usate:

- *tp* se non si vuole nessun oggetto in fondo alla pagina;
- *htp* se si vuole che \LaTeX cerchi come prima cosa di mettere l'oggetto esattamente lì dove lo si è inserito (se è sufficientemente piccolo, di solito si viene accontentati).

Ecco ora le opzioni da evitare *sempre*:

- *h* o, peggio, *h!* *possono* funzionare solo con oggetti molto piccoli; in caso contrario, l'oggetto viene messo alla fine del capitolo (o del documento) portandosi dietro tutti gli altri inseriti successivamente (si tenga *ben* presente questo comportamento);
- *t* e *b* da sole, perché è buona regola dare al programma almeno un paio di possibilità (ma è ammessa la sola *p*).

In casi estremi, e solo per ottenere effetti particolari, si può forzare l'oggetto nella posizione desiderata con la preferenza *H* del pacchetto *float* (da usare *sempre* da sola).

```
\dots
qui finisce un capoverso.

\begin{figure}[H]
\centering
\includegraphics[width=0.5\textwidth]{Formica}
\caption{Figura collocata a mano}
\label{fig:float}
\end{figure}

La figura~\vref{fig:float} è
un esempio di figura mobile
collocata a mano.
```

...qui finisce un capoverso.



Figura 6.1: Figura collocata a mano

La figura 6.1 è un esempio di figura mobile collocata a mano.

6.3 Tabelle

Comporre tabelle di altissima qualità è una delle specialità di \LaTeX , ma i suoi comandi standard sono piuttosto limitati. Numerosi pacchetti, però, ne definiscono di nuovi e più avanzati con cui si può personalizzare finemente il proprio lavoro. Questo paragrafo spiega come usare gli uni e gli altri e affronta gli aspetti principali dell'argomento.

Tabella 6.2: Tabella che non rispetta le regole generali

D	P	u	β	G
.500 m	269.8 kg	.000674 m	1.79	.04089 Pa
1.50 m	421.0 kg	.001035 m	3.59	"
10.0 m	640.2 kg	.001565 m	7.18	"

Tabella 6.3: Tabella che rispetta le regole generali

D (m)	P (kg)	u (m)	β	G (Pa)
0,500	269,8	0,000 674	1,79	0,040 89
1,50	421,0	0,001 035	3,59	0,040 89
10,0	640,2	0,001 565	7,18	0,040 89

6.3.1 Indicazioni generali

Regole generali di composizione

Le regole seguenti permettono di ottenere una tabella ben composta:

- non usare *mai* filetti verticali;
- evitare filetti doppi;
- scrivere *sempre* le unità di misura nell'intestazione della colonna e non nel corpo della tabella;
- non usare *mai* le virgolette per ripetere il contenuto di una cella;
- far precedere *sempre* il separatore decimale da almeno una cifra.

Così vuole la tradizione tipografica, in contrasto con la cattiva abitudine, purtroppo oggi molto diffusa, di comporre le tabelle come se fossero parti di un foglio elettronico. Per capire quanto sia importante rispettare queste regole, si confrontino le tabelle 6.2 e 6.3.

Separare le celle e chiudere le righe

Le celle di una tabella vanno separate tra loro con il carattere separatore & e le righe *devono* terminare con il comando `\\`, pena un errore. Si noti che se una riga ha meno celle piene di quante sono le colonne, può essere chiusa dopo l'ultima cella riempita.

Filetti professionali

Filetti migliori di quelli che si ottengono con il comando standard `\hline`, dalla resa tipografica insoddisfacente per via dello spazio troppo risicato che risulta tra filetti e testo nelle celle, sono prodotti da tre comandi definiti dal pacchetto `booktabs`. Questi comandi non vogliono `\\` dopo di sé, producono filetti di spessore differente (le righe prodotte da `\midrule`, infatti, sono più sottili delle altre) e vanno dati secondo un ordine rigoroso:

1. `\toprule` produce il primo filetto della tabella;
2. `\midrule` produce il filetto interno (o, ripetendolo, i filetti, ma non se ne abusi);
3. `\bottomrule` produce l'ultimo filetto.

Tabella 6.4: Descrittori standard delle colonne

Descrittore	Spiegazione
l	Allinea il contenuto della cella a sinistra (<i>left</i>)
c	Centra il contenuto della cella (<i>center</i>)
r	Allinea il contenuto della cella a destra (<i>right</i>)
p	Giustifica un testo lungo entro una $\langle larghezza \rangle$

Codice ordinato

Anche se \LaTeX non richiede di incolonnare le celle nel sorgente, si consiglia di farlo ugualmente: un codice ordinato facilita eventuali modifiche, diminuisce la probabilità di commettere errori e aumenta quella di scovarli.

