

The Not So Short
Introduction to $\text{\LaTeX 2}_{\epsilon}$

$\text{\LaTeX 2}_{\epsilon}$ 입문

85분 동안 익히는 $\text{\LaTeX 2}_{\epsilon}$

by Tobias Oetiker

Hubert Partl, Irene Hyna and Elisabeth Schlegl

Version 3.7, 14. April, 1999

김강수, 강운배, 장대훈, 김재우 옮김

3.7-kr, 2000년 2월 27일

Copyright ©1999 Tobias Oetiker and all the Contributors to LShort. All rights reserved.

This document is free; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This document is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this document; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

이 문서는 프리(free)입니다. 자유 소프트웨어 재단(FSF)에 의해 제출된 GNU GPL(일반 공개 라이선스) 제2판 또는 그 이후 버전이 정하는 바에 따라 자유롭게 재배포하고 수정할 수 있습니다.

이 문서는 유용하게 쓰이기를 바라는 마음으로 배포합니다. 그러나 아무런 보증도 하지 않습니다. 심지어 상업성이나 특정 목적에 적합하다는 보증도 하지 않습니다. 자세한 사항은 GNU GPL을 참조하십시오.

이 문서와 함께 GNU GPL 사본을 받으셨을 것입니다. 그렇지 않다면 Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA로 연락하십시오.

한국어판 저작권

이 문서의 한국어판 저작권은 GNU GPL을 따릅니다.

감사의 말

이 책에는 오스트리아에서 나온 독일어로 쓰여진 L^AT_EX 2.09 안내 책자로부터 인용한 것이 많다. 저자들은 다음과 같다.

Hubert Partl <partl@mail.boku.ac.at>
Zentraler Informatikdienst der Universität für Bodenkultur Wien
Irene Hyna <Irene.Hyna@bmwf.ac.at>
Bundesministerium für Wissenschaft und Forschung Wien
Elisabeth Schlegl <no_email>
in Graz

이 독일어 문서가 보고 싶으면, 아래 사이트에서 Jörg Knappen이 L^AT_EX 2_ε에 맞게 개선한 버전을 찾아보면 된다. CTAN:/tex-archive/info/lkurz

이 책을 쓰면서, comp.text.tex에 글을 올리는 사람들에게 자문을 구하였는데 많은 분들이 답변을 보내 주셨다. 아래에 기록한 분들은 교정, 제안, 자료들을 보내주셔서 이 글이 더 좋은 글이 되도록 도와주신 분들이다. 이 글이 현재 모양을 갖추는 데 이들의 도움이 컸다. 이 분들 모두에게 진심으로 감사드리고자 한다. 당연히, 이 책에서 잘못된 부분이 있다면 그건 순전히 내 책임이다. 제대로 기록된 것이 있다면 그건 틀림없이 아래 있는 분들이 내게 몇 자 적어주신 것 중에 있었을 것이다.

Rosemary Bailey, Friedemann Brauer David Carlisle, Christopher Chin, Chris McCormack, Wim van Dam, David Dureisseix, Elliot, David Frey, Robin Fairbairns, Erik Frisk, Alexandre Guimond, Cyril Goutte, Greg Gamble, Neil Hammond, Rasmus Borup Hansen, Martien Hulsen, Werner Icking, Jakob, Eric Jacoboni, Alan Jeffrey, Byron Jones, David Jones, Johannes-Maria Kaltenbach, Andrzej Kawalec, Alain Kessi, Christian Kern, Jörg Knappen, Kjetil Kjernsmo, Maik Lehradt, Martin Maechler, Claus Malten, Hubert Partl, John Reffling, Mike Ressler, Brian Ripley, Young U. Ryu, Chris Rowley, Hanspeter Schmid, Craig Schlenter, Josef Tkadlec, Didier Verna, Fabian Wernli, Fritz Zaucker, Rick Zacccone, and Mikhail Zotov

서문

L^AT_EX [1]은 과학 및 수학 문서를 작성하는 데 적당한 조판 시스템으로서 대단히 높은 조판 품질을 얻을 수 있게 한다. 또한 단순한 편지글에서 완전한 단행본에 이르기까지 여러 종류의 문서를 만드는 데도 적합하다. L^AT_EX은 T_EX [2]을 조판 엔진으로 사용한다.

이 길지 않은 입문서는 L^AT_EX 2_ε에 대해서 설명한다. 그리고 L^AT_EX을 응용해서 쓰는 방법을 충분히 소개할 생각이다. L^AT_EX 시스템에 대한 완전한 설명을 보려면 [1, 3]을 참조하면 된다.

L^AT_EX은 PC, Mac, UNIX, VMS 시스템 등 대부분의 컴퓨터에서 동작한다. 대학의 컴퓨터실에는 대개 L^AT_EX이 이미 설치되어 있어서 즉시 사용할 수 있는 경우가 많을 것이다. 현재 자신이 이용할 수 있는 시스템에 L^AT_EX이 어떻게 설치되어 있고 어떻게 사용해야 하는지에 대한 안내는 *Local Guide* [4]에 나와 있을 것이다. 만약 L^AT_EX를 사용하는 데 문제가 있다면 이 책을 권한 사람에게 물어 보라. 이 책은 시스템에 L^AT_EX를 설치하거나 설정하는 방법에 대해서가 아니라, L^AT_EX으로 문서 작성하는 방법을 알려주고자 하는 것이다.

이 책은 다섯 개의 장으로 이루어져 있다.

제 1 장 L^AT_EX 2_ε 문서의 기본 구조를 설명한다. 그리고 L^AT_EX의 역사에 관해서도 간략히 알게 될 것이다. 이 장을 읽고 나면 L^AT_EX에 대해서 개략의 지식을 얻게 될 것인데, 이것은 골격에 불과하지만 이어지는 장에서 설명되는 내용과 연결하면 L^AT_EX 전체에 대해 잘 알 수 있게 될 것이다.

제 2 장 문서를 조판하는 세부사항에 대한 내용이 시작된다. 필수적인 L^AT_EX 명령과 환경 거의 대부분을 설명한다. 이 장을 읽은 후에는 처음으로 간단한 문서를 작성할 수 있게 될 것이다.

제 3 장 L^AT_EX으로 수식을 표현하는 방법을 설명한다. 또, 많은 예제를 통해 L^AT_EX의 가장 주요한 장점인 수식 표현을 어떻게 할 것인지 쉽게 이해하도록 해줄 것이다. 이 장의 끝에는 L^AT_EX에서 사용할 수 있는 수학 기호들의 목록을 표로 만들어 두었다.

제 4 장 찾아보기와 참고문헌 만들기, EPS 그림 파일 포함하기, 그림의 다른 유용한 기능 확장에 대하여 설명한다.

제 5 장 L^AT_EX이 만들어내는 표준 문서 레이아웃을 변경하는, 약간 위험스러울지도 모르는 내용이 들어 있다. 즉, L^AT_EX이 만들어 내는 아름다운 결과물을 어떻게 하면 엉망으로 만들 수 있는지 알려준다.

순서대로 각 장을 읽도록 하는 것이 중요하다. 이 책은 그다지 크지 않은 규모이다. 특히 예제를 주의깊게 읽어야 한다. 왜냐하면 이 책 전체에 걸쳐 나타나는 예제에 아주 중요한 정보들이 많이 담겨 있기 때문이다.

L^AT_EX 관련 문서나 자료가 필요하다면 CTAN ftp 아카이브를 살펴보면 된다. 예를 들면, ctan.tug.org (미국), ftp.dante.de (독일), ftp.tex.ac.uk (영국) 가운데 한 곳을 찾아보라. 이 곳들이 아니라도 자신에게 가장 가까운 곳에서 운영하는 미러 사이트를 이용할 수 있다.

만약 자신의 개인 시스템에서 L^AT_EX를 운영하고 싶다면 CTAN:/tex-archive/systems에서 적당한 것을 찾아보면 될 것이다.

만약 이 책의 내용에 추가하거나, 삭제하거나, 변경해야 할 내용이 있다는 생각이 들면 나(저자)에게 알려 달라. 특히 L^AT_EX 초보자가, 이 개설서 중 어떤 부분이 이해하기 쉽다거나, 더 잘 설명할 수 있는 방법이 있을 것 같다는 등의 제안을 해주는 데 관심이 있다.

Tobias Oetiker <oetiker@ee.ethz.ch>

Department of Electrical Engineering,
Swiss Federal Institute of Technology

이 책의 최신 버전은 CTAN:/tex-archive/info/lshort에서 찾을 수 있다.

차례

감사의 말	iii
서 문	v
제 1 장 알아두어야 할 기본사항	1
1.1 어떻게 불러야 하는가?	1
1.1.1 T _E X	1
1.1.2 L ^A T _E X	1
1.2 기본사항	2
1.2.1 저자, 디자이너, 조판사	2
1.2.2 레이아웃 디자인	3
1.2.3 장점과 단점	3
1.3 L ^A T _E X 입력 파일	4
1.3.1 공백	4
1.3.2 특수문자	5
1.3.3 L ^A T _E X 명령	5
1.3.4 주석	6
1.4 입력 파일의 구조	6
1.5 레이아웃	8
1.5.1 문서 클래스	8
1.5.2 패키지	8
1.5.3 쪽 양식	11
1.6 대규모 프로젝트	11
제 2 장 텍스트의 조판	13
2.1 텍스트와 언어의 구조	13
2.2 줄바꿈과 문단나눔	15
2.2.1 문단 정렬	15
2.2.2 하이픈	16
2.3 특수 문자와 기호 문자	17
2.3.1 따옴표	17
2.3.2 대시와 하이픈	18
2.3.3 말줄임표 (. . .)	18
2.3.4 합자(ligature)	18
2.3.5 억양 표시와 특수 문자	19

2.4	국제 언어 지원	19
2.5	단어 간 간격	21
2.6	제목과 장, 절	21
2.7	상호 참조	23
2.8	각주	23
2.9	강조	24
2.10	환경	24
2.10.1	Itemize, Enumerate, Description 환경	25
2.10.2	Flushleft, Flushright, Center 환경	25
2.10.3	Quote, Quotation, Verse 환경	26
2.10.4	그대로 보이기(Verbatim 환경)	26
2.10.5	Tabular 환경	27
2.11	떠다니는 표와 그림	29
제 3 장	수식의 조판	33
3.1	일반사항	33
3.2	수식 모드의 그룹짓기	35
3.3	수학 기호 및 수식의 표현	35
3.4	수식 모드의 간격	38
3.5	수식의 수직 정렬	39
3.6	수학 글꼴의 크기	41
3.7	정리(theorem), 법칙,...	42
3.8	굵은 기호 문자	43
3.9	수식 기호문자	44
제 4 장	특별한 기능	51
4.1	EPS 그림 포함하기	51
4.2	참고문헌	53
4.3	찾아보기	54
4.4	멋진 쪽머리글(Fancy Headers)	55
4.5	Verbatim 패키지	55
제 5 장	L^AT_EX을 자기에 맞게 바꾸기	57
5.1	새로운 명령, 환경, 패키지	57
5.1.1	새로운 명령	58
5.1.2	새로운 환경	59
5.1.3	사용자 패키지	59
5.2	글꼴과 크기	60
5.2.1	글꼴 바꾸기 명령	60
5.2.2	경고	62
5.2.3	조언	63
5.3	간격	63
5.3.1	줄 간격	63
5.3.2	문단의 형식	63
5.3.3	수평 간격	64
5.3.4	수직 간격	65

5.4	페이지 레이아웃	65
5.5	길이 문제, 몇 가지 더	67
5.6	상자	68
5.7	선 그리기	70
	참고문헌	73
	역자후기	83

그림 차례

1.1	TEX System의 구성	2
1.2	최소한의 L ^A T _E X 파일	7
1.3	실제 학술지 논문의 보기	7
4.1	fancyhdr 설정 예제	56
5.1	패키지의 예	59
5.2	페이지 레이아웃의 변수	66

표 차례

1.1 문서 클래스들	9
1.2 문서 클래스의 옵션	9
1.3 L ^A T _E X과 함께 배포되는 패키지 몇 가지	10
1.4 L ^A T _E X에서 미리 정의된 쪽 양식	11
2.1 억양표시와 특수문자	19
2.2 개체가 놓일 수 있는 허용 범위	29
3.1 수학 모드의 억양 표시 붙은 글자	44
3.2 그리스 소문자	44
3.3 그리스 대문자	44
3.4 이항 관계 연산 기호	45
3.5 이항 연산 기호	45
3.6 큰 기호	46
3.7 화살표	46
3.8 시작과 끝을 나타내는 기호	46
3.9 시작과 끝을 나타내는 큰 기호	46
3.10 기타 기호 문자	47
3.11 수학과 관계없는 기호	47
3.12 AMS의 시작과 끝을 나타내는 기호	47
3.13 AMS 그리스 문자와 히브리 문자	47
3.14 AMS 이항 관계 연산 기호	48
3.15 AMS 화살표	48
3.16 AMS 부정 관계 연산 기호와 화살표	49
3.17 AMS 이항 연산 기호	49
3.18 AMS 기타 기호 문자	50
3.19 수학적 알파벳	50
4.1 graphicx 패키지의 key 명칭	52
4.2 Index Key 사용 예제	54
5.1 글꼴	60
5.2 글꼴 크기	61
5.3 표준 클래스의 글꼴 실제 크기	61
5.4 수식용 글꼴	61
5.5 T _E X 단위	65

1	각 장별 번역자와 교열자	84
---	-------------------------	----

제 1 장

알아두어야 할 기본사항

이 첫 장의 앞부분은 $\text{\LaTeX} 2_{\epsilon}$ 의 철학과 역사에 대하여 짧막하게 소개하고, 뒷부분에서 \LaTeX 문서의 기본 구조를 알아보겠다. 이 장을 읽고나면 \LaTeX 이 어떻게 작동하는지에 대하여 대강 알게 될 것이다. 이 개략적 지식을 지침으로 하여 계속 읽어가면서, 이어지는 각 장의 새로운 정보들을 합치면 전체를 이해하기가 훨씬 쉬워질 것이다.

1.1 어떻게 불러야 하는가?

1.1.1 \TeX

\TeX 은 Donald E. Knuth [2]가 만든 컴퓨터 프로그램이다. 텍스트와 수학적식을 조판하기 위해 만들어졌다. Knuth는 \TeX 조판 엔진을 1977년에 쓰기 시작했는데, 미국수학회가 자신의 논문을 출판할 때 저지른 일에 짜증이 나서였다. 실제 그는 1974년에 얼마 동안 아무 논문도 기고하지 않았는데, “최종 출판물이 보기에 아주 고통스러웠기” 때문이었다 한다. 오늘날 우리가 사용하는 \TeX 은 1982년에 발표되었고 그 후 몇 년간 조금씩 개선되었다. \TeX 은 계속 발전하여 최근 들어서 매우 안정화되었고, 마침내 Knuth는 이것이 버그가 없는(virtually bug-free) 프로그램이라고 선언하게 되었다. \TeX 의 판 번호는 원주율(π)을 따라서 들어가는데, 현재 3.14159까지 전개되었다.

\TeX 은 “테흐”(독일어 “Ach”나 스코틀랜드어 “Loch”의 “ch”처럼)로 발음한다.¹ 텍스트(ASCII)로 써야 할 때는 \TeX 을 TeX 이라고 쓴다.

1.1.2 \LaTeX

\LaTeX 은 \TeX 의 매크로 패키지이다. 이것은 글 쓰는 이로 하여금 미리 정의된 전문적인 레이아웃을 이용하여 고품위의 타이포그래피(조판조정)로 자신의 저작을 조판하고 인쇄하도록 해 준다. \LaTeX 은 원래 Leslie Lamport [1]에 의하여 작성되었다. \TeX 포맷기를 조판 엔진(typesetting engine)으로 사용한다.

1994년에 \LaTeX 패키지는 Frank Mittelbach가 주도하는 $\text{\LaTeX} 3$ 팀에 의하여 개선되었다. 여기서 그 동안의 오랜 개선 요구가 받아들여졌고 그 몇 해 전

¹우리나라에서는 “텍”이라고 읽는 것이 관례이다. 이 번역본에서 그렇게 썼다.[역자]

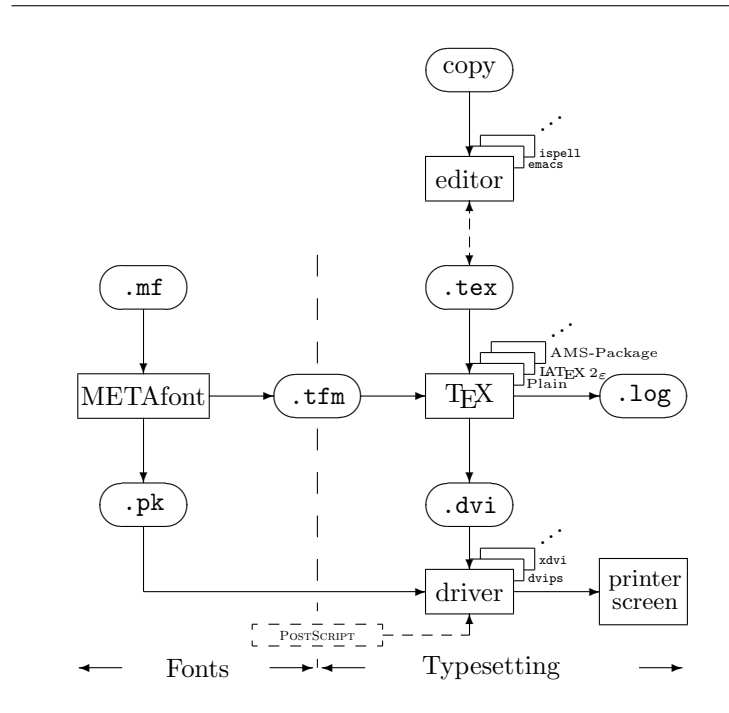


그림 1.1: TeX System의 구성

L^AT_EX 2.09가 릴리즈된 이래 산발적으로 쌓여 온 패치 버전을 전부 재정비하였다. 새로운 버전은 이전의 것과 구별하기 위하여 L^AT_EX 2_ε라고 부른다. 이 책은 L^AT_EX 2_ε에 대한 것이다.

L^AT_EX은 “레이텍(Lay-Tech)” 또는 “라텍(Lah-Tech)”으로 읽는다. 텍스트 환경에서는 LaTeX이라고 쓴다. L^AT_EX 2_ε는 “레이텍 투 이(Lay-Tech Two e)”라고 읽고 LaTeX2e라고 쓴다.

위의 그림 1.1은 TeX과 L^AT_EX 2_ε가 어떻게 함께 동작하는지를 보여준다. 이 그림은 Kees van der Laan의 `wots.tex`에서 가져온 것이다.

1.2 기본사항

1.2.1 저자, 디자이너, 조판사

어떤 것을 출판하려면, 저자는 타자기로 원고를 작성하여 출판사에 넘겨준다. 그 회사의 복디자이너 가운데 한 명이 이 문서의 레이아웃(외양: 문단폭, 글꼴, 제목줄 앞뒤의 여백...)을 결정한다. 디자이너는 자신의 지시사항을 원고에 써 넣은 다음 조판사에게 넘긴다. 조판사는 이 지시사항에 따라 책을 조판한다.

디자이너는 저자가 원고를 쓰면서 염두에 둔 것이 무엇인가를 알아내고자 한다. 그 원고의 내용과 자신의 전문지식에 입각하여 장의 제목줄, 인용문, 예문, 수식 등등을 결정하는 것이다.

L^AT_EX을 사용하는 경우, L^AT_EX이 디자이너 역할을 한다. 그리고 TeX을 조

판기로 이용한다. 그러나 \LaTeX 은 컴퓨터 프로그램에 지나지 않으므로, (사람인 디자이너에게 맡기는 경우보다) 더 많은 지시사항을 알려주어야 할 것이다. 저자는 자기 글의 논리적 구조를 나타내는 정보를 텍스트와 함께 추가로 제공하여야 하는데, 이 정보는 “ \LaTeX 명령”이라는 형태로 텍스트에 써넣는다.

이러한 방식은 이른바 WYSIWYG²라고 알려진, *MS Word*나 *아래아한글*³ 같은 요즘 거의 모든 워드 프로세서들이 취하는 방법과는 상당히 다른 것이다. WYSIWYG 방식은, 저자가 컴퓨터에 텍스트를 타이핑하면서 대화식으로 문서의 레이아웃을 조정해간다. 즉, 자신의 최종 작업 결과가 출력되었을 때의 모양을 화면으로 보면서 작업할 수 있는 것이다.

\LaTeX 을 사용할 때는 텍스트를 입력하는 동안 최종 출력물의 모양을 보지는 못한다. 그러나 그 입력 파일을 \LaTeX 으로 처리(컴파일)한 후에는 화면으로 최종 출력결과를 미리볼 수 있다. 문서를 실제 프린트하기 전에 화면으로 결과를 보면서 수정하는 것도 가능하다.

1.2.2 레이아웃 디자인

타이포그래피 디자인은 전문적인 일이다. 이 분야에 익숙지 않은 저자들이 자주 저지르는 심각한 실수는, 책을 디자인하는 것이 대부분 미관의 문제라고 생각하는 것이다. “디자인이 잘 된 거? 문서가 보기 좋으면 되는 거 아냐?” 그러나, 문서라는 것은 읽히기 위해 있는 것이지 화랑에 걸어두려고 있는 것이 아니다. 가독성과 이해가능성은 그 외관의 아름다움보다 훨씬 더 중요하다. 예를 들어보자.

- 글꼴 크기와 제목 번호붙이기는 독자에게 장/절의 구조가 명확하게 전달 되도록 선택되어야 한다.
- 행 길이는 독자의 시야를 넘어서지 않을 정도로 짧아야 하지만, 쪽를 보기 좋게 채울 정도로 길어야 한다.

WYSIWYG 시스템을 사용하는 저자들은 외관상 보기에는 좋지만 도대체 구조적이지 못하고 일관성도 없는 문서를 만들어내는 경우가 많다. \LaTeX 은 저자 자신의 문서에 논리적인 구조를 선언하게 함으로써 이러한 구조 오류를 피하게 해준다. 그런 다음 이 구조에 가장 적당한 레이아웃을 \LaTeX 이 선택하는 것이다.

1.2.3 장점과 단점

WYSIWYG에 익숙한 사람들이 \LaTeX 을 사용하는 사람을 만나거나 하면 “일반 워드 프로세서에 비해 \LaTeX 이 가진 장점이 무엇이나”거나, 반대로 워드 프로세서의 좋은 점이 무엇이나에 대해 논쟁이 벌어지는 일이 종종 있다. 이런 논쟁이 벌어지면 자리를 피하는 것이 상책이다. 왜냐하면 이런 식의 논의는 건잡을 수 없게 비화하기 일쑤이기 때문인데, 그래도 가끔은 피할 수 없을 때가 있는 법이다...

그래서 여기 탄약 몇 발을 마련해 둔다. \LaTeX 이 일반 워드 프로세서에 비해서 더 좋은 점은 다음과 같다:

²What you see is what you get.

³원문에는 *Corel Word-Perfect*[역주]

- 전문적으로 디자인된 레이아웃을 사용할 수 있다. 그래서 문서가 “인쇄된” 것과 거의 동일하다.
- 수학적 조판이 매우 쉽다.
- 사용자는 문서의 논리적 구조를 지시하는 몇 가지 기억하기 쉬운 명령어들만 익히면 된다. 이 명령들은 대부분 눈으로 보이는 문서의 모양을 주물러야 하는 것이 아니다.⁴
- 각주, 상호참조, 목차, 참고문헌 등 매우 복잡한 구조들도 아주 쉽게 만들어진다.
- L^AT_EX 기본 패키지 이외에도 여러가지 조판상의 필요를 충족시키는 무료의 추가 패키지들이 많다. POSTSCRIPT 그림을 포함하게 하거나 표준을 정확하게 지키는 참고문헌을 작성하게 하거나 하는 패키지들을 예로 들 수 있겠다. 이 추가 패키지들은 대부분 *The L^AT_EX Companion* [3]에 설명되어 있다.
- L^AT_EX은 구조화가 잘 된 문서를 작성하도록 저자를 유도한다. L^AT_EX의 동작 방식이 바로 그것이다—구조를 지정하는 것.
- L^AT_EX 2_ε의 조판 엔진인 T_EX은 매우 이식성이 높고, 무료이다. 그러므로 T_EX 시스템은 거의 모든 하드웨어 플랫폼에서 실행가능하다.

L^AT_EX에도 단점도 있다. 내 입장에서 그 중 사리에 맞아 보이는 것은 별로 없지만, 그런 단점을 수백 가지 지적해 내는 사람도 틀림없이 있는 것이다. ;-))

- L^AT_EX은 아무 생각 없는 머리가 빈 사람들하고는 잘 맞지 않는다...
- 미리 만들어진 레이아웃 인자의 고정값을 조절해가면서 쓸 수 있기는 하지만, 완전히 새로운 레이아웃을 설계하는 것은 어렵고 시간이 많이 드는 일이다.⁵
- 구조화되지 않고 비체계적인 문서를 작성하기가 너무나 어렵다.

1.3 L^AT_EX 입력 파일

L^AT_EX에 의하여 처리되는 것은 텍스트 파일(plain ASCII text)이다. 어떤 텍스트 편집기(에디터)로도 만들 수 있다. 여기에는 문서의 내용은 물론이고, L^AT_EX에게 텍스트를 조판하는 방법에 관하여 알려주는 지시사항들도 기록된다.

1.3.1 공백

“공백문자(whitespace characters)”, 즉 빈 칸(blank), 탭(tab) 등은 L^AT_EX에서 모두 동일하게 “스페이스”로 처리된다. 계속되는 여러 개의 공백문자들은 하

⁴즉, 논리적 구조를 선언하는 것이지 시각적 모양을 조절하는 것이 아니라는 뜻.[역주]

⁵소문에 의하면 앞으로 나올 L^AT_EX3 시스템에서는 이것이 주안점이 될 거라 한다.

나의 “스페이스”로 취급된다. 행의 첫머리에 있는 공백문자들은 무시되고 한 번의 줄바꿈(개행:linebreak) 역시 “공백문자”로 간주된다.

두 줄 사이에 빈 줄을 하나 넣으면 문단(paragraph)의 끝을 나타낸다. 여러 개의 빈 줄은 하나의 빈 줄과 같다. 아래 예를 보자. 왼쪽은 입력 파일에 적힌 텍스트의 내용이고 오른쪽은 그 출력된 형태를 나타낸다.

```
It does not matter whether you
enter one or several      spaces
after a word.
```

```
An empty line starts a new
paragraph.
```

It does not matter whether you enter one or
several spaces after a word.

An empty line starts a new paragraph.

1.3.2 특수문자

다음 기호들은 유보된 문자들로서 L^AT_EX에서 특별한 의미를 갖거나 어떤 글꼴로도 나타낼 수 없는 것들이다. 이것을 텍스트에 직접 써넣는다면 대개 프린트되지 않을 것이며 그럼에도 L^AT_EX에게 강제로 실행하도록 하면 원치 않는 결과를 초래할 것이다.

