

초보자를 위한 **L^AT_EX** 안내서

\mathcal{M}^2 KAIST 수학기초연구회
L^AT_EX 안내서 제작 프로젝트 TF

최초 작성: 2022-11-03 03:42

최종 수정: 2023-11-02 15:15

\mathcal{M}^2 KAIST 수학기초연구회

\LaTeX 안내서 제작 프로젝트

작성	강우현	수리과학과 학사 21
	김다인	수리과학과 학사 21
감수	강우현	수리과학과 학사 21
	권도형	수리과학과 학사 22
	권순용	수리과학과 학사 22
	노희윤	수리과학과 학사 19
	문강연	수리과학과 학사 22
	박현수	새내기과정학부 학사 23
	심우용	산업및시스템공학과 학사 22
	어세훈	수리과학과 학사 20
	이종민	물리학과 학사 21
	이태영	수리과학과 학사 22
	장지연	수리과학과 학사 22
	조현준	수리과학과 학사 22
	최은수	수리과학과 학사 22
	하준안	전기및전자공학부 학사 21
편집	강우현	수리과학과 학사 21
	이종민	물리학과 학사 21

발행일 2023-11-02

발행처 KAIST 수학기초연구회

들어가며

오늘날 자연과학 및 공학 분야의 많은 논문은 \LaTeX 이라는 문서 작성 시스템을 사용해 작성됩니다. \LaTeX 이 자주 쓰이는 이유는 이 프로그램의 두 가지 특징 때문일 것입니다.

- 다른 워드프로세서에 비해 미려한 출판물을 생성함
- 수식 작성 기능이 다른 프로그램에 비해 편리하고 강력함

이 특징은 논문을 작성할 때가 아니더라도, 평범한 문서를 만들거나 책을 작성할 때에도 편리하게 작용합니다. 대학교에서는 수업 과제물을 \LaTeX 으로 작성할 것을 요구하는 교수님도 계십니다.

하지만, \LaTeX 을 처음 접하시는 분들은 \LaTeX 을 사용하는 것이 직관적이지 않다고 느낄 것입니다. 이는 \LaTeX 으로 문서를 작성하는 방식이 우리가 흔히 접하던 아래아한글이나 Word 등의 다른 워드프로세서와는 다르기 때문입니다. \LaTeX 을 사용하려면 문서의 실제 모습을 보지 못하는 채로 코드를 작성해야 하고, 이 코드를 컴파일 해야 실제 모습을 볼 수 있습니다. 그나마 프로그래밍이 익숙하거나 HTML 등으로 웹 디자인을 해 본 경험이 있는 분이라면 쉽게 적응하겠지만, 그런 경험이 없다면 \LaTeX 으로 문서를 작성하는 데 꽤 애를 먹을 것입니다.

이 책의 첫 번째 목표는 **\LaTeX 을 처음 접하시는 분들이 여기에 익숙해지고, 기본적인 문서를 손쉽게 작성할 수 있게 하는 것입니다.** 문서의 내용, 서식, 레이아웃, 메타데이터 등을 모두 플레인 텍스트로 입력하고 컴파일하여 PDF 파일을 얻는 과정이 익숙해지도록, 또 \LaTeX 의 여러 필수 문법이 손에 익도록 많은 예제를 제시하였습니다.

《The Not-so-short Introduction to \LaTeX 2 ϵ 》^[5] 등의 안내서가 이미 있고, 웹사이트에 검색하여 \LaTeX 의 사용법을 설명하는 많은 글을 찾을 수 있습니다. 하지만, 이는 어느 정도의 배경 지식이 있음을 전제하거나 대부분 영어로 쓰여 있기 때문에 아무것도 모르는 초보자가 이를 읽고 \LaTeX 을 배우기는 쉽지 않습니다(\LaTeX 의 많은 기능을 원활하게 사용하려면 결국에는 영어를 읽는 데 익숙해지는 것이 필요하긴 합니다). 그래서, \LaTeX 에 대해 아무 배경 지식이 없는 분들도 예제를 따라하며 문서 작성 과정을 이해할 수 있도록, 또 한국어를 사용해 쉽게 읽을 수 있도록 이 책을 작성하는 데 집중하였습니다.

이 책의 두 번째 목표는 **\LaTeX 을 어느 정도 다루어 보신 분들에게 참고 가이드가 되는 것입니다.** \LaTeX 은 가장 대표적인 수식 작성 기능 외에 목차 생성, 교차 참조, 표 작성, 색인 생성 등 수많은 기능을

가지고 있습니다. 여러 기능을 빠르게 사용할 수 있도록 중요한 것들을 일목요연하게 정리하여 소개할 것입니다. 특정 기능을 어떻게 사용하는지 알아보기 위해 웹 상의 수많은 글을 일일이 읽으면서 찾아낼 필요 없이, 이 책에서 간략히 정리해 둔 내용을 읽음으로써 그 기능을 쉽게 이용하실 수 있도록 하였습니다.

사실 \LaTeX 의 사용법을 익히는 가장 좋은 방법은 직접 부딪혀 보는 것이라고 생각합니다. 아무리 글을 열심히 읽어도 직접 사용해 보지 않으면 배운 코드를 어떻게 활용하는지 잘 와닿지 않을 뿐 아니라 금방 잊어버리기 때문입니다. 반면, 여러 문서를 작성하다 보면 자연스럽게 코드가 머릿속에 남게 되고, 매뉴얼을 찾아보지 않아도 문서를 빠르게 작성할 수 있게 됩니다. 따라서, \LaTeX 을 배우기 전 작성하고 싶은 문서를 생각해 보시기 바랍니다. 평소애 알고 있던 개념을 정리하면서 \LaTeX 을 배워도 좋고, 관심 분야의 글을 정리하거나 논문 또는 책의 내용을 따라서 써 보는 것도 좋습니다. 작성할 문서의 주제를 정하면 코드를 쉽게 익힐 수 있을 뿐 아니라, \LaTeX 의 기능을 배우는 데에도 동기부여가 될 것입니다.

이 책의 내용

\LaTeX 을 사용하는 방법을 본격적으로 설명하기에 앞서, 이 책이 담고 있는 내용을 간단히 소개하려고 합니다. 어떻게 이 책을 읽어야 하는지, 또 이 책이 어떤 내용을 담고 있는지 간단하게 훑는다면 내용을 이해하는 데 도움이 될 것입니다.

이 책은 제1장으로 구성된 도입부와 제I편부터 제III편까지의 기본 사용법, 제IV편부터 제VII편까지의 다양한 기능, 그리고 부록으로 나눌 수 있습니다. 가장 처음에 있는 도입부에서는 \TeX 과 \LaTeX 의 간단한 역사와 배경 지식을 설명합니다. 여기에서는 문법을 설명하지 않으므로 빠르게 문법을 배우고 싶으신 분은 제2장으로 넘어가셔도 좋습니다.

제I편부터 제III편까지는 기본적인 \LaTeX 사용법을 다룹니다. 제I편에서는 문서를 작성하는 과정과 기본 문법을 설명합니다. 제2장의 안내를 따르면 \LaTeX 으로 문서를 작성하는 과정을 익힐 수 있을 것이고, 제3장을 읽는다면 \LaTeX 에서 명령을 어떻게 내리는지 알게 될 것입니다. 제II편에서는 문서를 작성할 때 사용할 수 있는 여러 요소 중 수식을 제외한 것의 문법을 설명합니다. \TeX 의 용어를 사용하면 ‘텍스트 모드’에서 사용할 수 있는 기능을 설명한다고 말할 수 있습니다. 문자를 입력하고, 표지와 장절 표제를 만들고, 특수한 형태의 문단을 만들고 글꼴을 바꾸는 등, 다른 워드프로세서에 비유했을 때 ‘기본 기능’이라고 말할 수 있는 기능을 다룹니다. 제III편에서는 마침내 \LaTeX 의 대표적인 기능인 수식을 작성하는 방법을 설명합니다. 수식 입력 모드·텍스트 모드의 차이와 기본적인 수식 요소를 입력하는 방법을 설명하고, ‘큰 수식’은 어떻게 보기 좋게 입력하는지, 또 가환 도표는 어떻게 그리는지 설명합니다.

제IV편부터 제VII편까지는 \LaTeX 의 더 다양한 기능을 다룹니다. 먼저 제IV편에서는 표와 그림을 어떻게 문서에 넣고 배치하는지 설명합니다. 제V편에서는 \LaTeX 의 다양한 참조 기능을 다룹니다. 교차 참조 및 링크 삽입, 목차 생성, 참고 문헌 인용 및 목록 출력, 색인 생성을 어떻게 하는지 4개의 장에 걸쳐 설명합니다. 제VI편에서는 \LaTeX 문서를 효율적으로 작성하는 방법을 설명합니다. 매크로와 카운터 기능, 문서의 레이아웃을 조정할 때 많이 사용되는 ‘길이 변수 제어 문자열’, 그리고

여러 파일에 소스 코드를 분리해 작성해 컴파일 하는 방법을 설명합니다. 마지막으로, 제 VII 편에서는 문서의 모습을 입맛에 맞게 변경하는 여러 방법을 설명합니다. 문서의 레이아웃과 구조, 문서의 언어, 글꼴을 바꾸고 검은색과 흰색뿐이었던 문서에 색을 추가하는 방법을 설명합니다.

마지막 부록에서는 \LaTeX 의 문법 외적인 부분을 설명합니다. 컴퓨터에 \TeX 을 설치해 사용하는 방법, 문제가 생겼을 때 해결하는 방법을 설명하고, 본문에서 못 다 한 이야기와 문서를 작성하는 데 도움이 될 만한 정보를 여기에 적어 두었습니다. 책 곳곳에 있는 예제의 정답도 여기에서 확인할 수 있습니다.

\LaTeX 을 처음 사용하시는 분은 책을 처음부터 제 III 편까지 순서대로 읽는 것을 권장합니다. 제 I 편에서 기본적인 문법을 설명하고, 제 II 편과 제 III 편에 걸쳐 본문과 수식을 작성하는 기본적인 문법을 설명하고 있기 때문입니다. 뒤쪽의 내용은 \LaTeX 을 사용하면서 더 많은 기능이 필요할 때 읽어 나가면 됩니다.

한편, \LaTeX 을 이미 사용해 본 경험이 있다면 책의 내용을 순서대로 읽어도 좋고, 필요한 부분을 찾아 읽어도 좋습니다. 이 책에 담겨 있는 각 내용은 자연스럽게 연결되도록 구성되어 있습니다. 하지만, 이미 잘 알고 있는 내용을 다시 읽을 필요는 없으므로 그런 부분은 건너뛰며 읽는 것이 더 효율적일 것입니다.

독자 여러분의 상황에 맞게 이 책을 유용하게 사용하시기 바랍니다.

변경 사항

2023년 8월 30일 초안(main-week9.pdf)을 공개한 이후의 수정 사항은 여기에 작성됩니다. 작은 수정 사항은 작성하지 않았습니다. 최종본 생성 시에는 여기의 내용은 삭제됩니다.

2023년 11월 2일: main-week11.pdf

부분별 변경 사항

제4장: 4.1절에서 각 클래스를 사용해 문서를 조판했을 때의 모습을 추상화하여 나타냈습니다. 4.2절에서 각 클래스의 옵션이 문서의 조판 결과로 어떻게 나타나는지 자세한 설명과 예시를 추가하였습니다.

제8장: 8.2절에서 글꼴 종류와 크기를 바꾸는 예시를 추가하였습니다. 지원되지 않는 글꼴 스타일을 지정한 경우 다른 글꼴로 대체될 수 있다는 설명을 추가했습니다. 글꼴의 크기를 보여주는 예시를 표 8.2에 추가하였습니다.

제10장: 도입부의 설명 일부를 10.1절로 이동하였습니다. 10.1절에서 \LaTeX 에서 수식을 조판하는 방법과 함께 수식 입력 모드와 텍스트 모드의 차이를 더 자세히 설명했습니다. 번호 붙이는 수식 작성 환경인 equation 환경의 설명을 짧게 추가하였습니다. 10.2절에서 $\tfrac{}{}{}$ 를 사용하는 예시를 추가하였습니다. 첨자 문법의 설명을 부연하고 상황에 맞는 예시를 각각 추가하였습니다. 10.3절에서 법 연산자를 조판하는 제어 문자열을 더 자세히 설명하였습니다. 그리

스 문자를 입력할 때 사용할 수 있는 코드 일부를 삭제하였습니다. 이탤릭체 그리스 대문자를 입력하는 제어 문자열을 나열한 표를 추가하였습니다. 비슷하게 생긴 기호 사이의 차이를 더 잘 보여줄 수 있도록 예시 표시 방법을 변경하였습니다. 세로줄과 겹세로줄을 입력하는 제어 문자열 \lvert , \lVert , \rvert , \rVert 의 설명을 추가하였습니다.

제11장: 11.3절에서 표 11.2를 3열로 변경하였습니다. 11.4절에서 mathtools 패키지가 제공하는 cases 환경을 비교하는 표 11.3을 추가하였습니다.

제13장: 13.3절에서 화살표를 굽히는 bend left 및 bend right 옵션의 설명을 추가했습니다.

제14장: 표 문법을 더 쉽게 이해할 수 있도록 14.1절과 14.2절의 내용을 재구성하고, tabular 환경과 array 환경을 통합해 설명하였습니다. 기초적인 표 문법을 14.1절에, 환경에서 사용할 수 있는 전체 기능을 14.2절에 설명하였습니다. 14.3절에는 tabular* 환경의 설명을 추가하였습니다.

2023년 10월 6일

전반적 변경 사항

L^AT_EX 로고에서 ‘L’, ‘A’, ‘T’ 사이의 간격을 보기 좋게 수정하였습니다.

부분별 변경 사항

제6장: 6.2절에서 subsection, subsubsection, subparagraph의 번역을 각각 ‘부절 → 하위 절’, ‘소절 → 최하위 절’, ‘소문단 → 하위 문단’으로 변경하였습니다. 장절 표제를 생성하는 제어 문자열의 문법 설명 상자를 별표 붙은 버전과 별표 없는 버전을 합쳐서 나타냈습니다. 설명을 보강하고, book 클래스의 문서 구조를 설명하는 그림을 추가하였습니다.

제10장: 10.3절과 10.4절을 10.3절로 병합하고 같은 모양으로 생긴 기호에 대한 설명을 보강하였습니다. 10.5절과 10.6절의 내용 중 글꼴을 제외한 내용은 10.4절로, 글꼴에 관한 내용은 10.5절로 재구성하였습니다. 10.5절에서 표 10.33에

`\mathrm` 제어 문자열을 추가하였습니다.

제11장: 연산자와 적분 기호, 함수에 대한 설명을 11.1절로 추가하였습니다. `\operatorname`과 `\DeclareMathOperator`를 사용하는 방법을 여기로 이동하였습니다. 괄호를 입력하는 제어 문자열을 10.3절로 이동하였습니다. (괄호의 크기를 조정하는 방법은 11.2절에 그대로 있습니다.) 11.2절에서 괄호 크기를 수동으로 조절하는 방법을 더 자세히 설명하였습니다. 11.3절에서 더 큰 행렬을 조판하는 방법의 설명을 추가했습니다. 11.4절의 제목을 변경하였습니다.

제21장: 21.2절에서 제어 문자열의 두 가지 형태를 더 명확하게 설명하였습니다. `\newenvironment*`와 `\renewenvironment*`를 추가했습니다. 환경을 정의할 때 끝 부분의 정의에는 매개변수를 사용할 수 없다는 내용을 추가했습니다. 플레인 T_EX의 제어 문자열인 `\def`와 `\let`에 관한 설명을 추가했습니다.

2023년 9월 26일: main-week10~rev1.pdf

전반적 변경 사항

예시 상자에서 코드 부분과 실제 조판 모습 사이에 간격을 두어 구분했습니다.

일부 예시에 있는 수학적 오류를 바로잡았습니다.

부분별 변경 사항

서문을 새로 작성하였습니다.

제3장: 3.3절에서 공백 문자와 개행 문자에 관한 설명을 추가하고, 띄어쓰기와 줄바꿈에 관한 설명을 추가했습니다.

제5장: 5.1절에서 표 5.1의 로마자를 T_EX 기본 글꼴로 조판하였습니다. 에스체트에 관한 설명이 삭제되었습니다.

제6장: 제목을 변경하였습니다.

제7장: 7.1절과 7.2절의 순서를 변경하였습니다.

2023년 9월 19일: main-week10.pdf

전반적 변경 사항

코드의 글꼴을 ‘Computer Modern Typewriter + 유엔피플 고딕 UNI’ 조합에서 ‘본모노’로 변경하였습니다.

문서 내에서 예시 코드 조판 패키지를 listings에서 minted로 변경하였습니다. 이에 따라 코드에 여러 색상이 사용되고 더 미려하게 표시됩니다.

예시 코드와 조판 결과의 글꼴 크기를 본문 글꼴 크기와 일치시켰습니다(8 pt → 9 pt). 더불어 예시 코드의 조판 결과의 글꼴을 문서 본문에서 사용하는 ‘Crimson Pro + Sandoll 명조Neo1’ 조합에서 PDF_{La}TeX 기본값인 ‘Computer Modern Roman + 나눔명조’ 조합으로 변경하였습니다.