6.3.2 Tabelle standard

In linea generale, una tabella che contiene prevalentemente testo va composta dentro l'ambiente `tabular`; una tabella che contiene prevalentemente matematica va composta dentro l'ambiente `array`. Entrambi si comportano in modo molto simile, come si può osservare negli esempi seguenti:

```
\begin{tabular}{lcr}
\toprule
Grandezza & Simbolo & Unità \\
\midrule
forza      &  $\text{\$F\$}$     & newton \\
energia    &  $\text{\$E\$}$     & joule \\
tensione   &  $\text{\$V\$}$     & volt \\
\bottomrule
\end{tabular}
```

Grandezza	Simbolo	Unità
forza	F	newton
energia	E	joule
tensione	V	volt

```
\[
\begin{array}{cc}
\toprule
f(x) & \text{\text{Una primitiva}} \\
\midrule
e^x   & e^x \\
\cos x & \sin x \\
\sin x & -\cos x \\
\bottomrule
\end{array}
\]
```

$f(x)$	Una primitiva
e^x	e^x
$\cos x$	$\sin x$
$\sin x$	$-\cos x$

Alle osservazioni del paragrafo 6.1 si aggiungano le seguenti:

- `tabular` e `array` richiedono un argomento, detto *preambolo della tabella*, formato da un certo numero di *descrittori*, ciascuno dei quali definisce il comportamento di un *tipo di colonna* come spiegato nella tabella 6.4;
- in `tabular`, eventuali formule matematiche si scrivono con i comandi per le formule in linea, per esempio fra dollari $\text{\$...\$}$;
- in `array`, un eventuale testo si scrive nell'argomento del comando `\text` del pacchetto `amsmath`, che va dunque caricato.

Tabella 6.5: Tabella con colonna p

Forza	Una forza è una grandezza fisica che si manifesta nell'interazione di due o più corpi materiali, che cambia lo stato di quiete o di moto dei corpi stessi.
Momento polare	Il momento polare di una forza rispetto a una determinata origine è definito come il prodotto vettoriale tra il vettore posizione (rispetto alla stessa origine) e la forza.

6.3.3 Celle con testo troppo lungo

Le tabelle migliori si ottengono lasciando loro la propria larghezza naturale. È ciò che fanno i tre descrittori `l`, `c` e `r`, allargando automaticamente la cella in base al contenuto. Se quest'ultimo è costituito da un testo troppo lungo, però, la tabella eccede la giustezza della riga e \LaTeX lo notifica con il relativo avviso. Per queste celle non si possono più usare le colonne appena viste, ma bisogna ricorrere ad altri strumenti:

- a un descrittore `p{<larghezza>}`, che permette di stabilire a priori la larghezza di una sola colonna;
- al pacchetto `tabularx`, che permette di stabilire a priori la larghezza dell'intera tabella.

Si noti che in entrambi i casi le eventuali intestazioni di colonna vengono allineate a sinistra per impostazione predefinita: per cambiare questo risultato si usi il comando `\multicolumn` spiegato nel paragrafo 6.3.4.

Colonne di larghezza prefissata

Il codice seguente, che mostra all'opera il descrittore `p{<larghezza>}`, produce la tabella 6.5:

```
\begin{tabular}{lp{0.5\textwidth}}
\toprule
\textbf{Forza} & Una forza è una grandezza fisica che si manifesta
nell'interazione di due o più corpi materiali, che cambia lo stato
di quiete o di moto dei corpi stessi. \\
\midrule
\textbf{Momento polare} & Il momento polare di una forza rispetto a
una determinata origine è definito come il prodotto vettoriale tra
il vettore posizione (rispetto alla stessa origine) e la forza. \\
\bottomrule
\end{tabular}
```

Si noti che:

- `\textbf` produce il proprio argomento in nero;
- per impostazione predefinita, il contenuto di una colonna `p` viene giustificato e sillabato automaticamente;
- le eventuali colonne `l`, `c` e `r` rimangono della propria larghezza naturale;
- la cella è allineata alla riga a cui appartiene rispetto alla linea di base superiore.

Tabella 6.6: Tabella di larghezza prefissata ottenuta con `tabularx`

Forza	Una forza è una grandezza fisica che si manifesta nell'interazione di due o più corpi materiali, che cambia lo stato di quiete o di moto dei corpi stessi.
Momento polare	Il momento polare di una forza rispetto a una determinata origine è definito come il prodotto vettoriale tra il vettore posizione (rispetto alla stessa origine) e la forza.

Tabella 6.7: Tabella con due colonne della stessa larghezza ottenuta con `tabularx`

Periodo	Fenomeni geologici	Biosfera
Giurassico	Periodo caratterizzato da variazioni del livello del mare; prevalenza delle terre emerse in America, Asia, Australia.	Fauna: compaiono i primi marsupiali; dominano i grandi rettili (dinosauri). Flora: predominano le conifere.
Triassico	Intensa l'erosione dei continenti; profonde fratture da cui escono lave che originano altopiani estesi.	Fauna: si diffondono i rettili; nei mari prosperano pesci e invertebrati. Flora: si sviluppano alghe caratteristiche.

Tabelle di larghezza prefissata

Il pacchetto `tabularx` definisce l'omonimo ambiente `tabularx` e un nuovo tipo di colonna `X`, che alle caratteristiche delle colonne `p` appena esaminate aggiunge un vantaggio: è \LaTeX a calcolarne automaticamente la larghezza in base alla larghezza complessiva assegnata *all'intera tabella*. Infatti `tabularx` richiede *obbligatoriamente* un secondo argomento in cui indicarla (nei prossimi esempi è pari a `\textwidth`, ma sono ammessi anche altri valori, da impostare come già spiegato nel paragrafo 6.1). Si osserva il pacchetto all'opera nella tabella 6.6, identica alla 6.5 nel contenuto, ottenuta con il codice seguente:

```
\begin{tabularx}{\textwidth}{lX}
\toprule
\textbf{Forza} & Una forza è una grandezza fisica che si manifesta
nell'interazione di due o più corpi materiali, che cambia lo stato
di quiete o di moto dei corpi stessi. \\
\midrule
\textbf{Momento polare} & Il momento polare di una forza rispetto a
una determinata origine è definito come il prodotto vettoriale tra
il vettore posizione (rispetto alla stessa origine) e la forza. \\
\bottomrule
\end{tabularx}
```