`$ & % # _ { } ~ ^ \`

앞으로 보게 되겠지만, 이러한 문자를 문서에서 사용하려면 백슬래시(\)를 더해 주어야 한다.

`\$ \& \% \# _ \{ \}`

`$ & % # - { }`

그밖의 많은 기호와 부호, 수학적식에서 쓰이는 문자, 억양 표시가 붙은 문자들은 특별한 명령을 써서 출력한다. 백슬래시 문자 \는 백슬래시를 그 앞에 하나 더 붙인다고(\\) 얻어지는 아니다. 이 부호는 줄바꿈(개행)에 사용된다.⁶

1.3.3 L^AT_EX 명령

L^AT_EX 명령은 대소문자를 구별하며 다음과 같은 두 가지 형태 가운데 하나를 취한다:

- 백슬래시 \ 로 시작하여 문자만으로 이루어진 이름을 갖는다. 명령 이름은 공백이나 숫자 또는 ‘문자 아닌 것’이 오면 끝난다.
- 백슬래시 다음에 딱 한 개의 특별한 글자가 온다.

L^AT_EX은 명령 다음의 공백문자를 무시한다. 명령 다음에 빈 칸을 두고 싶으면 {}와 빈 칸을 써넣거나 명령 이름 뒤에 특별한 칸 띄우기 명령을 써야 한다.

⁶ ‘\’를 얻으려면 `\backslash` 명령을 사용한다.

`{}`는 \LaTeX 이 명령 이름 다음에 나오는 빈 칸들을 “먹어버리지”(무시하지) 않도록 만든다.

```
I read that Knuth divides the
people working with \TeX{} into
\TeX{}nicians and \TeX{} perts.\
Today is \today.
```

```
I read that Knuth divides the people working
with TEX into TEXnicians and TEXperts.
Today is February 27, 2000.
```

인자(parameter)가 필요한 명령도 있다. 인자는 명령 이름 다음의 중괄호 `{ }` 속에 써넣어야 한다. 어떤 명령에는 옵션 인자(optional parameters)가 필요한 경우도 있는데, 명령 바로 다음에 대괄호 `[]`를 쓰고 그 안에 써넣는 것이 이것이다. 다음은 \LaTeX 명령을 사용하는 보기이다. 무슨 말인지 모르겠더라도 걱정할 것 없다. 나중에 설명이 될 것이다.

```
You can \textsl{lean} on me!
```

```
You can lean on me!
```

```
Please, start a new line
right here!\newline
Thank you!
```

```
Please, start a new line right here!
Thank you!
```

1.3.4 주석

\LaTeX 은 입력 파일을 처리해가다가 `%` 문자를 만나면 그 줄(행)의 나머지 부분과 줄바꿈을 무시한다. 그리고 다음 줄 첫머리에 오는 공백문자도 모두 무시한다. 이것은 입력 파일 작성시 최종 인쇄판에는 나타나지 않는 주석문을 쓰고 싶을 때 사용할 수 있다.

```
This is an % stupid
% Better: instructive <----
example: Supercal%
          ifragilist%
          icexpialidocious
```

```
This is an example: Supercalifragilisticexpiali-
docious
```

`%` 문자는 공백문자나 줄바꿈 문자가 허용되지 않는 긴 줄을 나누어 입력할 수 있게 해준다.

1.4 입력 파일의 구조

\LaTeX 이 입력 파일을 처리할 때는 그 파일이 일정한 구조를 따르고 있다고 가정한다. 그러므로 모든 입력 파일은 다음 명령으로 시작해야 한다.

```
\documentclass{...}
```

이 명령은 지금 쓰려 하는 문서가 어떤 종류의 것인지 설정하는 것이다. 이 다음에 전체 문서의 모양(스타일)에 영향을 주는 명령들을 포함하거나 \LaTeX 시

시스템에 새로운 기능을 추가하는 패키지들을 포함할 수도 있다. 패키지를 포함할 때는 다음과 같은 형태의 명령을 쓴다.

```
\usepackage{...}
```

이 설정 과정이 모두 끝난 뒤에,⁷ 문서의 주요부가 시작된다. 다음이 문서 주요부의 시작을 나타내는 명령이다.

```
\begin{document}
```

이제 문장을 몇 가지 L^AT_EX 명령과 섞어가면서 입력하면 된다. 문서의 끝에는

```
\end{document}
```

명령을 추가한다. 이것은 L^AT_EX에게 처리해야 하는 부분의 끝임을 알려준다. 이 명령 이후에 나오는 것은 모두 무시된다.

그림 1.2에 보인 것은 최소한의 L^AT_EX 파일 내용이다. 이보다 좀 복잡한 입력 파일이 그림 1.3에 있다.

```
\documentclass{article}
\begin{document}
Small is beautiful.
\end{document}
```

그림 1.2: 최소한의 L^AT_EX 파일

```
\documentclass[a4paper,11pt]{article}
\usepackage{latexsym}
\author{H.~Partl}
\title{Minimalism}
\frenchspacing
\begin{document}
\maketitle
\tableofcontents
\section{Start}
Well, and here begins my lovely article.
\section{End}
\ldots{} and here it ends.
\end{document}
```

그림 1.3: 실제 학술지 논문의 보기

⁷\documentclass와 \begin{document} 사이에 오는 부분을 전처리 부분(*preamble*)이라 한다.

1.5 레이아웃

1.5.1 문서 클래스

L^AT_EX에게 입력된 파일을 처리하게 할 때 첫번째로 제공되어야 하는 정보는 저자가 만들려는 문서의 유형(type)이다. 이것은 `\documentclass` 명령으로 지정한다.

```
\documentclass[options]{class}
```

여기서 *class*라는 인자는 만들어질 문서의 유형을 지시한다. 표 1.1에 이 책에서 설명하는 문서 클래스들을 요약해 놓았다. L^AT_EX 2_ε 배포본에는 편지나 슬라이드 등의 문서를 위한 클래스가 더 포함되어 있다. *option* 인자는 문서 클래스의 동작을 사용자의 의도에 맞게 조절하는 데 쓰인다. 여러 개의 옵션은 쉼표로 구분한다. 표준적 문서 클래스에 사용되는 일반적 옵션을 표 1.2에 요약하였다.

보기 : L^AT_EX 문서의 첫 줄이 다음과 같이 시작되었다면,

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

이것은 L^AT_EX에게 문서를 *article*로 작성하고 기본 글꼴 크기를 11포인트로 하고, A4 용지에 양면 인쇄하기에 적당하도록 레이아웃을 잡으라는 것을 지시하고 있다.

1.5.2 패키지

글을 써가다 보면, 기본 L^AT_EX만으로는 해결할 수 없는 문제를 만날 때가 있을 것이다. 그림을 포함하려 하거나, 채색 글씨를 쓰고 싶을 때, 또는 문서에 소스 코드를 삽입해야 할 경우, L^AT_EX의 기능을 향상시켜야 할 필요를 느끼게 된다. 이러한 기능 향상은 패키지를 통해서 이루어진다. 다음 명령을 써서 패키지를 사용 가능하게 한다.

```
\usepackage[options]{package}
```

여기서 *package*란, 패키지의 이름을 가리키고, *options*란 패키지가 수행해야 할 특정 기능을 나타내는 지시어들을 말한다. L^AT_EX 2_ε 기본 배포판에 함께 따라오는 패키지들도 있다. (표 1.3을 참조.) 그밖에 독자적으로 제공되는 패키지도 있다. 자신의 컴퓨터에 어떤 패키지들이 설치되어 있는지 알고 싶으면 거기에 있는 *Local Guide* [4]를 찾아보면 된다. L^AT_EX 패키지에 대해서는 *The L^AT_EX Companion* [3]을 보면 많은 것을 알 수 있을 것이다. 이 책은 여러 패키지에 대한 설명은 물론이고 L^AT_EX 2_ε를 확장하기 위해 자신이 직접 패키지를 작성하는 방법에 대해서도 설명하고 있다.

표 1.1: 문서 클래스들

<code>article</code>	과학 학술지, 프레젠테이션, 짧은 보고서, 프로그램 문서, 초대장 등에 쓰이는 아티클용 클래스
<code>report</code>	여러 장으로 이루어진 긴 보고서, 작은 책, 박사학위 논문 등에 쓰이는 클래스
<code>book</code>	진짜 책을 만들기 위한 클래스.
<code>slides</code>	슬라이드 제작용 클래스. 이 클래스는 큰 산세리프 글꼴을 사용한다. 이것 대신 <code>FoilTeX</code> ^a 의 사용을 고려해 볼 만도 함.

^aCTAN:/tex-archive/macros/latex/contrib/supported/foiltex

표 1.2: 문서 클래스의 옵션

<code>10pt</code> , <code>11pt</code> , <code>12pt</code>	문서의 기본 글꼴 크기를 설정한다. 아무 옵션도 주지 않으면 <code>10pt</code> 로 간주된다.
<code>a4paper</code> , <code>letterpaper</code> , ...	종이 크기를 지정한다. 기본 크기는 <code>letterpaper</code> 이다. 이외에도, <code>a5paper</code> , <code>b5paper</code> , <code>executivepaper</code> , <code>legalpaper</code> 등을 지정할 수 있다..
<code>fleqn</code>	수식을 가운데 정렬하지 않고 왼쪽 정렬하도록 정한다.
<code>leqno</code>	수식의 번호를 오른쪽이 아니라 왼쪽에 붙이도록 한다.
<code>titlepage</code> , <code>notitlepage</code>	표지를 만든 다음 새로운 쪽으로 시작할 것인지 그렇지 않을 것인지를 지정한다. <code>article</code> 클래스는 새로운 쪽으로 시작하지 않는 것이 기본값이다. <code>report</code> 와 <code>book</code> 은 표지를 한 쪽으로 만드는 것이 기본값이다.
<code>twocolumn</code>	<code>L^AT_EX</code> 에게 2단으로 조판하라고 지시하는 것.
<code>twoside</code> , <code>oneside</code>	양면인쇄용 또는 단면인쇄용 출력물을 만들라고 지시하는 것이다. <code>article</code> 과 <code>report</code> 는 단면, <code>book</code> 클래스는 양면이 기본값이다. 이 옵션은 문서의 모양에 관한 것일 뿐이다. 즉, <code>twoside</code> 옵션을 주었다고 해서 프린터가 실제로 양면으로 출력해주는 것은 아니다.
<code>openright</code> , <code>openany</code>	새로운 장을 홀수쪽에서만 시작할 것인지 홀/짝 구분 없이 바로 다음 쪽에서 시작할 것인지를 지정하는 것이다. 이 옵션은 장(chapter)이라는 개념이 없는 <code>article</code> 클래스에서는 동작하지 않는다. <code>report</code> 클래스는 다음 쪽에서 시작하는 것이 기본값으로 되어 있고 <code>book</code> 클래스에서는 홀수쪽에서 시작하는 것이 기본값이다.

표 1.3: L^AT_EX과 함께 배포되는 패키지 몇 가지

doc L ^A T _E X 프로그램의 문서화를 가능하게 한다. doc.dtx ^a 와 <i>The L^AT_EX Companion</i> [3]에 설명.
exscale 확장 수학 글꼴을 크기별로 조절할 수 있게 하는 기능을 제공한다. ltxscale.dtx에 설명.
fontenc L ^A T _E X이 어떤 글꼴 인코딩을 사용할 것인지 지정한다. ltoutenc.dtx에 설명.
ifthen 다음과 같은 문장 형식을 지원한다. 'if...then do...otherwise do...' ifthen.dtx와 <i>The L^AT_EX Companion</i> [3]에 설명.
latexsym L ^A T _E X 기호문자(심볼) 글꼴 사용을 가능하게 하려면 latexsym 패키지를 사용해야 한다. latexsym.dtx와 <i>The L^AT_EX Companion</i> [3]에 설명.
makeidx 찾아보기(index)를 만드는 명령을 제공한다. 이 글 4.3 절과 <i>The L^AT_EX Companion</i> [3]에 설명.
syntonly 문서를 처리(컴파일)하지만 조판하지는 않도록 한다. syntonly.dtx와 <i>The L^AT_EX Companion</i> [3]에 설명되어 있다. 문서의 오류를 확인하고 싶을 때 쓸 수 있다.
inputenc 입력 파일의 인코딩 방법을 지정할 수 있게 해 준다. 예를 들면, ASCII, ISO Latin-1, ISO Latin-2, 437/850 IBM 코드 페이지, 애플 매킨토시, 넥스트, ANSI-윈도, 및 사용자 정의 인코딩 등. inputenc.dtx에 설명.

^a이 파일은 시스템에 틀림없이 설치되어 있을 것이다. *.dtx 파일로부터 dvi 파일을 얻으려면 쓰기 권한이 있는 디렉토리에서 명령행에 `latex doc.dtx`라고 쓰면 된다. 이 표에서 언급된 다른 파일도 마찬가지다.

1.5.3 쪽 양식

L^AT_EX에는 세 가지 쪽머리글/쪽바닥글 조합(이것을 쪽 양식(page style)이라 한다.)이 미리 정의되어 있다.

```
\pagestyle{style}
```

이 명령의 *style* 인자가 이 가운데 어느 것을 사용할지 지정한다. 표 1.4에 미리 정의된 쪽 양식을 요약해 두었다.

표 1.4: L^AT_EX에서 미리 정의된 쪽 양식

plain 쪽바닥글 중앙에 쪽 번호를 붙인다. 이것이 기본값이다.

headings 현재의 장 제목과 쪽 번호를 각 쪽머리글에 인쇄하고 쪽바닥글은 비워둔다. (이 책에서 사용된 양식이다.)

empty 쪽머리글/쪽바닥글을 모두 비운다.

현재 쪽의 쪽 양식을 바꾸는 데는 다음 명령을 쓴다.

```
\thispagestyle{style}
```

자신만의 쪽머리글과 쪽바닥글을 만드는 방법이 *The L^AT_EX Companion* [3]에 나와 있고, 이 책 55 쪽 4.4 절에서도 찾아볼 수 있다.

1.6 대규모 프로젝트

큰 문서를 작업하려면 입력 파일을 몇 개 부분으로 나누어 작업하고 싶을 때가 있다. L^AT_EX은 이를 위해 두 가지 명령을 제공한다.

```
\include{filename}
```

이 명령을 사용하면 문서의 일부에 다른 파일의 내용을 삽입할 수 있다. L^AT_EX이 이 명령을 만나면 *filename*으로 삽입되는 부분을 처리할 때 새로운 쪽으로 시작한다는 점에 주의하라.

또 하나의 명령은 전처리(preamble) 부분에서 쓰일 수 있다. 이것은 `\include` 명령에 지정된 파일이 여기에 열거된 것 중 하나일 때만 그 파일을 삽입하라고 L^AT_EX에게 지시하는 것이다.

```
\includeonly{filename,filename,...}
```

이 명령이 문서의 전처리 부분에서 실행된 후에는 `\include` 명령으로 삽입하는 파일들 가운데 여기서 지정된 것만이 실제로 삽입될 것이다. 파일 이름과 쉼표 사이에 빈 공간이 없어야 한다는 데 주의한다.

`\include` 명령은 파일을 삽입할 때 새로운 쪽을 만들고 거기에서부터 조판하기 시작한다. `\includeonly`를 쓰는 경우 이렇게 하는 것이 유리하다. 왜냐하면 `include`된 파일 중에 몇 개를 제외하더라도 쪽 나눔 위치가 이동하지 않기 때문이다. 가끔 이렇게 하는 것이 바람직하지 못할 때도 있다. 이럴 경우

`\input{filename}`

명령을 쓰면 된다. 이 명령은 다만 지정된 파일을 포함시킬 뿐이고 조절이나 코드 추가를 전혀 하지 않는다.

제 2 장

텍스트의 조판

앞 장에서 $\text{\LaTeX} 2_{\epsilon}$ 문서를 만들기 위한 기본적인 사항을 알게 되었을 것이다. 이 장에서는 진짜 문서를 만들기 위해서 알아야 할 ‘구조’의 나머지 사항에 관해 설명한다.

2.1 텍스트와 언어의 구조

글쓰기의 주요 초점(현대의 일부 ‘무작정 다르게’ 문학¹을 논외로 한다면)은 생각(아이디어)과 정보, 지식을 독자에게 전달하는 것이다. 독자들의 이해가 수월해지려면 이 생각(아이디어)들이 잘 구조화되어 있어야 할 것이다. 그리고 구조를 보다 쉽게 보고 느끼게 하려면 조판 출력(타이포그래피) 형태가 내용의 논리적·의미론적 구조를 잘 반영하고 있어야 한다.

\LaTeX 이 다른 조판 시스템과 다른 점은, 사용자가 텍스트의 논리적·의미론적 구조를 지시하여야만 한다는 것이다. 그러면 \LaTeX 은 지정된 문서 클래스 파일과 여러 스타일 파일에 정의된 “규칙”에 따라서 텍스트의 조판 출력(타이포그래피) 형식을 만들어낸다.

\LaTeX (타이포그래피 일반)에서 가장 중요한 텍스트 단위는 문단(paragraph)이다. 이것을 “텍스트 단위”라고 말하는 이유는, 문단이 한 가지 일관된 생각이나 개념을 반영하는 조판 편집상의 형식이기 때문이다. 다음 몇 절에서 배우게 될 것이 이와 관련된 사항들이다. 예를 들면, 줄바꿈을 하기 위해 `\\`를 쓰는 법, 문단 나누기 위해서 원본 입력 파일에 빈 줄을 넣는 방법 같은 것들이다.

따라서, 새로운 문단은 새로운 생각이 전개되는 곳에서 시작해야 한다. 그렇지 않다면 그냥 줄바꿈만이 쓰여야 한다. 문단을 나누어야 할지 어떨지 확신이 서지 않으면, 글이 사고와 사상의 매체(전달자)라는 관점에서 자신의 텍스트를 살펴보자. 문단을 나누었는데 이전의 생각이 연속되고 있다면 그 문단나눔을 없애야 한다. 같은 문단 안에서 완전히 새로운 생각이 전개되는 곳이 있다면 이 곳에서 문단을 나누어 주어야 한다.

많은 사람들이 문단 나누기를 제대로 하는 게 얼마나 중요한지를 과소평가하고 있다. 심지어 문단 나누기의 의미가 무엇인지조차 모르는 사람도 있고,

¹DAAC(Different At All Cost). 스위스식 독일어 UVA(Um's Verrecken Anders)의 영어 표현.

L^AT_EX을 쓰는 사람 중에는 자신이 문단을 나누고 있는지도 모르는 채로 문단을 나누는 경우도 많다. 이런 실수는 특히 글 속에 수식이 사용될 경우 저지르기 쉽다. 다음 예를 보면서, 왜 어떤 때는 수식 전후로 빈 줄(문단 나눔)이 사용되고 또 어떤 때는 사용되지 않는지 생각해 보라. (만약 여기서 쓰인 명령들을 아직 충분히 알지 못해서 이 예가 무엇을 하자는 것인지 이해할 수 없다면, 이 장과 다음 장을 읽은 다음에 이 절로 다시 돌아와서 읽어보기 바란다.)

Example 1

```
\ldots when Einstein introduced his formula
\begin{equation}
  e = m \cdot c^2 \; ,
\end{equation}
which is at the same time the most widely known
and the least well understood physical formula.
```

Example 2

```
\ldots from which follows Kirchoff's current law:
\begin{equation}
  \sum_{k=1}^n I_k = 0 \; .
\end{equation}

Kirchhoff's voltage law can be derived \ldots
```

Example 3

```
\ldots which has several advantages.
\begin{equation}
  I_D = I_F - I_R
\end{equation}
is the core of a very different transistor model. \ldots
```

(역자의 보충) 한국어로 글을 쓸 때도 새로운 문단을 시작하는 부분에 수식이 있을 때 주의해야 하는 것은 마찬가지로 하겠다. 다음 두 가지 예를 번역해 둔다.

예 1

```
\ldots 아인슈타인이 도입한 다음 공식,
\begin{equation}
  e = m \cdot c^2 \; ;
\end{equation}
은 가장 널리 알려진 것이면서 동시에 가장 이해하는 사람이
적은 공식이다.
```

예 2

```
\ldots로부터 키르히호프의 전류 법칙이 도출된다.
\begin{equation}
\sum_{k=1}^n I_k = 0 \ ; \ .
\end{equation}
```

키르히호프의 전압 법칙은 다음과 같이 유도할 수 있는데 \ldots

문단 다음으로 더 작은 글의 단위는 문장(sentence)이다. 영문 텍스트에서는 약어 기호로 쓰인 마침표보다 문장의 끝을 나타내는 마침표의 뒤에 더 큰 공백을 둔다. L^AT_EX은 저자가 실제로 의도한 것이 문장의 끝이었는지 약어 기호였는지 알아내려 한다. 만약 L^AT_EX이 그것을 잘못 판단한 경우라면, 저자가 직접 L^AT_EX에게 자신이 원하는 바가 무엇인지를 알려주어야 한다. 이 점에 대해 이 장의 후반부에서 설명될 것이다.

나아가, 텍스트의 구조화는 문장의 일부분에까지 적용된다. 대부분의 언어에서 구두점은 상당히 복잡하다. 그러나, 영어나 독일어는 물론이고 어떤 언어라도, 구두점이 무엇을 나타내는 것인지 생각하기만 한다면 그 구두점을 제 자리에 정확하게 찍는 것은 별로 어려운 일이 아니다. 만약 어디에다가 구두점을 찍어야 하는지 확신이 서지 않으면, 그 문장을 소리내어 읽어보고, 모든 구두점마다 짧게 한 숨 멈추어 보는 것이 좋겠다. 그것이 어색한 곳이 있으면, 그 부분의 구두점을 지우도록 하라. 또 만약 어떤 곳에서 숨을 쉬거나 짧은 기식(stop)을 두는 편이 낫겠다는 느낌이 든다면, 그 곳에 구두점을 찍으면 된다.

끝으로, 텍스트의 문단들은 다시 논리적으로 보다 높은 수준에서 구조화되어야 한다. 즉, 문단을 모아서 장(chapter), 절(section), 소절(subsection) 등으로 배치하여야 하는 것이다. 이러한 높은 수준의 구조화가 어떤 식으로 실현되는가는 자명하다. 왜냐하면 예를 들어, 절의 제목을 나타내기 위해 `\section{The Structure of Text and Language}`라고 입력한다면 그 조판 효과가 어떠하리라는 것은 그 자체로 분명하기 때문이다.

2.2 줄바꿈과 문단나눔

2.2.1 문단 정렬

단행본은 각 행이 똑같은 길이를 가지도록 조판되는 경우가 많다. L^AT_EX은 문단 내용 전체를 최적화하여 단어 사이에 적절한 줄바꿈과 공백을 삽입한다. 필요하다면 줄 끝에 잘 들어맞지 않는 단어를 잘라서 하이픈 처리 하기도 한다. 문단이 어떻게 조판되느냐는 문서 클래스에 따라 달라진다. 보통 많이 쓰는 방식은, 문단 첫 줄을 들여쓰기 하고, 문단과 문단 간격을 줄간격과 같이 잡아 더 넓게 벌리지 않는 것이다. 좀더 자세한 사항은 5.3.2를 참고하기 바란다.

가끔 L^AT_EX에게 줄을 바꾸거나 쪽을 나누라는 명령을 사용자가 내려야 하는 경우도 있다.

다음 명령은 줄을 바꾸어 새 줄로 시작하되 새로운 문단으로 시작하지는 말

라는 명령이다.

\\ 혹은 \newline

다음 명령은 강제 줄바꿈을 하되 쪽 나누기는 방지하는 명령이다.

*

쪽을 나누어 새로운 쪽으로 시작하려면 다음과 같이 한다.

\newpage

줄나눔이나 쪽 나눔에 쓰이는 명령은 다음과 같다.

\linebreak[n], \nolinebreak[n], \pagebreak[n] and \nopagebreak[n]

이 명령의 동작은 이름이 가리키는 바대로다(줄 나누기, 줄 나누지 말기, 쪽 나누기, 쪽 나누지 말기). 사용자가 n 인수를 지정하여 명령의 동작 방식을 선택적으로 바꿀 수 있게 되어 있다. n 값으로는 1에서 4까지의 숫자가 올 수 있다. n 을 4 미만으로 설정하면, L^AT_EX은 결과가 아주 나쁠 경우 사용자 명령을 무시할 수 있다. “break” 명령과 “new” 명령을 혼동하지 말기 바란다.² “break” 명령을 쓰더라도, L^AT_EX은 여전히 쪽의 오른쪽 끝과 쪽 전체 길이를 맞추려고 한다(다음 절에서 설명하듯이). 정말 “새로운 행”을 시작하고 싶으면, “줄 나누기”가 아니라 “새 줄로”를 의미하는 명령을 사용해야 할 것이다. 이름이 왜 다른지 생각해 보라!

L^AT_EX은 항상 가능한 한 최선의 줄바꿈을 하려고 한다. 이러한 상당히 정밀한 기준을 충족시키는 줄바꿈 방법을 찾지 못하면, 한 행을 문단 폭 범위보다 오른쪽으로 나가도록 배치한다. 입력 파일을 처리하던 L^AT_EX은 (“overfull hbox”)라고 불평한다. 이런 일이 일어나는 것은 대개 L^AT_EX이 단어를 나누어서 하이픈 처리할 적절한 위치를 찾지 못한 경우에 일어난다.³ \sloppy 명령을 써서 L^AT_EX의 기준을 낮출 수 있다. 이 명령은 단어 사이의 간격을 더 증가시켜서 오른쪽으로 밀려나는 행이 나타나지 않도록 한다—최종 출력물은 최선의 결과를 보여주지 않겠지만. 그럴 때는 (“underfull hbox”)라는 경고가 나타난다. 대부분의 경우, 결과물은 그리 썩 좋은 모양은 아니다. 반면, \fussy 명령은 반대로 동작한다. 모든 곳에서 L^AT_EX이 불평하는 것을 보고 싶을 때만 쓰도록 하라.

2.2.2 하이픈

L^AT_EX은 필요하다면 언제나 단어를 하이픈 처리 한다. 하이픈 처리 알고리즘이 적절한 하이픈 삽입 위치를 찾지 못하는 경우, 다음과 같은 명령을 사용하

²여기서, linebreak는 “줄 나누기”, newline은 “줄 바꾸기”, 또는 “새 줄로”라고 번역하였다.[역자]

³L^AT_EX이 (Overfull hbox)라는 경고를 보여주었다 해도, 어느 행인지를 찾기는 쉽지 않다. 만약 \documentclass 명령에 draft 옵션을 쓴다면 이런 행들 오른쪽 여백에 굵은 검은 선을 표시해준다.