첫 문단 들여쓰기 간격을 클래스 기본값인 14 pt에서 1 em으로 변경하였습니다.

부분별 변경 사항

검토 시 용이하도록 목차의 앞에 짧은 목차를 추가하였습니다. 이 목차는 최종본 생성 시 제거됩니다.

제2장: 예시 문서의 그림을 세로로 회전하여 삽입하였습니다.

제3장: 3.1절에서 제어 문자열을 사용할 때 중괄호를 생략하는 예시를 추가하였습니다. 표 3.1에서 물결표와 캐럿 기호에 설명을 추가하였습니다. 3.3절에서 행중수식과 별행수식의 설명 위치가 조정되었습니다.

제5장: 제목이 ‘문자 입력과 언어 처리’가 ‘문자 입력하기’로 변경되었습니다. 언어 처리와 관련된 내용은 초심자에게 불필요하다고 판단하여 제 VII 편 26장으로 옮겼으며, 이에 따라 기존 26장과 27장이 27장과 28장으로 밀렸습니다. 5장 전반에서 ‘현대 엔진’이라는 용어를 ‘유니코드 엔진’으로 변경하였습니다. 5.2절에서 음수 부호를 입력하는 부분의 설명을 보강하였습니다.

뒤집힌 물음표와 느낌표 부분의 설명을 보강하였습니다. 제대로 출력되지 않는 기호에 관한 설명이 교체되었습니다.

제6장과 제7장의 순서가 변경되었습니다. 원래 의도한 것과 반대 순서로 작성되어 있었습니다.

제6장: 표지 작성 예시를 변경하였습니다.

제7장: 7.2절에서 verse 환경 내에서 줄 바꿈이 일어나는 예시를 추가하였습니다. 7.1절에서 `\setlist` 제어 문자열과 관련된 설명을 보강하고 예시를 추가하였습니다.

제8장: 도입부에서 ‘서식 선언형’ 제어 문자열과 ‘범위 지정형’ 제어 문자열의 유효 범위 설명을 추가했습니다. 8.2절의 예제 8.2에서 불필요한 문장이 삭제되었습니다.

제9장: 9.1절에서 `\newtheorem` 제어 문자열의 문법 설명 상자를 별표 붙은 것과 통합하였습니다. 이에 따라 일부 항목의 설명 순서와 예시가 변경되었습니다. 9.3절에서 `\theoremstyle` 제어 문자열의 문법 설명 상자가 추가되었습니다. `\newtheoremstyle` 제어 문자열의 예시가 삭제되었습니다. 9.4절에서 증명 끝 기호를 바꾸는 방법이 추가되었습니다.

제10장: 표 10.1에서 각 명령어를 쉼표 대신 ‘또는’으로 구분하였습니다. 기존의 예제 10.1이 삭제되었습니다. `\frac`의 문법 설명 상자가 수정되고, `\binom`의 문법 설명 상자가 추가되었습니다. 위 첨자와 아래 첨자 문법 설명 상자가 분리되었습니다. 또, 여기에서 중괄호를 생략할 수 있음을 명시적으로 설명하였습니다. 플레인 TeX의 문법인 `\over`, `\root` 등에 관한 조언이 추가되었습니다. ‘*n*항 연산자’에 대해 설명을 추가하였습니다. 정체로 쓰는 함수의 입력 방법으로 `\operatorname` 제어 문자열을 추가하였습니다. 비슷하게 생긴 기호의 구분 부분에서 쉼표를 추가하였습니다. 화살표 위에 문자를 입

력하는 방법을 10.6절로 분리하였으며, 여기에 `\stackrel`과 `\stackop` 제어 문자열을 추가하였습니다.

제11장: 표 11.1을 4열로 변경하였습니다. 괄호 크기를 조정하는 예시를 변경하였습니다. `dcases` 환경을 더 적극적으로 사용하도록 설명과 예시를 변경했습니다. 몇 가지 예제를 추가하였습니다.

제13장: 불필요한 서식 지정 설명을 삭제했습니다. `\arrow`와 `\ar` 제어 문자열의 문법 설명 상자를 변경하였습니다. 예제 13.1을 추가하였습니다.

제14장: 여러 예시를 추가하고 수정하였습니다. `array` 패키지가 제공하는 `m`인자와 `b`인자의

예시를 추가하였습니다.

제16장: !인자에 부연 설명을 추가했습니다.

제19장: 표 19.3의 필수 항목을 강조 표시하였습니다. 표 19.4를 추가하였습니다.

제20장: 색인 항목 추가 부분의 설명을 일부 수정하였습니다.

제21장: `\newcommand`의 별표 붙은 버전의 설명을 추가했습니다. 유용한 매크로 중 연산자 부분을 약화했습니다. 환경 정의 부분의 예시를 교체하였습니다.

제 22 장: 164 쪽의 예시에서 `\themy-counter`에 대한 임시 설명을 추가했습니다.

부록 E를 신설하고 소개할 패키지의 목록 초안을 작성하였습니다.

변경 예정 사항

문서 내 모든 표의 디자인을 통일합니다. (줄 간격, 구분선 등)

`<argument>` 형태로 문법을 설명할 때 화살괄호 안에는 영어만 입력합니다. 한국어로 작성된 내용은 영어로 고칩니다.

제8장에서 코드 작성 패키지를 간략하게 소개합니다.

제12장의 설명을 전체적으로 되짚어봅니다.

제13장에서 화살표 위에 라벨을 붙이는 옵션 (`description`)의 설명과 예시를 추가합니다.

제16장에서 그림 16.1 내의 글꼴 크기를 조정합니다.

제17장에서 `xurl` 및 `cleveref` 패키지를 소개합니다.

제19장에서 여러 문헌 표시 스타일을 소개합니다.

제20장의 레이아웃이 망가지지 않도록 합니다.

제26장에 `babel`과 `polyglossia`를 사용하는 예시를 추가합니다. 중국어 문서 조판 예시를 추

가합니다.

제27장에 `unicode-math` 패키지의 설명을 추가합니다.

부록 D에 몇 가지 유용한 템플릿을 작성합니다.

부록 E에 몇 가지 중요한 패키지를 짧막하게 소개합니다.

`beamer` 클래스의 간단한 사용법을 추가합니다.

고민 중인 사항

그래픽 패키지의 기본적인 사용 방법을 추가합니다.

제12장에서 수식에 상자 치는 문법을 소개합니다.

제19장에서 `biblatex` 패키지, `natbib` 패키지, `biber` 프로그램을 소개합니다.

제22장에서 표 22.2처럼 \LaTeX 의 기본 카운터 표시 형식도 출력 예시를 제시합니다.

제24장에서 `list` 환경과 `tabbing` 환경의 설

명을 추가합니다.

제25장에서 채움선(leader)의 설명을 추가합니다.

제26장에서 RTL 언어에 관한 설명을 추가합니다.

줄바꿈과 새 줄 시작, 페이지 나눔과 새 페이지

시작을 구분해서 설명합니다.

상자에 관한 설명을 추가합니다. (`\mbox`, `\fbox`, `\parbox`, `\lrbox`, `\sbox`, `minipage` 환경)

패키지와 클래스 파일을 직접 작성하는 방법을 설명합니다.

lshort와 비교했을 때 이 책에 없는 내용

참고: `lshort.pdf`는 ctan.org/pkg/lshort-english(버전 6.4) 또는 tobi.oetiker.ch/lshort/lshort.pdf(버전 7.0, 시험 버전)에서 내려받을 수 있습니다. 다음 내용은 참고 사항이며, 여기의 모든 내용을 이 책에 추가하겠다는 의미는 아닙니다.

줄바꿈과 페이지 나눔 — `\newline`, `\(no)linebreak`, `\newpage`, `\(no)pagebreak`, `\sloppy`, `\fussy`

몇 가지 문자열 — `\TeX`, `\LaTeX`, `\LaTeXe`
유럽 언어의 몇 가지 타이포그래피 규칙 (하이픈 처리, 합자 처리, 마침표 뒤의 공백 등) — `\hyphenation`, `\-`, `\@`, `\(non)frenchspacing`, `\textcompwordmark`

줄바꿈을 허용하는 슬래시 — `\slash`
`polyglossia` 패키지 사용 방법, `fontspec` 패키지로 새로운 글꼴 가족 선언, RTL 언어 입력 방법
코드 작성 — `verbatim`, `listings`, `minted` 패키지

표 작성 — `booktabs`, `longtable` 패키지, `array` 패키지의 `\newcolumntype` 제어 문자열

유동 개체 — `newfloat` 패키지

파일 분리 — `syntonly` 패키지

수식 기능 — `\overset`, `\underset`, `multiline` 환경에서 `\shove[left/right]`

공백 — `\(v/p)phantom`

큰 행중수식이 입력됐을 때의 줄 간격 조정 — `\smash`

`BibLaTeX`과 `Biber`를 사용한 참고 문헌 목록 생성 및 인용

`hyperref` 패키지 — `\ref*`, `\autoref`, `\autopageref`, 링크 표시 속성, 문서의 메타데이터 지정

`tikz` 패키지를 사용한 그래픽 작성

조건문, 매크로 정의, 패키지 작성

글꼴 — 이탤릭 수정 `\/`, `fontspec` 패키지의 추가 기능, 수식 글꼴 변경 패키지

공백 — `\stretch`

페이지 스타일과 레이아웃 — `\pagestyle`, `\thispagestyle`, `\pagenumbering`, `geometry` 패키지, `fancyhdr` 패키지

상자 — `\mbox`, `\fbox`, `\parbox`, `minipage` 환경, `\raisebox`

간단한 차례

검토 작업 시의 편의를 위해 간단한 목차를 추가되었습니다. 장절 표제의 번호를 클릭 또는 터치하면 해당 부분으로 이동합니다.

1 \TeX 과 \LaTeX 소개

I \LaTeX 에 관한 기본적인 지식

2 문서를 작성하고 조판하는 과정

3 코드의 구조와 문법

II 본문 작성하기

4 문서의 클래스

5 문자 입력하기

6 문서의 형식 맞추기

7 문단 만들기

8 서식 지정하기

9 정의, 정리, 증명 작성하기

III 수식 작성하기

10 수식 입력하기

11 복잡한 수식 문법

12 별행수식 입력하기

13 가환 도표 그리기

IV 표와 그림 다루기

14 표 작성하기

15 그림 삽입하기

16 표와 그림 배치하고 캡션 달기

V 상호 참조 및 특수 기능

17 상호 참조 및 링크

18 목차

19 참고 문헌

20 색인

VI 효율적으로 \LaTeX 문서 작성하기

21 매크로 정의하기

22 카운터와 길이 정의하기

23 문서 파일 분리하기

VII 문서 사용자 지정하기

24 레이아웃 편집하기

25 공백과 구분선

26 다른 언어로 문서 작성하기

27 글꼴 바꾸기

28 색 사용하기

VIII 프레젠테이션 작성하기

29 소개와 기본 문서 구조

30 프레젠테이션의 형식 맞추기

부록

A 컴퓨터에 \TeX 설치해서 사용하기

B 패키지 및 프로그램 관리하기

C 문제 해결하기

D 템플릿

E 유용한 패키지

F 예제 정답

차례

들어가며	iii
변경 사항	vii
차례	xv
1 T_EX과 L^AT_EX 소개	1
1.1 T _E X이란?	1
1.2 L ^A T _E X이란?	2
1.3 연관된 프로그램	2
1.4 관련 글과 웹사이트	3
<hr/>	
I L^AT_EX에 관한 기본적인 지식	
2 문서를 작성하고 조판하는 과정	7
2.1 문서 작성을 위한 환경 갖추기	7
2.2 소스 파일 작성하고 컴파일 하기	8
2.3 컴파일 오류와 경고	8
2.4 두 번째 예제: 한국어 문서 작성하기	10
3 코드의 구조와 문법	13
3.1 제어 문자열과 환경	13
3.2 문서의 클래스와 전처리부	16
3.3 문서 작성	17
3.4 원시적 템플릿	19

II 본문 작성하기

4 문서의 클래스	23
4.1 표준 클래스	23
4.2 클래스 옵션	25
4.3 추가 클래스	29
5 문자 입력하기	31
5.1 로마자 입력하기	31
5.2 문장부호 입력하기	32
5.3 기호 입력하기	34
6 문서의 형식 갖추기	39
6.1 표지 만들기	39
6.2 장과 절 구성하기	40
6.3 각주	42
7 문단 만들기	45
7.1 목록	45
7.2 요약문, 운문과 인용문	47
7.3 문단 정렬 바꾸기	50
8 서식 지정하기	53
8.1 특정 부분 강조하기	53
8.2 글꼴 바꾸기	54
8.3 입력한 내용을 그대로 조판하기	57
9 정의, 정리, 증명 작성하기	59
9.1 \newtheorem으로 환경 정의하기	59
9.2 환경의 번호 다루기	60
9.3 문단 스타일 바꾸기	62
9.4 증명 작성하기	63

III 수식 작성하기

10 수식 입력하기	67
10.1 수식 입력 모드	67

10.2	기본 문법	70
10.3	기호 입력하기	74
10.4	장식 기호	86
10.5	수식 글꼴	89
11	복잡한 수식 문법	91
11.1	연산자, 적분 기호와 함수	91
11.2	괄호 크기 조절하기	93
11.3	행렬 작성하기	95
11.4	여러 식 묶어서 작성하기	97
11.5	바둑판 모양으로 식 배열하기	99
12	별행수식 입력하기	101
12.1	수식에 번호 붙이기	101
12.2	여러 줄에 걸쳐 수식 작성하기	103
12.3	여러 열의 수식 동시에 정렬하기	106
13	가환 도표 그리기	109
13.1	tikzcd 환경	109
13.2	대상 배치하기	110
13.3	화살표 그리기	110
13.4	화살표에 라벨 붙이기	115
<hr/>		
IV	표와 그림 다루기	
14	표 작성하기	119
14.1	기본적인 표 그리기	119
14.2	환경의 세부 문법	121
14.3	칸 합치고 구분선 그리기	127
14.4	표 전체 폭 조정하기	130
15	그림 삽입하기	133
16	표와 그림 배치하고 캡션 달기	135
16.1	유동 개체로써 표와 그림 배치	135
16.2	캡션 작성	137

V 상호 참조 및 특수 기능

17 상호 참조 및 링크	141
17.1 참조하기	141
17.2 하이퍼링크	143
18 목차	145
18.1 목차 만들기	145
18.2 목차 내용 수정하기	146
19 참고 문헌	149
19.1 \LaTeX 에서 문헌을 관리하는 방법	149
19.2 문헌 정보 파일 작성하기	150
19.3 \LaTeX 소스 파일에서 문헌 정보 파일 사용하기	157
19.4 소스 파일 컴파일 하기	159
20 색인	161
20.1 \LaTeX 에서 색인을 생성하는 과정	161
20.2 \LaTeX 소스 파일 작성하기	162
20.3 색인 파일 생성하고 색인 조판하기	166

VI 효율적으로 \LaTeX 문서 작성하기

21 매크로 정의하기	171
21.1 매크로란?	171
21.2 제어 문자열 정의하기	172
21.3 유용한 매크로들	175
21.4 환경 정의하기	178
22 카운터와 길이 정의하기	179
22.1 카운터	179
22.2 길이	184
22.3 길이 변수 제어 문자열	185
23 문서 파일 분리하기	189

VII 문서 사용자 지정하기

24 레이아웃 편집하기	195
24.1 페이지 설정 바꾸기 · 195	
24.2 여러 단으로 문서 작성하기 · 196	
24.3 문단의 모양 바꾸기 · 199	
24.4 오버풀과 언더풀 · 200	
25 공백과 구분선	201
25.1 공백 삽입하기 · 201	
25.2 구분선 넣기 · 203	
26 다른 언어로 문서 작성하기	205
26.1 로마자, 키릴 문자, 그리스 문자를 사용하는 언어 · 205	
26.2 한중일 언어 · 206	
27 글꼴 바꾸기	209
27.1 글꼴을 지정하는 패키지로 글꼴 변경하기 · 209	
27.2 fontspec 패키지로 글꼴 변경하기 · 210	
27.3 한글·한자 글꼴 지정하기 · 212	
28 색 사용하기	215
28.1 색상 지정하기 · 215	
28.2 색상 정의하기 · 217	
28.3 색상 사용하기 · 218	

VIII 프레젠테이션 작성하기

29 소개와 기본 문서 구조	223
29.1 \LaTeX 으로 프레젠테이션 문서 만들기 · 223	
29.2 기본적인 문법 · 223	
29.3 프레임 만들기 · 226	
30 프레젠테이션의 형식 갖추기	227
31 프레젠테이션의 레이아웃 바꾸기	229
31.1 클래스 옵션 · 229	

31.2	테마	· 229
31.3	프레임의 구성 요소	· 229
32	오버레이 사용하기	· 231
32.1	오버레이 조건	· 231
32.2	명령어에 오버레이 조건 지정하기	· 232
32.3	가려진 항목을 흐리게 표시하기	· 232
33	유인물과 노트 만들기	· 233

부록

A	컴퓨터에 \TeX 설치해서 사용하기	· 237
A.1	\TeX 배포판 설치	· 237
A.2	편집기 및 뷰어	· 238
A.3	컴퓨터에서 문서 컴파일하기	· 239
B	패키지 및 프로그램 관리하기	· 241
B.1	\TeX Live	· 241
B.2	Mac \TeX	· 242
B.3	MiK \TeX	· 243
C	문제 해결하기	· 245
C.1	오류	· 245
C.2	경고	· 248
C.3	인터넷에 검색하기	· 249
D	템플릿	· 253
E	유용한 패키지	· 255
F	예제 정답	· 259
	참고 문헌	· 261
	제어 문자열과 환경 찾아보기	· 263
	패키지 찾아보기	· 273

제 1 장

TeX과 LaTeX 소개

이번 장에서는 TeX과 LaTeX이 무엇인지 알아봅니다. LaTeX을 사용해 문서를 작성하는 방법에 관한 내용은 담고 있지 않으므로, 문서를 작성하는 방법을 빠르게 익히고 싶으신 분들은 이번 장을 넘어 가져도 좋습니다.