La tabella 6.7, prodotta con il codice seguente, mostra come a *tutte* le colonne `X`, se più d'una, \LaTeX assegni la stessa larghezza *indipendentemente* dalle altre colonne presenti.

```
\begin{tabularx}{\textwidth}{lXX}
\toprule
Periodo & Fenomeni geologici & Biosfera \\
\midrule
\textbf{Giurassico} & Periodo caratterizzato da variazioni del
livello del mare; prevalenza delle terre emerse in America, Asia,
Australia. & Fauna: compaiono i primi marsupiali; dominano i grandi
rettili (dinosauri). Flora: predominano le conifere. \\
\midrule
\textbf{Triassico} & Intensa l'erosione dei continenti; profonde
fratture da cui escono lave che originano altopiani estesi.
```


Tabella 6.8: Tabella con cella multicolonna

Particella	Nome	
	Antiparticella	Carica (e)
elettrone	positrone	∓ 1
protone	antiprotone	± 1
neutrone	antineutrone	0

```
& Fauna: si diffondono i rettili; nei mari prosperano pesci e
invertibrati. Flora: si sviluppano alghe caratteristiche. \\
\bottomrule
\end{tabularx}
```

6.3.4 Celle multicolonna

Il comando

```
\multicolumn{<n>}{<descrittore>}{<testo>}
```

sostituisce a $\langle n \rangle$ celle successive un'unica cella, il cui $\langle testo \rangle$ viene organizzato nei modi specificati con il $\langle descrittore \rangle$. Lo si vede all'opera nel prossimo esempio, che produce la tabella 6.8:

```
\begin{tabular}{llc}
\toprule
\multicolumn{2}{c}{Nome} & Carica \\
Particella & Antiparticella & (e) \\
\midrule
elettrone & positrone &  $\mp 1$  \\
protone & antiprotone &  $\pm 1$  \\
neutrone & antineutrone & 0 \\
\bottomrule
\end{tabular}
```

Si può usare il comando anche per *una sola* cella, come spesso accade nell'intestazione della tabella: in questo caso, a $\langle n \rangle$ si sostituisce 1.

6.3.5 Spaziare a mano righe e colonne

Quando (molto di rado) i risultati di \LaTeX non soddisfano completamente, si può migliorare la resa tipografica della tabella con piccoli aggiustamenti manuali. I casi, di solito, sono due:

1. le righe della tabella appaiono troppo ravvicinate;
2. le colonne della tabella appaiono troppo ravvicinate.

Risolve il primo caso il comando

```
\[\langle altezza \rangle]
```

che, sostituito al delimitatore di riga standard \backslash nel corpo della tabella, abbassa la riga immediatamente successiva (e solo quella) di uno spazio verticale pari ad $\langle altezza \rangle$. Di solito s'inserisce questo spazio supplementare per rimediare a lievi sovrapposizioni (specialmente di formule matematiche in display) oppure per dare maggiore "respiro" alla tabella.

Si osservi come la spaziatura automatica nella tabella seguente è troppo risicata

```

\begin{array}{cc}
\toprule
f(x) & f'(x) \\
\midrule
\log x & \frac{1}{x} \\
\arctan x & \frac{1}{1+x^2} \\
\bottomrule
\end{array}

```

$f(x)$	$f'(x)$
$\log x$	$\frac{1}{x}$
$\arctan x$	$\frac{1}{1+x^2}$

e come la si può correggere molto semplicemente con

```

\begin{array}{cc}
\toprule
f(x) & f'(x) \\
\midrule
\log x & \frac{1}{x} \\
\arctan x & \frac{1}{1+x^2} \\
\bottomrule
\end{array}

```

$f(x)$	$f'(x)$
$\log x$	$\frac{1}{x}$
$\arctan x$	$\frac{1}{1+x^2}$

Il comando `\dfrac` richiede il pacchetto `amsmath`.

6.3.6 Tabelle con note

Il comando `\footnote`, che in un testo produce le note al piede, non funziona nell'ambiente `tabular` e, come per la maggior parte delle cose in \LaTeX , questa limitazione ha ottime ragioni per esserci. Come si sa, una tabella dovrebbe essere *sempre* inserita fuori testo in un documento: una nota al piede la vincolerebbe alla pagina in cui si trova la nota. Il luogo più adatto per mettere eventuali annotazioni è la didascalia.

6.3.7 Tabelle grandi

Se le dimensioni della tabella finita eccedono quelle della gabbia del testo in lunghezza, in larghezza o in entrambe, si prospettano soluzioni diverse a seconda della dimensione in eccesso. Se *troppo lunga* o *troppo larga*, si può ridurre il corpo del font o ruotarla (ma quest'ultima soluzione si applica solo se troppo larga). A ogni caso si possono applicare più soluzioni contemporaneamente.

Ridurre il corpo del font

Per ridurre il corpo del font in una tabella (ma non nell'eventuale didascalia) si usano le stesse dichiarazioni elencate nella tabella 4.5, con l'avvertenza di darle:

- *fuori* da `tabular` se la tabella è *in testo* (subito dopo la tabella una nuova dichiarazione ripristinerà il font corrente);
- *dentro* `table` se la tabella è *mobile* (si veda il paragrafo 6.2).