여기 TeX에게 예외 처리 방식을 알려줌으로써 해결할 수 있다.

```
\hyphenation{word list}
```

이 명령은 인자로 열거된 단어의 하이픈 삽입이 “-”으로 표시된 곳에서만 일어나도록 한다. 이 명령은 입력 파일의 전처리(preamble) 부분에서만 쓸 수 있고, 일반적인 문자로만 이루어진 단어여야 한다. 대소문자 구별은 무시된다. 다음 보기와 같이 했을 때, “hyphenation”의 하이픈 처리 방식은 “Hyphenation”과 같고, “FORTRAN”과 “Fortran”, “fortran”은 어떤 경우에도 하이픈 처리 되지 않을 것이다. 특수 문자나 기호(심볼 문자)는 인자로 올 수 없다.

보기:

```
\hyphenation{FORTRAN Hy-phen-a-tion}
```

명령어 \-는 분절 하이픈을 단어에 삽입한다. 동시에 이것은 그 단어에서 하이픈 처리가 허용되는 유일한 위치가 된다. 이 명령은 특히 특수 문자(예를 들면, 억양 표시 붙은 문자)가 포함된 단어를 하이픈 처리할 때 유용하다. L^AT_EX은 특수 문자가 들어 있는 단어의 하이픈 처리를 자동으로 하지 못하기 때문이다.⁴

```
I think this is: su\per\cal\-%
i\frag\i\lis\tic\ex\pi\-%
al\i\do\cious
```

```
I think this is: supercalifragilisticexpialido-
cious
```

여러 단어들을 같은 줄에 두고 싶을 때는 다음 명령을 사용한다.

```
\mbox{text}
```

이렇게 하면 어떤 경우에도 인자로 처리된 단어들이 함께 묶여 있게 된다.

```
My phone number will change soon.
It will be \mbox{0116 291 2319}.
```

```
The parameter
\mbox{\emph{filename}} should
contain the name of the file.
```

```
My phone number will change soon. It will be
0116 291 2319.
```

```
The parameter filename should contain the
name of the file.
```

2.3 특수 문자와 기호 문자

2.3.1 따옴표

타자기나 보통 에디터에서 글을 쓸 때처럼 따옴표로 "를 사용하면 안된다. 출판용으로 쓰이는 따옴표는, 여는 따옴표와 닫는 따옴표가 다른 부호를 사용한다. L^AT_EX에서는 여는 따옴표 표시로 두 개의 ‘를 사용하고 닫는 따옴표로 두 개의 ’를 사용한다.

⁴ 새로운 폰트인 DC 폰트를 쓰면 가능.

“Please press the ‘x’ key.”

“Please press the ‘x’ key.”

2.3.2 대시와 하이픈

L^AT_EX에는 네 종류의 대시가 있다. 그 중의 세 가지는 각각 몇 번씩 대시를 연이어 써서 얻을 수 있다. 네 번째 것은 사실은 대시가 아니다. 수학의 빼기(minus) 기호이다.

daughter-in-law, X-rated\\
pages 13--67\\
yes---or no? \\
\$0\$, \$1\$ and \$-1\$

daughter-in-law, X-rated
pages 13–67
yes—or no?
0, 1 and −1

세 가지 대시 각각의 이름은 다음과 같다.⁵ ‘-’ 하이픈, ‘—’ 대시, ‘—’ 긴 대시 그리고 ‘-’ 빼기 부호.

2.3.3 말줄임표 (...)

타자기에서는 쉼표나 마침표 간격이 한 글자 간격과 같다. 그러나 도서 인쇄에서 이 부호들은 더 짧은 폭을 갖고 앞 글자에 매우 가깝게 붙는다. 그러므로 “말줄임표”를 쓰려고 마침표를 세 개 연달아 찍으면 안된다. 이러한 점 세 개를 나타내기 위한 특별한 명령이 따로 존재한다.

\ldots

이 명령을 쓴 경우와 마침표를 연이어 찍은 경우를 비교해 보자.

Not like this ... but like this:\\
New York, Tokyo, Budapest, \ldots

Not like this ... but like this:
New York, Tokyo, Budapest, ...

2.3.4 합자(ligature)

합자(ligature)란 영문 조판에서 어떤 문자들이 결합할 때 그것을 한 글자씩 차례로 식자하는 것이 아니라 특별한 기호 문자를 써서 조판하는 것을 가리키는 말이다. 다음 예는 합자를 사용한 경우와 글자를 차례로 나열한 경우를 비교한 것이다.

합자를 이용한 것: ff fi fl ffi ...

합자를 이용하지 않은 것: ff fi fl ffi ...

⁵한국어에서 이 부호들은 영문과 조금 다른 용례로 쓰인다. 우선, 한국어에서는 하이픈이 없다. 대시에 해당하는 문장부호들이 어떻게 사용되는가는 85 페이지 ‘역자의 보충’을 참고하라.[역자]

합자 기능을 이용하고 싶지 않은 경우에는 합쳐지는 두 글자 사이에 `\mbox{}`를 삽입하여 떼어놓으면 된다. 두 단어가 합쳐져서 하나의 단어가 된 경우에 이렇게 할 필요가 있을 수 있다.

```
Not shelfful\\
but shelf\mbox{ }ful
```

```
Not shelfful
but shelfful
```

2.3.5 억양 표시와 특수 문자

L^AT_EX은 여러 언어의 억양 표시들과 특수 문자를 쓸 수 있도록 한다. ‘o’ 문자에 적용된 여러 가지 억양 표시들을 표 2.1에 보였다. 이 표시들은 다른 문자에도 마찬가지로 적용할 수 있다.

i와 j위에 억양표시를 붙이려면 이 두 문자 위의 점을 지워야 한다. 이것은 `\i`, `\j`라고 입력하면 된다.

```
H\^otel, na\"i ve, \'el\'eve,\\
sm\o rrebr\o d, !\'Se\'norita!,\\
Sch\"onbrunner Schlo\ss}
Stra\ss e
```

```
Hôtel, naïve, élève,
smørrebrød, ¡Señorita!,
Schönbrunner Schloß Straße
```

표 2.1: 억양표시와 특수문자

ò	\‘o	ó	\’o	ô	\^o	õ	\~o
ō	\=o	ô	\.o	ö	\"o	ç	\c c
ö	\u o	ö	\v o	ő	\H o	q	\c o
q	\d o	q	\b o	ôo	\t oo		
œ	\oe	Œ	\OE	æ	\ae	Æ	\AE
å	\aa	Å	\AA				
ø	\o	Ø	\O	l	\l	L	\L
ı	\i	j	\j	ı	!’	ı	?‘

2.4 국제 언어 지원

영어가 아닌 다른 언어를 사용해야 한다면, 다음 두 가지 경우에 해당하는 L^AT_EX 설정을 적절히 바꾸어야 한다.

1. 모든 자동으로 생성되는 문자열⁶을 새로운 언어에 적당하도록 맞추어야 한다. 이 변환은 Johannes Braams씨가 만든 babel 패키지를 사용하면 여러 언어에서 가능하다.
2. 새로운 언어의 하이픈 처리 규칙을 L^AT_EX에게 알려주어야 한다. L^AT_EX에게 하이픈 규칙을 알려주기 위해서는 약간 트릭을 쓰는데, 그것은 새로운 하이픈 패턴이 작동하도록 포맷 파일을 다시 만드는 것이다. 이 점에 대해서는 *Local Guide* [4]에 더 자세한 정보가 나와 있어야 할 것이다.

자신의 시스템이 이미 적절히 수정되어 있다면, `\documentclass` 명령 뒤에 다음과 같이 babel 패키지 사용 명령을 추가하면 된다.

```
\usepackage[language]{babel}
```

또, 자신이 사용하는 시스템에서 지원하는 *language*가 Local Guide에 열거되어 있어야 한다. Babel은 선택한 언어의 하이픈 처리 법칙을 자동적으로 활성화시킨다. 만약 현재 시스템에 설치된 L^AT_EX 포맷이 그 언어의 하이픈 처리 규칙을 지원하지 않더라도 babel은 작동한다. 그러나 조판된 문서의 모양은 그다지 좋은 결과를 얻기 어려울 것이다.

babel은 몇 종류의 언어에 대해서 특수 문자 입력을 쉽게 하는 새로운 명령을 지원한다. 독일어의 경우를 예로 들면, 많은 움라우트(äöü) 문자가 쓰인다. babel를 사용하면, `\"o`라고 쓰지 않고 `"o`라고 입력하는 것만으로 ö를 얻을 수 있다.

어떤 컴퓨터 시스템은 키보드로 직접 특수 문자를 입력할 수 있도록 되어 있다. L^AT_EX도 이러한 문자들을 다룰 수 있다. 1994년 12월 L^AT_EX 2_ε가 발표된 이후, 몇 가지 문자 입력 인코딩 지원이 L^AT_EX 2_ε 기본 배포판에 포함되게 되었다. inputenc 패키지를 살펴 보기 바란다. 이 패키지를 사용할 때는, 작성한 입력 파일을 다른 사람의 컴퓨터에서 읽을 수 없을지도 모른다는 사실에 주의해야 한다. 그 컴퓨터 시스템에서는 다른 인코딩 방식을 사용하기 때문이다. 한 예로, 독일어 움라우트 ä가 PC에서는 132로 인코딩되는 반면, ISO-LATIN 1 인코딩을 사용하는 몇몇 Unix시스템에서는 228로 인코딩된다. 따라서, 이 기능을 사용할 때는 주의를 요한다.

폰트 인코딩은 다른 문제이다. 이것은 T_EX 폰트 내의 어느 위치에 각 문자가 저장되지를 정의하는 것이다. 오리지널 컴퓨터 모던(Computer Modern) T_EX 글꼴은 예전 7비트 아스키 문자 집합인 128개 문자만을 포함하고 있다. 억양 표시 붙은 문자가 필요할 때, T_EX은 일반 문자와 억양 표시를 결합하여 문자를 만들어 낸다. 이런 방식은 보기에 좋은 출력물을 얻을 수 있게 하지만 억양 표시 붙은 문자가 사용된 단어에 대해 하이픈 자동 처리가 작동하지 못하는 결과를 초래한다.

다행히, 현재 대부분의 T_EX 배포본은 한 벌의 EC 폰트를 포함하고 있다. 이 폰트는 Computer Modern 폰트와 비슷한 모양을 하고 있지만, 유럽 언어에서 사용되는 억양 표시 붙은 문자를 특수 문자 형태로 포함하고 있는 것이다. 이러한 폰트를 사용하면 비영어권 언어로 쓰여진 문서들에서 하이픈 처리 기

⁶차례, 그림 차례,

능이 더 잘 작동하도록 만들 수 있다. EC 폰트를 쓰고 싶으면 문서의 전처리부(preamble)에 fontenc 패키지 사용을 선언하여 활성화한다.

```
\usepackage[T1]{fontenc}
```

2.5 단어 간 간격

출력물의 오른쪽 끝이 가지런하게 정렬되도록 하기 위해서, L^AT_EX은 단어와 단어 간 공백을 적절히 늘리거나 줄이거나 한다. 글을 보다 읽기 쉽게 하기 위해서 문장의 끝에 약간 더 넉넉한 공백을 삽입한다. L^AT_EX은 문장이 마침표나 물음표, 느낌표로 끝난다고 가정한다. 보통 대문자 다음에 오는 마침표는 약어를 표시하는 경우가 많으므로 마침표가 대문자 다음에 오면 약어를 표시하는 것으로 보고 문장이 끝난 것으로 간주하지 않는다.

L^AT_EX의 이런 가정에 맞지 않는 예외적인 경우라면 사용자가 스스로 그것을 명시해 주어야 한다. 백슬래시 다음에 공백 문자를 두면 폭이 일정한 공백을 만들어 낸다. 물결 표시(tilde) 문자 ‘~’는 줄바꿈이 일어나지 않는 고정폭 공백을 넣어준다. 마침표 앞에 \@ 명령을 쓰면 대문자가 오더라도 문장의 끝나는 곳이라는 사실을 명시한다.

```
Mr.~Smith was happy to see her\\
cf.~Fig.~5\\
I like BASIC\@. What about you?
```

```
Mr. Smith was happy to see her
cf. Fig. 5
I like BASIC. What about you?
```

다음 명령을 쓰면 마침표 뒤에 더 넓은 공백을 넣는 것을 방지할 수 있다.

```
\frenchspacing
```

이것은 L^AT_EX에게 일반 문자의 다음보다 마침표 다음에 더 넓은 공백을 삽입하지 않도록 지시한다. 참고문헌을 쓸 때를 제외하고 비영어권 언어에서 이런 방식을 흔히 볼 수 있다. \frenchspacing을 사용하는 경우 \@ 명령은 필요 없게 된다.

2.6 제목과 장, 절

독자가 글을 읽고 이해하기 쉽게 만들려면, 문서는 반드시 장(chapter)과 절(section), 그리고 소절(subsection) 등으로 나누어져 있어야 한다. L^AT_EX에서는 절 등의 제목을 인자로 취하는 특별한 명령으로 이것을 가능하게 한다. 이 명령을 올바른 순서로 사용하는 것은 전적으로 사용자의 책임이다.

article 클래스에서 사용할 수 있는 장/절 명령은 다음과 같다:

```
\section{...}      \paragraph{...}
\subsection{...}   \subparagraph{...}
\subsubsection{...} \appendix
```

또 report와 book 클래스에서는 다음 장/절 명령 두 개를 더 사용할 수 있다.

```
\part{...}           \chapter{...}
```

article 클래스에는 ‘장’이라는 개념이 없으므로, 이 article을 book에 하나의 장(chapter)처럼 삽입할 수도 있다. 절 제목과의 간격, 번호 붙이기, 제목의 글자 크기 등은 L^AT_EX이 알아서 처리해 준다.

장/절 명령 중 다음 두 가지는 약간 특별하다.

- \part 명령은 장의 번호 매기는 순서에 영향을 주지 않는다.
- \appendix 명령은 인자 없이 쓰인다. 다만 장의 번호 매기기를 숫자에서 문자로 바꾸어 준다.⁷

L^AT_EX은 문서를 마지막으로 컴파일할 때의 장/절 표제와 쪽 번호로부터 차례를 만든다. 다음 명령은 이 명령이 위치한 곳에 만들어진 차례를 넣는다.

```
\tableofcontents
```

새로운 문서의 올바른 차례를 얻기 위해서는 두 번 컴파일(“L^AT_EX을 실행”)해야 한다. 어떨 때는 세 번 컴파일 해야 할 때도 있다. 그럴 경우 L^AT_EX이 컴파일 이 더 필요하다는 것을 알려준다.

위에 언급된 모든 장/절 명령들은 “별표 붙은” 명령을 쓸 수 있다. “별표 붙은” 명령들이란 같은 명령 뒤에 *를 붙여서 사용하는 것을 말한다. 이 명령으로 만들어지는 장/절 표제는 차례에 나타나지도 않고 번호가 붙지도 않는다. 예를 들면 \section{Help} 명령에 이런 효과를 주고 싶으면 \section*{Help}처럼 쓴다.

보통 장/절 표제는 입력된 텍스트 그대로 차례에 나타난다. 그러나 표제가 너무 길어서 차례에 잘 들어맞지 않는 때에는 입력된 표제와는 다르게 차례에 표시할 수도 있다. 실제 표제 앞에 선택 인자를 써서 차례에 나타낼 짧은 제목을 지정하면 된다.

```
\chapter[Read it! It's Exciting]{This is a very long
and especially boring title}
```

문서 전체의 제목(title)을 붙이는 명령은 다음과 같다.

```
\maketitle
```

제목으로 만들어질 내용은 \maketitle 명령이 오기 전에, \title{...} 명령으로 제목줄을, \author{...} 명령으로 저자 이름을 표시하고, 또 선택적으로 \date{...}를 지정한다.

```
\title{...}, \author{...}, \date{...}
```

\author 명령의 인자 안에서는, \and 명령으로 여러 명의 저자 이름을 구분하여 쓸 수 있다.

⁷article에서는 절의 번호 매기기를 바꾼다.

7 쪽의 그림 1.3에서 위에 언급한 여러 가지 명령의 사용례를 보인 적이 있다.

위에서 설명한 장/절 명령 말고도, $\text{\LaTeX} 2_{\epsilon}$ 는 book 클래스에서 쓰기 위한 명령 세 개를 더 갖추고 있다.

```
\frontmatter, \mainmatter, \backmatter
```

이것은 책(출판물)을 몇 부분으로 나눌 때 쓰인다. 이 명령들은 보통 책에서 볼 수 있는 것처럼 장/절 제목 붙이기와 쪽번호 매기기가 각 부분별로 따로 이루어지도록 해준다.

2.7 상호 참조

단행본, 보고서, 논문(기사)에서는 그림, 표 또는 텍스트의 특정 부분을 상호 참조(cross-reference)하는 경우가 많다. \LaTeX 은 상호 참조를 위한 다음 명령을 제공한다.

```
\label{marker}, \ref{marker}, \pageref{marker}
```

여기서 *marker*는 사용자가 적절히 선택하면 되는 식별자이다. \LaTeX 은 \ref 를 만나면, 거기에 해당하는 \label 명령이 쓰인 곳을 찾아서, 장 · 절 · 그림 · 표 · 정리 등의 번호로 \ref 를 바꾸어 넣는다. \pageref 는 해당하는 \label 명령이 쓰인 쪽의 쪽수를 표시한다.⁸ 장/절 표제를 달 때와 마찬가지로, 이들도 직전에 컴파일될 때 생성된 번호를 사용한다.

```
A reference to this subsection
\label{sec:this} looks like:
“see section~\ref{sec:this} on
page~\pageref{sec:this}.”
```

A reference to this subsection looks like: “see section 2.7 on page 23.”

2.8 각주

각주를 붙이는 명령은 다음과 같다.

```
\footnote{footnote text}
```

각주는 현재 쪽의 하단에 인쇄된다. 각주는 항상 그 각주가 참조하는 단어나 문장의 뒤에 놓여야⁹ 한다.¹⁰

⁸ 이 명령들은 그들이 무엇을 참조하고 있는지 모른다. 단지 자동적으로 생성된 마지막 번호를 저장하고 있을 뿐이다.

⁹ “놓다(put)”라는 단어는 가장 일반적인 영어 단어이다.

¹⁰ 따라서, 한 문장 또는 문장의 일부를 참조하는 각주는 쉼표나 마침표 다음에 와야 한다.

Footnotes\footnote{This is
a footnote.} are often used
by people using \LaTeX.

Footnotes^a are often used by people using
L^AT_EX.

^aThis is a footnote.

2.9 강조

원고를 타자로 칠 때는 중요한 단어를 강조하려면 밑줄을 긋는다. 그러나 인쇄
시에는 원고에 밑줄이 그어진 부분을 *이탤릭* 글꼴로 처리한다.¹¹ L^AT_EX이 제공
하는 다음 명령은 텍스트를 강조 처리 하라는 것이다.

```
\emph{text}
```

이 명령이 실제로 어떻게 작용할 것인가는 전후 문맥에 의존한다. 다음 예에서
\emph 안의 \emph는 그 앞이 *이탤릭* 글꼴이므로 보통 글꼴로 바꾸어 강조를 나
타내고 있다.

```
\emph{If you use  
emphasizing inside a piece  
of emphasized text, then  
\LaTeX{} uses the  
\emph{normal} font for  
emphasizing.}
```

*If you use emphasizing inside a piece of empha-
sized text, then L^AT_EX uses the normal font for
emphasizing.*

즉, L^AT_EX에게 무언가를 강조하라고 하는 것과 다른 글꼴을 사용하라고 말
하는 것에는 차이가 있다는 것을 알아야 한다.

```
\textit{You can also  
\emph{emphasize} text if  
it is set in italics,}  
\textsf{in a  
\emph{sans-serif} font,}  
\texttt{or in  
\emph{typewriter} style.}
```

*You can also emphasize text if it is set in
italics, in a sans-serif font, or in typewriter
style.*

2.10 환경

텍스트 일부를 특별한 목적으로 조판하기 위해 L^AT_EX은 여러 종류의 환경(envi-
ronment)을 제공하여 다양한 텍스트 처리가 가능하게 한다.

```
\begin{name} text \end{name}
```

여기서 *name*은 환경의 이름이다. 환경은 그 호출 순서가 지켜 지는 한, 한 환
경 안에서 다른 환경을 여러 번 호출할 수 있다.

¹¹ 한글 서적의 출판 관행은 *이탤릭* 처리가 아니라 고딕체로 바꾸는 것이다. 이 번역본에서 그렇
게 하였다.[역주]

```
\begin{aaa}...\begin{bbb}...\end{bbb}...\end{aaa}
```

아래에서 중요한 환경들을 설명하겠다.

2.10.1 Itemize, Enumerate, Description 환경

`itemize` 환경은 단순 나열, `enumerate` 환경은 숫자 번호가 붙은 나열, `description` 환경은 사전에서 볼 수 있는 것 같은 설명을 위한 나열을 나타내는 데 쓰인다.

```
\flushleft
\begin{enumerate}
\item You can mix the list
environments to your taste:
\begin{itemize}
\item But it might start to
look silly.
\item[-] With a dash.
\end{itemize}
\item Therefore remember:
\begin{description}
\item[Stupid] things will not
become smart because they are
in a list.
\item[Smart] things, though, can be
presented beautifully in a list.
\end{description}
\end{enumerate}
```

1. You can mix the list environments to your taste:
 - But it might start to look silly.
 - With a dash.
2. Therefore remember:

Stupid things will not become smart because they are in a list.

Smart things, though, can be presented beautifully in a list.

2.10.2 Flushleft, Flushright, Center 환경

`flushleft`와 `flushright` 환경은 문단을 왼쪽 정렬시키거나 오른쪽 정렬시킨다. `center` 환경은 글을 가운데 정렬시킨다. 줄바꿈을 하기위해 `\\`를 써넣지 않으면 `LATEX`이 스스로 줄바꿈을 결정한다.

```
\begin{flushleft}
This text is\\ left-aligned.
\LaTeX{} is not trying to make
each line the same length.
\end{flushleft}
```

This text is
left-aligned. `LATEX` is not trying to make each
line the same length.

```
\begin{flushright}
This text is right-\\aligned.
\LaTeX{} is not trying to make
each line the same length.
\end{flushright}
```

This text is right-
aligned. `LATEX` is not trying to make each line
the same length.

```
\begin{center}
At the centre\of the earth
\end{center}
```

At the centre
of the earth

2.10.3 Quote, Quotation, Verse 환경

quote 환경은 중요한 구절이나 예문을 인용하는 데 쓴다.

```
A typographical rule of thumb
for the line length is:
\begin{quote}
No line should contain more than
66~characters.
```

```
This is why \LaTeX{} pages have
such large borders by default.
\end{quote}
That's why multicolumn print is
often used in newspapers.
```

A typographical rule of thumb for the line
length is:

No line should contain more than
66 characters.

This is why L^AT_EX pages have such
large borders by default.

That's why multicolumn print is often used in
newspapers.

이와 유사한 환경이 두 개 더 있다. quotation과 verse 환경이 그것인데, quotation 환경은 문단 들여쓰기를 하기 때문에 여러 문단에 걸친 긴 인용문을 처리할 때 쓴다. verse 환경은 줄바꿈이 중요한 시(詩) 등을 쓸 때 유용하다. 각 행은 \를 사용하여 나누어지고 각 연(verse) 뒤에는 빈 줄이 하나 따르게 된다.

```
I know only one English poem by
heart. It is about Humpty Dumpty.
\begin{flushleft}
\begin{verse}
Humpty Dumpty sat on a wall:\
Humpty Dumpty had a great fall.\
All the King's horses and all
the King's men\
Couldn't put Humpty together
again.
\end{verse}
\end{flushleft}
```

I know only one English poem by heart. It is
about Humpty Dumpty.

Humpty Dumpty sat on a wall:
Humpty Dumpty had a great fall.
All the King's horses and all the
King's men
Couldn't put Humpty together
again.

2.10.4 그대로 보이기(Verbatim 환경)

\begin{verbatim}과 \end{verbatim} 사이에 오는 텍스트는 마치 타자를 친 것처럼, L^AT_EX 명령이 전혀 수행되지 않고 모든 줄바꿈과 공백들이 입력된 그대로 출력된다.

문단 안에서는, 다음 명령으로 비슷한 효과를 얻을 수 있다.

```
\verb+text+
```

+는 처음과 끝을 나타내는 문자의 한 예이다. *와 빈 칸을 제외한 다른 문자를 + 대신 사용할 수도 있다. 이 책에 보이는 많은 L^AT_EX 예들은 이 명령을 사용하여 조판된 것이다.

The \verb|\ldots| command \ldots

```
\begin{verbatim}
10 PRINT "HELLO WORLD ";
20 GOTO 10
\end{verbatim}
```

The \ldots command ...

```
10 PRINT "HELLO WORLD ";
20 GOTO 10
```

```
\begin{verbatim*}
the starred version of
the      verbatim
environment emphasizes
the spaces in the text
\end{verbatim*}
```

```
the_starred_version_of
the_verbatim
environment_emphasizes
the_spaces_in_the_text
```

위에 나온 별표가 붙은 \verb 명령의 사용법도 비슷하다.

```
\verb*|like  this :-) |
```

```
like_this_:-)_
```

verbatim 환경과 \verb 명령은 다른 명령의 인자 안에서 사용되면 안 된다.

2.10.5 Tabular 환경

tabular 환경은 보기좋은 표를 만드는 데 사용한다. 원한다면 가로줄이나 세로줄도 그릴 수 있다. L^AT_EX은 열(column)의 폭을 자동으로 결정한다.

```
\begin{tabular}{table spec}
```

이 명령에서, *table spec* 인자는 표의 모양을 지정한다. l은 각 열(column)을 왼쪽 정렬하고, r은 오른쪽 정렬한다. 가운데 정렬을 하려면 c를 사용한다. p{width}를 쓰면 주어진 길이에 맞추어서 각 열 내의 텍스트를 정렬하고 자동 줄바꿈을 해준다. l는 세로줄을 나타낸다.

tabular 환경 내에서, &는 다음 열로 넘김, \\는 새로운 행의 시작, \hline는 가로줄 삽입에 쓰인다.

```
\begin{tabular}{|r|l|}
\hline
7C0 & hexadecimal \\
3700 & octal \\
11111000000 & binary \\
\hline
1984 & decimal \\
\hline
\end{tabular}
```

7C0	hexadecimal
3700	octal
11111000000	binary
1984	decimal

```
\begin{tabular}{|p{4.7cm}|}
\hline
Welcome to Boxy's paragraph.
We sincerely hope you'll
all enjoy the show.\\
\hline
\end{tabular}
```

Welcome to Boxy's paragraph. We sincerely hope you'll all enjoy the show.