1.1

TeX이란?

TeX은 도널드 커누스(Donald Ervin Knuth) 교수가 개발한 프로그램으로, 강력한 수식 작성 기능을 포함하고 있는 문서 조판 프로그램입니다. 1978년 첫 버전이 발표된 이후, 1982년 두 번째 버전, 1989년 세 번째 버전(TeX3)이 발표되었고, 이후로도 계속 업데이트되어 왔으며 2023년 기준 최신 버전은 2021년 1월 발표된 버전 3.141592653입니다. 값을 보면 알 수 있듯, TeX의 버전은 업데이트 될수록 원주율 π 에 수렴합니다. 1989년 발표된 버전은 매우 안정하며, 이 이후로는 큰 변경 사항 없이 버그 수정만 이루어지고 있습니다.

TeX을 사용해 문서를 작성하는 것은 XML 등의 마크업 언어를 사용해 문서를 작성하는 방식과 비슷합니다. 프린터로 출력하였을 때의 최종 결과물을 실시간으로 확인하면서 문서를 작성하는 방식이 아니고, ‘어떤 내용을 조판해라’, ‘여기에는 어떤 서식을 지정해라’, ‘여기에서 페이지를 넘겨라’ 등 최종 결과물이 어떻게 보여야 하는지를 명령어로 지정하고 이를 컴파일하여 결과물을 얻는 방식으로 문서를 작성합니다. 처음에는 ‘문서를 작성하려고 코딩을 해야 돼?’ 하는 의문이 생길 수도 있겠지만, 이 방식에 익숙해진다면 수많은 요소를 마우스 없이 키보드만으로 작성할 수 있어 편리할 것입니다.

TeX의 대표적인 장점은 강력한 수식 편집 기능입니다. 기호와 기호 사이의 간격을 스스로 알맞게 조정하고 수식에 자동으로 번호를 붙여 주며, 복잡한 수식을 보기 좋게 정렬해 줍니다. 또 다른 장점은 프로그래밍 기능으로, 자주 사용하는 상용구를 매크로로 정의하고, 반복문과 조건문도 사용할 수 있습니다. 또한, 가장 기본적인 문서 조판 능력에도 충실합니다. 문서를 작성하는 사람이 내용만을 입력하면, TeX이 글자와 글자, 단어와 단어, 문단과 문단 사이의 간격을 보기 좋게 조절하고, 적절한

곳에서 줄바꿈 및 페이지 나눔 처리를 해 주어 문서를 읽는 사람들로 하여금 편안하게 읽을 수 있게 해 줍니다. 이러한 장점 덕분에, 오늘날 \TeX 은 수학·과학·공학 분야에서 문서를 출판하는 데 거의 표준처럼 사용되고 있습니다.

이 프로그램의 이름 \TeX 은 어떻게 읽어야 할까요? 이 프로그램을 처음 본 한국인이라면, 아마 영어식 발음대로 ‘텍스’와 가깝게 읽을 것입니다. 하지만 이 세 글자 ‘T’, ‘E’, ‘X’는 로마자가 아닌 그리스 문자 타우 τ , 엡실론 ϵ , 카이 χ 로, 각 글자의 발음을 살려 ‘테크’ 내지는 ‘테흐’에 가깝게 발음하는 것이 옳습니다(IPA로 \TeX 의 올바른 발음을 나타내면 $[\text{t}\epsilon\chi]$ 이 됩니다). 많은 한국어 화자들은 ‘X’의 정확한 발음이 익숙하지 않아 보통 ‘텍’이라고 발음합니다. 한편, 이 프로그램의 이름은 ‘기술’을 뜻하는 그리스어 접두사 ‘ $\tau\epsilon\chi$ ’에서 유래했습니다.

1.2

\LaTeX 이란?

\LaTeX 은 레슬리 램포트(Leslie Lamport) 교수가 개발한 \TeX 매크로 집합입니다. 원래의 \TeX 이 정의한 명령어는 여러 문서를 작성하기에는 어렵고 복잡해서, 이를 더 쉽게 사용할 수 있도록 만들어진 것이 \LaTeX 입니다.

1984년 첫 버전(\LaTeX 2.09)이 공개되었으며, 10년 뒤인 1994년에 두 번째 버전(\LaTeX 2 ϵ)이 공개되었습니다. \LaTeX 3는 현재 개발 중이며, 기존의 \LaTeX 2 ϵ 에 다양한 프로그래밍 언어적 특징을 접목시킨 것이 큰 특징이라고 합니다. 오늘날 \LaTeX 2.09는 더 이상 사용되지 않으며, 오늘날 볼 수 있는 \LaTeX 문서는 일반적으로 \LaTeX 2 ϵ 를 사용해 작성된 것입니다.

\LaTeX 을 사용해 문서를 작성하는 경우, 문서의 논리적 구조에 집중하게 됩니다. 하나의 책은 여러 개의 문단으로 이루어져 있고, 각 문단은 비슷한 내용끼리 묶여 장과 절을 구성합니다. 책의 앞부분에는 이들의 제목을 모은 목차가 들어갈 것입니다. 책에 표와 그림이 많이 있다면 표와 그림의 차례도 책에 들어갈 수 있고, 본문을 작성할 때 표와 그림을 상호 참조할 일도 많을 것입니다. \LaTeX 을 사용하면 이들을 간단한 명령어 몇 개로 처리할 수 있게 됩니다. 여기에 \TeX 의 기본 기능인 매크로 정의, 조건문, 반복문까지 함께 사용한다면 문서 곳곳에 사용할 서식을 더 일관적으로 적용할 수 있습니다.

앞 절에서 \TeX 을 ‘테크’라고 발음한다고 설명하였습니다. 마찬가지로, \LaTeX 은 ‘라테크’나 ‘라테흐’ 정도로 발음합니다. 다만, 앞서 설명했듯 X의 발음이 익숙하지 않은 한국어권 사람들은 ‘라텍’ 또는 ‘레이텍’이라고 흔히 부릅니다. (절대 ‘라텍스’라고 읽으면 안 됩니다!)

1.3

연관된 프로그램

\LaTeX 을 사용해 문서를 작성하는 데에는 여러 프로그램이 필요합니다.

- 편집기(editor) \LaTeX 소스 코드를 작성하는 데 필요한 프로그램입니다. 메모장을 사용할 수도 있고, \LaTeX 문서 작성에 편리한 기능을 갖춘 \TeX works나 \TeX Shop 등을 사용할 수도 있습니다.

- 엔진(engine) 작성한 \LaTeX 문서를 컴파일 하는 데 사용하는 프로그램입니다. 앞에서 설명한 \TeX 이 가장 대표적인 엔진이며, 아래에서 설명할 다양한 파생 프로그램도 모두 엔진에 속합니다.
- 뷰어(viewer) 엔진을 사용해 컴파일 해서 얻은 DVI 또는 PDF 문서를 보기 위한 프로그램입니다 (오늘날 DVI 파일은 더 이상 사용되지 않으므로, PDF 뷰어만을 갖추셔도 충분합니다).

위의 세 가지 프로그램을 갖추시면 \LaTeX 문서를 작성할 수 있습니다. 이러한 프로그램을 설치하는 방법은 부록에서 설명합니다.

컴파일 엔진

\TeX 문서를 컴파일 하는 엔진에는 여러 종류가 있습니다. 1978년 첫 버전의 \TeX 이 발표된 이후, \TeX 에서 파생된 다양한 프로그램이 등장하였습니다. \TeX 이 업데이트되면서 다양한 엔진이 등장하고 사라졌으며, 그에 대응하는 \LaTeX 매크로 집합도 몇몇 등장하였습니다. 아래 목록에 오늘날 자주 사용되는 엔진인 pdf \TeX , X \LaTeX , Lua \TeX 이 무엇인지 정리하였습니다.

- \TeX 커누스 교수가 개발한 가장 기본적인 \TeX 엔진입니다. 다른 엔진들과 구분하여 ‘플레인 \TeX ’이라고 부르기도 합니다.
- PDF \TeX 플레인 \TeX 에서 생성하는 DVI 파일을 외부 프로그램을 사용해 PDF 파일로 변환하는 과정을 거치지 않고, 컴파일 과정에서 PDF 파일을 바로 생성하도록 합니다.
- X \LaTeX 다국어 지원을 위해 만들어진 \TeX 엔진입니다. UTF-8 인코딩을 사용한 \TeX 파일을 사용하며, TTF 및 OTF 글꼴 파일을 사용합니다. 이 엔진은 ‘지텍(*zee-tekh*)’이라고 읽습니다.
- Lua \TeX 스크립트 언어인 Lua를 사용할 수 있으며, X \LaTeX 과 같이 UTF-8 인코딩으로 저장된 \TeX 파일을 사용하고, TTF 및 OTF 글꼴을 사용합니다.

여기의 엔진에 \LaTeX 문법을 사용해 문서를 작성하는 경우, 각각 \LaTeX , pdf \LaTeX , X \LaTeX , Lua \LaTeX 으로 문서를 작성한다고 말하기도 합니다.

일반적으로는 pdf \TeX 으로 문서를 작성하며, 다양한 글꼴을 쉽게 사용할 수 있는 X \LaTeX 도 자주 사용됩니다. Lua를 사용해 복잡한 기능을 사용하려는 경우에는 Lua \TeX 을 사용합니다.

1.4

관련 글과 웹사이트

이 책에서 자주 언급될 글과 웹사이트를 소개하겠습니다. 글은 주로 \TeX 과 \LaTeX 의 문법, 여러 패키지와 관련된 내용을 설명하고 있으며, 웹사이트는 주로 \TeX 과 관련된 파일 및 프로그램을 개발 및 배포하는 곳입니다.

- The \TeX book^[2]: \TeX 을 개발한 도널드 커누스 박사가 직접 작성한 책으로, \TeX 이 어떤 방식으로 작동하는지, 문법은 어떻게 되고 어떻게 \TeX 을 사용하는지 자세히 설명되어 있습니다. (아래에서 설명할 웹사이트인) CTAN에서 이 책의 \TeX 소스 파일을 내려받을 수 있으며, 서점에서 출판된 책을 구매할 수도 있습니다.

- \LaTeX : A Document Preparation System (Second edition)^[3]: \LaTeX 을 개발한 레슬리 램포트 박사가 직접 작성한 책으로, $\text{\LaTeX} 2_{\epsilon}$ 에 대해 설명하고 있습니다. 서점에서 출판된 책을 구매할 수 있습니다.
- The \LaTeX Companion(Second edition)^[1]: \LaTeX 을 사용하는 방법에 더불어, 외부 패키지 등을 불러와 \LaTeX 에서 할 수 있는 작업을 여럿으로 구분하고 그에 따라 매우 상세한 설명을 적은 책입니다. 각 주제별로 사용할 수 있는 패키지뿐 아니라 각각의 특징도 자세히 설명되어 있습니다. 2023년에는 제3판이 출간되었습니다.
- The Not So Short Introduction to $\text{\LaTeX} 2_{\epsilon}$ ^[5]: $\text{\LaTeX} 2_{\epsilon}$ 를 사용하는 방법을 간략하게 설명한 문서입니다. 오스트리아에서 독일어로 작성한 $\text{\LaTeX} 2.09$ 입문서를 $\text{\LaTeX} 2_{\epsilon}$ 에 맞게 발전시킨 문서로, 영어를 비롯해 한국어를 포함한 수많은 언어로 번역되어 있습니다.
- CTAN(Comprehensive \TeX Archive Network): 여러 \TeX 엔진에서부터, \LaTeX 매크로 집합, 패키지, 문서 클래스 파일 등에 관한 모든 정보를 확인할 수 있습니다.
URL: <https://ctan.org/>
- TUG(\TeX User Group): \TeX 사용자 그룹으로, \TeX 자체에 관한 정보를 얻을 수 있으며, \TeX 관련 프로그램을 다운로드할 수 있습니다.
URL: <https://tug.org/>
- KTUG(Korean \TeX User Group): 한글 \TeX 사용자 그룹으로, 한글 및 한국어를 사용해 \TeX 문서를 조판하는 경우에는 이곳에서 정보를 얻는 것이 좋습니다.
URL: <https://www.ktug.org/>
- The \LaTeX Project: \LaTeX 및 $\text{\LaTeX} 3$ 의 개발을 진행하고 있는 웹사이트로, \LaTeX 에 대한 설명과 역사를 이곳에서 확인할 수 있습니다.
URL: <https://www.latex-project.org/>
- Overleaf: 온라인 \LaTeX 문서 작성 플랫폼이며, \LaTeX 을 사용하는 방법에 관한 많은 문서를 제공합니다.
URL: <https://www.overleaf.com/>

제 I 편

LaTeX에 관한 기본적인 지식

제 I 편에서는 문서를 작성하는 데 쓰이는 다양한 명령어가 아닌, LaTeX에 관한 기본적인 지식을 배웁니다. LaTeX을 처음 사용하는 초보자를 위한 것으로, LaTeX을 사용해 문서를 작성하는 과정을 알고 있고, LaTeX의 기본적인 문법을 알고 있다면 이 부분을 생략하셔도 좋습니다. 하지만 여기의 내용은 모든 LaTeX문서를 작성할 때 배경 지식으로 잊지 않고 있어야 하는 내용이므로, 잘 모르는 내용이 있다면 이 부분을 꼼꼼히 읽어 두기를 권장합니다.

제2장은 LaTeX을 사용해 문서를 만드는 전반적인 과정을 알아봅니다. 아주 기초적인 코드를 입력하고, 문서를 어떻게 얻으며, 컴파일 과정에서 발생한 오류는 어떻게 해석하는지 알 수 있습니다.

제3장은 LaTeX의 기본적인 문법과 문서 구조를 배웁니다. 제어 문자열과 환경이 무엇인지, 그 인자와 유효 범위는 어떻게 지정하는지 알아보고, 전처리부에서 어떤 작업을 하고, 본문에서 수식은 어떻게 작성하는지 알 수 있습니다.



제 2 장

문서를 작성하고 조판하는 과정

\LaTeX 의 문법은 어떻게 되고, 원하는 내용을 어떻게 작성하는지 알아보기에 앞서, 먼저 문서를 작성하기 위한 환경을 준비하고 소스 파일을 작성한 뒤 \LaTeX 을 실행하여 PDF 문서 파일을 얻는 과정을 알아보시다. 문서를 작성하는 과정을 그림 2.1에 간략히 나타냈습니다.

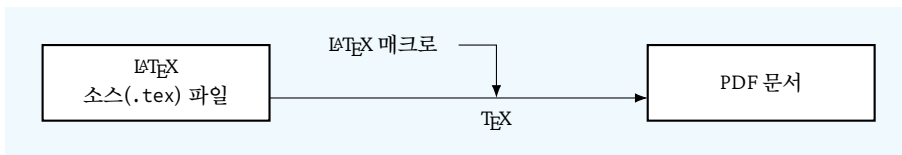


그림 2.1: \LaTeX 문서 작성 과정

2.1

문서 작성을 위한 환경 갖추기

문서를 작성하기 위해서는 먼저 소스 코드를 작성하는 편집기를 준비해야 합니다. 편집기에는 여러 가지가 있지만, 여기에서는 복잡한 준비 과정이 필요 없는 온라인 환경인 Overleaf를 사용하겠습니다. (컴퓨터에 \TeX 과 편집기를 설치해 사용할 수도 있으며, 이 방법은 부록 A에서 확인할 수 있습니다.) 아래의 과정을 따라해 보십시오.

- 1 | Overleaf에 접속합니다. URL은 overleaf.com입니다.
- 2 | 화면 오른쪽 위의 ‘Log in’ 버튼을 클릭해 로그인 해 줍니다. Overleaf 계정이 없다면 먼저 ‘Register’ 버튼을 눌러 회원 가입한 후 로그인합니다.
- 3 | 파일 목록이 나타나면, 좌상단에 있는 ‘New Project’ → ‘Blank Project’를 클릭합니다.
- 4 | 프로젝트 이름으로 ‘First Document’를 입력한 후 ‘Create’를 클릭합니다.
- 5 | 편집기가 열리면 문서를 작성할 준비가 끝납니다.