Il codice

Tabella 6.9: Tabella con corpo di carattere inferiore (`footnotesize`) a quello del testo principale (`normalsize`)

Forza	Una forza è una grandezza fisica che si manifesta nell'interazione di due o più corpi materiali, che cambia lo stato di quiete o di moto dei corpi stessi.
Momento polare	Il momento polare di una forza rispetto a una determinata origine è definito come il prodotto vettoriale tra il vettore posizione (rispetto alla stessa origine) e la forza.

```

\begin{table}[tb]
\footnotesize
\caption{\langle...\rangle}
\label{\langle...\rangle}
\centering
\begin{tabular}{lp{0.5\textwidth}}
...
\end{tabular}
\end{table}

```

produce la tabella 6.9. (Se ne confronti il risultato con la tabella 6.5.)

Ruotare una tabella

Per ruotare di 90° una tabella è utile il pacchetto `rotating`, che definisce l'ambiente `sidewaystable` da usare nel modo seguente:

```

\begin{sidewaystable}
\caption{\langle...\rangle}
\label{\langle...\rangle}
\centering
\begin{tabular}
...
\end{tabular}
\end{sidewaystable}

```

Si noti che una tabella di questo tipo è mobile e occupa *sempre* una pagina a sé, come mostra la tabella 3.5. Per ruotare le immagini il pacchetto definisce l'analogo ambiente `sidewaysfigure`.

6.4 Figure

Le figure rientrano tra gli argomenti più studiati dalle guide a \LaTeX , tanto che ne esistono di specifiche, cui si rimanda per gli approfondimenti.

Questi oggetti presentano almeno due problemi diversi, riguardanti:

- il *tipo di file* da introdurre nel documento (verrà trattato in questo paragrafo);
- la *collocazione* della figura sulla pagina (verrà trattato nel paragrafo successivo).

Di qui in avanti si dà per caricato il pacchetto `graphicx`.

6.4.1 Immagini vettoriali e bitmap

Si possono dividere le figure in due grandi classi: le immagini *vettoriali* e le immagini *bitmap*.

Immagini vettoriali

Le immagini vettoriali sono descritte da forme, possono essere scalate e deformate senza perdere in definizione e sono adatte soprattutto per schemi e grafici, argomento non considerato in questa guida per via della sua complessità. Molto più semplicemente, le si può aggiungere al documento dopo averle preparate a parte con programmi specifici. Il formato vettoriale più noto e diffuso è il PDF. Il paragrafo 6.4.4 raccoglie alcuni programmi per la grafica vettoriale.

Immagini bitmap

Le immagini bitmap sono matrici di pixel colorati, di solito perdono in definizione se ingrandite o rimpicciolite e sono più adatte a fotografie e icone. I formati bitmap sono numerosissimi e comprendono il JPEG, molto diffuso in ambito fotografico e nella grafica con colori morbidi, e il PNG, adatto per grafica con colori decisi.

6.4.2 Convertire i formati

Prima ancora di includere le immagini nel documento bisogna produrle nel formato più adatto al proprio scopo. È inutile registrare una figura come JPEG per poi convertirla in PDF, perché la conversione include semplicemente il file bitmap in una “cornice” PDF senza migliorarne in alcun modo la qualità. È sbagliato anche fare l’opposto, perché così si perdono le informazioni sulla geometria della figura, abbassandone la qualità. Nonostante questo, si potrebbero avere a disposizione soltanto immagini in formati *non* adatti a \LaTeX , e allora la conversione sarebbe davvero necessaria. A questo proposito si ricordi bene che \LaTeX accetta immagini PDF, JPEG e PNG. Il paragrafo 6.4.4 descrive alcuni programmi per convertire i diversi formati.

6.4.3 Ritagliare le immagini

Uno dei parametri più importanti di una figura è l’informazione sulle dimensioni del rettangolo circoscritto a essa (*bounding box*). Questo contorno determina la grandezza effettiva dell’immagine e serve a \LaTeX per calcolare lo spazio da riservarle sulla pagina. Idealmente, il contorno dovrebbe coincidere con il limite dell’immagine, ma talvolta le figure sono circondate da un invisibile bordo bianco più o meno ampio, fonte di non pochi problemi estetici: la figura appare sulla pagina troppo piccola o non centrata o circondata da eccessivi margini verticali, per esempio, anche se \LaTeX la sta trattando nel modo corretto.

La primissima cosa da verificare, quindi, è che le dimensioni della *bounding box* siano corrette, aprendo la figura con un programma opportuno (come Adobe Reader o GIMP) e attivando la visualizzazione del contorno che, se scorretto, va ridimensionato. Se il problema riguarda poche figure si può risolvere a mano, ma se i file da ottimizzare sono molti, vanno corretti all’origine (magari configurando ad hoc il programma usato per produrli).

6.4.4 Alcuni programmi utili

La composizione asincrona di \LaTeX ha anche un vantaggio da non trascurare: permette di usare sempre il prodotto migliore. Per ciascuna operazione sul documento, infatti,

Tabella 6.10: Alcuni programmi utili per lavorare con \LaTeX (le lettere G, C, R e V indicano rispettivamente le funzioni di grafica vettoriale, conversione dei formati, ritaglio immagini e visualizzazione; il simbolo • indica che la funzione è disponibile)

Programma	G	C	R	V
Adobe Acrobat			•	•
Adobe Reader				•
Anteprima		•	•	•
Asymptote	•			
Ghostview e GSview		•		•
GIMP		•	•	•
Gnuplot	•			
ImageMagick		•		
Inkscape	•	•		•
Mathematica	•			
OmniGraffle	•			
Xfig e WinFIG	•			

l'utente può usare un programma specializzato, ciò che non sarebbe permesso con un software "tuttofare" come un editor di testi tradizionale. La tabella 6.10 raccoglie, senza pretese di completezza, alcuni programmi utili per lavorare con \LaTeX (ma preziosi anche in molte altre occasioni), specificandone le funzioni principali. Una veloce ricerca in Rete permette di recuperarli.