열 분리자는 @{...} 형태로 구성을 지시할 수 있다. 이 명령은 열 사이의 공백을 없애고 소괄호 속에 있는 것(무엇이든 상관없다)으로 바꾸어 채운다. 이 명령의 일상적 용례 중의 하나를 다음 쪽에 설명했는데, 이것은 소수점을 기준으로 한 숫자 정렬의 보기이다. 또 다른 응용 가능성은, 표 안에서 각 칸의 안쪽 여백을 없애는 것이다. @{}으로 지정하면 된다.

```
\begin{tabular}{@{} l @{}}
\hline
no leading space\\
\hline
\end{tabular}
```

no leading space

```
\begin{tabular}{l}
\hline
leading space left and right\\
\hline
\end{tabular}
```

leading space left and right

숫자로 된 열을 소수점에 맞추어 정렬하는 방법이 L^AT_EX 내부명령으로 되어 있지는 않으므로,¹² 약간의 트릭을 사용하여 구현해보려 한다. 즉, 두 개의 칼럼을 사용하되 왼쪽 칼럼은 오른쪽 정렬하고 오른쪽 칼럼은 왼쪽 정렬한 다음 칼럼 분리자를 소수점에 해당하는 점으로 처리하는 것이다. \begin{tabular}와 같은 줄에 쓴 @{} 명령은 열 사이의 공백을 “.”로 바꾸어주므로, 이것을 마치 소수점처럼 보이게 하는 것이다. 열 구분자(&)로 정수부와 나머지 소수부를 분리하는 것을 잊어 먹지 말기 바란다! 그런 다음, 칼럼의 표제는 \multicolumn 명령을 사용해서 방금 만든 숫자열 “칼럼”(사실은 두 개의 칼럼) 위에 하나의 칼럼인 것처럼 놓을 수 있다.

¹²만약 사용자의 시스템에 ‘tools’ 패키지 꾸러미가 설치되어 있으면, dcolumn 패키지를 살펴보기 바란다.

```
\begin{tabular}{c r @{.} l}
Pi expression      &
\multicolumn{2}{c}{Value} \\
\hline
 $\pi$                 & 3.1416 & \\
 $\pi^\pi$              & 36.46  & \\
 $(\pi^\pi)^\pi$         & 80662.7 & \\
\end{tabular}
```

Pi expression	Value
π	3.1416
π^π	36.46
$(\pi^\pi)^\pi$	80662.7

2.11 떠다니는 표와 그림

요즘의 출판물들은 대부분 많은 그림과 표를 포함하고 있다. 이런 것들은 특별한 처리가 필요하다. 왜냐하면 쪽이 나누어져서는 안 되기 때문이다. 그림이나 표의 크기가 현재 쪽에 다 들어가지 않을 경우 생각해 볼 수 있는 방법은 표나 그림이 나오면 그 때마다 항상 새로운 쪽을 시작하는 것이다. 그러나 이렇게 하면 그 앞 페이지에 너무 많은 빈 공간이 남게 되고 이것은 보기에 좋지 않다.

이 문제의 해결책은 현재 페이지를 텍스트로 이어 채우면서, 이 페이지에 맞지 않는 그림이나 표는 ‘떠다니게’ 하는 것이다. L^AT_EX은 이러한 떠다니는 개체에 해당하는 두 가지 환경을 제공한다. 하나는 표이고, 또 하나는 그림이다. 이 환경을 최대한 유용하게 사용하려면 L^AT_EX이 이 떠다니는 개체를 어떻게 조작하는지를 대강이라도 알고 있는 것이 중요하다. 그렇지 않으면, L^AT_EX은 이것을 원하는 위치에 놓지 않기 때문에 오히려 커다란 혼란에 빠질 수도 있다.

먼저, L^AT_EX이 이런 떠다니는 개체를 위해 제공하는 명령을 살펴보자.

`figure`나 `table` 환경으로 둘러 싸인 것은 무엇이든 떠다니는 개체로 간주된다. 두 환경은 모두 *placement specifier*라 불리는 선택 인자를 지원한다.

```
\begin{figure}[placement specifier] 혹은 \begin{table}[placement specifier]
```

이 선택적 지시자들은 떠다니는 개체가 이동할 수 있도록 허락된 위치를 L^AT_EX에게 알려준다. *placement specifier*는 개체가 놓일 수 있는 허용 범위의 첫글자를 따서 만들어졌다. 표 2.2를 보기 바란다.

표는 예를 들면 다음과 같은 줄로 시작한다.

표 2.2: 개체가 놓일 수 있는 허용 범위

선택 사항	개체가 놓일 수 있는 허용 범위 ...
<code>h</code>	<i>here</i> 의 약자. 이 명령이 사용된 바로 그 위치. 작은 개체에 유용하다.
<code>t</code>	<i>top</i> 의 약자. 쪽의 윗부분.
<code>b</code>	<i>bottom</i> 의 약자. 쪽의 아래 부분.
<code>p</code>	<i>page</i> 의 약자. 그림이나 표만 모아둔 특별한 쪽에 위치.
<code>!</code>	개체의 위치를 제어하는 변수 ^a 를 거의 고려하지 않도록 함.

^a한 페이지에 올 수 있는 개체의 최대값 같은 것들

```
\begin{table}[!hbp]
```

선택 인자 `[!hbp]`는 \LaTeX 이 표를 바로 여기 놓든지(h), 바닥에 놓든지(b) 혹은 특별한 페이지에 놓든지(p)하라는 것이고, 이렇게 할 때 설명 보기 좋지 않더라도 상관없다는(!) 것이다. 기본값으로 설정되어 있는 것은 `[tbp]`이다.

\LaTeX 은 각각의 떠다니는 개체를 모두 사용자가 정의한 대로 위치시킨다. 만약 현재 쪽에 개체가 올 수 없다면, 그림 대기열이나 표 대기열(queue)¹³로 가서 출력을 기다리게 된다. 새로운 쪽이 시작되면, \LaTeX 은 먼저 이 쪽이 대기열의 떠다니는 개체들로 채울 특별한 ‘플로트’ 페이지가 될 수 있는지 여부를 확인한다. 만약 이것이 불가능하면, 각 대기열의 첫번째 개체를 텍스트의 현재 위치에서 나온 것으로 보고 처리하려 한다. 즉 \LaTeX 은 다시 그 개체의 위치 지정자(‘h’는 더이상 가능하지 않으므로 무시되지만)에 따라 놓으려고 해본다. 텍스트 중에 새로운 떠다니는 개체가 나오면 그것은 적절한 대기열로 보내진다. \LaTeX 은 각 그림과 표가 나타난 순서를 엄격하게 유지한다. 바로 이 이유 때문에 놓을 수 없는 그림 뒤에 나온 모든 그림들이 문서의 끝에 나타나는 것이다. 그러므로,

만약 \LaTeX 이 사용자가 원하는 대로 그림을 위치시키지 못한다면,
그것은 두 떠다니는 개체의 대기열 가운데 어떤 하나가 다른 하나
를 방해하기 때문이다.

약간 어렵게 설명을 해왔는데, 아직 표와 그림 환경에 대하여 언급할 것이 조금 더 남아 있다.

```
\caption{caption text}
```

위의 명령을 사용하여, 그림이나 표의 제목(캡션)을 달 수 있다. 그림이나 표의 최종 번호와 “Figure(그림)” 또는 “Table(표)”이라는 문자열은 \LaTeX 이 자동적으로 붙여준다.

```
\listoffigures 그리고 \listoftables
```

이 두 명령은 `\tableofcontents` 명령과 비슷하게 작동하는데, 각각 그림과 표의 차례를 출력해 준다. 이 차례에는 그림과 표의 캡션이 전부 다시 나타날 것이다. 만약 긴 제목을 사용하게 된다면, 이 목차에 나타날 짧은 제목을 따로 만들어야 한다. `\caption` 명령 뒤의 대괄호 내에 원하는 짧은 제목을 써주면 된다.

```
\caption[Short]{LLLLLoooooonnnnnngggg}
```

`\label`과 `\ref`로 문서 내의 그림이나 표를 참조할 수 있다.

다음의 예는 정사각형을 하나 그린 후, 그것을 문서에 삽입하는 것이다. 이것은 완성된 문서 내에 어떤 그림을 집어넣을 공간을 미리 확보하고 싶을 때 유용하다.

¹³이것은 먼저 들어간 것이 먼저 나오는(fifo – ‘first in first out’) 큐(queue)이다.

```
Figure~\ref{white} is an example of Pop-Art.
\begin{figure}[!hbp]
\makebox[\textwidth]{\framebox[5cm]{\rule{0pt}{5cm}}}
\caption{Five by Five in Centimetres.} \label{white}
\end{figure}
```

위에 예에서, L^AT_EX이 그림을 바로 이 자리에(h)¹⁴ 놓으려고 정말 어려운 (!) 시도를 할 것이다. 만약 이것이 불가능하면, 그림을 쪽의 바닥에 (b) 위치시키려 할 것이다. 그림에도 불구하고 그림을 현재 쪽에 위치 시키는 것이 불가능하면, L^AT_EX은 이 그림(과 표 대기열에서 대기 중인 표들)을 담는 어떤 특별한 쪽에 위치시키는 것이 가능한지 여부를 결정할 것이다. 특별한 플롯 페이지를 만들기에는 그림이나 표가 충분치 않다면¹⁵ L^AT_EX은 새로운 쪽을 만들면서 다시 한번 이 그림을 마치 텍스트의 바로 그 위치에서 생긴 것처럼 처리할 것이다.

어떤 경우 다음 명령들이 필요할지도 모른다.

`\clearpage` 혹은 `\cleardoublepage`

이것은 L^AT_EX으로 하여금 출력을 기다리고 있는 모든 개체들을 즉시 출력하고 새로운 쪽을 시작하게 한다. `\cleardoublepage`는 빈 쪽을 하나 만들어서라도 왼쪽 페이지로 가도록 한다.

L^AT_EX 2_ε 문서에 어떻게 POSTSCRIPT 그림을 삽입하는지에 대해서는 이 책의 뒷부분에서 배우게 될 것이다.

¹⁴기다리고 있는 그림 대기열이 비어 있다고 가정할 때

¹⁵플롯 페이지를 만들기에 충분한가의 여부는 각 대기열에서 대기 중인 표와 그림을 모은 세로 길이가 `\floatpagefraction`이라는 상수값을 충족하는가의 여부에 달려 있다. 표준 L^AT_EX 클래스에서는 일반적으로 이 값이 0.5로 지정되어 있는데, `\renewcommand`로 사용자가 조절하는 것도 가능하다.[역자]

제 3 장

수식의 조판

자, 이 장에서는 \TeX 의 가장 큰 장점 하나를 다루어보자. 바로 수식 조판이다. 여기서 다루는 것은 대략적인 사용법에 불과하다는 걸 알아두자. 하지만 대다수 사용자에게 필요한 사항은 이 장에서 다룬 내용이면 충분하리라 본다. 원하는 수학적 표기 방법을 여기서 찾을 수 없다 해도 실망할 것은 없다. \LaTeX ¹ 패키지나 다른 패키지를 이용하면 대부분 해결할 수 있을 것이다.

3.1 일반사항

\LaTeX 은 수식의 표현을 위해 “수식 모드”라는 특별한 모드를 사용한다. 문장 내에 포함되는 수식은 $\backslash($ 와 $\backslash)$, $\backslash$$ 와 $\backslash$$ 또는 \backslashbegin{math} 와 \backslashend{math} 사이에 넣어서 표현한다.

Add a^2 squared and b^2 squared
to get c^2 squared. Or, using
a more mathematical approach:
 $c^2 = a^2 + b^2$

Add a squared and b squared to get c squared.
Or, using a more mathematical approach: $c^2 =$
 $a^2 + b^2$

\TeX is pronounced as
 $\tau\epsilon\chi$.
100 m³ of water
This comes from my \heartsuit

\TeX is pronounced as $\tau\epsilon\chi$.
100 m³ of water
This comes from my \heartsuit

더 큰 수식이나 공식은 줄마다 \backslash 을 사용하여 하나씩 조판하는 것보다 수식 보여주기(*display*) 하는 것이 더 나운데, 이 경우 $\backslash[$ 와 $\backslash]$ 또는 $\backslashbegin{displaymath}$ 와 $\backslashend{displaymath}$ 로 수식을 둘러싸면 된다. 이렇게 하면 수식의 오른쪽에 수식번호가 표시되지 않는다. 만약 수식번호를 보이게 하고 싶으면 *equation* 환경을 사용한다.

¹CTAN:/tex-archive/macros/latex/required/amslatex

Add a squared and b squared
to get c squared. Or, using
a more mathematical approach:
$$c^2 = a^2 + b^2$$

And just one more line.

Add a squared and b squared to get c squared.
Or, using a more mathematical approach:

$$c^2 = a^2 + b^2$$

And just one more line.

수식에 붙인 `\label`을 `\ref`로 참조하려면 다음과 같이 한다.

$$\epsilon > 0$$

From $(\ref{eq:eps})$, we gather
$$\dots$$

$$\epsilon > 0 \quad (3.1)$$

From (3.1), we gather ...

수식 보이기(`displaymath`) 환경의 수식은 본문 중에 들어간 수식과 표현방
식이 다르다는 점에 주의하라.

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

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

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

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

수식 모드와 텍스트 모드 사이에는 다른 점이 있다. 예를 들면 수식 모드에
서는,

1. 대부분의 빈 칸과 줄바꿈 문자들은 아무런 효과를 갖지 않는다. 수식의
빈 칸과 줄바꿈은 수식 표현 자체에서 논리적으로 나타나야 할 자리에 나
타나거나, `\,`, `\quad` 혹은 `\qquad`과 같은 특수 명령어를 사용한 경우에
나타날 뿐이다.
2. 빈 줄은 허용하지 않는다. 즉 각 수식마다 한 개의 문단을 이루어야 한다.
3. 모든 글자는 변수 이름으로 간주되어 입력된 대로 식자된다. 수학적식의 변
수명은 기울인 글꼴을 사용하고 수식 자체의 독특한 간격을 스스로 결정
하는데, 만약 수식 내에서 일반 텍스트(곧게 선 로마 글꼴에 일반 텍스트
의 간격 주기를 적용한)를 식자하고자 하는 부분이 있으면 `\textrm{...}`
명령어를 사용해서 입력해야 한다.

$$\forall x \in \mathbf{R} : x^2 \geq 0$$

$$\forall x \in \mathbf{R} : x^2 \geq 0 \quad (3.2)$$


```
\begin{equation}
x^{2} \geq 0 \quad \quad
\text{for all } x \in \mathbf{R}
\end{equation}
```

$$x^2 \geq 0 \quad \text{for all } x \in \mathbf{R} \quad (3.3)$$

수학자는 기호(심볼)를 사용할 때 무척 세심하게 신경을 쓴다. 아래 예에서 보듯이 amsfonts 혹은 amssymb 패키지에서 `\mathbb` 명령을 사용하여 얻어지는 ‘블랙보드 볼드’(blackboard bold) 문자를 사용하는 것이 관행인 경우도 있다.

```
\begin{displaymath}
x^{2} \geq 0 \quad \quad
\text{for all } x \in \mathbb{R}
\end{displaymath}
```

$$x^2 \geq 0 \quad \text{for all } x \in \mathbb{R}$$

3.2 수식 모드의 그룹짓기

대부분의 수식 모드 명령어는 오직 명령어 다음에 오는 한 문자에 대해서만 효력을 갖는다. 만약 여러 문자에 대해 효력이 있게 하려면 중괄호 {...}를 사용하여 함께 묶으면 된다.

```
\begin{equation}
a^{x+y} \neq a^{x+y}
\end{equation}
```

$$a^x + y \neq a^{x+y} \quad (3.4)$$

3.3 수학 기호 및 수식의 표현

이 절에서는 수식 조판에서 가장 중요하게 쓰이는 명령 몇 가지를 설명한다. 수학 기호를 조판하는 데 쓰이는 명령을 44 쪽 3.9 절에 목록으로 작성해 두었으므로 참고하기 바란다.

그리스 소문자는 `\alpha`, `\beta`, `\gamma`, ... 와 같이 나타내며, 그리스 대문자는 `\Gamma`, `\Delta`, ...²와 같이 써 넣는다.

```
$\lambda, \xi, \pi, \mu, \Phi, \Omega$
```

$$\lambda, \xi, \pi, \mu, \Phi, \Omega$$

윗첨자와 아래첨자는 각각 `^`와 `_` 문자를 사용하여 나타낸다.

```
$a_{1}$ \quad $x^{2}$ \quad
$e^{-\alpha t}$ \quad
$a^{3}_{ij}$ \quad
$e^{x^2} \neq e^x{}^2$
```

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

²대문자 알파(Alpha)는 로마 A와 모양이 동일하므로 L^AT_EX 2_ε는 이것을 정의하지 않았다.

제곱근은 `\sqrt`로 입력하며, n -제곱근(n^{th} root)은 `\sqrt[n]`이라고 쓰면 만들어진다. 근호의 크기는 L^AT_EX이 자동으로 결정한다. 근호만 필요하다면 `\surd`를 사용한다.

```
$\sqrt{x}$ \quad
$\sqrt{x^2+\sqrt{y}}$
\quad $\sqrt[3]{2}$\[\[3pt]
$\surd[x^2 + y^2]$
```

$$\sqrt{x} \quad \sqrt{x^2 + \sqrt{y}} \quad \sqrt[3]{2} \\ \sqrt{[x^2 + y^2]}$$

`\overline`, `\underline` 명령은 수평 줄을 그린다.

```
$\overline{m+n}$
```

$$\overline{m+n}$$

`\overbrace`, `\underbrace` 명령은 지정된 범위에 걸치는 긴 수평 중괄호 (horizontal braces)를 만든다.

```
$\underbrace{a+b+\cdots+z}_{26}$
```

$$\underbrace{a+b+\cdots+z}_{26}$$

작은 화살표나 물결표(tilde)와 같은 억양 표시를 변수에 첨가하려면, 44 쪽 표 3.1의 명령들을 사용한다. 여러 글자에 걸치는 넓은 모자형 심볼이나 큰 물결표는 `\widetilde` `\widehat`으로 만들 수 있다. ' 심볼은 프라임(prime)을 붙여준다.

```
\begin{displaymath}
y=x^2\quad y'=2x\quad y''=2
\end{displaymath}
```

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

벡터는 일반적으로 변수 상단에 작은 화살표 기호를 붙이는데, 이것을 얻으려면 `\vec` 명령을 쓴다. `\overrightarrow`와 `\overleftarrow` 명령을 쓰면 A 에서 B 로 가는 벡터를 나타낼 수 있다.

```
\begin{displaymath}
\vec{a}\quad\overrightarrow{AB}
\end{displaymath}
```

$$\vec{a} \quad \overrightarrow{AB}$$

변수들은 기울어진 글자체로 나타내는 데 비해, log 같은 함수 이름은 곧게 선 (로만) 글자체로 쓰는 것이 일반적이다. L^AT_EX은 주요 함수 이름들을 조판하기 위하여 다음과 같은 명령을 제공한다.

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

```
\[ \lim_{x \rightarrow 0} \frac{\sin x}{x} = 1 ]
```

$$\lim_{x \rightarrow 0} \frac{\sin x}{x} = 1$$

나머지 연산(modulo) 함수에는 두 가지 명령이 있다. 이항 연산 “ $a \bmod b$ ”는 `\bmod` 명령을 사용하여 표현하며, “ $x \equiv a \pmod{b}$ ”와 같은 표현은 `\pmod` 명령을 쓴다.

분모 · 분자가 아래위로 붙는 보통 분수 표현은 `\frac{...}{...}` 명령으로 만든다. 가끔 $1/2$ 처럼 슬래시로 분수를 나타내는 것이 더 낫다는 사람도 있는데, 서너 개 정도의 ‘분수’만 쓰는 경우 이쪽이 더 보기 좋기 때문이다.

```
$1\frac{1}{2}$~hours
\begin{displaymath}
\frac{x^2}{k+1} \quad
x^{\frac{2}{k+1}} \quad
x^{1/2}
\end{displaymath}
```

$1\frac{1}{2}$ hours

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

상하 수식을 조판하려면 `{... \choose ...}` 또는 `{... \atop ...}` 명령을 쓸 수 있다. 후자는 전자와 결과가 동일하지만 아래위를 묶는 괄호가 붙지 않는다.

```
\begin{displaymath}
\{n \choose k\} \quad \{x \atop y+2\}
\end{displaymath}
```

$$\binom{n}{k} \quad x \atop y+2$$

적분 기호는 `\int` 명령으로 생성할 수 있으며, 합을 나타내는 기호는 `\sum`으로 만들 수 있다. 적분이나 \sum 의 상한과 하한 값은 윗첨자와 아래 첨자를 나타낼 때 쓰는 `^`와 `_`로 지정할 수 있다.

```
\begin{displaymath}
\sum_{i=1}^n \quad
\int_0^{\frac{\pi}{2}}
\end{displaymath}
```

$$\sum_{i=1}^n \quad \int_0^{\frac{\pi}{2}}$$

괄호나 시작과 끝을 나타내는 문자는 TeX에서 정의된 심볼을 그대로 쓴다.(보기, $[\langle \| \uparrow]$). 동근괄호(소괄호)와 각진괄호(대괄호)는 키보드에서 그대로 입력하면 된다. 중괄호는 `\{`으로 입력한다. 그밖의 다른 시작과 끝을 나타내는 문자들은 특별한 명령을 쓴다.(예. `\updownarrow`). 46 페이지의 표 3.8에 그 개요가 있으므로 참고하라.

```
\begin{displaymath}
\{a,b,c\} \neq \{a,b,c\}
\end{displaymath}
```

$$a,b,c \neq \{a,b,c\}$$


```
\newcommand{\ud}{\mathrm{d}}
\begin{displaymath}
\int\!\!\!\int\limits_D g(x,y)
\quad \mathrm{d}x \, \mathrm{d}y
\end{displaymath}
instead of
\begin{displaymath}
\int\!\!\!\int\limits_D g(x,y)\mathrm{d}x\mathrm{d}y
\end{displaymath}
```

$$\iint_D g(x,y) \, dx \, dy$$

instead of

$$\iint_D g(x,y) dx dy$$

미분의 ‘d’기호는 관행상 로마 글자체로 식자한다.⁵

$\mathcal{M}\mathcal{S}$ - $\mathrm{I}\mathrm{T}\mathrm{E}\mathrm{X}$ 을 포함하면 쓸 수 있는 `\iint`, `\iiint`, `\iiint`, `\idotsint` 명령은, 다중적분의 적분기호 사이 간격을 정밀하게 조정할 수 있는 다른 방법을 제공한다. `amsmath` 패키지 사용을 선언하였다면 위의 예는 아래와 같이 나타낼 수 있다.

```
\newcommand{\ud}{\mathrm{d}}
\begin{displaymath}
\iint\limits_D \quad \mathrm{d}x \, \mathrm{d}y
\end{displaymath}
```

$$\iint_D dx \, dy$$

좀 더 자세한 내용을 알려면 $\mathcal{M}\mathcal{S}$ - $\mathrm{I}\mathrm{T}\mathrm{E}\mathrm{X}$ 과 함께 배포되는 전자문서 `test-math.tex`를 보거나 “The LaTeX Companion” 제8장⁶의 내용을 참조하면 된다.

3.5 수식의 수직 정렬

배열은 `array` 환경으로 조판한다. 이 `array` 환경은 `tabular` 환경과 비슷하게 동작한다. 줄 바꿈은 `\\` 명령을 쓴다.

```
\begin{displaymath}
\mathbf{X} =
\left( \begin{array}{ccc}
x_{11} & x_{12} & \ldots \\
x_{21} & x_{22} & \ldots \\
\vdots & \vdots & \ddots
\end{array} \right)
\end{displaymath}
```

$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \dots \\ x_{21} & x_{22} & \dots \\ \vdots & \vdots & \ddots \end{pmatrix}$$

`array` 환경에서는 오른쪽 닫는 문자를 보이지 않게 하는 문자로 `.`을 쓸 수 있어서, 왼쪽에만 시작을 나타내는 문자를 크게 표시하고 오른쪽에는 쓰지 않으려는 경우에 사용할 수도 있다.

⁵ 우리나라에서 만들어진 수학서적은 대부분 미분의 ‘d’도 기울어진 글꼴로 식자하는 경우가 많다.[역자]

⁶CTAN:/tex-archive/info/companion-rev/ch8.{pdf,ps}

```
\begin{displaymath}
y = \left\{ \begin{array}{ll}
a & \text{if } d > c \\
b+x & \text{in the morning} \\
l & \text{all day long}
\end{array} \right.
\end{displaymath}
```

$$y = \begin{cases} a & \text{if } d > c \\ b+x & \text{in the morning} \\ l & \text{all day long} \end{cases}$$

여러 줄에 걸친 수식이나 수식군을 나타내기 위해, `equation` 환경 대신에 `eqnarray`와 `eqnarray*`를 사용할 수 있다. `eqnarray`를 사용하면 줄 마다 오른 쪽에 수식 번호가 표시된다. `eqnarray*`에서는 번호가 어느 줄에도 나타나지 않는다.

`eqnarray`와 `eqnarray*` 환경은 `{rc1}` 형식의 세 칸(column) 짜리 표처럼 동작한다. 이들 세 개의 칸 중 가운데 칸은 등호나 부등호 기호를 나타내는 데 사용된다. 혹은 사용자가 원하는 기호를 사용하여 표현할 수도 있다. `\` 명령어는 줄바꿈을 나타낸다.

```
\begin{eqnarray}
f(x) & = & \cos x & \\
f'(x) & = & -\sin x & \\
\int_0^x f(y)dy & = & \sin x & \\
\end{eqnarray}
```

$$f(x) = \cos x \quad (3.5)$$

$$f'(x) = -\sin x \quad (3.6)$$

$$\int_0^x f(y)dy = \sin x \quad (3.7)$$

등호의 좌우측 간격은 다소 넓다.

이것은 다음 예제에서 쓰인 것처럼, `\setlength\arraycolsep{2pt}`라고 설정하여 줄일 수 있다.

긴 수식은 자동으로 줄바꿈이 되지 않으므로 사용자는 이런 긴 식의 줄바꿈할 부분과 들여쓰기할 부분을 지정해야 한다. 제일 많이 쓰이는 방법을 아래 두 예제에서 보았다.

```
{\setlength\arraycolsep{2pt}}
\begin{eqnarray}
\sin x & = & x - \frac{x^3}{3!} + \frac{x^5}{5!} - \\
& & \frac{x^7}{7!} + \cdots
\end{eqnarray}
```

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots \quad (3.8)$$

```
\begin{eqnarray}
\lefteqn{ \cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots }
\end{eqnarray}
```

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots \quad (3.9)$$

`\nonumber` 명령은 수식의 오른쪽에 수식 번호를 넣지 않도록 한다.

위에 설명한 방법으로는 수식을 수직 정렬하여 가지런하게 배치하기 어려울 수 있다. `amsmath` 패키지는 좀더 강력한 표현 방법을 제공한다.