로그인 후 나타나는 파일 목록 화면에서는 기존에 생성했던 프로젝트를 열어 수정할 수도 있습니다.

다. 또, ‘New Project’를 누른 후에는 가지고 있는 소스 코드를 올려 편집할 수도 있고, Overleaf에서 제공하는 템플릿을 사용해 새로운 문서를 만들 수도 있습니다.

2.2

소스 파일 작성하고 컴파일 하기

편집기가 준비되었다면, 아래의 과정을 따라서 직접 소스 파일을 작성해 봅시다.

- 1 | 기존에 작성되어 있던 코드를 삭제합니다.
- 2 | 아래 예시 문서 2.1의 코드를 입력합니다. 제6행 `\author` 옆의 ‘Woohyun Kang’은 자신의 (로마자) 이름으로 바꾸십시오.
- 3 | 입력한 코드가 자동으로 컴파일 되어 오른쪽 화면에 나타날 것입니다. 자동으로 컴파일이 되지 않으면 화면 위쪽의 ‘Recompile’ 버튼을 클릭하거나 단축키 `command return`을 누릅니다.

이렇게 함으로써, 첫 번째 \LaTeX 소스 파일이 작성되었습니다. 그림 2.2와 비슷한 문서를 오른쪽 화면에서 볼 수 있을 것입니다. 생성한 PDF 파일은 Recompile 버튼 오른쪽의 버튼을 클릭해 내려받을 수 있습니다. 화면 왼쪽 위의 Menu 버튼을 클릭하면 프로젝트 파일을 내려받을 수 있으며, 컴파일 할 때 사용할 엔진도 선택할 수 있습니다.

2.3

컴파일 오류와 경고

소스 파일에 문제가 있다면, 컴파일 도중 오류(error)와 경고(warning) 메시지가 나타납니다. 오류는 컴파일이 불가능할 때 발생하고, 경고는 컴파일은 진행할 수 있는 비교적 작은 문제가 생겼을 때 발생합니다.

예시 문서 2.1 First Document

```

1 \documentclass[a4paper,12pt]{article}
2
3 \usepackage[T1]{fontenc}
4
5 \title{First \LaTeX\ Document}
6 \author{Woohyun Kang}
7 \date{\today}
8
9 \begin{document}
10 \maketitle
11
12 \textbf{Hello, \LaTeX !} You have just
   created your \emph{first} \LaTeX\
   document! You can type out most of the
   characters you want here freely.
13
14 To start a new paragraph, insert empty
   line(s) between paragraphs in the source
   code. Each paragraph will be indented
   automatically.
15
16 % In each line, any letters at the
   right side of the percent symbol will
   be considered as comments.
17
18 You can also directly enter in math
   formulae like \((a^2 + b^2 = c^2)\) or
   \((e^{i\pi} + 1 = 0)\), or
19 \[ i\hbar \frac{\partial}{\partial t}
   \Psi = H \Psi. \]
```

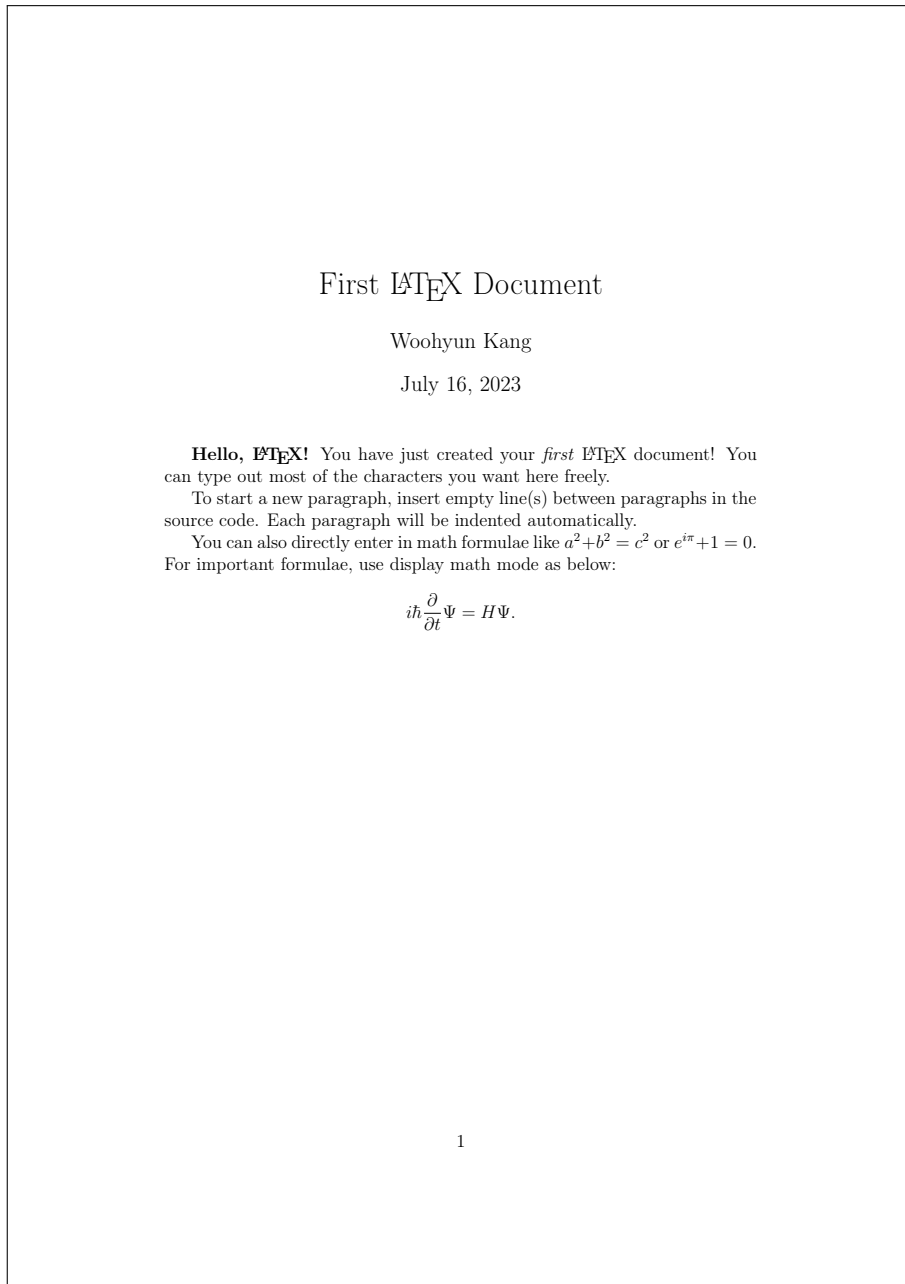


그림 2.2: 예시 문서 2.1의 컴파일 결과.

앞에서 작성했던 First Document 프로젝트를 가지고 계속 진행해 봅시다. 10행의 `\maketitle`을 `\Maketitle`로 바꾸어 의도적으로 오류를 일으켜 봅시다. 컴파일을 진행하면 Recompile 버튼 오른쪽의 ‘Show logs and messages’ 버튼에 숫자 ‘1’이 표시될 것입니다. 버튼을 클릭하면 PDF 파일 대신 오류 메시지가 표시됩니다.

오류 메시지는 문제가 발생한 파일이 무엇인지, 문제가 몇 번째 줄에서 발생했는지, 문제의 원인은 무엇인지 알려줍니다. 여기에서는 `main.tex` 파일의 제10행에서 오류가 발생했으며, `\Maketitle`이 정의되지 않은 제어 문자열(control sequence)이기 때문에 오류가 발생했음을 알 수 있습니다. 이처럼 소스 파일에 문제가 있으면 컴파일 과정에서도 문제가 발생하며, 이때 발생했던 문제에 대한 정보는 오류 메시지를 통해 알 수 있습니다.

Overleaf에서 오류 메시지는 빨간색으로, 경고 메시지는 노란색으로 표시됩니다. \LaTeX 으로 문서를 작성하다 보면 여러 메시지를 보게 될 것입니다. 어떤 상황에서 어떤 메시지가 발생하는지, 또 어떻게 해결할 수 있는지는 부록 C에서 확인할 수 있습니다.

2.4

두 번째 예제: 한국어 문서 작성하기

앞선 예시에서는 영어 문서를 작성하였습니다. 이번에는 한국어 문서를 작성한 후, 같은 과정을 통해 컴파일 해 봅시다. ‘First Korean Document’라는 새로운 프로젝트를 만든 후, 예시 문서 2.2의 코드를 입력하고 컴파일 하여 그림 2.3과 동일한 문서가 나타나는지 확인해 보십시오. 이번에도 이름 부분에는 자신의 이름을 입력하십시오.

한국어 문서를 작성하는 데에는 `kotex` 패키지가 필요합니다. 예시 문서 2.2의 제4행에 있는 코드가 바로 이 패키지를 불러오는 코드입니다. 이 패키지를 불러옴으로써 문서의 한글을 올바른 글꼴로 조판하고, 날짜나 문서의 각종 요소가 한국어로 올바르게 바뀌어 문서가 만들어집니다.

예시 문서 2.2 First Korean Document

```

1 \documentclass[a4paper,12pt]{article}
2
3 \usepackage[T1]{fontenc}
4 \usepackage[hangul]{kotex}
5
6 \title{첫 번째 한국어 \LaTeX\ 문서}
7 \author{강우현}
8 \date{\today}
9
10 \begin{document}
11 \maketitle
12
13 \textbf{Hello, \LaTeX !} 직접 \dotemph{첫
  번째} 한국어 \LaTeX\ 문서를 만들어 보았습니다.
  여기에 대부분의 문자들을 자유롭게 입력해 조판할 수
  있습니다.
14
15 새로운 문단을 시작하려면, 소스 코드에서 각 문단 사
  이에 빈 줄(들)을 입력하세요. 각 문단은 자동으로 들
  여쓰기 처리됩니다.
16
17 % 각 줄에서, 퍼센트 기호의 오른쪽에 있는 문자들은
  주석으로 처리됩니다.
18
19 \((a^2 + b^2 = c^2)\), \((e^{i\pi} + 1 =
  0)\), 또는
20 \[ i\hbar \frac{\partial}{\partial t}
  \Psi = H \Psi ]
21
22
23 \end{document}
  
```

처럼 다양한 수식을 자유롭게 조판할 수도 있습니다.

첫 번째 한국어 L^AT_EX 문서

강우현

2023년 7월 16일

Hello, L^AT_EX! 직접 첫 번째 한국어 L^AT_EX 문서를 만들어 보았습니다. 여기에 대부분의 문자들을 자유롭게 입력해 조판할 수 있습니다.

새로운 문단을 시작하려면, 소스 코드에서 각 문단 사이에 빈 줄(들)을 입력하세요. 각 문단은 자동으로 들여쓰기 처리됩니다.

$$a^2 + b^2 = c^2, e^{i\pi} + 1 = 0, \text{ 또는}$$

$$i\hbar \frac{\partial}{\partial t} \Psi = H\Psi$$

처럼 다양한 수식을 자유롭게 조판할 수도 있습니다.

그림 2.3: 예시 문서 2.2의 컴파일 결과.

제 3 장

코드의 구조와 문법

이번 장에서는 \LaTeX 의 문법과 소스 코드의 구조를 알아볼 것입니다.

- 제어 문자열(control sequence)과 환경(environment)이 무엇인지,
- 전처리부(preamble)는 무엇인지,
- 문서의 클래스(class)를 지정하고 패키지(package)를 불러오는 방법이 어떻게 되는지,
- \TeX 이 특수하게 처리하는 문자에는 무엇이 있는지,
- 수식과 주석은 어떻게 입력하는지

알고 있는 독자는 이 부분을 넘어가도 좋습니다.

3.1

제어 문자열과 환경

\LaTeX 에서 대부분의 명령은 제어 문자열(control sequence)을 사용해 내립니다. 역슬래시로 시작해 로마자가 이어지는 문자열이나, 역슬래시 뒤에 숫자 또는 기호 한 개가 있는 문자열을 제어 문자열이라고 하며, 컴파일 과정에서 \LaTeX 이 제어 문자열을 마주치면 그 제어 문자열의 정의에 따라 특정 동작을 수행합니다. 예를 들어, 다음은 모두 제어 문자열입니다.

```
 $\backslash\text{\LaTeX}$        $\backslash\text{\documentclass}$      $\backslash\text{\usepackage}$      $\backslash\text{\P}$        $\backslash\text{\[}$        $\backslash\text{\@}$ 
```

역슬래시 뒤에 로마자가 이어지는 제어 문자열은 뒤에 오는 공백을 무시합니다. 의도적으로 제어 문자열 뒤에 공백을 삽입하려면 $\backslash_$ 를 입력하면 됩니다. 예를 들어, ' \TeX '을 출력하는 제어 문자열 $\backslash\text{\TeX}$ 은 다음과 같이 사용할 수 있습니다.

```
 $\backslash\text{\TeX}$  nicians use  $\backslash\text{\TeX}$  well.
```

```
 $\text{\TeX}$ nicians use  $\text{\TeX}$  well.
```

많은 제어 문자열은 인자(argument)를 입력받습니다. 인자를 지정할 때에는 중괄호를 사용하며, 중괄호 안에 인자로 지정할 내용을 입력하면 됩니다. 예를 들어, 분수를 입력하는 제어 문자열 $\backslash\text{\frac}$ 은 분자와 분모를 각각 인자로 입력받습니다.

$$\backslash[\ \pi\ \approx\ \frac{355}{113}\ \backslash]$$

$$\pi \approx \frac{355}{113}$$

인자로 지정할 내용이 문자 하나 또는 제어 문자열 하나일 때에는 중괄호를 생략해도 됩니다.

$$\backslash[\ \frac{ab}{bc} = \frac{a\{bc\}}{ad}\{b\}\ \backslash]$$

$$\frac{a}{b} = \frac{a}{bc} = \frac{ad}{b}$$

인자 중에는 필수로 입력할 필요가 없는 ‘선택 인자’도 있습니다. 이러한 인자는 대괄호를 사용해 지정합니다. 선택 인자를 특별히 지정하지 않으려면 대괄호를 포함해 해당 인자를 생략하면 됩니다. 예를 들어, 근호를 입력하는 제어 문자열 `\sqrt`는 선택 인자인 근지수와 필수 인자인 피제곱근수를 입력받습니다.

$$\backslash[\ 2 = \sqrt{4} = \sqrt[3]{8}\ \backslash]$$

$$2 = \sqrt{4} = \sqrt[3]{8}$$

선택 인자는 필수 인자와 달리 대괄호를 생략해 입력할 수 없습니다.

많은 제어 문자열의 인자에서는 문단을 나누면(리턴 키를 두 번 연속으로 눌러 빈 줄을 만들거나 `\par`를 입력하면) 안 됩니다. 컴파일 과정에서 오류 메시지로

Runaway argument?

! Paragraph ended before $\langle control\ sequence \rangle$ was complete.

등이 나타나면 인자를 지정하는 중괄호 안에서 문단을 나누지는 않았는지 확인해 보십시오.

환경

복잡한 명령을 입력하는 경우에는 환경(environment)을 사용하게 됩니다. 환경은 시작하는 코드와 끝내는 코드로 이루어지며, 두 코드 사이에 해당 환경이 처리할 코드를 입력합니다. 이름이 $\langle env \rangle$ 인 환경은 `\begin{ $\langle env \rangle$ }`로 시작해 `\end{ $\langle env \rangle$ }`로 끝냅니다. 예를 들어, 목록을 작성하는 `itemize` 환경은 아래와 같이 사용합니다.

```
1 \begin{itemize}
2   \item 첫 번째 항목
3   \item 두 번째 항목
4 \end{itemize}
```

- 첫 번째 항목
- 두 번째 항목

환경도 경우에 따라 인자를 입력받습니다. 이때에는 각 인자를 `\begin{ $\langle env \rangle$ }` 뒤에 중괄호나 대괄호를 사용해 지정합니다. 예를 들어, 표를 작성하는 `tabular` 환경은 표의 열 개수와 각 열의 정렬 방법을 지정하는 인자를 입력받습니다.

```

1 \begin{tabular}{c|cc}
2   \hline
3   & Midterm Score & Final Score \\ \hline
4   Analysis I & 73 & 84 \\
5   Analysis II & 76 & 91 \\ \hline
6 \end{tabular}

```

	Midterm Score	Final Score
Analysis I	73	84
Analysis II	76	91

`\newcommand`와 `\newenvironment` 제어 문자열을 사용하면 제어 문자열과 환경을 직접 정의할 수도 있습니다. 이에 대해서는 제21장에서 다룹니다.

명령의 유효 범위

제어 문자열 중 일부는 ‘선언(declaration)’을 하기도 합니다. 제어 문자열이 입력된 곳부터 글꼴을 바꾸도록 명령하는 `\bfseries`나 `\itshape`, 문단을 가운데에 맞춰 정렬시키는 `\centering`, 새로운 명령어를 정의하는 `\newcommand` 등이 선언하는 제어 문자열의 대표적인 예시입니다. 이들은 특별히 제한 범위를 지정하지 않으면 문서의 끝부분까지 영향을 미치므로, 해당 명령의 효과를 지정할 부분을 설정해 주어야 합니다.