6.4.5 Includere le immagini nel documento

Il pacchetto `graphicx`, che in genere non richiede opzioni, gestisce il trattamento delle immagini con \LaTeX . Il comando `\includegraphics`, la cui sintassi completa è

```
\includegraphics[⟨chiave⟩=⟨valore⟩,⟨...⟩]{⟨immagine⟩}
```

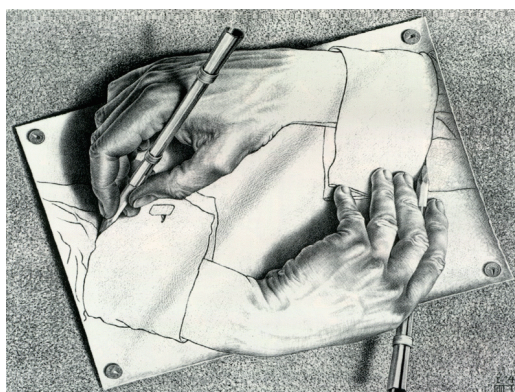
le include nel documento. Si osservi che:

- nell'argomento facoltativo ci vanno le opzioni che regolano l'aspetto della figura sulla pagina nella forma `⟨chiave⟩=⟨valore⟩`;
- nell'argomento obbligatorio ci va il nome dell'immagine *senza* specificarne l'estensione.

Gli esempi seguenti mostrano il pacchetto all'opera.

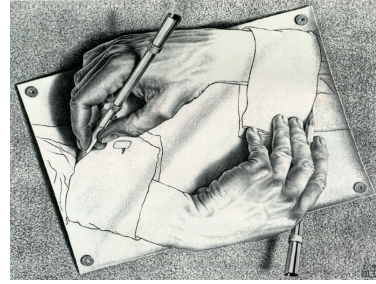
L'utente può assegnare all'immagine una larghezza (`width`)

```
\includegraphics[width=%  
\textwidth]{mani}
```



o un'altezza (`height`) a piacere:

```
\includegraphics[height=%  
0.15\textheight]{mani}
```



Si noti che, per i motivi già spiegati nel paragrafo 6.1, le figure devono avere dimensioni *relative*, cioè essere larghe una frazione di `\textwidth` e alte una frazione di `\textheight` (l'altezza della gabbia del testo).

Capitolo 7

Bibliografia

La bibliografia è da sempre uno degli aspetti più delicati di un documento, e \LaTeX aiuta anche in questo caso, definendo gli strumenti per realizzarla e gestirla con semplicità.

L'ambiente `thebibliography` gestisce la bibliografia di un documento molto facilmente. La sintassi generale è:

```
\begin{thebibliography}{\langle etichetta più lunga \rangle}
\bibitem[\langle etichetta personalizzata \rangle]{\langle chiave di citazione \rangle}
...
\end{thebibliography}
```

dove:

- $\langle etichetta \text{ più lunga} \rangle$ può essere un numero (di solito 9 se la bibliografia comprende meno di dieci opere, 99 se almeno dieci ma meno di cento, eccetera) oppure del testo (nel caso di etichette personalizzate: si scrive allora l'etichetta più lunga);
- `\bibitem` va premesso a ogni riferimento bibliografico;
- $\langle etichetta \text{ personalizzata} \rangle$ è un'etichetta personalizzata che *eventualmente* comparirà nella bibliografia e nelle citazioni al posto del numero predefinito;
- $\langle chiave \text{ di citazione} \rangle$ è l'etichetta *univoca* per citare la fonte nel documento (si consiglia di usare la sintassi $\langle autore \rangle : \langle titolo \rangle$ analoga a quella di `\label`).

Lo si vede all'opera nell'esempio seguente:

```
\begin{thebibliography}{9}
\bibitem{eco:tesi}
Eco, Umberto (1977),
\emph{Come si fa una tesi di
laurea}, Bompiani, Milano.

\bibitem{mori:tesi}
Mori, Lapo Filippo (2007),
“Scrivere la tesi di laurea
con  $\text{\LaTeX}$ ”,  $\text{\Ars}$ (3).
\end{thebibliography}
```

Bibliografia

- [1] Eco, Umberto (1977), *Come si fa una tesi di laurea*, Bompiani, Milano.
- [2] Mori, Lapo Filippo (2007), “Scrivere la tesi di laurea con \LaTeX ”, \ArsTeX nica (3).

Si noti che:

- `thebibliography` si comporta in modo molto simile a un ambiente per elenchi;
- ciascun riferimento bibliografico va scritto per intero, regolandone *a mano* tutti gli aspetti (corsivo, virgolette, eccetera), compresa la posizione nell'ordine alfabetico;

- nel documento finito i riferimenti saranno contrassegnati con un numero tra parentesi quadre sia nella bibliografia sia nelle citazioni;
- thebibliography produce la sezione contenente la bibliografia con relativi titolo e testatina.