3.6 수학 글꼴의 크기

수식 모드에서는 T_EX이 전후 맥락에 맞추어 가장 적절한 글꼴 크기를 선택한다. 예를 들면 윗첨자로 들어가는 문자는 작은 글자체로 조판한다. 만약 로만 글자체로 수식의 일부분을 작성하고자 한다면, `\textrm` 명령을 사용하면 안 된다. 왜냐하면 글꼴 크기 조절 메커니즘이 작동하지 않기 때문이다. `\textrm` 명령은 일시적으로 텍스트 모드로 빠져나가게 하는 것이다. 수식 모드에서 글꼴 크기 조절 메커니즘이 동작하도록 하려면 `\mathrm`을 써야 하는데, 약간의 주의가 필요하다. `\mathrm`은 짧은 수식항목에서만 잘 동작할 것이다. 공백은 여전히 조절되지 않을 것이고, 억양 표시 붙은 문자들은 작동하지 않을 것이다.⁷

```
\begin{equation}
2^{\text{\texttt{\textbackslash textrm{nd}}}} \quad \backslash quad
2^{\text{\texttt{\textbackslash mathrm{nd}}}}
\end{equation}
```

$$2^{\text{nd}} \quad 2^{\text{nd}} \quad (3.10)$$

TeX이 자동으로 크기를 선택하기는 하지만, 간혹 L^AT_EX에게 글자 크기를 지정해주어야 할 필요가 있는 경우도 있다. 수식 모드에서, 글꼴 크기는 네 가지 명령으로 지시한다.

`\displaystyle (123)`, `\textstyle (123)`, `\scriptstyle (123)`와
`\scriptscriptstyle (123)`.

스타일을 바꾸면 상하한값의 표시방식에도 영향을 미친다.

$$\mathop{\mathrm{corr}}(X,Y)=\frac{\sum_{i=1}^n(x_i-\overline{x})(y_i-\overline{y})}{\sum_{i=1}^n(x_i-\overline{x})^2\sum_{i=1}^n(y_i-\overline{y})^2}^{\frac{1}{2}}$$

$$\text{corr}(X, Y) = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\left[\sum_{i=1}^n (x_i - \bar{x})^2 \sum_{i=1}^n (y_i - \bar{y})^2 \right]^{1/2}}$$

이 예제에서는 기본 $\left[\right]$ 이 만드는 것보다 더 큰 각진괄호(대괄호)가 사용되었다.

⁷The $\mathcal{A}\mathcal{M}\mathcal{S}$ - $\mathcal{L}\mathcal{A}\mathcal{T}\mathcal{E}\mathcal{X}$ 패키지는 `\texttt{textrm}` 명령어와 크기 조절이 함께 작동하게 해준다.

3.7 정리(theorem), 법칙, ...

수학 문서를 작성하다 보면, 종종 “보조 정리(Lemmas)”, “정의(Definitions)”, “공리(Axiom)” 내지 이와 유사한 구조를 조판해야 할 필요가 생긴다. L^AT_EX이 지원하는 이에 관한 명령은 다음과 같다.

```
\newtheorem{name}[counter]{text}[section]
```

여기서 *name* 인자는 “정리(theorem)”를 구분하기 위해 붙이는 짧은 명칭(식별자)이다. *text* 인자는 최종 출력물에서 인쇄될 “정리(theorem)”의 실제 이름을 지정한다.

대괄호내의 인자는 옵션이다. 위의 두 대괄호는 “정리(theorem)”에 붙일 번호 매기기에 사용된다. *counter* 인자에는 앞에서 사용된 “정리(theorem)”의 *name*을 기입할 수 있다. 그렇게 하면 새로운 “정리”는 앞의 “정리”에서부터 번호가 연속되어 붙는다. *section* 지시어는 “정리”에 번호를 붙일 때 정리가 들어가기로 원하는 절(section) 번호를 함께 표시하도록 한다.

`\newtheorem` 명령이 T_EX 문서의 전처리부에서 실행된 후라면, 문서 본문에서 다음 명령을 사용할 수 있다.

```
\begin{name}[text]
This is my interesting theorem
\end{name}
```

위의 예로써 충분히 이해하고 쓸 수 있을 것이다. 그밖의 의문점들을 해소하고 너무 복잡해 보이는 `\newtheorem` 환경에 대한 이해를 명확하게 했으면 하는 바람에서 다음 몇 가지 예를 보이겠다.

```
% definitions for the document
% preamble
\newtheorem{law}{Law}
\newtheorem{jury}[law]{Jury}
%in the document
\begin{law} \label{law:box}
Don't hide in the witness box
\end{law}
\begin{jury}[The Twelve]
It could be you! So beware and
see law~\ref{law:box}\end{jury}
\begin{law}No, No, No\end{law}
```

Law 1 *Don't hide in the witness box*

Jury 2 (The Twelve) *It could be you! So beware and see law 1*

Law 3 *No, No, No*

“Jury” 정리는 “Law” 정리와 동일한 카운터를 사용한다. 그러므로 다른 “Law” 정리들과 함께 일련번호가 붙게 된다. 괄호(대괄호) 안의 인자는 정리의 제목이나 참조사항 등을 써넣는 데 이용된다.


```
\flushleft
\newtheorem{mur}{Murphy}[section]
\begin{mur}
If there are two or more
ways to do something, and
one of those ways can result
in a catastrophe, then
someone will do it.\end{mur}
```

Murphy 3.7.1 *If there are two or more ways to do something, and one of those ways can result in a catastrophe, then someone will do it.*

“Murphy” 정리에 붙는 번호는 현재의 절(section) 번호와 연동된다. 필요하다면 다른 단위들, 예를 들어 장(chapter)이나 소절(subsection)에 대해서도 마찬가지로 할 수 있다.

3.8 굵은 기호 문자

L^AT_EX에서는 두꺼운 기호(볼드 심볼)을 얻기가 상당히 어렵다. 아마 아마추어 사용자가 이것을 남용하는 경향이 있어 의도적으로 그렇게 해 놓은 것 같다. 글꼴 바꿈 명령인 `\mathbf`는 두꺼운 글자를 만들지만, 이것은 로마 글꼴(바로 선 글씨체)로 나오게 된다. 하지만 수학 기호 문자는 일반적으로 이탤릭(기울인) 글꼴이 대부분이다. `\boldmath`란 명령이 있으나, 이것은 수식 모드를 빠져나가서 써야 한다. 기호 문자에도 영향을 미친다.

```
\begin{displaymath}
\mu, M \quad \mathbf{M} \quad \mu, M
\mbox{\boldmath $\mu, M$}
\end{displaymath}
```

$$\mu, M \quad \mathbf{M} \quad \mu, M$$

그래서 이 예에서는 쉼표(콤마)가 두꺼운 글자로 나오게 되었다. 그러나 이것은 원하는 바가 아니다.

`amsmath`에 포함되어 있는 `amssbsy` 패키지를 쓰면 두꺼운 기호 문자를 훨씬 쉽게 해결할 수 있는데, 다음과 같이 `\boldsymbol` 명령을 사용하면 된다.

```
\begin{displaymath}
\mu, M \quad \mu, M
\boldsymbol{\mu}, \boldsymbol{M}
\end{displaymath}
```

$$\mu, M \quad \boldsymbol{\mu}, \boldsymbol{M}$$

3.9 수식 기호문자

수식 모드에서 일반적으로 사용할 수 있는 기호 문자(심볼)들을 모아서 다음에 몇 개의 표로 정리하였다.

표 3.12-3.16⁸의 기호문자(심볼)를 사용하기 위해서는 `amssymb` 패키지가 전 처리부에서 포함되어야 하며, AMS 수식 글꼴이 시스템에 설치되어 있어야 한다. AMS 패키지와 글꼴이 설치되어 있지 않다면,

CTAN:/tex-archive/macros/latex/required/amslatex를 참조하라.

표 3.1: 수학 모드의 역양 표시 붙은 글자

\hat{a}	<code>\hat{a}</code>	\check{a}	<code>\check{a}</code>	\tilde{a}	<code>\tilde{a}</code>	\acute{a}	<code>\acute{a}</code>
\grave{a}	<code>\grave{a}</code>	\dot{a}	<code>\dot{a}</code>	\ddot{a}	<code>\ddot{a}</code>	\breve{a}	<code>\breve{a}</code>
\bar{a}	<code>\bar{a}</code>	\vec{a}	<code>\vec{a}</code>	\widehat{A}	<code>\widehat{A}</code>	\widetilde{A}	<code>\widetilde{A}</code>

표 3.2: 그리스 소문자

α	<code>\alpha</code>	θ	<code>\theta</code>	o	<code>o</code>	υ	<code>\upsilon</code>
β	<code>\beta</code>	ϑ	<code>\vartheta</code>	π	<code>\pi</code>	ϕ	<code>\phi</code>
γ	<code>\gamma</code>	ι	<code>\iota</code>	ϖ	<code>\varpi</code>	φ	<code>\varphi</code>
δ	<code>\delta</code>	κ	<code>\kappa</code>	ρ	<code>\rho</code>	χ	<code>\chi</code>
ϵ	<code>\epsilon</code>	λ	<code>\lambda</code>	ϱ	<code>\varrho</code>	ψ	<code>\psi</code>
ε	<code>\varepsilon</code>	μ	<code>\mu</code>	σ	<code>\sigma</code>	ω	<code>\omega</code>
ζ	<code>\zeta</code>	ν	<code>\nu</code>	ς	<code>\varsigma</code>		
η	<code>\eta</code>	ξ	<code>\xi</code>	τ	<code>\tau</code>		

표 3.3: 그리스 대문자

Γ	<code>\Gamma</code>	Λ	<code>\Lambda</code>	Σ	<code>\Sigma</code>	Ψ	<code>\Psi</code>
Δ	<code>\Delta</code>	Ξ	<code>\Xi</code>	Υ	<code>\Upsilon</code>	Ω	<code>\Omega</code>
Θ	<code>\Theta</code>	Π	<code>\Pi</code>	Φ	<code>\Phi</code>		

⁸이 표들은 David Carlisle이 처음 작성한 후 Josef Tkadlec이 제안한 바를 반영하여 확장한 `symbols.tex` 문서에서 얻은 것이다.

표 3.4: 이항 관계 연산 기호

아래 심볼은 `\not` 명령어를 이용해서 부정 관계 기호로 바꿀 수 있다.

$<$	<code><</code>	$>$	<code>></code>	$=$	<code>=</code>
\leq	<code>\leq</code> or <code>\le</code>	\geq	<code>\geq</code> or <code>\ge</code>	\equiv	<code>\equiv</code>
\ll	<code>\ll</code>	\gg	<code>\gg</code>	$\dot{=}$	<code>\doteq</code>
\prec	<code>\prec</code>	\succ	<code>\succ</code>	\sim	<code>\sim</code>
\preceq	<code>\preceq</code>	\succeq	<code>\succeq</code>	\simeq	<code>\simeq</code>
\subset	<code>\subset</code>	\supset	<code>\supset</code>	\approx	<code>\approx</code>
\subseteq	<code>\subseteq</code>	\supseteq	<code>\supseteq</code>	\cong	<code>\cong</code>
\sqsubset^a	<code>\sqsubset^a</code>	\sqsupset^a	<code>\sqsupset^a</code>	\bowtie	<code>\Join^a</code>
\sqsubseteq	<code>\sqsubseteq</code>	\sqsupseteq	<code>\sqsupseteq</code>	\bowtie	<code>\bowtie</code>
\in	<code>\in</code>	\ni	<code>\ni</code> , <code>\owns</code>	\propto	<code>\propto</code>
\vdash	<code>\vdash</code>	\dashv	<code>\dashv</code>	\models	<code>\models</code>
\mid	<code>\mid</code>	\parallel	<code>\parallel</code>	\perp	<code>\perp</code>
\smile	<code>\smile</code>	\frown	<code>\frown</code>	\asymp	<code>\asymp</code>
$:$	<code>:</code>	\notin	<code>\notin</code>	\neq	<code>\neq</code> or <code>\ne</code>

^a이 기호를 사용하려면 `latexsym` 패키지가 포함되어야 한다.

표 3.5: 이항 연산 기호

$+$	<code>+</code>	$-$	<code>-</code>	
\pm	<code>\pm</code>	\mp	<code>\mp</code>	\triangleleft <code>\triangleleftleft</code>
\cdot	<code>\cdot</code>	\div	<code>\div</code>	\triangleright <code>\trianglerightright</code>
\times	<code>\times</code>	\setminus	<code>\setminusminus</code>	\star <code>\star</code>
\cup	<code>\cup</code>	\cap	<code>\cap</code>	\ast <code>\ast</code>
\sqcup	<code>\sqcup</code>	\sqcap	<code>\sqcap</code>	\circ <code>\circ</code>
\vee <code>\lor</code>	<code>\vee , \lor</code>	\wedge <code>\land</code>	<code>\wedge , \land</code>	\bullet <code>\bullet</code>
\oplus	<code>\oplus</code>	\ominus	<code>\ominus</code>	\diamond <code>\diamond</code>
\odot	<code>\odot</code>	\oslash	<code>\oslash</code>	\uplus <code>\uplus</code>
\otimes	<code>\otimes</code>	\bigcirc	<code>\bigcirc</code>	\amalg <code>\amalg</code>
\triangleup	<code>\bigtriangleup</code>	\triangledown	<code>\bigtriangledown</code>	\dagger <code>\dagger</code>
\lhd ^{<i>a</i>}	<code>\lhd</code> ^{<i>a</i>}	\rhd ^{<i>a</i>}	<code>\rhd</code> ^{<i>a</i>}	\ddagger <code>\ddagger</code>
\unlhd ^{<i>a</i>}	<code>\unlhd</code> ^{<i>a</i>}	\unrhd ^{<i>a</i>}	<code>\unrhd</code> ^{<i>a</i>}	\wr <code>\wr</code>

표 3.6: 큰 기호

\sum	<code>\sum</code>	\bigcup	<code>\bigcup</code>	\bigvee	<code>\bigvee</code>	\bigoplus	<code>\bigoplus</code>
\prod	<code>\prod</code>	\bigcap	<code>\bigcap</code>	\bigwedge	<code>\bigwedge</code>	\bigotimes	<code>\bigotimes</code>
\coprod	<code>\coprod</code>	\bigsqcup	<code>\bigsqcup</code>			\bigodot	<code>\bigodot</code>
\int	<code>\int</code>	\oint	<code>\oint</code>			\biguplus	<code>\biguplus</code>

표 3.7: 화살표

\leftarrow	<code>\leftarrow</code> or <code>\gets</code>	\longleftarrow	<code>\longleftarrow</code>	\uparrow	<code>\uparrow</code>
\rightarrow	<code>\rightarrow</code> or <code>\to</code>	\longrightarrow	<code>\longrightarrow</code>	\downarrow	<code>\downarrow</code>
\leftrightarrow	<code>\leftrightarrow</code>	\longleftrightarrow	<code>\longleftrightarrow</code>	\updownarrow	<code>\updownarrow</code>
\Leftarrow	<code>\Leftarrow</code>	\Longleftarrow	<code>\Longleftarrow</code>	\Uparrow	<code>\Uparrow</code>
\Rightarrow	<code>\Rightarrow</code>	\Longrightarrow	<code>\Longrightarrow</code>	\Downarrow	<code>\Downarrow</code>
\Leftrightarrow	<code>\Leftrightarrow</code>	\Longleftrightarrow	<code>\Longleftrightarrow</code>	\Updownarrow	<code>\Updownarrow</code>
\mapsto	<code>\mapsto</code>	\longmapsto	<code>\longmapsto</code>	\nearrow	<code>\nearrow</code>
\hookrightarrow	<code>\hookrightarrow</code>	\hookrightarrow	<code>\hookrightarrow</code>	\searrow	<code>\searrow</code>
\leftharpoonup	<code>\leftharpoonup</code>	\rightharpoonup	<code>\rightharpoonup</code>	\swarrow	<code>\swarrow</code>
\leftharpoondown	<code>\leftharpoondown</code>	\rightharpoondown	<code>\rightharpoondown</code>	\nwarrow	<code>\nwarrow</code>
\rightleftharpoons	<code>\rightleftharpoons</code>	\iff (bigger spaces)	<code>\iff</code> (bigger spaces)	\leadsto	<code>\leadsto</code> ^a

^a이 기호를 사용하기 위해서는 `latexsym` 패키지를 포함해야 한다.

표 3.8: 시작과 끝을 나타내는 기호

$($	<code>(</code>	$)$	<code>)</code>	\uparrow	<code>\uparrow</code>	\Uparrow	<code>\Uparrow</code>
$[$	<code>[</code> or <code>\lbrack</code>	$]$	<code>] or \rbrack</code>	\downarrow	<code>\downarrow</code>	\Downarrow	<code>\Downarrow</code>
$\{$	<code>\{ or \lbrace</code>	$\}$	<code>\} or \rbrace</code>	\updownarrow	<code>\updownarrow</code>	\Updownarrow	<code>\Updownarrow</code>
\langle	<code>\langle</code>	\rangle	<code>\rangle</code>	$ $	<code> or \vert</code>	$\ $	<code>\ or \Vert</code>
\lfloor	<code>\lfloor</code>	\rfloor	<code>\rfloor</code>	\lceil	<code>\lceil</code>	\rceil	<code>\rceil</code>
$/$	<code>/</code>	\backslash	<code>\backslash</code>	.	(dual. empty)		

표 3.9: 시작과 끝을 나타내는 큰 기호

$\left($	<code>\lgroup</code>	$\right)$	<code>\rgroup</code>	$\left[$	<code>\lmoustache</code>	$\right]$	<code>\rmoustache</code>
\uparrow	<code>\arrowvert</code>	\Uparrow	<code>\Arrowvert</code>	\uparrow	<code>\bracevert</code>		

표 3.10: 기타 기호 문자

...	<code>\dots</code>	...	<code>\cdots</code>	⋮	<code>\vdots</code>	⋯	<code>\ddots</code>
\hbar	<code>\hbar</code>	\imath	<code>\imath</code>	\jmath	<code>\jmath</code>	ℓ	<code>\ell</code>
\Re	<code>\Re</code>	\Im	<code>\Im</code>	\aleph	<code>\aleph</code>	\wp	<code>\wp</code>
\forall	<code>\forall</code>	\exists	<code>\exists</code>	\mho ^a	<code>\mho</code>	∂	<code>\partial</code>
'	<code>'</code>	'	<code>\prime</code>	\emptyset	<code>\emptyset</code>	∞	<code>\infty</code>
∇	<code>\nabla</code>	\triangle	<code>\triangle</code>	\Box ^a	<code>\Box</code>	\Diamond ^a	<code>\Diamond</code>
\bot	<code>\bot</code>	\top	<code>\top</code>	\angle	<code>\angle</code>	\surd	<code>\surd</code>
\diamondsuit	<code>\diamondsuit</code>	\heartsuit	<code>\heartsuit</code>	\clubsuit	<code>\clubsuit</code>	\spadesuit	<code>\spadesuit</code>
\neg	<code>\neg</code> or <code>\lnot</code>	\flat	<code>\flat</code>	\natural	<code>\natural</code>	\sharp	<code>\sharp</code>

^a이 기호를 사용하려면 `latexsym` 패키지가 포함되어야 한다.

표 3.11: 수학과 관계없는 기호

이 기호들은 텍스트 모드에서도 사용할 수 있다.

\dagger	<code>\dag</code>	\S	<code>\S</code>	\copyright	<code>\copyright</code>
\ddagger	<code>\ddag</code>	\P	<code>\P</code>	\pounds	<code>\pounds</code>

표 3.12: AMS의 시작과 끝을 나타내는 기호

\ulcorner	<code>\ulcorner</code>	\urcorner	<code>\urcorner</code>	\llcorner	<code>\llcorner</code>	\lrcorner	<code>\lrcorner</code>
-------------	------------------------	-------------	------------------------	-------------	------------------------	-------------	------------------------

표 3.13: AMS 그리스 문자와 히브리 문자

\digamma	<code>\digamma</code>	\varkappa	<code>\varkappa</code>	\beth	<code>\beth</code>	\daleth	<code>\daleth</code>	\gimel	<code>\gimel</code>
------------	-----------------------	-------------	------------------------	---------	--------------------	-----------	----------------------	----------	---------------------

표 3.14: AMS 이항 관계 연산 기호

\lessdot	<code>\lessdot</code>	\gtrdot	<code>\gtrdot</code>	\doteqdot or \Doteq	<code>\doteqdot or \Doteq</code>
\leqslant	<code>\leqslant</code>	\geqslant	<code>\geqslant</code>	\risingdotseq	<code>\risingdotseq</code>
\eqslantless	<code>\eqslantless</code>	\eqslantgtr	<code>\eqslantgtr</code>	\fallingdotseq	<code>\fallingdotseq</code>
\leqq	<code>\leqq</code>	\geqq	<code>\geqq</code>	\eqcirc	<code>\eqcirc</code>
\lll or \llless	<code>\lll or \llless</code>	\ggg or \gggtr	<code>\ggg or \gggtr</code>	\circeq	<code>\circeq</code>
\lesssim	<code>\lesssim</code>	\gtrsim	<code>\gtrsim</code>	\triangleq	<code>\triangleq</code>
\lessapprox	<code>\lessapprox</code>	\gtrapprox	<code>\gtrapprox</code>	\bumpeq	<code>\bumpeq</code>
\lessgtr	<code>\lessgtr</code>	\gtrless	<code>\gtrless</code>	\Bumpeq	<code>\Bumpeq</code>
\lesseqgtr	<code>\lesseqgtr</code>	\gtreqless	<code>\gtreqless</code>	\thicksim	<code>\thicksim</code>
\lesseqqgtr	<code>\lesseqqgtr</code>	\gtreqqless	<code>\gtreqqless</code>	\thickapprox	<code>\thickapprox</code>
\preccurlyeq	<code>\preccurlyeq</code>	\succcurlyeq	<code>\succcurlyeq</code>	\approxeq	<code>\approxeq</code>
\curlyeqprec	<code>\curlyeqprec</code>	\curlyeqsucc	<code>\curlyeqsucc</code>	\backsim	<code>\backsim</code>
\precsim	<code>\precsim</code>	\succsim	<code>\succsim</code>	\backsimeq	<code>\backsimeq</code>
\precapprox	<code>\precapprox</code>	\succapprox	<code>\succapprox</code>	\Vdash	<code>\Vdash</code>
\subseteqq	<code>\subseteqq</code>	\supseteqq	<code>\supseteqq</code>	\Vdash	<code>\Vdash</code>
\Subset	<code>\Subset</code>	\Supset	<code>\Supset</code>	\Vvdash	<code>\Vvdash</code>
\sqsubset	<code>\sqsubset</code>	\sqsupset	<code>\sqsupset</code>	\backepsilon	<code>\backepsilon</code>
\therefore	<code>\therefore</code>	\because	<code>\because</code>	\varpropto	<code>\varpropto</code>
\shortmid	<code>\shortmid</code>	\shortparallel	<code>\shortparallel</code>	\between	<code>\between</code>
\smallsmile	<code>\smallsmile</code>	\smallfrown	<code>\smallfrown</code>	\pitchfork	<code>\pitchfork</code>
\vartriangleleft	<code>\vartriangleleft</code>	\vartriangleright	<code>\vartriangleright</code>	\blacktriangleleft	<code>\blacktriangleleft</code>
\trianglelefteq	<code>\trianglelefteq</code>	\trianglerighteq	<code>\trianglerighteq</code>	\blacktriangleright	<code>\blacktriangleright</code>

표 3.15: AMS 화살표

\dashleftarrow	<code>\dashleftarrow</code>	\dashrightarrow	<code>\dashrightarrow</code>	\multimap	<code>\multimap</code>
\leftleftarrows	<code>\leftleftarrows</code>	\rightrightarrows	<code>\rightrightarrows</code>	\upuparrows	<code>\upuparrows</code>
\leftrightarrows	<code>\leftrightarrows</code>	\rightleftarrows	<code>\rightleftarrows</code>	\downdownarrows	<code>\downdownarrows</code>
\Lleftarrow	<code>\Lleftarrow</code>	\Rrightarrow	<code>\Rrightarrow</code>	\upharpoonleft	<code>\upharpoonleft</code>
\twoheadleftarrow	<code>\twoheadleftarrow</code>	\twoheadrightarrow	<code>\twoheadrightarrow</code>	\upharpoonright	<code>\upharpoonright</code>
\leftarrowtail	<code>\leftarrowtail</code>	\rightarrowtail	<code>\rightarrowtail</code>	\downharpoonleft	<code>\downharpoonleft</code>
\leftrightharpoons	<code>\leftrightharpoons</code>	\rightleftharpoons	<code>\rightleftharpoons</code>	\downharpoonright	<code>\downharpoonright</code>
\Lsh	<code>\Lsh</code>	\Rsh	<code>\Rsh</code>	\rightsquigarrow	<code>\rightsquigarrow</code>
\looparrowleft	<code>\looparrowleft</code>	\looparrowright	<code>\looparrowright</code>	\leftrightsquigarrow	<code>\leftrightsquigarrow</code>
\curvearrowleft	<code>\curvearrowleft</code>	\curvearrowright	<code>\curvearrowright</code>		
\circlearrowleft	<code>\circlearrowleft</code>	\circlearrowright	<code>\circlearrowright</code>		

표 3.16: AMS 부정 관계 연산 기호와 화살표

\nless	\ngtr	\varsubsetneqq
\lneq	\gneq	\varsupsetneqq
\nleq	\ngeq	\nsubseteqq
\nleqslant	\ngeqslant	\nsupseteqq
\lneqq	\gneqq	\nmid
\lvertneqq	\gvertneqq	\nparallel
\nleqq	\ngeqq	\nshortmid
\lnsim	\gnsim	\nshortparallel
\lnapprox	\gnapprox	\nsim
\nprec	\nsucc	\ncong
\npreceq	\nsucceq	\nvdash
\precneqq	\succneqq	\nvDash
\precnsim	\succnsim	\nVdash
\precnapprox	\succnapprox	\nVDash
\subsetneq	\supsetneq	\ntriangleleft
\varsubsetneq	\varsupsetneq	\ntriangleright
\nsubseteq	\nsupseteq	\ntrianglelefteq
\subsetneqq	\supsetneqq	\ntrianglerighteq
\nleftarrow	\rightarrow	\nleftrightarrow
\nLeftarrow	\Rightarrow	\nLeftrightarrow

표 3.17: AMS 이항 연산 기호

\dotplus	\centerdot	\intercal
\ltimes	\rtimes	\divideontimes
\Cup or \doublecup	\Cap or \doublecap	\smallsetminus
\veebar	\barwedge	\doublebarwedge
\boxplus	\boxminus	\circleddash
\boxtimes	\boxdot	\circledcirc
\leftthreetimes	\rightthreetimes	\circledast
\curlyvee	\curlywedge	

표 3.18: AMS 기타 기호 문자

\hbar	<code>\hbar</code>	\hslash	<code>\hslash</code>	\Bbbk	<code>\Bbbk</code>
\square	<code>\square</code>	\blacksquare	<code>\blacksquare</code>	\textcircled{S}	<code>\circledS</code>
\triangle	<code>\vartriangle</code>	\blacktriangle	<code>\blacktriangle</code>	\complement	<code>\complement</code>
∇	<code>\triangledown</code>	\blacktriangledown	<code>\blacktriangledown</code>	\Game	<code>\Game</code>
\lozenge	<code>\lozenge</code>	\blacklozenge	<code>\blacklozenge</code>	\bigstar	<code>\bigstar</code>
\angle	<code>\angle</code>	\measuredangle	<code>\measuredangle</code>	\sphericalangle	<code>\sphericalangle</code>
\diagup	<code>\diagup</code>	\diagdown	<code>\diagdown</code>	\backprime	<code>\backprime</code>
\nexists	<code>\nexists</code>	\Finv	<code>\Finv</code>	\varnothing	<code>\varnothing</code>
\eth	<code>\eth</code>	\mho	<code>\mho</code>		

표 3.19: 수학용 알파벳

예제	명령	포함되어야 할 패키지
ABCdef	<code>\mathrm{ABCdef}</code>	
ABCdef	<code>\mathit{ABCdef}</code>	
\mathnormal{ABCdef}	<code>\mathnormal{ABCdef}</code>	
\mathcal{ABC}	<code>\mathcal{ABC}</code>	
\mathscr{ABC}	<code>\mathcal{ABC}</code>	<code>mathrsfs</code>
\mathcal{ABC}	<code>\mathcal{ABC}</code>	<code>eucal</code> with option: <code>mathcal</code> or
	<code>\mathscr{ABC}</code>	<code>eucal</code> with option: <code>mathscr</code>
\mathfrak{ABCdef}	<code>\mathfrak{ABCdef}</code>	<code>eufrak</code>
\mathbb{ABC}	<code>\mathbb{ABC}</code>	<code>amsfonts</code> or <code>amssymb</code>

제 4 장

특별한 기능

대규모 문서를 \LaTeX 으로 작성하는 경우, 찾아보기 생성이나 참고문헌 관리와 같은 특별한 기능을 활용하면 많은 도움이 된다. 이와같은 \LaTeX 의 특별한 기능에 대해서는 *LaTeX Manual* [1]과 *The LaTeX Companion* [3]에 보다 자세히 설명되어 있다.