제어 문자열의 유효 범위는 그 제어 문자열이 입력된 곳부터 제어 문자열이 속한 영역(group)이 끝나는 곳까지입니다. 일반적으로 영역은 중괄호를 사용해 만들 수 있습니다. 예를 들어, 글꼴을 볼드체로 조판하도록 선언하는 `\bfseries`는 중괄호의 유무에 따라 다음의 결과를 생성합니다.

```

1 글꼴을 {\bfseries 볼드체로} 조판한다. \\
2 글꼴을 \bfseries 볼드체로 조판한다.

```

글꼴을 **볼드체로** 조판한다.
글꼴을 **볼드체로** 조판한다.

한편, 제어 문자열이나 환경의 인자를 입력한 부분도 하나의 영역으로 취급됩니다. `\textsf`는 인자로 지정된 내용을 산세리프체로 조판하도록 합니다. 이 인자 안에 `\bfseries`를 입력하면 글자를 볼드 산세리프체로 조판할 수 있습니다.

```

글꼴을 \textsf{\bfseries 볼드 산세리프체로} 조판하
기

```

글꼴을 **볼드 산세리프체로** 조판하기

환경의 내부도 영역으로 취급됩니다. 환경 안에 입력된 글자를 크게 조판하도록 하는 `large` 환경을 시작할 때 `\bfseries`를 입력하면 환경 안의 글자가 큰 볼드체로 조판됩니다. 환경이 끝나면 다시 본래 크기로 돌아옵니다.

```

글꼴을 \begin{Large}\bfseries 큰 볼드체
로\end{Large} 조판하기

```

글꼴을 **큰 볼드체로** 조판하기

특히 영역을 지정하지 않고 선언하는 문자열을 사용하면, 해당 제어 문자열의 기능을 상쇄하는

다른 제어 문자열이 쓰이지 않는 한 문서의 끝부분까지 영향을 미칩니다. 필요에 따라 선언형 제어 문자열과 영역을 적절하게 사용하십시오.

주석

문서를 작성하다 보면 주석(comment)을 사용할 일이 생깁니다. 주석을 사용하면 특정 코드의 목적이 무엇인지 설명을 적어 둘 수도 있고, 컴파일 오류의 원인을 분석하기 위해 코드 일부분을 임시로 작동하지 않도록 할 수도 있습니다. \LaTeX 에서 주석은 퍼센트 기호를 사용해 입력합니다. 퍼센트 기호부터 그 줄 끝까지의 내용이 주석으로 처리되어 컴파일 과정에서 무시됩니다. 필요에 따라 주석을 적절히 사용하면 코드를 이해하는 데 도움이 됩니다.

¹ 여기에 입력한 문자열은 % 무시된다.

² 무시되지 않는다.

여기에 입력한 문자열은 무시되지 않는다.

3.2

문서의 클래스와 전처리부

모든 \LaTeX 문서의 소스 코드는 다음 세 가지 코드를 이 순서로 반드시 포함합니다.

(1) `\documentclass` (2) `\begin{document}` (3) `\end{document}`

(1)은 문서의 클래스(class)를 지정합니다. (2)는 문서의 본문이 시작됨을 알립니다. (3)은 문서의 모든 내용이 입력되었음을 알립니다. (1)과 (2) 사이를 전처리부(preamble)라고 하며, 종이 크기나 여백, 글꼴의 사전 설정 등 생성될 PDF 파일의 페이지에 입력되는 내용에 직접적으로 관련되지 않은 모든 설정을 이곳에서 지정합니다. 문자, 수식, 표와 그림, 목차 등을 포함한 출력 문서에 표시되는 모든 내용은 (2)와 (3) 사이에, 즉 document 환경 안에 입력합니다. (3) 뒤에 입력된 모든 내용은 컴파일 과정에서 무시됩니다. 이를 정리해 그림으로 나타내면 그림 3.1과 같습니다.

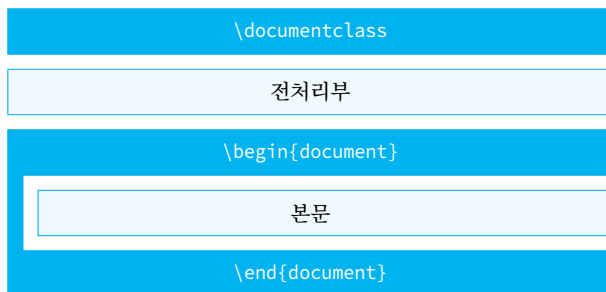


그림 3.1: \LaTeX 문서의 구조

문서의 클래스

문서를 작성할 때에는 단면 편집·양면 편집, 장절 표제의 형식 등 페이지 레이아웃에 따라 문서의 종류가 구분됩니다. 이를 \LaTeX 은 ‘클래스’라고 구분하며, 문서를 작성하기 위한 세 가지 클래스

article, report, book을 제공합니다. 문서의 클래스를 `\documentclass` 제어 문자열의 인자로 지정하면 됩니다. 또, 본문의 글자 크기, 용지 크기, 양면 편집 여부, 수식 표시 방식 등의 옵션을 지정할 수도 있습니다. 이는 `\documentclass` 제어 문자열의 선택 인자로 지정합니다. 예를 들어, 다음 예시를 생각할 수 있습니다.

```
1 \documentclass[a4paper,12pt]{article}
2 \documentclass[b5paper,leqno]{book}
```

지정할 수 있는 클래스와 다양한 추가 옵션은 제4장에서 알아봅니다.

전처리부

문서의 클래스를 지정했다면 전처리부에서 다양한 사전 설정을 지정해 주어야 합니다. 예를 들어, 패키지를 불러와 다양한 명령어를 정의하거나 추가 기능을 사용할 수 있도록 하고, 페이지의 레이아웃을 바꾸고, 본문에서 사용할 글꼴을 지정합니다. 새로운 매크로를 정의하기도 합니다.

전처리부에서 하는 주된 작업은 `\usepackage` 제어 문자열을 사용해 패키지를 불러 오고 패키지의 옵션을 설정하는 것입니다. 패키지(package)는 문서에서 사용할 수 있는 다양한 명령을 제공하고 기능을 확장해 줍니다.

```
\usepackage[<options>]{<package name>}
```

`<options>`: 패키지를 불러올 때 지정할 옵션

`<package name>`: 불러올 패키지의 이름

옵션을 지정하는 방법은 불러온 패키지에 따라 천차만별입니다. 각 패키지에는 사용 방법을 설명하는 패키지 문서가 있으며, 이는 CTAN에서 검색해 확인할 수 있습니다. 예를 들어, 한국어 입출력 패키지 kotex를 hangul 옵션을 지정하여 불러오려면

```
\usepackage[hangul]{kotex}
```

를 입력하며, 정의·정리·증명을 작성하는 기능을 제공하는 패키지 amsthm을 불러온 뒤 theorem이라는 환경을 정의하려면

```
1 \usepackage{amsthm}
2 \newtheorem{theorem}{Theorem}
```

을 입력하면 됩니다.

3.3

문서 작성

전처리부에서 설정을 마쳤다면 document 환경에 문서를 작성합니다. 문서의 내용을 이곳에 자유롭게 입력하면 되지만, 주의해야 할 점이 몇 가지 있습니다. 첫째로, \LaTeX 에서 영어가 아닌 언어의 문서를 작성하려면 추가 설정을 해 주어야 합니다. 일반적인 한국어 문서를 작성하려면 전처리부에

`\usepackage[hangul]{kotex}`을 입력해야 합니다. 여러 언어로 문서 작성하는 방법은 제26장에서 자세히 다룹니다.

둘째로, 역슬래시 `\`, 중괄호 `{ }`, 달러 기호 `$`, 앰퍼샌드 `&`, 해시 `#`, 물결표 `~`, 캐럿 기호 `^`, 밑줄 문자 `_`, 퍼센트 기호 `%`는 특별한 목적으로 예약되어 있습니다. 잘못 입력하면 컴파일 과정에서 오류가 발생하거나 의도하지 않은 결과를 얻을 수 있습니다. 이 기호를 입력해야 한다면 표 3.1의 제어 문자열을 대신 사용해 입력합니다.

표 3.1: L^AT_EX 특수 문자와 대체 입력 방법

역슬래시 <code>\</code>	<code>\textbackslash</code>	해시 <code>#</code>	<code>\#</code>
여는 중괄호 <code>{</code>	<code>\{</code>	닫는 중괄호 <code>}</code>	<code>\}</code>
퍼센트 기호 <code>%</code>	<code>\%</code>	달러 기호 <code>\$</code>	<code>\\$</code>
앰퍼샌드 <code>&</code>	<code>\&</code>	밑줄 문자 <code>_</code>	<code>_</code>
물결표 <code>~</code>	<code>\~{}</code> ¹ 또는 <code>\textasciitilde</code>	캐럿 기호 <code>^</code>	<code>\^{}</code> 또는 <code>\textasciicircum</code> ²

¹ 설정에 따라 구분 기호(diacritic) 틸데(tilde, `~`)가 출력될 수 있습니다.

² 설정에 따라 구분 기호 서킴플렉스(circumflex, `^`)가 출력될 수 있습니다.

세 번째로 주의할 사항은 공백 문자와 개행 문자에 관한 내용입니다. 이 책에서는 스페이스 키를 눌렀을 때 입력되는 문자를 ‘공백 문자’, 리턴(엔터) 키를 눌렀을 때 입력되는 문자를 ‘개행 문자’라고 부르겠습니다. 이들을 코드 상에서 시각적으로 나타내야 할 때에는 각각 ‘`\`’, ‘`<`’로 나타내겠습니다.

L^AT_EX은 공백 문자나 한 개의 개행 문자를 단어 사이의 띄어쓰기로 조판합니다. 공백 문자는 여러 개가 입력되더라도 한 개가 입력된 것처럼 처리되므로, 공백 문자를 실수로 여러 개 입력하여 단어 사이의 간격이 불균일하게 조판되는 일을 피할 수 있습니다. 또, 코드에서 각 줄의 첫머리에 입력된 공백 문자는 모두 무시됩니다. 따라서, 코드를 보기 좋도록 적절히 들여쓰기 처리를 해 주면 좋습니다.

개행 문자를 두 번 이상 입력해 코드 상에서 빈 줄이 생기면 문단이 나뉩니다. 공백 문자와 마찬가지로, 개행 문자를 세 번 이상 입력하더라도 문단 사이에 추가 공백이 만들어지지 않습니다. 단어 사이의 띄어쓰기는 `\space`로, 문단 나눔은 `\par`를 입력해 대신할 수도 있습니다.

물결표 ‘`~`’도 띄어쓰기를 조판하지만, 이때에는 앞뒤의 단어 사이에서 줄이 바뀌지 않도록 합니다. 역슬래시 두 개 `\\`를 입력하면 줄을 강제로 바꿀 수 있고, 여기에 별표를 붙여 `*`를 입력하면 줄을 바꾸되 여기에서 페이지는 나뉘지 않도록 합니다. 단어 사이나 문단 사이에 넓은 공간을 만들고 싶으면 이 물결표나 역슬래시 두 개를 사용할 수 있지만, 이 방법보다는 25.1절에서 다루는 `\hspace`와 `\vspace` 제어 문자열을 사용하는 방법이 바람직합니다.

수식 조판

수식은 `\`(와 `\`) 사이에 수식을 작성하는 코드를 입력하여 조판합니다. 중요하거나 긴 수식은 `\[`와 `\]`의 사이에 코드를 입력하여 별도의 글줄에 표시할 수 있습니다. 전자의 방식으로 입력한 수식은 글줄에 섞여 나타나므로 ‘행중수식(inline math)’이라고 부르며, 후자의 방식으로 입력한 수식은 ‘별행수식(displayed math)’이라고 부릅니다. 예를 들어 다음과 같이 사용할 수 있습니다.

- 1 자연로그의 밑 $\backslash(e\backslash)$, 허수 단위 $\backslash(i\backslash)$, 원주율 $\backslash(\pi\backslash)$, 곱셈의 항등원 $\backslash(1\backslash)$ 과 덧셈의 항등원 $\backslash(0\backslash)$ 사이의 관계는 오일러 등식
- 2 $\backslash[e^{i\pi} + 1 = 0 \backslash]$
- 3 이 설명한다.

자연로그의 밑 e , 허수 단위 i , 원주율 π , 곱셈의 항등원 1과 덧셈의 항등원 0 사이의 관계는 오일러 등식

$$e^{i\pi} + 1 = 0$$

이 설명한다.

위 예시에서 행중수식은 처음 두 줄에 있는 ‘ $e, i, \pi, 1, 0$ ’이 있으며, 별행수식은 셋째 줄에 있는 ‘ $e^{i\pi} + 1 = 0$ ’입니다. 수식과 관련된 더 다양한 내용은 제10장에서 설명하겠습니다.

3.4

원시적 템플릿

이상의 내용을 종합하여, 문서를 작성할 수 있는 원시적인 템플릿을 아래에 제공합니다. 모든 \LaTeX 문서는 이 템플릿을 바탕으로 하여 작성할 수 있습니다.

주석에서 설명하듯, `mathtools` 패키지와 `amssymb` 패키지는 수식 작성 기능을 보강합니다. 수식을 작성할 일이 없다면 제3행의 두 패키지를 불러오지 않아도 됩니다. 마찬가지로, 한국어 문서를 작성하지 않는다면 제7행의 `kotex` 패키지를 불러오지 않아도 됩니다. 이후, 제3행과 제7행 사이에서 패키지를 불러오고, 각종 설정을 해 주면 됩니다. 설정이 끝났다면, 제9~11행에 문서의 제목, 저자, 작성 날짜를 지정한 후 `document` 환경의 `\maketitle` 제어 문자열 아래에 문서의 내용을 작성하면 됩니다. 뒤쪽의 내용을 익힐수록 더 다채로운 문서를 작성할 수 있을 것입니다.

예시 문서 3.1 Primitive Template

```

1 \documentclass[a4paper]{article}
  % 문서의 클래스를 지정합니다
2
3 \usepackage{mathtools,amssymb}
  % 수식 작성 기능을 보강합니다
4
5 % 패키지를 불러오고 문서의 설정을 조정합니다
6
7 \usepackage[hangul]{kotex}
  % 한국어 문서를 작성하도록 설정합니다
8
9 \title{Title}
  % 문서의 제목을 중괄호 안에 입력합니다
10 \author{Author}
  % 저자의 이름을 중괄호 안에 입력합니다
11 \date{\today}
  % 문서가 작성된 날짜를 중괄호 안에 입력합니다
12
13 \begin{document}
14 \maketitle % 문서의 표지를 생성합니다
15
16 % 문서의 내용을 입력합니다
17
18 \end{document}
```


제 II 편

본문 작성하기

제 II 편은 문서의 내용을 입력하고 기본적인 서식을 지정하는 기본적인 내용을 다룹니다. 이후 문서의 레이아웃과 관련된 내용은 뒤쪽 제 VII 편에서 다룹니다. 대부분의 문서를 작성할 때에는 여기에서 언급하는 내용을 사용할 것입니다. 그만큼, 이 부분의 문법은 잘 익혀 두기 바랍니다.

제4장은 문서의 클래스에 대해 설명합니다. 문서의 클래스는 아주 기본적인 형식을 맞추어 주는 템플릿이라고 할 수 있습니다.

제5장은 여러 글자와 문장 부호, 기호를 입력하는 방법을 알아봅니다.

제6장은 기본적인 문서 구조, 즉 표지와 장절 표제를 생성하는 방법을 알아봅니다. 또, 필요한 경우 주석을 다는 방법도 알아봅니다.

제7장은 목록을 비롯해 다양한 형식의 문단을 입력하기 위한 환경을 다룹니다. 문단의 좌우 정렬을 설정하는 방법도 알아봅니다.

제8장은 글꼴을 지정하는 방법을 설명합니다. \LaTeX 의 네 가지 글꼴 속성을 설명하고, 속성을 변경하는 방법을 알아봅니다. 추가로, \LaTeX 을 사용해 프로그램의 코드를 작성하는 방법도 알아봅니다.

제9장은 정의, 정리, 증명 등 수학적 문단을 작성하는 방법을 설명합니다. `amsthm` 패키지를 불러온 뒤 문단을 작성하는 환경을 만들고, 문단에 기본적인 서식을 지정하는 방법을 알아봅니다.

제 4 장

문서의 클래스

문서의 클래스(class)는 장절 표제의 스타일과 페이지 레이아웃 등을 설정하는 아주 최소한의 템플릿이라고 할 수 있습니다. \LaTeX 에서는 문서를 작성하기 위한 기본 클래스로 article, report, book, letter를 사용할 수 있습니다. 편지를 작성하기 위한 클래스인 letter는 이 책에서 다루지 않겠습니다.

4.1

표준 클래스

앞서 3장에서 클래스는 `\documentclass` 제어 문자열로 지정함을 설명하였습니다. 더 자세히 설명하면, 이 제어 문자열은 다음과 같이 사용합니다.