Per citare un riferimento bibliografico nel testo si usa il comando `\cite`:

```
\cite[⟨informazioni⟩]{⟨chiave di citazione⟩}
```

dove:

- *⟨informazioni⟩* sono ulteriori indicazioni (numero di pagina o di capitolo) che eventualmente si possono dare per completare la citazione;
- *⟨chiave di citazione⟩* si spiega da sé.

I prossimi esempi lo mostrano all'opera:

```
Si veda~\cite{eco:tesi} per  
maggiori dettagli.
```

Si veda [1] per maggiori dettagli.

```
Si veda~\cite[p.~7]{eco:tesi}  
per maggiori dettagli.
```

Si veda [1, p. 7] per maggiori dettagli.

Per mandare nell'indice generale il titolo della bibliografia del documento, *immediatamente prima* di aprire thebibliography vanno date sequenze di comandi diverse in base alla classe in uso:

```
\cleardoublepage  
\phantomsection  
\addcontentsline{toc}{chapter}{\bibname}
```

se la classe è book o report, e

```
%\clearpage  
\phantomsection  
\addcontentsline{toc}{section}{\refname}
```

se la classe è article, dove:

- `\cleardoublepage` fa cominciare la bibliografia in una pagina nuova dispari e assegna alla corrispondente voce nell'indice il numero di pagina corretto;
- `\clearpage` va dato per assicurare la corretta assegnazione del numero di pagina alla voce nell'indice *solo se* a fine composizione il corpo del documento terminasse esattamente a fine pagina e la bibliografia cominciasse all'inizio di una pagina nuova (in tal caso si decommenti la riga corrispondente);
- `\phantomsection` va dato *solo se* è caricato anche hyperref (in tal caso si decommenti la riga corrispondente);
- `chapter` e `section` indicano il livello della sezione bibliografica (un capitolo e un paragrafo, rispettivamente);
- `\bibname` e `\refname` producono nell'indice generale del documento le voci *Bibliografia* e *Riferimenti bibliografici* rispettivamente.

Capitolo 8

Indice analitico

L'indice analitico è un elenco alfabetico di parole o espressioni (anche con alcuni livelli di subordinazione) dette *voci*, posto di regola alla fine di un documento; accanto a ogni voce ci sono i numeri delle pagine in cui la voce in questione compare. In molti lavori questo indice è utilissimo, e \LaTeX è capace di gestirlo automaticamente e con grande efficienza, come si mostra in questo capitolo.

Per creare l'indice analitico con \LaTeX bisogna innanzitutto eseguire due operazioni preliminari:

1. caricare il pacchetto `makeidx` per abilitare il programma alla composizione dell'indice;
2. dare il comando `\makeindex` *nel preambolo*, per attivare i comandi dedicati che verranno inseriti nel corpo del testo.

A questo punto, *immediatamente dopo* ogni parola o espressione da indicizzare basterà dare il comando

```
\index{<voce>}
```

la cui sintassi è mostrata nella tabella 8.1, nel cui argomento in teoria si può scrivere ciò che si vuole.

Per produrre la sezione dell'indice analitico e mandarne il titolo nell'indice generale, *immediatamente prima* di `\end{document}` vanno date le seguenti sequenze di comandi:

```
\cleardoublepage
%\phantomsection
\addcontentsline{toc}{chapter}{\indexname}
\printindex
```

se la classe in uso è `book` o `report`, e

```
%\clearpage
%\phantomsection
\addcontentsline{toc}{section}{\indexname}
\printindex
```

se la classe è `article`. Il codice

```
\documentclass{article}
...
\usepackage{makeidx}
\makeindex
...
\begin{document}
```

Tabella 8.1: Sintassi del comando `\index`

Tipo di voce e codice	Risultato
Primaria <code>\index{Artisti}</code>	Artisti, 2
Sottovoce <code>\index{Artisti!Escher}</code>	Artisti, 2 Escher, 3

Si possono inserire nell'indice analitico singole parole come `\emph{arte}\index{arte}`, oppure intere espressioni come `questa\index{intere espressioni come questa}`.

...

```
\addcontentsline{toc}{section}{\indexname}
\printindex
\end{document}
```

crea un indice analitico con le voci *arte* e *intere espressioni come questa*.

Per generare effettivamente l'indice analitico, questa è la sequenza di composizione da seguire:

1. si compone il documento con \LaTeX una prima volta;
2. si lancia il programma `MakeIndex` premendo l'apposito pulsante dell'editor;
3. si compone il documento altre *due* volte con \LaTeX .

Si ricorda infine il pacchetto `imakeidx`, con cui si possono ottenere automaticamente indici semplici e multipli *già alla prima composizione*.

Capitolo 9

Personalizzazioni

Prima o poi, capita a tutti di aver bisogno di “cose” non contemplate da \LaTeX standard. In questo capitolo s’impara come farlo.

9.1 Comandi e ambienti personali

9.1.1 Nuovi comandi

S’immagini di scrivere un libro di botanica e di volere tutti i nomi di pianta in corsivo. Il modo più veloce per farlo è scrivere ogni nome nell’argomento di `\textit`. Successivamente l’editore chiede i nomi di pianta in nero. Che fare? Si potrebbero sostituire automaticamente tutti i `\textit` con altrettanti `\textbf`, ma ci sarebbe qualche problema, perché si potrebbe aver usato il corsivo anche in altre parti del libro, che invece devono rimanere tali. Non resta che sostituire un comando dopo l’altro perdendo un sacco di tempo.