4.1 EPS 그림 포함하기

\LaTeX 은 `figure`나 `table` 환경으로 떠다니는 표와 그림, 이미지와 그래픽을 다룰 수 있게 해 주는 기본 기능을 제공한다.

기본 \LaTeX 이나 \LaTeX 확장 패키지만으로도 실제 그림을 생성할 수 있는 방법이 몇 가지 있다. 하지만 불행하게도, 대부분의 사용자에게는 너무 난해하다는 게 문제다. 따라서 이 방법은 여기에서 더이상 다루지 않겠다. 이 주제에 관심이 있는 사용자들은 *The LaTeX Companion* [3]이나 *LaTeX Manual* [1]을 참고하기 바란다.

문서 안에 그림을 포함하는 훨씬 쉬운 방법은, 그림 그리기는 전문 그래픽 소프트웨어¹를 이용하고, 이렇게 해서 완성된 그림을 \LaTeX 문서에 삽입하는 것이다. 다양한 그림 삽입 방법을 지원하는 \LaTeX 패키지들이 있다. 이 책에서는 그 중 널리 쓰이고 있고 비교적 사용하기 쉬운 Encapsulated PostScript (EPS) 그래픽에 대해서만 다룬다. EPS 형식의 그림을 처리하기 위해서는 포스트스크립트 프린터²가 필요하다.

D. P. Carlisle의 `graphicx` 패키지에는 쓸모있는 명령이 많다. 이는 “`graphics`”³ 패키지 꾸러미(package bundle)의⁴ 일부분이다.

¹XFig, CorelDraw!, Freehand, Gnuplot, ... 등.

²PS 출력을 얻는 방법으로 GHOSTSCRIPT를 사용하는 방법도 있는데 이 프로그램은 CTAN:/tex-archive/support/ghostscript에서 얻을 수 있다. 윈도우(Windows) 사용자는 GSVIEW를 이용해서 출력할 수 있을 것이다.

³CTAN:/tex-archive/macros/latex/required/graphics

⁴ \LaTeX ‘패키지’는 \LaTeX `\texinput{}` 아래 있는 ‘기본 패키지 꾸러미’ (graphics, tools, amslatex, psnfss, ...)와 \LaTeX `\texinput{}` `contrib/` 아래 놓이는 ‘추가 패키지 군’이 있다. ‘패키지 꾸러미’란 한 것은 다른 지시가 없으면 ‘기본 패키지 꾸러미’ 가운데 하나를 가리킨다.[역자]

출력 장치로 PostScript 프린터를 사용하고 `graphicx` 패키지가 설치되어 있는 경우, 그래픽을 문서에 붙여 넣으려면 다음과 같은 순서를 차례로 따라하면 된다.

1. 사용하고 있는 그래픽 프로그램에서 그림을 EPS 형식으로 저장한다.⁵
2. 입력 파일의 전처리부(preamble)에서 다음과 같이 `graphicx` 패키지 사용을 선언한다.

```
\usepackage[driver]{graphicx}
```

여기서 *driver*는 사용자의 “dvi를 postscript로” 변환하는 프로그램의 명칭이다. 많이 사용되는 프로그램은 `dvips`이다. \TeX 에는 그림을 붙여넣기 위한 표준이 없으므로 *driver* 이름의 지정이 필요하다. `graphicx` 패키지는 *driver* 명칭을 알아야 프린터가 처리할 수 있도록 그래픽에 관한 정보를 `.dvi` 파일에 제대로 집어넣어, `.eps` 파일을 정확하게 처리할 수 있게 되는 것이다.

3. 다음의 명령으로 *file*을 사용자 문서에 붙여넣는다.

```
\includegraphics[key=value, ...]{file}
```

선택적인 인자 *key*들과 해당 변수값인 *value*들은 쉼표(콤마)로 구분하여 입력한다. 변수 *key*들로는 삽입될 그래픽의 폭과 높이 및 회전값 등을 지정한다. 표 4.1에 중요한 *key*를 정리해 두었다.

표 4.1: `graphicx` 패키지의 *key* 명칭

<code>width</code>	그래픽의 폭을 지정
<code>height</code>	그래픽의 높이를 지정
<code>angle</code>	그래픽을 반시계방향으로 회전
<code>scale</code>	그래픽의 배율을 지정

다음의 예제를 살펴보면 좀 더 분명하게 이해할 수 있을 것이다.

```
\begin{figure}
\begin{center}
\includegraphics[angle=90, width=0.5\textwidth]{test.eps}
\end{center}
\end{figure}
```

⁵만약 그래픽 소프트웨어가 EPS 저장을 지원하지 않는다면 포스트스크립트 프린터 드라이버를 설치하여 얻는 방법이 있다. 예를 들면 Apple LaserWriter 드라이버를 설치한 다음, 이 드라이버를 이용해서 파일로 인쇄한다. 대개 이렇게 저장한 파일은 EPS 포맷일 것이다. 참고로, EPS 파일은 한 페이지밖에 포함되지 않는다. 프린터 드라이버 설정값을 조절하여 EPS 포맷으로 출력하도록 할 수 있게 되어 있는 경우가 많다.

이 코드는 `test.eps`라는 이름으로 저장된 그래픽을 문서에 삽입한다. 삽입되는 그림은 먼저 90도 각도로 회전된 다음, 폭이 표준 문단의 0.5배로 조정된다. 그래픽의 높이를 특별히 지정하지 않았으므로 가로×세로 비율은 원래 상태가 유지된다. 폭과 높이를 각각 개별적으로 지정할 수도 있음은 물론이다. 자세한 내용은 65 쪽에 있는 표 5.5를 참조하기 바란다. 이 문제에 대해 좀더 자세히 알고 싶다면 [8]과 [11]을 살펴보자.

4.2 참고문헌

`thebibliography` 환경을 이용하여 참고문헌 목록을 만들 수 있다. 각 참고문헌 항목은

```
\bibitem{marker}
```

으로 시작한다. 문서 본문에서 단행본이나 논문, 기사를 인용하려면 *marker*라는 명칭으로 참조한다.

```
\cite{marker}
```

항목 번호는 자동적으로 붙는다. `\begin{thebibliography}` 명령에 이어 나오는 인자는 자동으로 붙는 번호의 최대 자리수를 지정한다. 아래의 예제에서 {99}라고 한 것은 참고문헌 목록의 항목 갯수가 숫자 99의 자리수(두 자리)를 넘지 않도록 한다는 것을 \LaTeX 에게 알려주는 것이다.⁶

```
Partl~\cite{pa} has proposed,
that \ldots

\begin{thebibliography}{99}
\bibitem{pa} H.~Partl:
\emph{German \TeX}, TUGboat Vol.~9,
No.~1 ('88)
\end{thebibliography}
```

Partl [1] has proposed, that ...

Bibliography

[1] H. Partl: *German T_EX*, TUGboat Vol. 9,
No. 1 ('88)

⁶이 인자로 숫자 이외에 *widest-label*을 지정할 수 있다. *widest-label*은 참조할 레이블이 가질 대강의 길이를 보여주는 텍스트인데, `\bibitem`의 옵션 인자로 주어진 문자열 중에서 제일 긴 것을 쓰면 된다. 이렇게 하면 참고문헌 목록이나 본문의 `\cite`가 쓰인 위치에 숫자가 붙지 않고 이 문자열이 온다.[역자]

표 4.2: Index Key 사용 예제

Example	Index Entry	Comment
<code>\index{hello}</code>	hello, 1	Plain entry
<code>\index{hello!Peter}</code>	Peter, 3	Subentry under ‘hello’
<code>\index{Sam@\textsl{Sam}}</code>	<i>Sam</i> , 2	Formatted entry
<code>\index{Lin@\textbf{Lin}}</code>	Lin , 7	Same as above
<code>\index{Jenny textbf}</code>	Jenny, 3	Formatted page number
<code>\index{Joe textit}</code>	Joe, <i>5</i>	Same as above

대규모 작업일 경우 BibTeX 사용을 고려해 보는 것도 좋을 것이다. 대부분의 TeX 배포판에는 BibTeX이 포함되어 있다. BibTeX을 이용하면 참고문헌 데이터베이스를 관리할 수 있고, 그 데이터베이스로부터 사용자 문서에서 인용된 것들만 적절히 추출하여 참고문헌으로 제시할 수 있다. BibTeX이 생성하는 참고문헌 목록의 외관(표시방식)은, 사용자들이 참고문헌 목록을 널리 쓰이는 다양한 기존 디자인에 맞추어 작성할 수 있게 하는 유형 양식(style sheets) 개념을 따르고 있다.

4.3 찾아보기

대부분의 책에서 찾아보기는 매우 쓸모있는 도구이다. L^AT_EX과 그 응용 패키지인 `makeindex`⁷를 이용하면 매우 손쉽게 찾아보기를 작성할 수 있다. 이 책에서는 기본적인 찾아보기 목록 생성 명령어에 대해서만 설명한다. 보다 상세한 사항은 *The L^AT_EX Companion* [3]을 참고하라.

L^AT_EX의 찾아보기 만들기 기능이 작동하게 하려면, 문서의 전처리부(preamble)에서 `makeidx` 패키지의 사용을

```
\usepackage{makeidx}
```

와 같이 선언하고 다음과 같은 특별한 찾아보기 생성 명령어

```
\makeindex
```

를 전처리부(preamble)에 써넣어야 한다.

찾아보기의 내용은

```
\index{key}
```

로 작성되며, *key*는 찾아보기 항목이다. 최종 찾아보기 목록에서 참조될 본문 위치에서 `index` 명령을 사용한다. 표 4.2에 몇 가지 예제를 통하여 *key* 인자의 사용법을 설명하였다.

⁷파일명에 8자 이상을 사용할 수 없는 시스템에서는 `makeidx`으로 불리기도 한다.

L^AT_EX은 입력파일을 처리하다가 `\index` 명령어가 나타나면 찾아보기 항목과 현재의 페이지 번호를, L^AT_EX 입력파일과 이름이 같고 확장자가 `.idx`인 특별한 파일에 기록한다. 이 `.idx`파일은 `makeindex` 프로그램을 실행하여 처리한다.

```
makeindex filename
```

프로그램 `makeindex`는 찾아보기를 정렬하여, 파일이름이 같고 확장자가 `.ind`인 파일에 저장한다. L^AT_EX이 입력파일을 다시 처리할 때, 이렇게 정렬된 색인은 본문내의

```
\printindex
```

명령이 위치한 곳에 삽입된다.

L^AT_EX 2_ε와 함께 배포되는 `showidx` 패키지는 찾아보기 항목을 본문의 좌측 여백에 인쇄한다. 이것은 문서의 검토와 찾아보기 확인에 매우 유용하다.

4.4 멋진 쪽머리글(Fancy Headers)

Piet van Oostrum이 작성한 `fancyhdr` 패키지⁸는 간단한 몇개의 명령어로 문서의 쪽머리글과 쪽바닥글을 사용자가 설정할 수 있게 해 준다. 이 문서의 페이지 상단 쪽머리글이 바로 이 패키지를 활용하여 작성한 것이다.

쪽머리글과 쪽바닥글을 사용자가 조정할 때 실제 장·절의 이름을 포함시키고 싶을 때가 있다. 이것은 약간 교묘한 방법을 써야 하는데, L^AT_EX에서는 2단계 접근 방식으로 이 문제를 해결하고 있다. 머리글과 바닥글을 정의하면서 현재의 장·절 이름을 표현하기 위해 `\rightmark`와 `\leftmark` 명령어를 사용한다. 이 두 명령어의 값은 `chapter`나 `section` 명령어가 수행될 때마다 변화하게 된다.

최대한 유연성을 유지하려고, `\chapter` 계열의 명령이 직접 `\rightmark`, `\leftmark`를 재정의하지 않는다. 그 대신 다른 명령 `\chaptermark`, `\sectionmark` 및 `\subsectionmark`를 호출하는데, 이 명령들이 `\rightmark`와 `\markleft`를 재정의하도록 되어 있다.

따라서, 쪽머리글에서 `chapter` 이름을 변경하고자 한다면, `\chaptermark`를 “새로 지정(renew)”하기만 하면 된다.

`fancyhdr` 패키지를 써서 이 책의 쪽머리글을 만든 설정 내용을 그림 4.1에 나타내었다. 각주 8에 언급된 주소에서 이 패키지에 관한 문서를 구해볼 것을 권한다.

4.5 Verbatim 패키지

앞에서 `verbatim` 환경에 대해 알아본 적이 있다. 이 절에서는 `verbatim` 패키지에 대해 알아보도록 하자. 기본적으로 `verbatim` 패키지는 원래의 `verbatim` 환경이 갖는 몇가지 한계를 극복할 수 있도록 `verbatim`을 재구현한 것이다. 이것 자체는 별로 대단한 것이 아니지만, `verbatim` 패키지를 구현하면서 몇 가지

⁸CTAN:/macros/latex/contrib/supported/fancyhdr/에서 구할 수 있다.

```

\documentclass{book}
\usepackage{fancyhdr}
\pagestyle{fancy}
% with this we ensure that the chapter and section
% headings are in lowercase.
\renewcommand{\chaptermark}[1]{\markboth{#1}{}}
\renewcommand{\sectionmark}[1]{\markright{\thesection\ #1}}
\fancyhf{} % delete current setting for header and footer
\fancyhead[LE,R0]{\bfseries\thepage}
\fancyhead[LO]{\bfseries\rightmark}
\fancyhead[RE]{\bfseries\leftmark}
\renewcommand{\headrulewidth}{0.5pt}
\renewcommand{\footrulewidth}{0pt}
\addtolength{\headheight}{0.5pt} % make space for the rule
\fancypagestyle{plain}{%
    \fancyhead{} % get rid of headers on plain pages
    \renewcommand{\headrulewidth}{0pt} % and the line
}

```

그림 4.1: fancyhdr 설정 예제

기능들이 추가되었는데, 이 추가된 기능 때문에 이 패키지를 언급하는 것이다. `verbatim` 패키지는 다음 명령, 즉,

`\verbatiminput{filename}`

를 제공한다. 이 명령은 일반 아스키 텍스트를 마치 `verbatim` 환경에 들어 있는 것처럼 불러올 수 있게 해준다.

`verbatim` 패키지는 ‘tools’ 패키지 꾸러미에 포함되어 있으므로, 대부분의 시스템에 이미 설치 되어 있을 것이다. 이 패키지에 대하여 좀더 자세히 알고 싶으면 [9]를 참조하라.

제 5 장

L^AT_EX을 자기에게 맞게 바꾸기

앞 장까지 배운 명령으로도, 볼 만한 문서를 만들어낼 수 있다. 멋을 부린 모양은 아니지만, 읽기 쉽고 보기 좋게 만드는 알려진 좋은 조판 규칙을 잘 따르는 문서가 된다.

그런데 L^AT_EX에서 사용자가 원하는 결과를 얻을 수 있는 명령어나 환경을 제공하지 않는 경우도 있고, 이미 있는 명령을 써서 얻은 출력 결과가 사용자의 요구에 적합하지 않은 경우도 있다.

이 장에서는, L^AT_EX에서 새로운 값을 정의하는 방법, 그리고 L^AT_EX 기본값을 이용했을 때 나오는 결과와 좀 다른 모양을 얻고 싶을 때 어떻게 하면 되는지에 대해 약간의 힌트를 제공하려 한다.

5.1 새로운 명령, 환경, 패키지

이 책에서 소개한 명령은 모두 상자 안에 표시하고 책 뒤의 찾아보기에 모아두었다. 이렇게 하기 위해 L^AT_EX 필수 명령을 사용하지 않고 새로운 명령과 환경을 정의하여 패키지를 만들었다. 그런 다음 아래와 같이 간단하게 입력하여 이 일을 쉽게 처리할 수 있게 한 것이다.

```
\begin{command}  
\ci{dum}  
\end{command}
```

\dum

이 예제는 `command`라는 이름의 새로운 환경을 사용하고 있다. 이 새로운 환경은 소개하려는 명령어 이름을 쓰고 그것을 감싸는 상자를 그리는 것이다. 그리고 명령어는 `\ci`라는 새로운 명령으로 지정하였는데, 이것은 명령을 식자하고 동시에 자동으로 찾아보기에 들어가도록 한다. 책 마지막의 찾아보기(index)를 보면 `\dum` 명령어가 나와 있고 `\dum` 명령을 언급한 쪽의 번호가 표시되어 있는 것을 확인할 수 있을 것이다.

만약 명령을 상자 안에 넣는 방식에 싫증이 났다면 `command` 환경 정의를 새로운 모양이 되도록 살짝 바꾸면 된다. 그러면 이 환경을 사용한 모든 곳의 모

양이 다 바뀌는데, 매번 박스를 그리고 명령을 써넣는 식으로 문서를 작성했을 때 그것을 일일이 다 찾아서 바꿔야 하는 것과 비교해보면, 이 방법이 얼마나 편리한 것인지 알 수 있다.

5.1.1 새로운 명령

새로운 명령을 추가하려면 다음과 같이 한다.

```
\newcommand{name}[num]{definition}
```

여기에는 두 개의 인자가 기본적으로 필요하다. 하나는 이 명령의 명칭 *name*이고, 다른 하나는 이 명령의 정의 *definition*이다. 대괄호 안에 있는 *num* 인자는 옵션으로 여러 개의 인자를 가진 명령어를 만들 때 쓰는데, 최대 9개까지의 인자를 가지게 할 수 있다.

아래 두 개의 예제는 새로운 명령을 만드는 예를 나타낸 것이다. 첫번째 예제에서는 `\tnss`라고 하는 새로운 명령을 만들었다. 이것은 “The Not So Short Introduction to L^AT_EX 2_ε”를 줄여 쓰게 한다. 만약 이 책의 제목을 문서 내에서 자주 사용한다면 이와 같이 줄여서 쓰면 편리하다.

```
\newcommand{\tnss}{The not
  so Short Introduction to
  \LaTeXe}
This is ‘‘\tnss’’ \ldots{}
```

This is “The not so Short Introduction to L^AT_EX 2_ε” ... “The not so Short Introduction to L^AT_EX 2_ε”

다음 예제는 *num* 인자의 사용법을 나타낸다. #1은 사용자가 넣는 인자의 내용으로 치환되어 들어가게 된다. 만약 한 개 이상의 인자를 사용하려면, #2 ... 등으로 쓸 수 있다.

```
\newcommand{\txsit}[1]
{This is the \emph{#1} Short
  Introduction to \LaTeXe}
% in the document body:
\begin{itemize}
\item \txsit{not so}
\item \txsit{very}
\end{itemize}
```

- This is the *not so* Short Introduction to L^AT_EX 2_ε
- This is the *very* Short Introduction to L^AT_EX 2_ε

새로운 명령을 만들었을 때 그 명령이 이미 정의되어 있는 것이라면, L^AT_EX는 이를 허용하지 않는다. 그러나 `\renewcommand` 명령을 사용하면 기존에 정의된 내용은 무시되고 새로 정의한 명령이 유효하게 된다. `\renewcommand` 명령의 사용법은 `\newcommand`와 동일하다.

이따금, `\providecommand` 명령을 써야 할 때도 있다. 이것은 `\newcommand`와 유사하지만, 만약 명령어가 이미 정의되어 있다면 L^AT_EX 2_ε는 새로 정의한 것을 무시하고 이미 정의되어 있는 것을 취한다.

5.1.2 새로운 환경

`\newcommand` 명령처럼, 새로운 환경을 만드는 `\newenvironment` 명령이 있다. 이 명령의 사용법은 다음과 같다.

```
\newenvironment{name}[num]{before}{after}
```

`\newcommand` 명령과 비슷하게, `newenvironment`에서도 옵션 인자 *num*은 있어도 되고 없어도 된다. *before* 인자에는 이 환경 안에 있는 텍스트가 시작되기 전에 처리할 내용을 넣는다. *after* 인자는 `\end{name}` 명령을 만났을 때 실행될 내용이다.

아래 예제는 `\newenvironment` 명령의 사용법을 보여준다.

```
\newenvironment{king}
{\rule{1ex}{1ex}%
 \hspace{\stretch{1}}}
{\hspace{\stretch{1}}%
 \rule{1ex}{1ex}}
```

■ My humble subjects ... ■

```
\begin{king}
My humble subjects \ldots
\end{king}
```

num 인자의 사용법은 `\newcommand` 명령의 경우와 동일하다. \LaTeX 은 정의되어 있는 환경이 또다시 정의되는 것을 허용하지 않는데, 만약 이미 있는 환경을 재정의하려면, `\renewenvironment` 명령을 사용해야 한다. 이 명령의 문법은 `\newenvironment` 명령과 동일하다.

위의 예제에서 사용된 `\rule` 명령어는 70 쪽에서 설명하며, `\stretch`는 64 쪽에서 설명한다. `\hspace`에 관한 더 자세한 내용은 64 쪽에서 볼 수 있다.

5.1.3 사용자 패키지

만약 새로운 환경과 명령을 많이 정의한다면, 문서의 전처리부가 무척 길어질 수 있다. 이런 경우, 환경과 명령의 정의를 담고 있는 \LaTeX 패키지를 만들어 쓰면 매우 유용하다. 패키지로 만든 다음에는, `\usepackage` 명령으로 이 패키지를 문서에 포함하여 사용할 수 있다.

```
% Demo Package by Tobias Oetiker
\ProvidesPackage{demopack}
\newcommand{\tnss}{The not so Short Introduction to \LaTeXe}
\newcommand{\txsit}[1]{The \emph{#1} Short
    Introduction to \LaTeXe}
\newenvironment{king}{\begin{quote}}{\end{quote}}
```

그림 5.1: 패키지의 예

패키지를 만드는 것은 기본적으로 문서의 전처리부에 있는 내용을 독립 파일로 복사하는 것이다. 이 파일의 확장명은 `.sty`가 된다. 다만 한 가지 특별한 명령이 있는데,

```
\ProvidesPackage{package name}
```

이 명령은 패키지 파일의 가장 앞 부분에서 사용된다. `\ProvidesPackage`는 패키지의 이름을 L^AT_EX에 알리는 기능을 함으로써, 컴파일 시 두 번 이상 삽입하려 할 때 이를 알리는 에러 메시지를 내는데 사용된다. 그림 5.1은 위의 예제에서 정의된 명령들을 패키지로 만든 것이다.

5.2 글꼴과 크기

5.2.1 글꼴 바꾸기 명령

L^AT_EX은 문서의 논리적 구조(장/절, 각주 ...)에 따라 적절한 글꼴 및 글꼴 크기를 선택한다. 하지만, 사용자가 임의로 글꼴의 모양과 크기를 바꾸고 싶을 경우가 있다. 이에 관한 명령들이 표 5.1과 5.2에 있다. 각 글꼴의 실제 크기는 설계하기 나름이다. 그것은 문서 클래스와 클래스 옵션에 따라 달라진다. 각 표준 문서 클래스에서 이용되는 글꼴 선택 명령의 절대 포인트 크기를 표 5.3에 요약해 두었다.

```
{\small The small and
\textbf{bold} Romans ruled}
{\Large all of great big
\textit{Italy}.}
```

The small and **bold** Romans ruled all of great big *Italy*.

L^AT_EX 2_ε의 한가지 중요한 특징은 글꼴의 속성이 비의존적이라는 것이다. 달리 말하자면, 이전에 설정한 볼드(굵게)/슬랜트(기울임) 속성을 유지하면서 크기는 물론 글꼴 자체도 바꿀 수 있다는 뜻이다.

수식 모드를 쓰는 경우, 수식 모드를 잠시 빠져 나가 텍스트를 입력하면서 글꼴 바꾸기 명령을 사용할 수도 있다. 수식 조판용 글꼴을 바꾸려면 수식 글꼴 바꾸기를 위하여 특별히 마련된 명령을 써야 한다. 이것은 표 5.4를 참조하라.