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

<options>: 문서의 클래스에 지정할 옵션

<class>: 문서의 클래스

<class> 부분에 사용하려는 클래스의 이름, 즉 article, report, book 등을 입력하면 됩니다. 클래스가 지정되면 이에 따라 몇 가지 명령어가 정의되며, 여러 가지 레이아웃 옵션이 설정됩니다. *<options>*에 지정할 수 있는 옵션은 다음 절에서 설명하고, 이번 절에서는 각 클래스의 특징을 설명하겠습니다.

각 클래스의 특징을 설명할 때에는 뒤쪽 장에서 설명할 내용을 몇 가지 언급하기도 합니다. 문서의 표지를 생성하는 `\maketitle` 제어 문자열이나 초록을 작성하기 위한 abstract 환경에 대해서는 제5장을, \LaTeX 의 장절 위계 시스템과 장절 표제를 조판하기 위한 `\part`, `\chapter`, `\section` 등의 제어 문자열은 제6장을 참조하시기 바랍니다.

article 클래스

article 클래스는 작은 규모의 문서를 작성할 때 사용합니다. report나 book 클래스에서는 사용할 수 있는 `\chapter`를 사용하지 못하고 `\section`과 그 하위 수준의 제어 문자열, 그리고 `\part`만을

사용할 수 있으며, `\maketitle` 제어 문자열로 생성하는, 문서의 제목과 저자, 작성 날짜가 적힌 표지도 첫 페이지의 위쪽 1/3 정도만을 차지합니다. 또, 편이나 절 표제를 삽입해도 페이지가 바뀌지 않습니다. `abstract` 환경을 사용해 문서의 표지 아래에 초록을 작성할 수 있습니다.

그림 4.1에서는 `article` 클래스를 사용해 조판한 문서의 예시 모습을 확인할 수 있습니다.

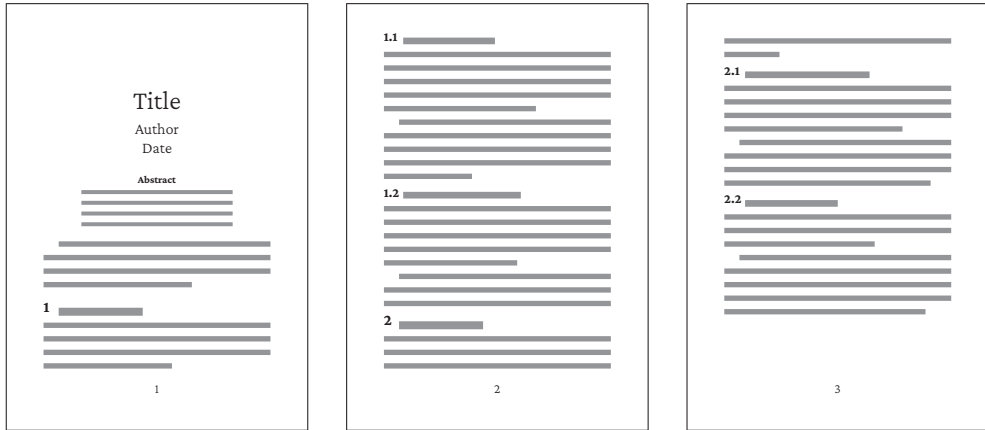


그림 4.1: `article` 클래스의 문서 조판 예시

특히 큰 규모의 문서를 작성하는 것이 아니라면 `article` 클래스를 사용해 문서를 작성하는 것이 일반적입니다.

report 클래스

`report` 클래스는 `article`에 비해 더 큰 규모의 문서를 작성할 때 사용합니다. `\maketitle` 제어 문자열로 생성되는 표지가 별도의 페이지에 조판되고, `article` 클래스에서는 사용할 수 없었던 `\chapter`를 사용할 수 있습니다. `\part`나 `\chapter` 수준의 표제를 사용하면 새로운 페이지가 만들어지고 이 표제로 페이지가 시작됩니다. 또, `article` 클래스와 마찬가지로 `abstract` 환경을 사용해 초록을 작성할 수 있습니다. 이때 `article` 클래스의 `abstract` 환경과는 달리 별도의 페이지가 만들어지고, 이 페이지에 초록이 조판됩니다. 표지와 초록이 작성된 페이지는 페이지 번호를 계산할 때 제외되고, 이후의 내용이 작성되는 페이지가 1페이지가 됩니다.

그림 4.2에서 `report` 클래스를 사용해 조판한 문서의 예시 모습을 확인할 수 있습니다.

book 클래스

`book` 클래스는 단행본을 작성할 때 사용합니다. `report` 클래스와 마찬가지로 `\maketitle` 제어 문자열로 만들어지는 표지가 별도의 페이지에 조판되고, `\chapter`를 표제로 사용할 수 있습니다. 또, 양면 편집이 적용되므로 책으로 조판했을 때 왼쪽 페이지와 오른쪽 페이지의 레이아웃이 다르게 설정됩니다. `\part`와 `\chapter` 수준의 표제를 사용하면 새로운 홀수 페이지가 생성되고, 각 페이지의 위쪽에는 현재 장 또는 절 표제가 머리말로 설정됩니다. 한편, `article` 클래스나 `report` 클래스와는 달리, 초록을 작성하기 위한 `abstract` 환경은 제공되지 않습니다. 마지막으로, 많은 외국 책을 보면



그림 4.2: report 클래스의 문서 조판 예시

서문과 목차 등이 적힌 부분은 페이지 번호를 로마 숫자로 적고, 본문이 시작되는 첫 페이지에서 아라비아 숫자로 페이지 번호를 1로 적기도 합니다. book 클래스를 사용하면 이런 기능도 사용할 수 있습니다.

그림 4.3에서는 book 클래스를 사용해 조판한 문서의 예시 모습을 확인할 수 있습니다.

article 클래스의 문서를 모아 각 문서를 하나의 장으로 하는 report 또는 book 클래스의 문서를 만들 수 있습니다.

4.2

클래스 옵션

앞서 알아보았듯, 세 가지 \LaTeX 의 표준 클래스인 article, report, book은 레이아웃을 사전 설정하는 역할을 합니다. `\documentclass`의 *(options)* 인자를 사용하면 레이아웃을 클래스의 기본값과 다르게 만들거나 더 다양한 설정을 사용할 수 있습니다. 이번 절에서는 세 표준 클래스에서 사용할 수 있는 옵션을 자세히 알아보겠습니다. 지정할 수 있는 옵션은 표 4.1에서 모두 확인할 수 있습니다.

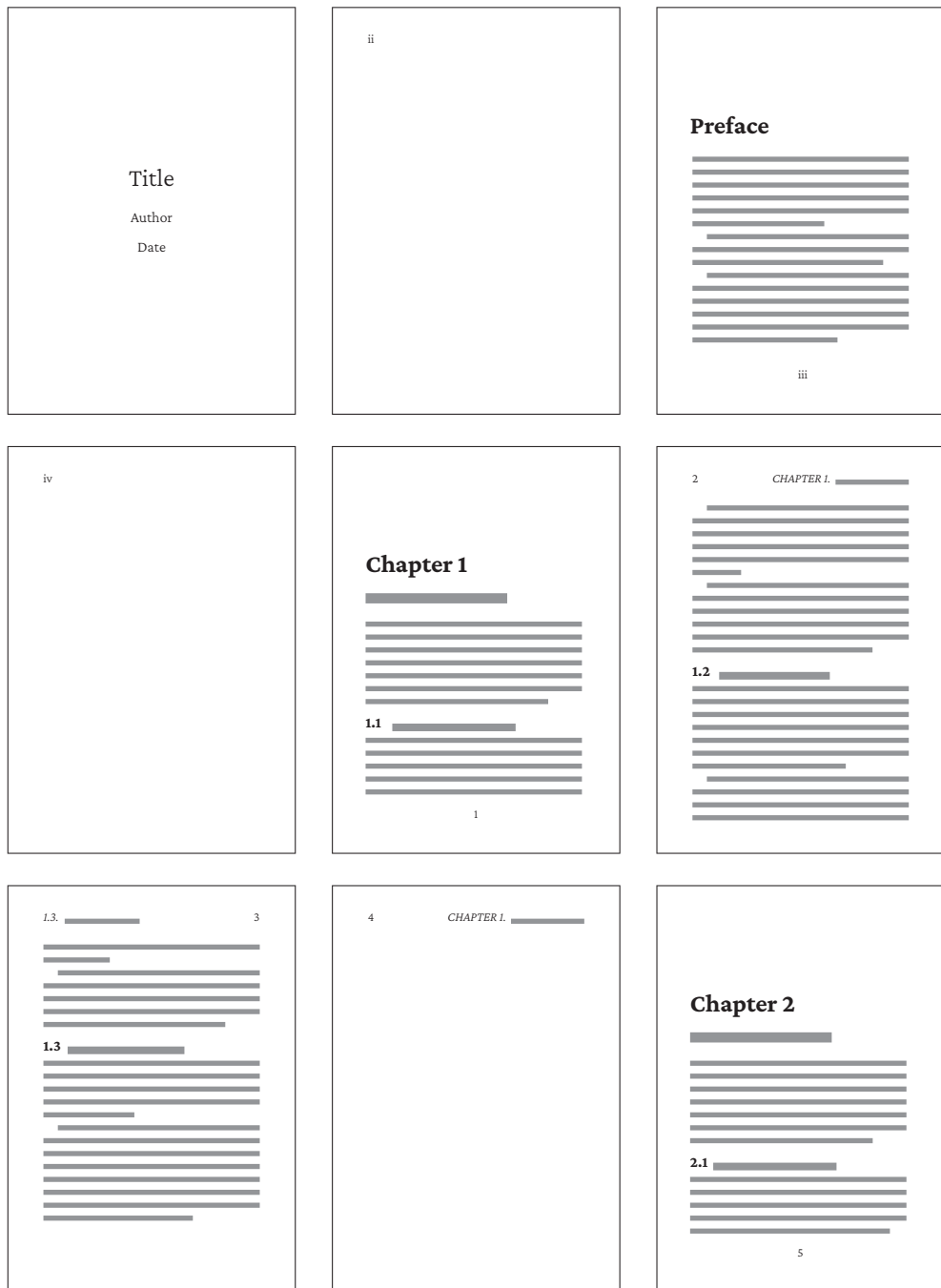


그림 4.3: book 클래스의 문서 조판 예시

이 표에 없는 옵션이 입력되면, 이는 문서 클래스의 설정을 바꾸는 데 쓰이지 않고 불러오는 각 패키지의 옵션으로써 사용됩니다.

표 4.1: 기본 문서 클래스에서 지정할 수 있는 옵션

옵션 종류	사용 가능한 옵션
페이지 크기	letterpaper(기본값), legalpaper, executivepaper, a4paper, a5paper, b5paper
페이지 방향	landscape
기본 글꼴 크기	10pt(기본값), 11pt, 12pt
양면 편집 여부	oneside(article, report 기본값), twoside(book 기본값)
최종본 여부	draft, final(기본값)
표지 페이지 분리 여부	notitlepage(article 기본값) titlepage(report, book 기본값)
편·장 표제 시작 위치	openany(report 기본값), openright(book 기본값)
2단 편집 여부	onecolumn(기본값), twocolumn
수식 번호의 위치	leqno
별행수식의 정렬 위치	fleqn
참고 문헌 목록 형식	openbib

여러 옵션을 동시에 지정할 수도 있으며, 이때 각 옵션은 쉼표를 사용하여 구분합니다. 예를 들어, article 클래스를 사용하는 문서의 페이지 크기를 A4로 바꾸고 기본 글꼴 크기를 12포인트로 하려면

```
\documentclass[a4paper,12pt]{article}
```

을 소스 파일의 맨 처음에 입력하면 됩니다.

페이지 크기와 방향

문서의 전체 페이지 크기와 방향을 결정합니다. 페이지 크기로 사용할 수 있는 옵션에는 미국 용지 규격인 letterpaper(8.5 × 11 in), legalpaper(8.5 × 14 in), executivepaper(7 × 10.5 in)와 국제 표준 용지 규격인 a4paper(210 × 297 mm), a5paper(148.5 × 210 mm), b5paper(176 × 250 mm)¹가 있으며, 기본값은 letterpaper입니다.

특별한 옵션을 지정하지 않으면 페이지는 세로로 긴 방향으로 조판됩니다. 이때 landscape 옵션을 지정하면 페이지의 가로와 세로 길이가 바뀌어 페이지가 가로로 긴 방향으로 설정됩니다.

사용할 수 있는 옵션 이외에 다른 크기의 페이지를 사용하고자 한다면 페이지 레이아웃과 관련된 인자의 값을 변경하거나 geometry 패키지를 사용할 수도 있습니다.

기본 글꼴 크기

문서의 본문을 조판할 때 사용할 글꼴 크기를 지정합니다. 이 글꼴 크기에 따라 줄 간격, 표지, 장절 표제, 각주 등에서 쓰이는 글꼴의 크기가 결정됩니다. 사용할 수 있는 옵션은 10pt, 11pt, 12pt이며,

¹우리나라에서 사용하는 B5 용지의 크기는 일본의 규격을 따른 182 × 257 mm이로, 국제 표준 규격인 이 크기와 약간의 차이가 있습니다.

기본값은 10pt입니다. 더 다양한 크기의 글꼴을 사용하고 싶다면 `extarticle`, `extreport`, `extbook` 클래스를 사용할 수도 있습니다.

양면 편집 여부

문서를 단면으로 편집할지, 양면으로 편집할지 결정합니다. 단면으로 편집한 문서는 좌우 여백의 폭과 머리말의 내용이 페이지의 홀짝 여부에 따라 달라지지 않으며, 양면으로 편집한 문서는 (책을 제본해서 보았을 때) 안쪽 여백이 바깥쪽 여백보다 넓어지고, 머리말의 내용도 왼쪽 페이지와 오른쪽 페이지에서 각각 다르게 지정할 수 있습니다. 사용할 수 있는 옵션은 `oneside`(단면 편집)와 `twoside`(양면 편집)이며, `article`과 `report` 클래스의 기본값은 `oneside`, `book` 클래스의 기본값은 `twoside`입니다.

그림 4.4: 단면 편집 디자인과 양면 편집 디자인의 비교

최종본 여부

현재 편집 중인 문서가 초안인지 최종본인지 지정합니다. 초안으로 설정하는 경우에는 \TeX 이 문서를 보기 좋게 조판할 수 없어 문서 내용 중 일부가 여백을 침범하는 경우 이를 검은색 사각형을 사용해 표시해 줍니다. 사용할 수 있는 옵션은 `draft`(초안)와 `final`(최종본)이며, 기본값은 `final`입니다.

문서에서 사용하는 패키지에서 `draft` 옵션을 인식한다면 문서 전체의 레이아웃은 변하지 않도록 하면서 컴파일은 더 빠르게 진행되도록 할 수도 있습니다. 예를 들어, 외부 이미지를 문서에 포함하는 `graphicx` 패키지는 `draft` 옵션이 지정된 경우, 외부 이미지를 문서에 삽입하지 않고 이미지와 같은 크기의 상자를 대신 그려 컴파일 속도를 높입니다.

표지 페이지 분리 여부

앞 절의 설명에서 확인할 수 있듯, `article` 클래스에서는 문서의 표지를 별도의 페이지에 조판하지 않고 첫 페이지의 윗부분만을 차지하지만, `report`와 `book` 클래스에서는 별도의 페이지에 표지를 조판합니다. `notitlepage` 옵션을 사용하면 표지를 페이지의 일부로써 조판하고, `titlepage` 옵션을 사용하면 표지를 별도의 페이지로 조판합니다.

편·장 표제 시작 위치

`report`와 `book` 클래스에서는 편과 장 표제를 조판하기 전에 새로운 페이지를 시작합니다. `report` 클래스에서는 새로 시작하는 페이지 번호의 홀짝을 구분하지 않지만, `book` 클래스에서는 새로운 페이지의 번호가 반드시 홀수가 되도록, 필요한 경우에는 빈 페이지를 삽입합니다. `openany` 옵션을 사용하면 `report` 클래스처럼 편과 장 표제가 만드는 새로운 페이지의 번호의 홀짝을 구분하지 않고, `openright` 옵션을 사용하면 `book` 클래스처럼 새로 만들어지는 페이지의 번호가 홀수가 되도록 합니다.

2단 편집 여부

문서에 더 많은 내용을 작성하고 싶다면, 한 페이지에 글을 2단으로 작성할 수도 있습니다. 본문을 한 단으로 조판하는 `onecolumn` 옵션이 기본값이지만, `twocolumn` 옵션을 지정하면 본문을 2단으로 조판합니다. 한편, 클래스 옵션에 관계없이 `\twocolumn` 제어 문자열을 사용하면 새로운 페이지가 만들어지고 그 페이지부터는 본문이 2단으로 조판되며, 반대로 `\onecolumn` 제어 문자열을 사용하면 새로운 페이지가 만들어지고 그 페이지부터는 본문이 1단으로 조판됩니다.

수식 번호의 위치

`equation` 환경을 사용하면 수식을 조판할 수 있으며, 페이지의 오른쪽 끝에 수식의 번호가 조판됩니다. `leqno`를 클래스 옵션으로 지정하면 이 수식 번호를 페이지의 왼쪽 끝에 오도록 할 수 있습니다. 클래스 기본값에서는 이 옵션을 사용하지 않습니다.