Con \LaTeX , anziché agire a mano si può definire *una volta per tutte* un nuovo comando, `\pianta`, che produce il proprio argomento nello stile deciso dall’utente. Se l’imposizione del nero avviene a documento completato, basterà modificare *una volta per tutte* la definizione del comando.

I comandi personali si definiscono *nel preambolo* con `\newcommand`:

```
\newcommand{<nome>}[<numero di argomenti>]{<definizione>}
```

Dove:

- `<nome>` è il nome che si dà al nuovo comando.
- `<numero di argomenti>` è il numero di argomenti obbligatori che gli si assegna.
- `<definizione>` sono le istruzioni che specificano ciò che si vuole che il nuovo comando “faccia”. Gli argomenti eventualmente assegnati al comando s’indicano nella `<definizione>` con # seguito da un numero progressivo: #1, #2, eccetera.

Di seguito si mostra la sintassi di un comando personale *con argomenti*:

```
\newcommand{\pianta}[1]{\textit{#1}}
```

Si è assegnato al nuovo comando un solo argomento obbligatorio ([1]). Quando si dà il comando, il testo nell’argomento di `\pianta` viene “passato” a #1 e trattato secondo la `<definizione>`. In questo caso verrà reso in corsivo, come si vede nell’esempio seguente:

```
\pianta{Rosa canina}
```

Rosa canina

Tabella 9.1: Parole fisse italiane di babel

Comando	Voce	Comando	Voce
<code>\abstractname</code>	Sommario	<code>\indexname</code>	Indice analitico
<code>\appendixname</code>	Appendice	<code>\listfigurename</code>	Elenco delle figure
<code>\bibname</code>	Bibliografia	<code>\listtablename</code>	Elenco delle tabelle
<code>\chaptername</code>	Capitolo	<code>\partname</code>	Parte
<code>\contentsname</code>	Indice	<code>\refname</code>	Riferimenti bibliografici

9.1.2 Nuovi ambienti

Un ambiente personale si definisce con il comando `\newenvironment` nel *preambolo*:

```
\newenvironment{<nome>}[<numero di argomenti>]{<comandi di apertura>}{<comandi di chiusura>}
```

la cui sintassi si spiega da sé.

L'esempio seguente mostra la sintassi di un ambiente personale *senza argomenti*. Scrivendo nel *preambolo*

```
\newenvironment{itaitemize}{\begin{itemize}\itshape}{\end{itemize}}
```

si potrà usare il nuovo ambiente `itaitemize` così:

```
\begin{itaitemize}
\item Un elenco con voci\dots
\item \dots automaticamente corsive.
\end{itaitemize}
```

- *Un elenco con voci...*
- *...automaticamente corsive.*

9.2 Specialità

Questo paragrafo descrive alcune delle numerose possibilità offerte da \LaTeX per personalizzare il proprio documento.

9.2.1 Colori

Il pacchetto `xcolor` permette di usare i colori in un documento scritto con \LaTeX . Con `xcolor` si possono usare i colori con il loro nome, scelto in tavolozze predefinite.

Per racchiudere del testo **in un riquadro colorato** si usa il comando

```
\colorbox{<colore>}{<testo>}
```

la cui sintassi si spiega da sé.

9.2.2 Modificare le parole fisse

Per modificare le parole fisse generate da babel (sostituire *Capitolo* con *Unità*, per esempio), si scrive nel *preambolo* il codice

```
\addto\captions<lingua>{<testo>}
```

che comanda a \LaTeX di aggiungere alle definizioni specifiche della *<lingua>* il *<testo>*.

Volendo sostituire la voce *Capitolo* con *Unità* basta scrivere

```
\addto\captionsitalian{\renewcommand{\chaptername}{Unità}}
```

La tabella 9.1 elenca alcuni comandi di queste voci con la relativa traduzione italiana.

Capitolo 10

Revisione finale

Anche se \LaTeX induce a privilegiare la struttura logica del documento e a trascurarne l'aspetto finale, nemmeno lui risolve *tutti* i problemi tipografici evitando d'intervenire a mano. La revisione finale del documento è un'arte complicata ma ricca di soddisfazioni, che dà i propri frutti migliori solo quando il documento è nella sua forma *definitiva*.

10.1 Problemi orizzontali

I difetti d'impaginazione orizzontali riguardano la formazione dei capoversi: di solito consistono in righe sporgenti nel margine destro.

Titoli problematici

Se troppo lunghi, i titoli creano problemi almeno in due luoghi:

- nel corpo del documento, perché \LaTeX potrebbe spezzarli in modo insoddisfacente;
- nell'indice generale e nelle testatine, riproducendo il problema appena visto.


Un titolo di sezione non dovrebbe *mai* andare a capo. Si risolvono tutti i problemi riformulandolo: si può fare praticamente sempre. Se invece il titolo lungo fosse davvero necessario, lo si mandi a capo con un `\` esplicito nell'argomento del comando di sezionamento e se ne usi l'argomento facoltativo per indice generale e testatine:

```
\chapter[Titolo breve per indice e testatine]{Titolo lungo\\ da mandare a capo}
```

Capoversi problematici

Un documento di una certa lunghezza conterrà quasi certamente qualche riga che \LaTeX non è riuscito a comporre "bene" e che si vedrà sporgere (ma non sempre, si noti) nel margine destro. Per essere sicuri di "curarle" tutte, si possono evidenziare dando subito prima dell'inizio del documento il comando

```
\overfullrule=<lunghezza>
```

che accanto a ciascuna riga che sporge dal margine stampa un  (più o meno grande a seconda del valore assegnato a `<lunghezza>`, esprimibile in una delle unità di misura tipografiche accettate da \LaTeX).