표 5.1: 글꼴

<code>\textrm{...}</code>	roman	<code>\textsf{...}</code>	sans serif
<code>\texttt{...}</code>	typewriter		
<code>\textmd{...}</code>	medium	<code>\textbf{...}</code>	bold face
<code>\textup{...}</code>	upright	<code>\textit{...}</code>	<i>italic</i>
<code>\textsl{...}</code>	<i>slanted</i>	<code>\textsc{...}</code>	SMALL CAPS
<code>\emph{...}</code>	<i>emphasized</i>	<code>\textnormal{...}</code>	document font

표 5.2: 글꼴 크기

<code>\tiny</code>	tiny font	<code>\Large</code>	larger font
<code>\scriptsize</code>	very small font	<code>\LARGE</code>	very large font
<code>\footnotesize</code>	quite small font		
<code>\small</code>	small font	<code>\huge</code>	huge
<code>\normalsize</code>	normal font		
<code>\large</code>	large font	<code>\Huge</code>	largest

표 5.3: 표준 클래스의 글꼴 실제 크기

size	10pt (default)	11pt option	12pt option
<code>\tiny</code>	5pt	6pt	6pt
<code>\scriptsize</code>	7pt	8pt	8pt
<code>\footnotesize</code>	8pt	9pt	10pt
<code>\small</code>	9pt	10pt	11pt
<code>\normalsize</code>	10pt	11pt	12pt
<code>\large</code>	12pt	12pt	14pt
<code>\Large</code>	14pt	14pt	17pt
<code>\LARGE</code>	17pt	17pt	20pt
<code>\huge</code>	20pt	20pt	25pt
<code>\Huge</code>	25pt	25pt	25pt

표 5.4: 수식용 글꼴

Command	Example	Output
<code>\mathcal{...}</code>	<code>\$\mathcal{B}=c\$</code>	$\mathcal{B} = c$
<code>\mathrm{...}</code>	<code>\$\mathrm{K}_2\$</code>	K_2
<code>\mathbf{...}</code>	<code>\$\sum x=\mathbf{v}\$</code>	$\sum x = \mathbf{v}$
<code>\mathsf{...}</code>	<code>\$\mathsf{G\times R}\$</code>	$G \times R$
<code>\mathtt{...}</code>	<code>\$\mathtt{L}(b,c)\$</code>	$L(b, c)$
<code>\mathnormal{...}</code>	<code>\$\mathnormal{R_{19}}\neq R_{19}\$</code>	$R_{19} \neq R_{19}$
<code>\mathit{...}</code>	<code>\$\mathit{ffi}\neq ffi\$</code>	$\mathit{ffi} \neq ffi$

글꼴 크기 명령에서 중괄호({})는 중요한 역할을 한다. 이렇게 중괄호로 묶으면 그 안의 내용이 하나의 그룹을 이루게 된다. 대부분의 L^AT_EX 명령은 이 그룹 안에서 쓰이면 효력이 이 범위 안으로 제한된다.

```
He likes {\LARGE large and
\small small} letters}.
```

He likes large and small letters.

글꼴 크기 명령은 줄 간격도 바꾼다. 다만 글꼴 크기 명령의 효력 범위 안에 문단 끝(\par 또는 \\\)이 올 때만 줄 간격이 바뀐다. 따라서 범위의 끝을 나타내는 중괄호}가 너무 일찍 나타나면 줄 간격이 바뀌지 않는다. 다음 두 예에서 \par 명령의 위치를 주의깊게 살펴보라.

```
{\Large Don't read this! It is not
true. You can believe me!}\par}
```

Don't read this! It is not true.
You can believe me!

```
{\Large This is not true either.
But remember I am a liar.}\par}
```

This is not true either. But remember I am a liar.

한 문단 전체나, 그보다 더 긴 텍스트의 일정 부분에 대해 글꼴 크기를 바꾸려 한다면, ‘환경(environment)’ 형태의 구문법을 사용하여 글꼴을 바꾸는 것이 나올 수도 있다.

```
\begin{Large}
This is not true.
But then again, what is these
days \ldots
\end{Large}
```

This is not true. But then again, what is these days ...

이렇게 하면 범위를 정해주기 위한 중괄호를 덜 사용해도 된다.

5.2.2 경고

이 장을 시작할 때 말한 바와 같이, 문서의 각 부분에 이와 같이 구체적이고 개별적인 명령을 어지럽게 삽입하는 것은 위험스런 일이다. 이런 방식은 L^AT_EX의 기본 개념과 상충하기 때문이다. L^AT_EX은 문서의 논리적 구성과 그 결과물의 실제 모양을 구별해서 취급한다는 개념을 가지고 있다. 다시 말하면 똑같은 내용의 글꼴 바꾸기 명령이 여러 번 나오는 경우라면 그 명령을 매번 써넣을 것이 아니라 글꼴 바꾸기를 위한 \newcommand를 정의해서 사용해야 한다는 뜻이다.

```
\newcommand{\oops}[1]{\textbf{#1}}
Do not \oops{enter} this room,
it's occupied by a \oops{machine}
of unknown origin and purpose.
```

Do not **enter** this room, it's occupied by a **ma-**
chine of unknown origin and purpose.

이 예제는 위험에 대한 경고문이다. 훗날 위험(물)을 나타내는 데 좀 다른 시각적 표현을 하고 싶을 때, `\textbf`를 일일이 써넣어서 만든 문서라면 그것을 매번 찾아서 확인하고 바꾸어야 할 것이다. 게다가 `\textbf`로 표시된 모든 단어들이 꼭 위험(물)을 나타내는 단어일 것이라는 법이 없으므로 그것들을 일일이 식별해서 바꾸어야 한다. 이에 비하면 `\oops`를 정의해서 한번에 위험물에 대한 표현을 바꾸는 것이 훨씬 쉽고 편리하다.

5.2.3 조언

글꼴과 글자크기에 대한 이야기를 끝내기 전에 조언 한 마디.¹

Remember! *The MORE fonts you use in a document, the more READABLE and beautiful it becomes*

5.3 간격

5.3.1 줄 간격

만약 문서의 줄 간격을 더 띄우고 싶다면, 다음과 같이 한다.

```
\linespread{factor}
```

이 명령은 문서의 전처리부에 넣는다. “한 줄 반”의 줄 간격을 사용하려면 `\linespread{1.3}`으로 하고, “두 줄” 줄 간격으로 벌리려면(영문 원고의 double space) `\linespread{1.6}`을 사용한다. 여기서 *factor* 인자의 기본값은 1이다.

5.3.2 문단의 형식

LaTeX에서 문단 모양(레이아웃)에 영향을 주는 변수가 두 개 있는데. 이 변수의 값을 지정하는 다음 예를 보자.

```
\setlength{\parindent}{0pt}
\setlength{\parskip}{1ex plus 0.5ex minus 0.2ex}
```

이 두 명령은 입력 파일의 전처리부에 들어간다. 앞의 것은 문단의 첫 줄 들여쓰기 값을 0에 맞추는 것이고, 다른 하나는 문단과 문단 사이의 간격을 더 넓히는 것이다. 유럽에서는 일반적으로 문단과 문단 사이를 넓히고 들여쓰기를 하지 않는다.² 이 두 명령어는 차례에도 영향을 미쳐서 차례의 줄 사이를 너무 때

¹ “더 많은 글꼴을 사용하면 문서는 더욱 더 읽기 쉽고 아름다워진다.” (역자의 사족. 역자는 개인적으로 이 조언이 말하는 바와는 다른 견해를 가지고 있다. 너무 많은 글꼴을 사용한 한글 문서는 오히려 가독성과 품위를 낮추는 결과를 낳을 수도 있는 것이다. 물론 이 문장은 지은이가 그렇게 생각하고 있다는 말이라기보다 글꼴 사용의 예제에 불과하지만.)

² 우리나라의 책들은 첫 줄 들여쓰기를 한글 한 글자 정도로 하는 것이 관행인 듯하다. (`\parindent=1em`) LaTeX 표준 클래스들은 영문자 세 글자 정도를 들여쓰는 것을 기본값으로 하고 있다.[역자]

어놓는다. 이것을 피하려면, 이 두 명령을 전처리부에 놓지 않고 문서 본문의 `\tableofcontents` 다음에 넣거나 아예 사용하지 않을 수 있다. 전문적으로 조판이 잘 된 책을 보면, 첫 줄 들여쓰기를 하고 문단과 문단 사이는 그대로 두는 것이 일반적인 관행이다.

첫 줄 들여쓰기 되지 않은 문단을 강제로 들여쓰기 하려면 다음 명령을 문단 시작 부분에 넣는다.³

```
\indent
```

말할 것도 없이 `\parindent`의 값은 0 이 아닌 값으로 설정되어야 제대로 동작한다.

첫 줄 들여쓰기를 하지 않으려면 다음 명령을 문단의 시작 부분에 넣는다.

```
\noindent
```

`\section` 명령 없이 바로 본문을 시작하고 싶을 때 이 명령을 쓰면 편리하다.

5.3.3 수평 간격

L^AT_EX은 단어간 간격과 문장 사이 간격을 자동으로 설정한다. 이 간격(여백)을 넓히려면 다음과 같이 한다.

```
\hspace{length}
```

`\hspace` 대신 (별표 붙은) `\hspace*` 명령어를 사용하면 여백을 넣어야 하는 곳이 줄 끝이나 줄 처음이라도 간격을 유지한다. *length* 인자는 보통 숫자 하나에 단위가 붙은 꼴인데, 주요한 단위를 표 5.5에 나타내었다.

This`\hspace{1.5cm}`is a space
of 1.5 cm.

This is a space of 1.5 cm.

줄의 끝까지 수평 간격을 넣어 채우는 가변 길이 명령어가 있다.

```
\stretch{n}
```

두 개의 `\hspace{\stretch{n}}` 명령어가 같은 줄에서 사용되면 stretch 인수 *n* 값에 비례해서 간격이 늘어나게 된다.

x`\hspace{\stretch{1}}`
x`\hspace{\stretch{3}}`x

x x x

³각 절의 첫 문단을 들여쓰기 하려면 `indentfirst` 패키지를 사용한다. 이 패키지는 ‘tools’에 포함되어 있다.

표 5.5: T_EX 단위

mm	millimetre $\approx 1/25$ inch	□
cm	centimetre = 10 mm	□
in	inch = 25.4 mm	□
pt	point $\approx 1/72$ inch $\approx \frac{1}{3}$ mm	□
em	사용되고 있는 글꼴에서 ‘M’자의 폭	□
ex	사용되고 있는 글꼴에서 ‘x’자의 높이	□

5.3.4 수직 간격

문단(paragraph), 절(section), 소절(subsection) 등의 수직 간격은 L^AT_EX에 의해 자동으로 계산된다. 두 문단 사이의 수직 간격을 늘려야 할 필요가 있으면 다음 명령을 쓴다.

`\vspace{length}`

이 명령의 전후에 각각 빈 줄을 하나씩 두는 것이 보통이다. 만약 페이지의 첫 줄 또는 마지막 줄에 걸치더라도 주어진 간격만큼 벌려야 한다면 `\vspace` 대신에 별표 붙은 `\vspace*` 명령을 사용한다.

`\stretch` 명령을 `\pagebreak`와 함께 쓰면 텍스트를 쪽의 마지막 줄에 놓거나 한 페이지의 중앙에 둘 수 있다.

```
Some text \ldots
```

```
\vspace{\stretch{1}}
```

```
This goes onto the last line of the page.\pagebreak
```

문단을 바꾸지 않고 두 줄 사이를 띄고 싶거나 표에서 두 줄 사이의 간격을 넓히려면 다음 명령

`\[length]`

를 사용한다.

5.4 페이지 레이아웃

L^AT_EX에서는 `\documentclass` 명령의 인자로 용지 크기를 지정할 수 있다. 그러면 용지의 오른쪽 여백과 본문 영역의 길이를 자동으로 설정한다. 그러나 이렇게 자동 설정된 기정값이 마음에 들지 않는 때도 있을 것이다. 당연히, 이 값을 변경하는 것도 가능하다. 그림 5.2는 변경할 수 있는 변수를 모두 보여준다. 이 그림은 ‘tool’ 묶음의 layout 패키지를 써서 만든 것이다.⁴

⁴CTAN:/tex-archive/macros/latex/required/tools

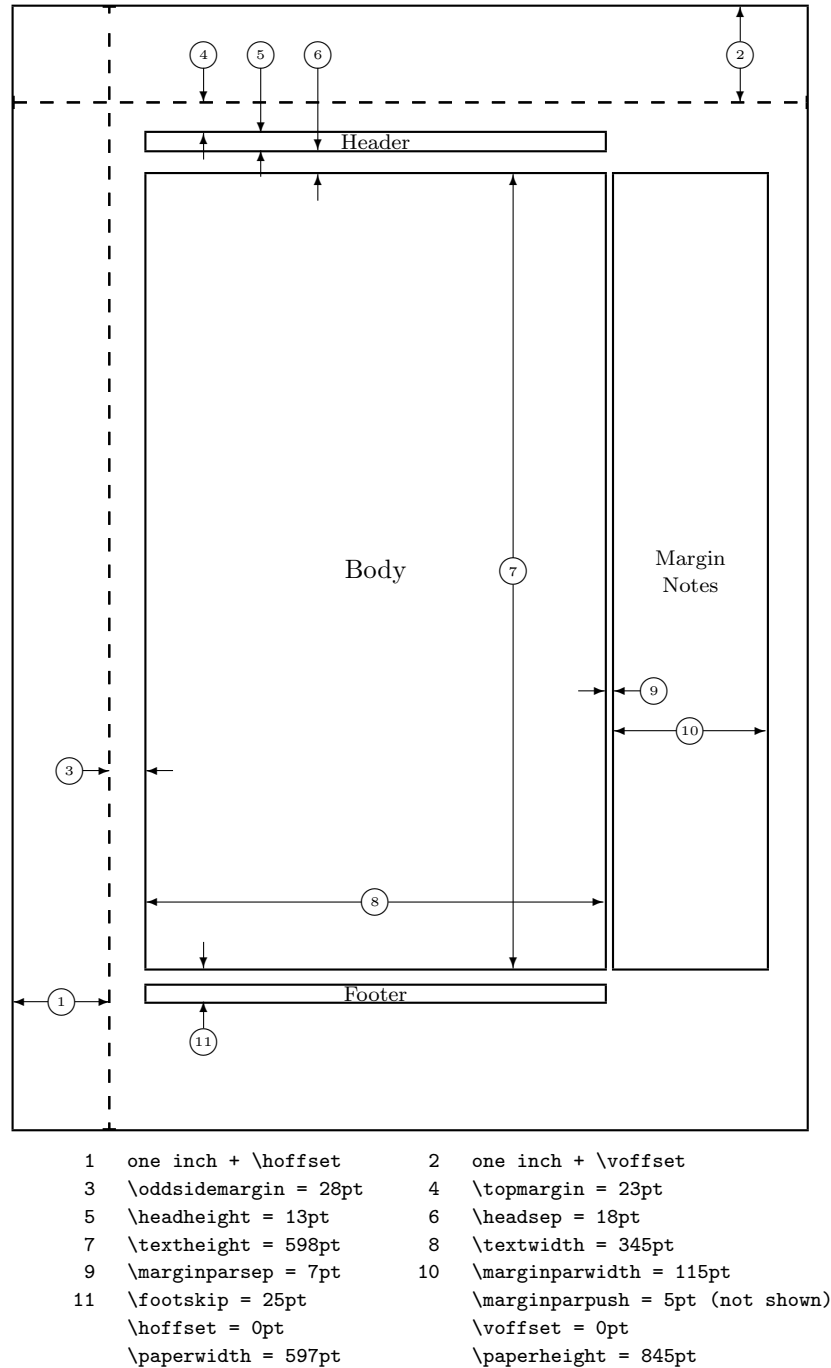


그림 5.2: 페이지 레이아웃의 변수

잠깐! ... 이 레이아웃이 너무 좁아 보여서 좀 넓게 만들고 싶은 분에게 한 마디 하겠다. 잠시 생각해 보자. 다른 L^AT_EX 기능들이 대부분 그렇듯이, 이처럼 페이지를 좁게 짜는 것이 기본값으로 되어 있는 데는 그럴 만한 이유가 있다.

MS 워드의 기본 규격 레이아웃과 비교해보면, 확실히 L^AT_EX의 기본 레이아웃이 상당히 좁다. 그러나 책⁵을 살펴보자. 한 줄에 몇 글자나 들어 있는가? 한 줄에 많아야 66자(영문)를 넘지 않을 것이다. L^AT_EX 문서를 만들어서 한 줄에 몇 자나 들어가는지 세어보라. 역시 약 66자 정도가 들어가는 것을 알 수 있을 것이다. 실험을 해보면 한 줄에 문자가 많을수록 읽기가 어려워진다는 걸 알 수 있다. 글자가 많을수록 사람의 눈이 줄 끝에서 다음 줄 첫 문자로 이동하기가 어려워지기 때문이다. 신문이 여러 단으로 조판되는 것도 이러한 이유에서이다.

그러므로, 만약 행의 나비를 늘리면, 독자들이 괴로워진다는 것을 명심하라. 그래도 굳이 문단폭을 넓혀야겠다면, 어떻게 하는 건지 가르쳐주겠다 ...

L^AT_EX은 이 변수 값을 바꿀 수 있는 두 개의 명령어를 제공한다. 이 명령들은 문서의 전처리부에서 사용된다.

첫번째는 어떤 고정값을 특정 변수에 할당하는 것이다.

```
\setlength{parameter}{length}
```

두번째 것은 어떤 변수의 기존값에 일정한 값을 더하는 것이다.

```
\addtolength{parameter}{length}
```

이 두번째 명령은 기존값에 대하여 상대 설정이 가능하므로 `\setlength` 명령어보다 더 유용하다. 예를 들어, 텍스트 영역의 문단폭을 1 센티미터 더 늘리려면 전처리부에 다음과 같이 쓰면 된다.

```
\addtolength{\hoffset}{-0.5cm}
\addtolength{\textwidth}{1cm}
```

이런 일을 하는 데 `calc` 패키지가 도움이 될 수도 있겠다. 그것은 `\setlength`의 인수에 대해 산술연산을 할 수 있게 해준다. 길이 설정뿐 아니라 그밖의 다른 함수의 경우에도 인자로서 숫자를 쓸 때는 이 패키지의 도움을 받을 수 있다.

5.5 길이 문제, 몇 가지 더

L^AT_EX 문서에서는 되도록 절대 길이를 사용하지 않도록 한다. 길이를 지정할 때는 쪽 레이아웃을 구성하는 다른 요소(예를 들면 문단 폭과 같은)의 값을 기초로 해서 상대값을 지정하는 것이 낫다. 그림을 쪽 나비에 맞게 채우고 싶으면 `\textwidth`를 그림 폭으로 쓰면 될 것이다.

⁵저명한 출판사에서 나온 진짜 인쇄본의 책.

다음 3개의 명령은 어떤 문자열(*text*)의 폭, 높이, 깊이를 계산하여 *command*의 값으로 할당하는 것이다.⁶

```
\settoheight{command}{text}
\settodepth{command}{text}
\settowidth{command}{text}
```

이 명령이 어떻게 활용될 수 있는지 보여주는 예제는 다음과 같다.

```
\flushleft
\newenvironment{vardesc}[1]{%
  \settowidth{\parindent}{#1:\ }
  \makebox[0pt][r]{#1:\ }}{}

\begin{displaymath}
a^2+b^2=c^2
\end{displaymath}

\begin{vardesc}{Where}$a$,
$b$ -- are adjunct to the right
angle of a right-angled triangle.

$c$ -- is the hypotenuse of
the triangle and feels lonely.

$d$ -- finally does not show up
here at all. Isn't that puzzling?
\end{vardesc}
```

$$a^2 + b^2 = c^2$$

Where: a , b – are adjunct to the right angle of a right-angled triangle.

c – is the hypotenuse of the triangle and feels lonely.

d – finally does not show up here at all. Isn't that puzzling?

5.6 상자

L^AT_EX이 조판하는 원리는 페이지에 상자를 하나씩 배치하는 방식이다. 우선, 글자 하나하나가 작은 상자이고, 단어는 문자 상자들을 잘 붙여서 만든다. 이 단어들을 다시 다른 단어와 붙인다. 단어와 단어를 붙일 때는 약간 특별한 방법을 쓰는데, 이어지는 단어들이 쪽과 줄에 딱 맞아들어가도록 신축성 있게 그 간격을 늘리거나 줄이면서 붙여가는 것이다.

실제로 이루어지는 일은 이보다 더 복잡하지만, 요는 T_EX이 상자를 붙여가는 방법으로 식자한다는 것이다. 글자만이 상자인 것은 아니다. 상자에는 무엇이든지 들어갈 수 있다. 심지어 상자 그 자체도 상자에 담을 수 있다. 이렇게 채워진 상자를 L^AT_EX은 각각 한 글자인 것처럼 취급한다.

상자라고 말하진 않았어도, 앞 장에서 이미 상자를 다루어 본 적이 있다. 예를 들면, `tabular` 환경이나 `\includegraphics` 명령은 상자를 생성한다. 즉, 두 개의 표나 그림을 나란히 붙도록 배열하는 것도 가능하다는 뜻이다. 단지 이렇게 할 때 두 개의 그림이나 표의 폭을 합한 값이 문단 나비(`textwidth`)보다 더 넓어서는 안 된다.

⁶폭, 높이, 깊이에 대해서는 85 페이지의 ‘역자의 보충’을 볼 것.[역자]

한 문단을 상자 안에 넣으려면 다음 두 가지 방법이 있다. 명령을 이용해서 넣고 싶으면 다음과 같이 하고,

```
\parbox[pos]{width}{text}
```

‘환경’ 구문을 쓰고 싶으면 다음과 같이 한다.

```
\begin{minipage}[pos]{width} text \end{minipage}
```

`pos` 인자로 올 수 있는 것은 상자의 수직 정렬을 조절하는 `c`, `t` 혹은 `b`의 값 중 하나이다. `width` 인자는 상자의 폭을 지정하는 것이다. `minipage`와 `parbox`의 가장 중요한 차이점은, `parbox` 안에서는 명령과 환경 전부를 사용할 수 있는 것이 아니지만, `minipage` 안에서는 거의 모든 명령이나 환경을 사용할 수 있다는 점이다.

`\parbox`는 줄바꿈 등등이 포함된 문단 전체를 상자에 넣는다. 수평으로 나열된 것들만을 하나로 묶는 상자 만들기 명령도 있다. 앞에서 본 바 있는 `\mbox`라는 것이 그 가운데 하나인데, 이 명령은 여러 개의 상자를 한 개의 상자로 묶어주기만 할 뿐이다. 그래서 `\mbox`로 묶인 단어들은 줄바꿈이 일어나지 않는다. 상자를 또다른 상자에 넣을 수 있기 때문에, 다음과 같은 수평 상자 묶음 명령을 잘 활용하면 매우 유연하게 써먹을 수 있다.

```
\makebox[width][pos]{text}
```

`width` 값은 밖에서 보았을 때 그려지는 상자의 폭이다.⁷ 직접 길이값을 써넣지 않고 `\width`, `\height`, `\depth`, `\totalheight`라고 지정할 수도 있다. 이렇게 하면 이 값들은 `text`를 식자한 결과 얻어지는 값을 계산하여 취한다. `pos` 인자는 `center`(중앙으로), `left flush`(왼쪽으로 몰기), `right flush`(오른쪽으로 몰기), `spread`(크기에 맞추어 펼치기) 중 한가지가 들어간다.

`\framebox` 명령의 기능은 `\makebox`와 동일한데, 텍스트 둘레에 상자를 그려준다.

다음 예제는 `\makebox`와 `\framebox` 명령을 활용하여 어떤 일을 할 수 있는지 보여주는 것이다.

```
\makebox[\textwidth]{%
  c e n t r a l}\par
\makebox[\textwidth][s]{%
  s p r e a d}\par
\framebox[1.1\width]{Guess I'm
  framed now!} \par
\framebox[0.8\width][r]{Bummer,
  I am to wide} \par
\framebox[1cm][l]{never
  mind, so am I}
Can you read this?
```

⁷ 즉, 이 값이 상자 안의 내용보다 작은 값일 수도 있다. 이 폭을 `0pt`로 설정할 수도 있는데, 이 때는 박스 안의 텍스트는 바깥쪽 상자와는 관계없이 그냥 출력될 것이다.

여기까지 수평(horizontal) 위치를 조절하는 방법을 보았다. 다음으로 수직(vertical) 위치를 조절하는 방법을 알아보자.⁸ 이것도 L^AT_EX에서는 어렵지 않다.

```
\raisebox{lift}[depth][height]{text}
```

이 명령은 상자의 수직 속성을 정의하게 한다. 첫 세 개의 인자에는 고정값을 넣어도 되지만, `\width`, `\height`, `\depth`, `\totalheight`를 써서 `text`의 크기를 계산하여 그 값으로 동작하게 할 수도 있다.

```
\raisebox{0pt}[0pt][0pt]{\Large%
\textbf{Aaaa}\raisebox{-0.3ex}{a}%
\raisebox{-0.7ex}{aa}%
\raisebox{-1.2ex}{r}%
\raisebox{-2.2ex}{g}%
\raisebox{-4.5ex}{h}}
he shouted but not even the next
one in line noticed that something
terrible had happened to her.
```

Aaaaaa he shouted but not even
the next one in line noticed that something ter-
rible had happened to her.

5.7 선 그리기

이 앞 몇 페이지 전에 아래와 같은 명령을 만난 적이 있을 것이다.

```
\rule[lift]{width}{height}
```

이 명령은 보통 간단한 검정 상자를 그리는 데 사용한다.

```
\rule{3mm}{.1pt}%
\rule[-1mm]{5mm}{1cm}%
\rule{3mm}{.1pt}%
\rule[1mm]{1cm}{5mm}%
\rule{3mm}{.1pt}
```



이것은 수직선이나 수평선을 그릴때 유용하다. 예를 들면 이 책 겉표지의 선은 `\rule` 명령을 사용해서 만든 것이다.

높이는 있으나 폭이 없는 특별한 선(`\rule{0pt}{height}`)이 쓰일 때가 있다. 전문 조판 용어로 `strut`라는 것이다. 이것은 페이지 상의 어떤 요소가 최소한 일정한 높이를 갖게 하기 위해서 쓰인다. `tabular` 환경에서 행(row) 높이가 적어도 일정값 이상을 갖도록 하는 데 쓰일 수 있다.

⁸수직, 수평 위치를 모두 다루려면 이 각각을 조절하면 된다.

```
\begin{tabular}{|c|}  
 \hline  
 \rule{1pt}{4ex}Pitprop \ldots\\  
 \hline  
 \rule{0pt}{4ex}Strut\\  
 \hline  
 \end{tabular}
```

Pitprop ...
Strut

참고문헌

- [1] Leslie Lamport. *L^AT_EX: A Document Preparation System*. Addison-Wesley, Reading, Massachusetts, second edition, 1994, ISBN 0-201-52983-1.
- [2] Donald E. Knuth. *The T_EXbook*, Volume A of *Computers and Typesetting*, Addison-Wesley, Reading, Massachusetts, second edition, 1984, ISBN 0-201-13448-9.
- [3] Michel Goossens, Frank Mittelbach and Alexander Samarin. *The L^AT_EX Companion*. Addison-Wesley, Reading, Massachusetts, 1994, ISBN 0-201-54199-8.
- [4] L^AT_EX을 설치할 때는, *L^AT_EX Local Guide*라는 이름의 문서를 제공하여, 각 로컬 시스템에 특정한 사항들을 설명해야 한다. 그 내용은 `local.tex`이라는 파일에 들어 있도록 되어 있다. 가끔 게으른 시스템 관리자는 이 문서를 제공하지 않기도 하는데, 이런 경우 주변의 L^AT_EX 도사(L^AT_EX guru)에게 물어보는 수밖에 없다.
- [5] L^AT_EX3 Project Team. *L^AT_EX 2_ε for authors*. `usrguide.tex`라는 이름으로 L^AT_EX 2_ε 배포본에 함께 딸려오는 문서이다.
- [6] L^AT_EX3 Project Team. *L^AT_EX 2_ε for Class and Package writers*. `clsguide.tex`라는 이름으로 L^AT_EX 2_ε 배포본에 함께 딸려오는 문서이다.
- [7] L^AT_EX3 Project Team. *L^AT_EX 2_ε Font selection*. `fntguide.tex`라는 이름으로 L^AT_EX 2_ε 배포본에 함께 딸려오는 문서이다.
- [8] D. P. Carlisle. *Packages in the ‘graphics’ bundle*. `grfguide.tex`이라는 이름으로 ‘graphics’ 패키지에 딸려오는 것으로서 L^AT_EX 배포본을 얻었던 곳에서 찾을 수 있다.
- [9] Rainer Schöpf, Bernd Raichle, Chris Rowley. *A New Implementation of L^AT_EX’s verbatim Environments*. `verbatim.dtx`라는 이름으로 ‘tools’ 패키지 꾸러미와 함께 딸려오는 것이다. L^AT_EX 배포본을 구했던 곳에서 찾을 수 있다.
- [10] Graham Williams. *The T_EX Catalogue*는 T_EX과 L^AT_EX 관련 패키지 리스트를 거의 전부 망라하고 있다.
CTAN:/help/Catalogue/catalogue.html에서 구할 수 있다.