그림 4.5에는 `leqno` 옵션을 지정하지 않았을 때와 지정했을 때 수식 번호가 조판된 모습을 비교하여 표시하였습니다.

<p>문서 클래스 옵션으로 <code>leqno</code>를 지정하지 않으면 수식 번호가 페이지 오른쪽에 조판된다.</p> $\int_a^b f(x) dx = F(b) - F(a) \quad (1)$	<p>문서 클래스 옵션으로 <code>leqno</code>를 지정하면 수식 번호가 페이지 왼쪽에 조판된다.</p> $(1) \quad \int_a^b f(x) dx = F(b) - F(a)$
--	---

그림 4.5: 수식 번호의 위치 비교

별행수식의 정렬 위치

참고 문헌 목록 형식

- article 클래스
`letterpaper, 10pt, onside, final, notitlepage, onecolumn`
- report 클래스
`letterpaper, 10pt, onside, final, titlepage, openany, onecolumn`
- book 클래스
`letterpaper, 10pt, twoside, final, titlepage, openright, onecolumn`

4.3

추가 클래스

기본 클래스 외에도 다른 클래스를 사용하여 \LaTeX 문서를 작성할 수 있습니다. \LaTeX 에서 사용 가능한 모든 문서 클래스의 목록은 CTAN의 Class 페이지(ctan.org/topic/class)에서 확인할 수 있습니다. 범용 문서를 작성할 수 있는 클래스, 단체에 글을 투고하기 위한 클래스, 특수한 목적으로 작성된 클래스 등이 이곳에 모두 모여 있습니다.

특기할 만한 클래스로는 memoir, beamer, minimal, standalone이 있습니다. memoir는 book 클래스를 기반으로 자주 사용되는 많은 패키지를 불러오고 문서의 레이아웃을 더 쉽게 수정할 수 있도록 한 클래스입니다. 이를 바탕으로 한국어 문서 작성 환경에 적합하도록 기능을 발전시킨 oblivoir 클래스도 사용할 수 있습니다. 한편, beamer 클래스는 \LaTeX 으로 프레젠테이션을 작성하도록 하는 클래스입니다. 일반적인 발표보다는 학술적 발표에 적합하도록 설계되어 있으며, 다양한 편의 기능을 제공합니다.

한편, minimal과 standalone 클래스는 독립된 문서를 만드는 데 사용되지 않습니다. minimal 클래스는 아주 최소한의 명령어만을 정의하고 있어, 컴파일 중 오류가 발생했을 때 디버깅을 하는 용도로 사용할 수 있습니다. standalone 클래스는 종이의 크기가 정의되지 않으며, 문서에 입력되는 내용에 맞춰집니다. tikz 등의 패키지로 그린 그림을 이 클래스로 컴파일한 뒤, 최종본 문서에는 그림을 PDF 파일로 삽입하여 컴파일 시간을 줄일 수 있습니다. 이들 클래스에 대한 자세한 사항은 각 클래스의 안내서를 읽으시기 바랍니다.

제 5 장

문자 입력하기

이번 장에서는 \LaTeX 문서에 다양한 문자를 입력하는 방법을 다룹니다. 다양한 언어를 표기하기 위한 문자들과 다양한 문장 부호, 기호들을 입력하는 방법을 알아볼 것입니다.

1970년대에 개발된 \TeX 을 기반으로 만들어진 \LaTeX 과 $\PDF\LaTeX$ 은 문자를 처리하고 글꼴을 사용하는 방식이 오늘날의 다른 워드프로세서에 비해 복잡합니다. 이 문제점을 해결하기 위해 만들어진 컴파일 엔진으로 $X\LaTeX$ 과 $\Lua\LaTeX$ 이 있으며, 이들은 앞의 두 엔진에 비해 더 유연하게 문자를 처리할 수 있습니다. 다만, 경우에 따라 \LaTeX 또는 $\PDF\LaTeX$ 을 사용해 문서를 컴파일해야 하는 경우도 있습니다. 이번 절에서는 엔진을 구분하여 내용을 설명할 것입니다. \LaTeX 과 $\PDF\LaTeX$ 을 ‘레거시 엔진’, $X\LaTeX$ 과 $\Lua\LaTeX$ 을 ‘유니코드 엔진’이라고 부르겠습니다.

5.1

로마자 입력하기

모든 \LaTeX 엔진은 A부터 Z, 또 a부터 z까지의 기본 로마자 52자를 아무 문제 없이 조판할 수 있습니다. 또, 구분 기호(diacritic)가 붙은 문자나 변형된 로마자도 원활하게 조판할 수 있습니다.

- 각종 구분 기호가 붙은 로마자: â, ç, é, î, ñ, ô, ü 등
- 특수한 형태의 로마자: Ææ, Ðð, Ðđ, Iĳ, Łł, Ŋŋ, Œœ, Øø, ß, þþ
- 점이 없는 i와 j: ı, ȷ

영어 키보드를 사용하는 경우에는 이러한 문자를 키보드로 입력하기 어려울 수 있습니다. 예를 들어, 린델뢰프(Lindelöf)나 로피탈(L'hôpital), 가우스(Gauß) 등의 이름에 있는 ‘ö’, ‘ô’, ‘ß’는 영어 키보드로 바로 입력할 수 없습니다. 이런 경우에는 표 5.1의 제어 문자열을 사용해 입력할 수 있습니다.

유니코드 엔진에서는 특별한 설정을 하지 않아도 위 문자를 입력할 수 있습니다. 한편, 레거시 엔진을 사용할 때에는 T1 옵션을 지정해 fontenc 패키지를 불러와야 합니다.

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

일부 언어는 표의 제어 문자열로 입력할 수 없는 형태의 로마자를 사용합니다. 아프리카의 여러

표 5.1: 다양한 로마자를 입력하는 명령어

Üü \{"U}\{"u}	Ôô \^{O}\^{o}	Çç \c{C}\c{c} ¹	Åå \r{A}\r{a}
Éé \'{E}\'{e}	Èè \`{E}\`{e}	Àà \d{A}\d{a}	Ââ \t{A}\t{a}
Ĥĥ \.{H}\.{h}	Ññ \~{N}\~{n}	Õõ \H{O}\H{o}	Ğğ \u{G}\u{g}
İı \={I}\={i}	Kk \b{K}\b{k}	Ėė \k{E}\k{e}	Žž \v{Z}\v{z} ¹
Ää \AA\aa	Đđ \DJ\dj	Łł \L\l	Œœ \OE\oe
Ææ \AE\ae	IJij \IJ\ij	Ŋŋ \NG\ng	Šš \SS\ss
Ðð \DH\dh	ıı \i\j	Øø \O\o	Þþ \TH\th

¹ 레거시 엔진에서 fontenc 패키지를 불러왔거나 현대 텍 엔진을 사용하는 경우, \c는 입력한 인자에 따라 cedilla와 comma below를 자동으로 골라 출력합니다. 마찬가지로, \v도 입력한 인자에 따라 caron과 comma right above를 자동으로 골라 출력합니다.

² 레거시 엔진에서 \k, \DH, \dh, \DJ, \dj, \IJ, \ij, \NG, \ng, \TH, \th는 fontenc 패키지를 T1 옵션을 지정하여 불러온 경우에만 작동합니다. 현대 T_εX 엔진을 사용하는 경우에는 패키지를 불러오지 않아도 됩니다.

언어나 베트남어에서 사용하는 문자가 이러한 예로, 이러한 문자를 표기하려면 fontenc 패키지에 T4, T5 등의 옵션을 지정해야 합니다. 자세한 내용은 L^AT_εX 글꼴 인코딩 안내서를 확인하십시오.

구 버전 L^AT_εX을 사용하는 경우

컴퓨터에 T_εX을 설치한 후 레거시 엔진을 사용해 문서를 작성하는 경우 엔진의 버전을 확인해야 합니다. 연도에 따라 프로그램의 기본값이 달라지기 때문입니다. 2017년 이전 버전의 T_εX을 사용하는 경우 소스 파일에 기본 52자 외의 로마자가 입력되면 문자가 정상적으로 조판되지 않을 수 있습니다. 이때에는

```
\usepackage[utf8]{inputenc}
```

를 전처리부에 입력해 inputenc 패키지를 불러와야 합니다.

5.2

문장부호 입력하기

L^AT_εX 문서를 만들 때에는 세 가지 문장부호를 주의해서 입력해야 합니다. 바로 따옴표, 대시와 줄임표입니다.

따옴표

따옴표를 입력할 때는 여는 것과 닫는 것을 구분해 입력해야 합니다. L^AT_εX 코드에서는 키보드에서 **1** 키의 왼쪽에 있는 **'** 키를 사용해 여는 따옴표를 입력하고, **;**의 오른쪽에 있는 **'**를 사용해 닫는 따옴표를 입력합니다. 같은 기호를 두 번 연속해서 입력하면 큰따옴표가 입력됩니다. L^AT_εX에서는 큰따옴표 키 **shift**+**'**를 눌러 큰따옴표를 입력하지 않음에 주의하세요.

- 1 ``큰따옴표'와 `작은따옴표' \\
- 2 {``}`큰따옴표 안에 작은따옴표{''}` \\
- 3 `{'}`작은따옴표 안에 큰따옴표{''}`

“큰따옴표”와 ‘작은따옴표’
 “큰따옴표 안에 작은따옴표”
 “작은따옴표 안에 큰따옴표”

위의 예시에서 볼 수 있듯, 작은따옴표와 큰따옴표가 연달아 나온다면 큰따옴표를 중괄호로 묶어 컴파일 엔진이 올바르게 해석하도록 해야 합니다.

언어에 따라 다른 모양의 따옴표를 사용하기도 합니다. 영어나 한국어와 달리, 유럽의 여러 언어는 다음과 같은 여러 스타일을 사용하기도 합니다.

,예시 1‘ „예시 2“ <예시 3> «예시 4» >예시 5< »예시 6«

이러한 모양의 따옴표는 앞과 같이 T1 옵션과 함께 fontenc 패키지를 불러온 뒤, 아래 표 5.2의 명령어를 입력하면 됩니다.

표 5.2: 추가 따옴표 입력 명령어

« \guillemetleft	< \guilsinglleft	„ \quotedblbase
» \guillemetright	> \guilsinglright	, \quotesinglbase

«는 <<로, »는 >>로, „는 ,,로도 입력할 수 있습니다. 다만, 설정에 따라 이 명령이 제대로 동작하지 않을 수 있습니다.

대시

한국어 문법에서와 달리, 영어 문서에서는 하이픈과 대시를 구분해 사용합니다. 단어를 이을 때 사용하는 하이픈은 키보드에서 숫자 0 오른쪽의 -를 한 번 눌러서 입력하며, 수의 범위를 나타낼 때 사용하는 엔 대시는 -를 두 번 눌러서, 문장의 부연 설명을 할 때 사용하는 엠 대시는 -를 세 번 눌러 입력합니다.

hy-phen; en--dash; em---dash

hy-phen; en-dash; em—dash

한국어에서 사용하는 줄표도 엠 대시와 같은 방법으로 입력하면 됩니다. 한편, 수치를 나타낼 때의 음의 부호를 넣을 때에는 수식 입력 모드를 사용하거나 \textminus를 입력하십시오. 하이픈이나 대시로 입력하는 것이 아닙니다.

- 1 옳은 예시: \(-1\)점, \textminus 2점. \\
- 2 잘못된 예시: -3점, --4점, ---5점.

옳은 예시: -1점, -2점.
 잘못된 예시: -3점, -4점, -5점.

줄임표

마지막으로, 줄임표를 입력하는 방법을 알아보겠습니다. 영어 문법에서는 마침표를 조금 넓은 간격으로 세 번 찍어 말줄임표를 나타냅니다. 이는 `\textellipsis` 또는 간단히 `\dots` 제어 문자열을 사용해 입력합니다. 유니코드 엔진에서는 줄임표가 다르게 출력되는 문제가 있어, 다음 명령을 전처리부에 입력해야 올바른 형태로 출력됩니다.

```
\UndeclareTextCommand\textellipsis{TU} % For XeLaTeX and LuaLaTeX
```

한국어 문법에서는 가운데점을 여섯 번 찍어 말줄임표를 나타냅니다. 현대 \LaTeX 엔진에서는 kotex 패키지를 불러온 후 `\hellipsis` 제어 문자열을 사용하면 위와 같은 여섯 점 말줄임표를 출력해 줍니다. 한편, 레거시 엔진에서는 이 제어 문자열을 지원하지 않으므로, 전처리부에 다음 명령을 입력해 직접 정의합니다.

```
1 \newcommand\hellipsis{%
2   \textperiodcentered\textperiodcentered\textperiodcentered
3   \textperiodcentered\textperiodcentered\textperiodcentered
4 } % For LaTeX and PDFLaTeX
```

적절한 설정을 마쳤다면, 아래처럼 코드를 입력해 말줄임표를 올바르게 조판할 수 있습니다.

- 1 The ellipsis in the English language looks like this\ldots. \\\
- 2 한국어의 말줄임표는 이렇게 생겼습니다\hellipsis.

The ellipsis in the English language looks like this....
한국어의 말줄임표는 이렇게 생겼습니다…….

뒤집힌 물음표와 느낌표

한국어나 영어로 문서를 작성하는 경우에는 중요한 사항이 아니지만, 스페인어 문서를 작성하는 경우에는 뒤집힌 물음표(`¿`)와 뒤집힌 느낌표(`¡`)를 입력해야 합니다. 뒤집힌 물음표는 `?`` 또는 `\textquestiondown`으로, 뒤집힌 느낌표 `¡`는 `!`` 또는 `\textexclamdown`으로 입력할 수 있습니다. 한편, 앞서 설명한 적절한 설정을 갖추었다면(레거시 엔진을 사용하며 fontenc 패키지를 T1 옵션을 적용하여 불러오거나 간단히 유니코드 엔진을 사용한다면) 이 문자를 소스 파일에 직접 입력하여도 문제 없이 문서를 조판할 수 있습니다.

문서의 설정에 따라 `?``와 `!``가 작동하지 않을 수도 있습니다. 이때에는 위의 제어 문자열을 사용하거나, 해당 문자를 키보드로 직접 입력하십시오.

5.3

기호 입력하기

위에서 설명한 문장부호 외에도, \LaTeX 에서는 다양한 기호를 입력하기 위한 제어 문자열을 제공합니다. 표 5.13의 제어 문자열을 제외하고는 이번 절에서 소개하는 제어 문자열은 텍스트 모드(수식

입력 모드 바깥)에서만 사용할 수 있습니다. 이번 절에서 설명하는 기호들 중 자주 쓰이는 것은 극히 일부이지만, \LaTeX 이 기본 제공하는 모든 기호를 나열하였습니다.

먼저, 문장 부호를 입력해 주는 제어 문자열을 표 5.3에 정리하였습니다. 앞 절에서 설명한 문장 부호들은 이 표에 있는 제어 문자열 외에도, 앞서 설명한 명령어를 사용해서도 입력할 수 있습니다. 그 다음의 표 5.4에는 화살표를 입력하는 제어 문자열을 나열하였습니다.

표 5.3: 문장부호

\backslash	\textbackslash	\textordfeminine	\textquoteright
\dots	\textellipsis^1	\textordmasculine	\textquotedbl
$—$	\textemdash	$\text{\textperiodcentered}$	\textquotesingle
$-$	\textendash	\textquestiondown	$\text{\textquotestraightbase}$
\textexclamdown	\textquotedblleft	$\text{\textquotestraightdblbase}$	
\textinterrobang	$\text{\textquotedblright}$	$\text{\textthrequartersemdash}$	
$\text{\textinterrobangdown}$	\textquoteleft	$\text{\texttwelvewardash}$	

¹ 표 5.13의 더 간단한 제어 문자열로 입력할 수도 있습니다.

표 5.4: 화살표

\leftarrow	\textleftarrow	\uparrow	\textuparrow	\downarrow	\textdownarrow	\rightarrow	\textrightarrow
--------------	-------------------------	------------	-----------------------	--------------	-------------------------	---------------	--------------------------

다음 페이지의 표 5.5는 통화 기호를 입력하는 제어 문자열을 설명하고 있습니다. 또, 표 5.6, 표 5.7, 표 5.8은 각각 지적재산권 관련 기호, 계보학 관련 기호, 괄호를 입력하는 방법을 설명합니다.

표 5.5: 통화 기호

\textbaht	\textdollar^1	\textguarani	\textwon
\textcent	$\text{\textdollaroldstyle}$	\textlira	\texttyen
\textcentoldstyle	\textdong	\textnaira	
$\text{\textcolonmonetary}$	\texteuro	\textpeso	
\textcurrency	\textflorin	\textsterling^1	

¹ 표 5.13의 더 간단한 제어 문자열로 입력할 수도 있습니다.

표 5.6: 지적재산권 관련 기호

\textcircledP	\textcopyright^1	\textservice mark
\textcircleft	\textregistered	\texttrademark

¹ 표 5.13의 더 간단한 제어 문자열로 입력할 수도 있습니다.