Si possono risolvere situazioni problematiche eliminando qualche spazio indivisibile non necessario o mettendo in display alcune formule matematiche poco leggibili in linea. Infine, si usino il meno possibile lunghi URL nel corpo del testo: li si metta in una nota al piede (se sono pochi) oppure, se molti, in un elenco alla fine del documento.

10.2 Problemi verticali

I difetti d’impaginazione verticali riguardano la divisione in pagine del documento.

Oggetti in testo e fuori testo

Gli oggetti in testo causano problemi d’impaginazione a volte irrisolvibili perché, di regola, non possono essere spezzati tra due pagine. Basta evitarli il più possibile.

Gli oggetti fuori testo, invece, sono molto più flessibili. Tenendo presente che si può considerare “ottimale” il risultato fintanto che oggetto e relativo riferimento si trovano sulla stessa pagina o al massimo in due pagine opposte, qualche oggetto potrebbe comunque non piacere dove \LaTeX ha pensato di metterlo:

- potrebbe essere finito un po’ troppo lontano dal punto “ottimale”;
- ci si potrebbe trovare con due oggetti in una pagina e nessuno in quella successiva, quando li si preferirebbe distribuiti più omogeneamente;
- si potrebbe volere in una pagina di soli oggetti un oggetto che invece \LaTeX ha messo in una pagina con del testo.

Come fare? I primi due casi si risolvono arretrando il codice dell’oggetto di qualche capoverso, mentre il terzo dandogli la preferenza `p`. Si possono sbrogliare tutte le altre situazioni intricate usando oculatamente le preferenze di collocazione e facendo qualche prova. A volte può risolvere la situazione `\clearpage`, spiegato di seguito.

Orfani e vedove nel corpo del documento

In tipografia si usa chiamare *orfano* la prima e unica riga di un capoverso in fondo alla pagina e *vedova* l’ultima riga di un capoverso in cima a una pagina nuova. Sono orrori tipografici da evitare con cura certissima. \LaTeX è programmato per evitare automaticamente queste due situazioni: quando non ce la fa, si ricorra ai consigli descritti di seguito.

Il metodo più semplice per risolvere la faccenda è riformulare il capoverso in questione per diminuirlo o aumentarlo di una riga oppure valutare la possibilità di spezzare un capoverso in due o di riunirne due in uno nei pressi del problema.

Interrompere la pagina corrente

Di seguito si descrivono brevemente i principali comandi per cambiare pagina. Tutti interrompono la pagina corrente *nel punto in cui vengono dati*, ma:

- `\newpage` comincia semplicemente una pagina nuova;
- `\pagebreak` prima di cominciare la nuova pagina stiracchia in verticale il contenuto di quella in cui viene dato per riempirla al meglio;
- `\clearpage` prima di cominciare la nuova pagina dice a \LaTeX di stampare tutti gli oggetti già definiti e che non hanno ancora trovato posto sulle pagine, se ce ne sono;
- `\cleardoublepage` si comporta come il comando precedente ma inserendo, se necessario, una pagina bianca prima di cominciare quella nuova (utile nei documenti impostati per la stampa in fronte/retro).

E per finire...

Infine, non rimane che rileggere il tutto *più volte* per stanare i refusi: è praticamente impossibile non scovarne in ogni documento che superi la decina di pagine.

Bibliografia

Beccari, Claudio e Tommaso Gordini

- 2016 *Codifiche in T_EX e L^AT_EX. Dal sorgente al PDF. Guida pratica per lavorare con successo*, <http://www.guitex.org/home/images/doc/GuideGuIT/introcodifiche.pdf>.

Fairbairns, Robin

- 2014 *The UK T_EX FAQ*, <http://texdoc.net/texmf-dist/doc/generic/FAQ-en/newfaq.pdf>.

Goossens, Michel, Frank Mittelbach e Johannes Braams

- 2004 *The L^AT_EX Companion*, Addison-Wesley, Reading (Massachusetts).

Gregorio, Enrico

- 2010 *L^AT_EX. Breve guida ai pacchetti di uso più comune*, <http://profs.sci.univr.it/~gregorio/breveguida.pdf>.
2011 *Introduzione a T_EXworks*, <http://profs.sci.univr.it/~gregorio/introtextworks.pdf>.
2013 *Installare T_EXLive 2012 su Ubuntu*, <http://profs.sci.univr.it/~gregorio/textlive-YEAR-ubuntu.pdf>.

Guiggiani, Massimo e Lapo Filippo Mori

- 2008 «Consigli su come *non* maltrattare le formule matematiche», *Ar_ST_EXnica*, 5, http://www.guit.sssup.it/arstexnica/download_ars/arstexnica05.pdf.

Knuth, Donald Ervin

- 1984 *The T_EXbook*, Addison-Wesley, Reading (Massachusetts).

Lamport, Leslie

- 1994 *L^AT_EX. A Document Preparation System*, Addison-Wesley, Reading (Massachusetts).

Pakin, Scott

- 2015 *The Comprehensive L^AT_EX Symbol List*, <http://www.ctan.org/tex-archive/info/symbols/comprehensive/symbols-a4.pdf>.

Pantieri, Lorenzo e Tommaso Gordini

- 2017 *L'arte di scrivere con L^AT_EX*, http://www.lorenzopantieri.net/LaTeX_files/ArteLaTeX.pdf.