- [11] Keith Reckdahl. *Using EPS Graphics in L^AT_EX 2_ε Documents*, 이것은 EPS 그림 파일과 L^AT_EX에서의 사용에 대해 필요한 내용 대부분과 그 이상의 것을 설명하고 있는 문서이다. CTAN:/info/epslatex.ps에서 구할 수 있다.

찾아보기

$\backslash!$, 38
", 17
\$, 33
, 36
 $\backslash($, 33
 $\backslash)$, 33
 $\backslash,$, 34, 38
-, 18
—, 18
 $\backslash-$, 17
—, 18
—, 18
., space after, 21
..., 18
 $\backslash:$, 38
 $\backslash;$, 38
 $\backslash\emptyset$, 21
 $\backslash[$, 33
L^AT_EX이 가진 장점, 3
 $\backslash\backslash$, 16, 25–27, 65
 $\backslash\ast$, 16
 $\backslash]$, 33
^, 35
_, 35
|, 27
~, 21
2단으로, 9

acute, 19
 $\backslash addtolength$, 67
æ, 19
amssbsy, 43
amsmath, 38, 39, 41, 43
amssymb, 35, 44
 $\backslash and$, 22
 $\backslash appendix$, 21, 22

array, 39
article 클래스, 9
 $\backslash atop$, 37
 $\backslash author$, 22

babel, 20
 $\backslash backmatter$, 23
 $\backslash backslash$, 5
 $\backslash begin$, 24
 $\backslash bibitem$, 53
 $\backslash Big$, 38
 $\backslash big$, 38
 $\backslash Bigg$, 38
 $\backslash bigg$, 38
 $\backslash bmod$, 37
bold face, 60
 $\backslash boldmath$, 43
 $\backslash boldsymbol$, 43
book 클래스, 9

calc, 67
 $\backslash caption$, 30
 $\backslash cdots$, 38
center, 25
 $\backslash chapter$, 22
 $\backslash chaptermark$, 55
 $\backslash choose$, 37
 $\backslash ci$, 57
 $\backslash cite$, 53
 $\backslash cleardoublepage$, 31
 $\backslash clearpage$, 31
command, 57

$\backslash date$, 22
DC 폰트, 17
dcolumn, 28
 $\backslash ddots$, 38

`\depth`, 69, 70
 description, 25
 displaymath, 33
`\displaystyle`, 41
 doc, 10
`\documentclass`, 8, 16
 dotless i and j, 19
`\dum`, 57

`\emph`, 24, 60
 empty, 11
 Encapsulated PostScript, 51
`\end`, 24
 enumerate, 25
 eqnarray, 40
 equation, 33
 eucal, 50
 eufrak, 50
 exscale, 10, 38

 fancyhdr, 55, 56
 figure, 29
`\floatpagefraction`, 31
 flushleft, 25
 flushright, 25
 foiltex, 9
 fontenc, 10, 21
`\footnote`, 23
`\footnotesize`, 61
`\frac`, 37
`\framebox`, 69
`\frenchspacing`, 21
`\frontmatter`, 23
`\fussy`, 16

 GhostScript, 51
 graphicx, 51
 grave, 19

 textttheadings, 11
`\height`, 69, 70
`\hline`, 27
`\hspace`, 59, 64
`\Huge`, 61
`\huge`, 61
`\hyphenation`, 17

`\idotsint`, 39
 ifthen, 10
`\iiiint`, 39
`\iiint`, 39
`\iint`, 39
`\include`, 11
`\includegraphics`, 52, 68
`\includeonly`, 11, 12
`\indent`, 64
 indentfirst, 64
`\index`, 54
`\input`, 12
 inputenc, 10, 20
`\int`, 37
 italic, 60
`\item`, 25
 itemize, 25

 Knuth, Donald E., 1

`\label`, 23, 34
 Lamport, Leslie, 1
`\LARGE`, 61
`\Large`, 61
`\large`, 61
 \LaTeX 2.09, 2
 \LaTeX 2 ϵ , 2
 \LaTeX 3, 1, 4
 latexsym, 10
 layout, 65
`\ldots`, 18, 38
`\left`, 38
`\leftmark`, 55
`\linebreak`, 16
`\linespread`, 63
`\listoffigures`, 30
`\listoftables`, 30

`\mainmatter`, 23
`\makebox`, 69
 makeidx, 10, 54
`\makeindex`, 54
`\maketitle`, 22
`\markleft`, 55
 math, 33
`\mathbb`, 35

<code>\mathbf</code> , 61	<code>\providecommand</code> , 58
<code>\mathcal</code> , 61	<code>\ProvidesPackage</code> , 60
<code>\mathit</code> , 61	
<code>\mathnormal</code> , 61	<code>\qqquad</code> , 34, 38
<code>\mathrm</code> , 41, 61	<code>\quad</code> , 34, 38
<code>mathrsfs</code> , 50	quotation, 26
<code>\mathsf</code> , 61	quote, 26
<code>\mathtt</code> , 61	
<code>\mbox</code> , 17, 19, 69	<code>\raisebox</code> , 70
<code>minipage</code> , 69	<code>\ref</code> , 23, 34
Mittelbach, Frank, 1	<code>\renewcommand</code> , 31, 58
<code>\multicolumn</code> , 28	<code>\renewenvironment</code> , 59
	report 클래스, 9
<code>\newcommand</code> , 58	<code>\right</code> , 38
<code>\newenvironment</code> , 59	<code>\right.</code> , 38
<code>\newline</code> , 16	<code>\rightmark</code> , 55
<code>\newpage</code> , 16	roman, 60
<code>\newtheorem</code> , 42	<code>\rule</code> , 59, 70
<code>\noindent</code> , 64	
<code>\nolinebreak</code> , 16	sans serif, 60
<code>\nonumber</code> , 40	<code>\scriptscriptstyle</code> , 41
<code>\nopagebreak</code> , 16	<code>\scriptsize</code> , 61
<code>\normalsize</code> , 61	<code>\scriptstyle</code> , 41
	<code>\section</code> , 21
œ, 19	<code>\sectionmark</code> , 55
option, 8	<code>\setlength</code> , 63, 67
<code>\overbrace</code> , 36	<code>\settodepth</code> , 68
overfull hbox, 16	<code>\settoheight</code> , 68
<code>\overleftarrow</code> , 36	<code>\settowidth</code> , 68
<code>\overline</code> , 36	showidx, 55
<code>\overrightarrow</code> , 36	slanted, 60
	slides 클래스, 9
<code>\pagebreak</code> , 16	<code>\sloppy</code> , 16
<code>\pageref</code> , 23	<code>\small</code> , 61
<code>\pagestyle</code> , 11	small caps, 60
<code>\par</code> , 62	<code>\sqrt</code> , 36
<code>\paragraph</code> , 21	<code>\stretch</code> , 59, 64
<code>\parbox</code> , 69	strut, 70
<code>parbox</code> , 69	<code>\subparagraph</code> , 21
<code>\parindent</code> , 63	<code>\subsection</code> , 21
<code>\parskip</code> , 63	<code>\subsectionmark</code> , 55
<code>\part</code> , 22	<code>\subsubsection</code> , 21
plain, 11	<code>\sum</code> , 37
<code>\pmod</code> , 37	syntonly, 10
prime, 36	
<code>\printindex</code> , 55	table, 29

-
- \tableofcontents, 22
 - tabular, 27, 68
 - \textbf, 60
 - \textit, 60
 - \textmd, 60
 - \textnormal, 60
 - \textrm, 41, 60
 - \textsc, 60
 - \textsf, 60
 - \textsl, 60
 - \textstyle, 41
 - \texttt, 60
 - \textup, 60
 - thebibliography, 53
 - \thispagestyle, 11
 - tilde (~), 21
 - \tiny, 61
 - \title, 22
 - \tnss, 58
 - \totalheight, 69, 70
 - \underbrace, 36
 - underfull hbox, 16
 - \underline, 36
 - upright, 60
 - \usepackage, 8, 20, 21, 59
 - \vdots, 38
 - \vec, 36
 - \verb, 27
 - verbatim, 55, 56
 - verbatim, 26, 55, 56
 - \verbatiminput, 56
 - verse, 26
 - \vspace, 65
 - \widehat, 36
 - \widetilde, 36
 - \width, 69, 70
 - WYSIWYG, 3
 - 공백문자, 4, 5
 - 명령 다음의, 5
 - 괄호, 37
 - 구조, 6
 - 국제 언어 지원, 19
 - 굵은 기호 문자, 43
 - 그리스 대문자, 35
 - 그리스 소문자, 35
 - 그림, 8, 51
 - 그림, 30
 - 글꼴, 60
 - 글꼴 인코딩, 10
 - 글꼴 크기, 60, 61
 - 기본 글꼴 크기, 9
 - 긴 대시, 18
 - 나머지 연산(modulo), 37
 - 단면, 9
 - 단위, 65
 - 대괄호, 6
 - 대시, 18
 - 독일어, 20
 - 따옴표, 17
 - 떠다니는 표와 그림, 29
 - 마침표, 18
 - 말줄임표, 18
 - 명령, 5
 - \!, 38
 - \(, 33
 - \), 33
 - \,, 34, 38
 - \-, 17
 - \:, 38
 - \;, 38
 - \@, 21
 - \[, 33
 - \\, 16, 25–27, 65
 - *, 16
 - \], 33
 - \addtolength, 67
 - \and, 22
 - \appendix, 21, 22
 - \atop, 37
 - \author, 22
 - \backmatter, 23
 - \backslash, 5
 - \begin, 24
 - \bibitem, 53

<code>\Big</code> , 38	<code>\indent</code> , 64
<code>\big</code> , 38	<code>\index</code> , 54
<code>\Bigg</code> , 38	<code>\input</code> , 12
<code>\bigg</code> , 38	<code>\int</code> , 37
<code>\bmod</code> , 37	<code>\item</code> , 25
<code>\boldmath</code> , 43	<code>\label</code> , 23, 34
<code>\boldsymbol</code> , 43	<code>\LARGE</code> , 61
<code>\caption</code> , 30	<code>\Large</code> , 61
<code>\cdots</code> , 38	<code>\large</code> , 61
<code>\chapter</code> , 22	<code>\ldots</code> , 18, 38
<code>\chaptermark</code> , 55	<code>\left</code> , 38
<code>\choose</code> , 37	<code>\leftmark</code> , 55
<code>\ci</code> , 57	<code>\linebreak</code> , 16
<code>\cite</code> , 53	<code>\linespread</code> , 63
<code>\cleardoublepage</code> , 31	<code>\listoffigures</code> , 30
<code>\clearpage</code> , 31	<code>\listoftables</code> , 30
<code>\date</code> , 22	<code>\mainmatter</code> , 23
<code>\ddots</code> , 38	<code>\makebox</code> , 69
<code>\depth</code> , 69, 70	<code>\makeindex</code> , 54
<code>\displaystyle</code> , 41	<code>\maketitle</code> , 22
<code>\documentclass</code> , 8, 16	<code>\markleft</code> , 55
<code>\dum</code> , 57	<code>\mathbb</code> , 35
<code>\emph</code> , 24, 60	<code>\mathbf</code> , 61
<code>\end</code> , 24	<code>\mathcal</code> , 61
<code>\floatpagefraction</code> , 31	<code>\mathit</code> , 61
<code>\footnote</code> , 23	<code>\mathnormal</code> , 61
<code>\footnotesize</code> , 61	<code>\mathrm</code> , 41, 61
<code>\frac</code> , 37	<code>\mathsf</code> , 61
<code>\framebox</code> , 69	<code>\mathtt</code> , 61
<code>\frenchspacing</code> , 21	<code>\mbox</code> , 17, 19, 69
<code>\frontmatter</code> , 23	<code>\multicolumn</code> , 28
<code>\fussy</code> , 16	<code>\newcommand</code> , 58
<code>\height</code> , 69, 70	<code>\newenvironment</code> , 59
<code>\hline</code> , 27	<code>\newline</code> , 16
<code>\hspace</code> , 59, 64	<code>\newpage</code> , 16
<code>\Huge</code> , 61	<code>\newtheorem</code> , 42
<code>\huge</code> , 61	<code>\noindent</code> , 64
<code>\hyphenation</code> , 17	<code>\nolinebreak</code> , 16
<code>\idotsint</code> , 39	<code>\nonumber</code> , 40
<code>\iiiint</code> , 39	<code>\nopagebreak</code> , 16
<code>\iiint</code> , 39	<code>\normalsize</code> , 61
<code>\iint</code> , 39	<code>\overbrace</code> , 36
<code>\include</code> , 11	<code>\overleftarrow</code> , 36
<code>\includegraphics</code> , 52, 68	<code>\overline</code> , 36
<code>\includeonly</code> , 11, 12	<code>\overrightarrow</code> , 36

<code>\pagebreak</code> , 16	<code>\textnormal</code> , 60
<code>\pageref</code> , 23	<code>\textrm</code> , 41, 60
<code>\pagestyle</code> , 11	<code>\textsc</code> , 60
<code>\par</code> , 62	<code>\textsf</code> , 60
<code>\paragraph</code> , 21	<code>\textsl</code> , 60
<code>\parbox</code> , 69	<code>\textstyle</code> , 41
<code>\parindent</code> , 63	<code>\texttt</code> , 60
<code>\parskip</code> , 63	<code>\textup</code> , 60
<code>\part</code> , 22	<code>\thispagestyle</code> , 11
<code>\pmod</code> , 37	<code>\tiny</code> , 61
<code>\printindex</code> , 55	<code>\title</code> , 22
<code>\providecommand</code> , 58	<code>\tnss</code> , 58
<code>\ProvidesPackage</code> , 60	<code>\totalheight</code> , 69, 70
<code>\qqquad</code> , 34, 38	<code>\underbrace</code> , 36
<code>\quad</code> , 34, 38	<code>\underline</code> , 36
<code>\raisebox</code> , 70	<code>\usepackage</code> , 8, 20, 21, 59
<code>\ref</code> , 23, 34	<code>\vdots</code> , 38
<code>\renewcommand</code> , 31, 58	<code>\vec</code> , 36
<code>\renewenvironment</code> , 59	<code>\verb</code> , 27
<code>\right</code> , 38	<code>\verbatiminput</code> , 56
<code>\right.</code> , 38	<code>\vspace</code> , 65
<code>\rightmark</code> , 55	<code>\widehat</code> , 36
<code>\rule</code> , 59, 70	<code>\widetilde</code> , 36
<code>\scriptscriptstyle</code> , 41	<code>\width</code> , 69, 70
<code>\scriptsize</code> , 61	
<code>\scriptstyle</code> , 41	문단, 13
<code>\section</code> , 21	문서에 소스 코드를 삽입, 8
<code>\sectionmark</code> , 55	물결표(tilde), 36
<code>\setlength</code> , 63, 67	백슬래시, 5
<code>\settodepth</code> , 68	
<code>\settoheight</code> , 68	블랙보드 볼드, 35
<code>\settowidth</code> , 68	
<code>\sloppy</code> , 16	빼기 부호, 18
<code>\small</code> , 61	상호 참조, 23
<code>\sqrt</code> , 36	
<code>\stretch</code> , 59, 64	세 점, 38
<code>\subparagraph</code> , 21	세로점, 38
<code>\subsection</code> , 21	소수점 정렬, 28
<code>\subsectionmark</code> , 55	수식, 33
<code>\subsubsection</code> , 21	공식, 33
<code>\sum</code> , 37	수식군, 40
<code>\tableofcontents</code> , 22	수직 공백, 65
<code>\textbf</code> , 60	수평
<code>\textit</code> , 60	여백, 64
<code>\textmd</code> , 60	점, 38
	줄, 36

- 중괄호, 36
- 수학
- 간격, 38
 - 긴 수식, 40
 - 벡터, 36
 - 빼기 기호, 18
 - 수학 글꼴 크기, 41
 - 역양 표시, 36
 - 처음과 끝을 나타내는 문자, 38
 - 함수, 36
- 섬표, 18
- 스칸디나비아 문자, 19
- 스페이스, 4
- 시작과 끝을 나타내는 문자, 37
- 아래첨자, 35
- 양면, 9
- 역양 표시, 19
- 언어, 19
- 옵션 인자, 6
- 왼쪽 정렬, 25
- 용지
- A4 paper, 9
 - A5 paper, 9
 - B5 paper, 9
 - executive paper, 9
 - legal paper, 9
 - letter paper, 9
- 용지 크기, 9
- 움라우트(umlaut), 19
- 유보된 문자들, 5
- 인자(parameter), 6
- 입력 파일, 7
- 적분, 37
- 전처리, 7
- 제곱근, 36
- 주석, 6
- 줄 간격, 63
- 줄바꿈, 15
- 지수, 35
- 쪽 양식, 11
- 쪽머리글, 11
- 쪽바닥글, 11
- 차례, 22
- 찾아보기, 54
- 채색 글씨, 8
- 타이틀, 9
- 특수 문자, 19
- 패키지, 7, 8, 57
- ambsy, 43
 - amfonts, 35, 50
 - amsmath, 38, 39, 41, 43
 - amssymb, 35, 44
 - babel, 20
 - calc, 67
 - dcolum, 28
 - doc, 10
 - eucal, 50
 - eufrak, 50
 - exscale, 10, 38
 - fancyhdr, 55, 56
 - fontenc, 10, 21
 - graphicx, 51
 - ifthen, 10
 - indentfirst, 64
 - inputenc, 10, 20
 - latexsym, 10
 - layout, 65
 - makeidx, 10, 54
 - mathrsfs, 50
 - showidx, 55
 - syntonly, 10
 - verbatim, 55, 56
- 페이지 레이아웃, 65
- 페이지 스타일, 11
- empty, 11
 - headings, 11
 - plain, 11
- 포스트스크립트, 51
- 표, 27, 30
- 표지, 9
- 프로그램
- makeindex, 54
- 하이픈, 18

합, 37

환경, 24

array, 39

center, 25

command, 57

description, 25

displaymath, 33

enumerate, 25

eqnarray, 40

equation, 33

figure, 29

flushleft, 25

flushright, 25

itemize, 25

math, 33

minipage, 69

parbox, 69

quotation, 26

quote, 26

table, 29

tabular, 27, 68

thebibliography, 53

verbatim, 26, 55, 56

verse, 26

그림, 30

표, 27, 30

역자후기

이 책은 사용법이 쉽지만은 않은 L^AT_EX의 입문서로 이미 정평이 있다.

당연히 L^AT_EX 2_ε를 제대로 이해하고 쓰기 위해서는 *The L^AT_EX Companion* [3]이 있어야 할 것이고, 좀더 고급의 독자들은 *The T_EXbook*을 보아야 하겠지만, 논문 작성 등 일반적 용도에는 이 책이 제공하는 정도의 기능만 숙지 하더라도 충분히 자신의 목적을 달성할 수 있을 것이다.

이 책을 번역해야겠다는 생각은 오래 전부터 가지고 있었는데, 그것을 실행에 옮길 엄두를 내기가 어려웠다. 우선, 한글판 lshort가 과연 필요할 것인가도 확실하기 어려웠고(왜냐하면 어차피 이 책을 한국어로 옮긴다 하더라도 예제는 여전히 영어 예제를 쓸 수밖에 없으며, 한글 구현에 관한 사항은 이 글의 ‘번역’에서는 다룰 수 없었기 때문이다.), 사실 초창기 lshort는 영문판도 컴파일 잘 되지 않는 경우가 있어서, 이것이 과연 한글로 제대로 동작할 것인지 확실할 수 없는 상태였기 때문에, 그냥 영문판을 보는 것으로 만족하고 지낸 것이 사실이다.

그러던 차에, 나의 개인 홈페이지¹ 게시판에서 이 문제를 제기했더니 강운배 · 장대훈 님이 흔쾌히 돕겠다는 의사를 밝혀 주셨다. 이렇게 의기투합하여, 대부분의 본문을 한글로 옮기는 일을 두 분이 하고, 나는 한글L^AT_EX으로 컴파일 되도록 맞추는 일을 주로 하면서 초벌번역이 이루어졌다. 초벌번역이 끝날 무렵, 김재우 님께서는 다른 경로로 나에게 연락을 해오셨는데, 3.1의 번역을 이미 해두신 적이 있다는 것이었다. 이렇게 전체의 번역이 이루어진 후, 내가 각 장을 다시 읽으면서 교열하고 오역을 수정하는 작업을 거쳐 마침내 한국어판 lshort를 출판(!)하게 되었다.

각 장별로 최종적으로 사용된 텍스트의 초벌 역자와 교열자는 표 1과 같다.

이 책이 L^AT_EX에 입문하는 분들에게 좋은 선물이 되기를 바란다. 사실 한글로 이루어진 T_EX 관련서적이 거의 없다 해도 좋을 정도의 상황에서, 이 글이 가치있는 입문서 구실을 충분히 할 것으로 믿는다.

번역상 주의한 것은 다음과 같다.

- 예제들은 영문을 그냥 노출시켰다. 이렇게 한 이유는, 이 책이 L^AT_EX 2_ε에 대한 설명이지 한글L^AT_EX에 대한 설명이 아니라는 점 때문이었다. 다시 말하면 이 예제들을 한글화했을 때, 그것은 L^AT_EX 2_ε를 통해 실행되는 것이 아니라 한글L^AT_EX을 통해서 실행되는 것이므로, 이 책의 원래 의도와는 동떨어진 것이 된다. L^AT_EX에서의 한글 사용에 대한 좋은 입문서가 나

¹<http://www.doeun.pe.kr>

감사의 말, 서문	강윤배·장대훈	김강수
제1장	김강수	장대훈·김강수
제2장	강윤배	장대훈·김강수
제3장	장대훈	김강수
제4장	김재우*	김강수
제5장	장대훈	김강수
참고문헌	장대훈	김강수

* 장대훈 님의 번역본도 있었는데, 김재우 님의 텍스트를 주로 살리면서 장대훈 님의 번역을 참조하여 교열하였음.

표 1: 각 장별 번역자와 교열자

오기를 바라는 마음 간절하다. 아니 그보다, 안심하고 쓸 수 있는 한글 \TeX 이 하나 있었으면 하는 생각도 든다.

- 문장의 번역은 무엇보다도 \LaTeX 입문자들이 가장 잘 이해할 수 있게 하는 데 초점을 맞추었다. 필요하다면 설명을 길게 덧붙이기도 했고 몇 가지 역사에 의한 보충도 추가하였다. 이런 시도가 도움이 되기를 바란다.
- 용어는 공동번역자들이 통일하기 위해서 여러번 시도했지만 완전히 일치하지는 못했을 수도 있을 것이다. 이 문제는 차차 고쳐가겠다.
- 최근 CTAN의 디렉토리 구조가 바뀌면서 이 책에 나오는 URL 정보가 달라진 것이 있어서 그것을 바로잡았다.

책을 옮기는 일은 솔직히 말하면 쉽지 않았다. 그 과정에서 격려해 준 김도현 님, 이현호 님, ‘무식인’ 님을 비롯한 모든 분들에게 특별히 감사의 말을 전한다. 초벌 번역본과 교열본을 읽고 어색한 표현을 지적해 준 \LaTeX 을 전혀 모르는 나의 학생들에게도 고맙다는 인사를 전한다. 한글 \LaTeX 의 저자인 은광희 님께 감사한다. 한글 \LaTeX 이 없었으면 이 글의 번역은 불가능했을 것이다.

이 번역본의 모든 책임은 김강수에게 있다. 다른 공동역자들은 오역에 대하여 책임이 없다.

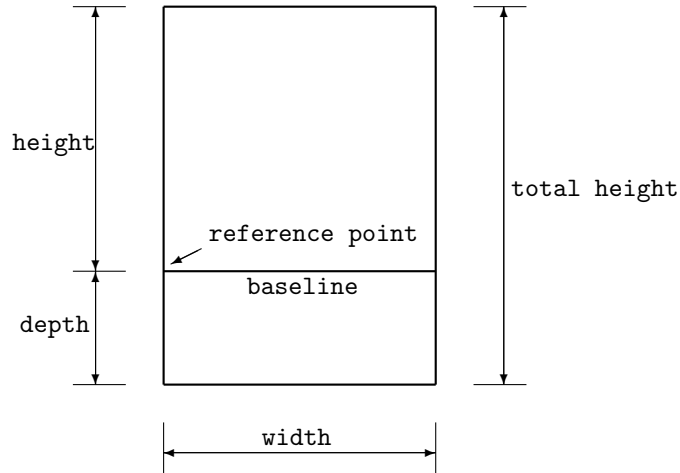
번역자 : 김강수(karnes@doeun.pe.kr), 강윤배(pear@postech.ac.kr), 장대훈(webmaster@texworld.pe.kr), 김재우(dbunix@kitinet.co.kr)

공동번역자를 대표하여...
김강수 karnes@doeun.pe.kr

역자의 보충

68 페이지

폭(width), 높이(height), 깊이(depth)는 각각 다음 그림에서 보인 것과 같이 박스의 크기를 나타내는 값들이다.²



18 페이지

한글 맞춤법의 ‘문장부호’ 규정에 따르면,

V. 이음표[連結符]

1. 줄표(—)

이미 말한 내용을 다른 말로 부연하거나 보충함을 나타낸다.

(1) 문장 중간에 앞의 내용에 대해 부연하는 말이 끼여들 때 쓴다.

[예] 그 신동은 네 살에—보통 아이 같으면 천자문도 모를 나이에—벌써 시를 지었다.

(2) 앞의 말을 정정 또는 변명하는 말이 이어질 때 쓴다.

[예] 어머님께 말했다가—아니, 말씀드렸다가—꾸중만 들었다.

이건 내 것이니까—아니, 내가 처음 발견한 것이니까—절대로 양보할 수 없다.

2. 붙임표(-)

(1) 사전, 논문 등에서 합성어를 나타낼 적에, 또는 접사나 어미임을 나타낼 적에 쓴다.

(2) 외래어와 고유어 또는 한자어가 결합되는 경우에 쓴다.

²김도현 님이 원래의 그림을 수정해 주셨음.