표 5.9는 다양한 형태의 숫자를 입력하는 명령어를, 표 5.10은 몇 가지 수학 기호를 텍스트 모드에서 입력하는 명령어를 나열하고 있습니다. 이들 기호는 특별한 경우가 아니라면 사용하지

표 5.7: 계보학 관련 기호

★ <code>\textborn</code>	† <code>\textdied</code>	⚭ <code>\textdivorced</code>	🍃 <code>\textleaf</code>	∞ <code>\textmarried</code>
--------------------------	--------------------------	------------------------------	--------------------------	-----------------------------

표 5.8: 괄호

{ <code>\textbraceleft</code> ¹	< <code>\texttriangle</code>	{ <code>\textlquill</code>]] <code>\textbrackdbl</code>
} <code>\textbraceright</code> ¹	[[<code>\textlbrackdbl</code>	> <code>\textriangle</code>	} <code>\textrquill</code>

¹ 표 5.13의 더 간단한 제어 문자열로 입력할 수도 있습니다.

않는 것이 좋습니다. 분수와 수학 기호는 \LaTeX 의 수식 입력 모드를 사용하고, 첨자 형태의 숫자는 `<string>`을 사용해 입력하는 것이 바람직합니다. 고전 형태의 숫자¹는 `\oldstylenums{<digits>}`을 사용해 입력할 수 있습니다.

표 5.9: 숫자

½ <code>\textonehalf</code>	¼ <code>\textonequarter</code>	¾ <code>\textthreequarters</code> / <code>\textfractionsolidus</code>
¹ <code>\textonesuperior</code>	² <code>\texttwosuperior</code>	³ <code>\textthreesuperior</code>
o <code>\textzerooldstyle</code>	4 <code>\textfouroldstyle</code>	8 <code>\texteightoldstyle</code>
1 <code>\textoneoldstyle</code>	5 <code>\textfiveoldstyle</code>	9 <code>\textnineoldstyle</code>
2 <code>\texttwooldstyle</code>	6 <code>\textsixoldstyle</code>	
3 <code>\textthreeoldstyle</code>	7 <code>\textsevenoldstyle</code>	

표 5.10: 텍스트 모드에서 사용하는 수학 기호

÷ <code>\textdiv</code>	< <code>\textless</code>	- <code>\textminus</code>	√ <code>\textsurd</code>
> <code>\textgreater</code>	¬ <code>\textlnot</code>	± <code>\textpm</code>	× <code>\texttimes</code>

다음 표 5.11은 일반 악센트 기호를 출력하는 명령어를 소개합니다. 이들은 표 5.1과 다르게 로마자에 악센트를 붙이는 데 사용할 수 없으며, 물결표를 출력하는 `\textasciitilde`를 제외하고는 일반적으로 사용되지 않습니다.

마지막으로, 다음 페이지의 표 5.12에는 \LaTeX 에서 입력할 수 있는 나머지 기호들을 나열하였습니다. 이 표의 명령어들 중 특기할 만한 것은 `\textmu`와 `\textohm`, `\textmho`로, 마이크로미터(μm)나 옴(Ω) 등 SI 단위를 입력할 때 사용합니다.

¹고전 형태의 숫자는 영어로 ‘old-style figure’라고 부르며, 흔히 사용하는 숫자와 다르게, 로마자의 소문자처럼 위아래로 들쭉날쭉한 모습입니다. 일반적으로 사용하는 숫자는 로마자의 대문자처럼 크기가 일정하며, 이를 영어로 ‘lining figure’라고 부릅니다. 두 꼴의 숫자를 아래에 제시합니다.

old-style figures: o 1 2 3 4 5 6 7 8 9

lining figures: 0 1 2 3 4 5 6 7 8 9

표 5.11: 일반 악센트 기호

“ \textacutedbl	˘ \textasciicaron	` \textasciigrave	” \textgravedbl
´ \textasciiacute	^ \textasciicircum	— \textasciimacron	
˘ \textasciibreve	¨ \textasciidieresis	~ \textasciitilde	

표 5.12: 기타 기호

\textbar	ℓ \textdiscount	‰ \textperthousand
\textbardbl	ℰ \textestimated	‡ \textpilcrow
○ \textbigcircle	ℴ \textmho	℞ \textrecipe
␣ \textblank	μ \textmu	※ \textreferencemark
! \textbrokenbar	♯ \textmusicalnote	§ \textsection ¹
• \textbullet	№ \textnumero	˜ \texttildelow
† \textdagger ¹	Ω \textohm	_ \textunderscore ¹
‡ \textdaggerdbl ¹	◦ \textopenbullet	␣ \textvisiblespace
= \textdblhyphen	¶ \textparagraph ¹	
° \textdegree	‰ \textpertenthousand	

¹ 표 5.13의 더 간단한 제어 문자열로 입력할 수도 있습니다.

앞서 소개한 표들에 제시된 명령어는 매우 길므로, 자주 사용하는 기호의 명령어는 반복해 입력하는 것이 귀찮을 것입니다. 몇 가지 자주 사용되는 기호는 표 5.13의 명령어를 사용해 비교적 간단하게 입력할 수 있습니다. 이 표의 명령어들은 수식 모드와 텍스트 모드에서 모두 사용할 수 있습니다.

표 5.13: 수식 안팎에서 모두 사용 가능한 기호 입력 명령어

{ \{	_ _	‡ \ddag	£ \pounds
} \}	© \copyright	... \dots ¹	§ \S
\$ \\$	† \dag	¶ \P	

¹ \ldots도 같은 결과를 출력합니다.

주의 사항

2019년 이전 버전의 레거시 엔진을 사용하거나 2016년 이전 버전의 유니코드 엔진을 사용하는 경우, 표 5.3 ~ 5.13의 일부 기호가 올바르게 표시되지 않을 수 있습니다. 이때에는 전처리부에

```
\usepackage{textcomp}
```

를 입력해 textcomp 패키지를 불러오십시오. 2020년 이후 버전의 레거시 엔진에서는 textcomp 패키지를 불러오는 것이 기본값이고, 2017년 이후 버전의 유니코드 엔진은 글꼴을 이전과 다른 방식으로 처리하므로 패키지를 불러올 필요가 없습니다.

제 6 장

문서의 형식 갖추기

어느 정도 형식을 갖춘 문서의 첫 페이지에는 제목과 저자, 날짜가 있고, 본문의 내용을 장과 절 등으로 나누어 문서의 내용을 쉽게 파악할 수 있도록 합니다. 필요한 경우 각주 등을 사용해 부연 설명을 하기도 합니다. 이번 장에서는 이때 사용할 수 있는 명령어를 설명합니다.

6.1

표지 만들기

모든 문서는 문서의 정보를 담은 표지로 시작합니다. \LaTeX 의 표준 클래스의 표지는 문서의 제목, 저자, 날짜로 구성되며 이들을 각각 `\title`, `\author`, `\date` 제어 문자열의 인자로 지정합니다. 전처리부에서 이들을 지정한 후 document 환경의 첫머리에 `\maketitle`을 입력하면 문서의 표지가 생성됩니다. article 클래스에서는 첫 페이지의 윗부분에 표지가 삽입된 뒤 본문의 글이 이어져 조판되며, report와 book 클래스에서는 표지가 첫 페이지를 차지하고 본문의 글은 다음 페이지에서부터 조판됩니다.

```

1 \title{\LaTeX: 문서 작성 도구}
2 \author{수리과학과 21학번 강우현}
3 \date{\today}
4 \begin{document}
5 \maketitle
6 ...
7 \end{document}

```

\LaTeX : 문서 작성 도구

수리과학과 21학번 강우현

2023-11-02

여기에서부터 본문이 시작됩니다.

`\maketitle`을 사용할 때, `\title`을 지정하지 않았으면 컴파일 과정에서 오류가 발생하고, `\author`를 지정하지 않으면 경고 메시지를 띄웁니다. `\date`는 필수로 지정하지 않아도 되며, 이때에는 컴파일 하는 시점의 날짜가 입력됩니다. 한편, 여러 명의 저자를 입력할 때에는 `\and`를 각 저자 이름 사이에 입력해 각 저자를 구분하며, 표지에서의 각주는 `\thanks` 제어 문자열로 입력하면

됩니다. 각주의 내용을 `\thanks`의 인자로 지정하십시오.

직접 표지 만들기

`titlepage` 환경을 사용하면 문서의 표지를 직접 만들 수 있습니다. 표지에 입력하고 싶은 내용을 환경 안에 입력하고 컴파일하면 그 내용만으로 구성된 한 페이지가 만들어집니다. 이 환경 바깥의 내용은 표지와 구분된 페이지에 조판됩니다. 25.1절의 `\vspace`, `\hspace` 제어 문자열과 제8장의 서식을 지정하는 명령어를 적절히 사용해 문서의 내용을 잘 설명하는 표지를 만들어 보십시오.

6.2

장과 절 구성하기

\LaTeX 에서는 문서의 내용을 분류하고, 편(part), 장(chapter), 절(section) 등 총 7개의 수준으로 구분하여 표제를 붙일 수 있습니다. 이 7개 수준의 장절 표제를 생성하기 위한 제어 문자열을 표 6.1에 정리하였습니다.

표 6.1: \LaTeX 의 장절 표제 명령어

(a) article 클래스			(b) report 및 book 클래스		
이름	수준	제어 문자열	이름	수준	제어 문자열
편	0	<code>\part</code>	편	-1	<code>\part</code>
—	—	—	장	0	<code>\chapter</code>
절	1	<code>\section</code>	절	1	<code>\section</code>
하위 절	2	<code>\subsection</code>	하위 절	2	<code>\subsection</code>
최하위 절	3	<code>\subsubsection</code>	최하위 절	3	<code>\subsubsection</code>
문단	4	<code>\paragraph</code>	문단	4	<code>\paragraph</code>
하위 문단	5	<code>\subparagraph</code>	하위 문단	5	<code>\subparagraph</code>

이 표에서 확인할 수 있듯, article 클래스에서는 `\chapter`를 사용할 수 없습니다. article 클래스로 작성한 문서 여러 개를 모아서, 각각의 문서를 하나의 장으로 하는 report 또는 book 클래스의 문서를 만들 수 있습니다.

각 제어 문자열은 다음과 같이 사용합니다.

```
\section[⟨short title⟩]{⟨title⟩}
\section*{⟨title⟩}
```

⟨short title⟩: 목차, 머리글, 바닥글 등에서 사용할 표제

⟨title⟩: 표제

`\part`, `\chapter`, …, `\subparagraph`와 각각의 별표 붙은 버전도 같은 방법으로 사용한다.

이 제어 문자열을 사용하면 각 수준에 맞는 서식이 자동으로 지정되며, 수준별로 표제의 번호도

자동으로 생성됩니다. 인자로 지정한 표제는 목차와 머리글에서도 사용됩니다. 이때 표제가 긴 경우 문서의 레이아웃이 흐트러질 수 있는데, 이 경우에는 선택 인자로 짧은 제목을 따로 지정해 줄 수도 있습니다.

별표 붙은 제어 문자열을 사용하면 표제의 번호가 출력되지 않고, 목차나 머리글에도 영향을 주지 않습니다. 특정 문자열을 장절 표제처럼 조판하고 싶지만 목차 등에 영향을 주고 싶지 않을 때 사용할 수 있습니다.

```
1 % article 클래스에서
2 \section{기본 개념}
3 이번 절에서는 군에 관한 기본적인 개념을 알아본다.
4
5 \subsection{군의 정의}
6 집합  $(G)$ 가 주어졌다고 가정하자.
```

1 기본 개념

이번 절에서는 군에 관한 기본적인 개념을 알아본다.

1.1 군의 정의

집합 G 가 주어졌다고 가정하자.

```
1 \subsection*{연습 문제}
2 \paragraph{연습 문제 1.} 군  $(G)$ ,  $(H)$ 와 군
   준동형  $(\varphi: G \rightarrow H)$ 가 주어졌다고
   가정하자. 다음을 증명하여라.
```

연습 문제

군 준동형의 성질. 군 G, H 와 군 준동형 $\varphi: G \rightarrow H$ 가 주어졌다고 가정하자. 다음을 증명하여라.

상위 수준 표제의 번호가 바뀌면 하위 수준 표제의 번호는 1부터 다시 시작됩니다. 즉, 절의 번호가 바뀌면 하위 절의 번호가 1로 초기화됩니다. 하지만 `\part`는 하위 수준의 표제 번호에 영향을 미치지 않습니다. 즉, 제 II 편에서 제 5장까지 작성하였다면 제 III 편의 첫 번째 장은 제 6장이 됩니다.

장절 표제의 서식은 `titlesec` 패키지를 사용해 바꿀 수 있습니다. 자세한 내용은 해당 패키지 안내서를 참조하십시오.

부록

본문과 관련된 부가적인 내용을 모아 부록(appendix)을 작성하는 경우도 있습니다. 부록이 시작되면 장절 표제의 번호가 처음부터 다시 시작되며, 이때 아라비아 숫자가 아닌 알파벳으로 바뀝니다.

부록을 작성할 때에는 부록이 시작되는 부분에서 `\appendix` 제어 문자열을 사용해 ‘여기에서 부록이 시작됨’을 선언하면 됩니다. 이 제어 문자열의 뒤에 등장하는 절(article 클래스) 또는 장(report 또는 book 클래스) 번호는 A부터 시작하도록 바뀝니다.

예를 들어, article 클래스 문서에서 `\appendix`를 다음과 같이 사용할 수 있습니다.

```
1 \documentclass[a4paper]{article}
2 ...
3 \title{Group Theory}
4 \author{...}
```

```

5 \begin{document}
6 \maketitle
7 \section{Groups and Some Examples} % Section 1
8 ...
9 \section{Normal Subgroups and Quotient Groups} % Section 2
10 ...
11 \section{Group Actions} % Section 3
12 ...
13 \section{Products of Groups} % Section 4
14 ...
15 \appendix
16 \section{Basic Number Theory} % Appendix A
17 ...
18 \end{document}

```

이 경우에 16행의 `\section{Basic Number Theory}`는

A Basic Number Theory

와 같은 형태로 조판됩니다.

전문, 본문, 부속물

어느 정도 규모가 되는 책은 내용을 크게 세 개의 부분으로 나눌 수 있습니다. ‘전문(front matter)’은 책에 대한 안내, 저자의 서문(preface), 목차 등이 등장하는 부분입니다. ‘본문(main matter)’은 책의 제목과 관련된 내용을 본격적으로 다루는 부분입니다. 부록도 본문에 포함됩니다. ‘부속물(back matter)’은 본문 뒤에 참고 문헌 목록, 색인 등이 쓰이는 부분입니다.

전문과 부속물 부분에 작성되는 내용의 장절 표제는 번호가 없이 조판되며, 목차에서도 번호 없이 나타납니다. 또, 전문의 내용을 조판할 때에는 페이지 번호가 소문자 로마 숫자(i, ii, ……)로 조판되며, 본문이 시작될 때 아라비아 숫자로 바뀌어 1부터 다시 시작합니다.

LaTeX 문서의 book 클래스에서는 `\frontmatter`, `\mainmatter`, `\backmatter` 제어 문자열을 사용해 전문, 본문, 부속물이 시작되는 부분을 지정할 수 있습니다. `\begin{document}`의 바로 뒤에 `\frontmatter`를 입력하고, 본문의 첫 장 또는 첫 편을 시작하는 부분 바로 앞에 `\mainmatter`를, 본문(부록 포함)의 내용이 모두 입력된 이후, 참고 문헌과 색인 등의 바로 앞에 `\backmatter`를 입력하면 됩니다. 그림 6.1을 참조하십시오.

6.3

각주

각주는 글의 내용 중 짧게 첨언할 내용이 있을 때, 첨언할 부분에 숫자 등으로 표시를 한 후 페이지의 아래쪽에 설명을 덧붙이는 형식의 주석을 뜻합니다. LaTeX에서 주석은 `\footnote` 제어 문자열을 사용해 조판합니다.



그림 6.1: book 클래스에서의 문서 구조

`\footnote[⟨number⟩]{⟨text⟩}`

⟨number⟩: 각주의 번호

⟨text⟩: 각주의 내용

각주를 덧붙이고 싶은 부분에 `\footnote`를 입력하고, 각주의 내용을 인자로 지정하면 됩니다.

각주는 부연 설명`\footnote{특히, 중요도가 낮은 내용을 적는 것이 좋습니다.}`을 하는 데 사용합니다.

각주는 부연 설명¹을 하는 데 사용합니다.

¹특히, 중요도가 낮은 내용을 적는 것이 좋습니다.

`\footnote`는 선택 인자를 입력받습니다. 선택 인자로써 출력하고 싶은 각주의 번호를 입력하면 됩니다. 각주의 번호는 원래 자동으로 계산되지만, 선택 인자를 지정한 경우에는 자동으로 계산되지 않습니다.

뒤에서 알아볼 표를 작성하는 tabular 환경에서는 각주 기능이 제대로 작동하지 않습니다. 표에서

각주를 올바르게 조판하기 위한 방법이 있지만, 복잡하여 이 책에서는 다루지 않겠습니다.

제 7 장

문단 만들기

앞 장에서는 \LaTeX 문서에 여러 문자를 입력하고 조판하도록 하는 방법을 알아보았습니다. 이번 장에서는 여러 가지 형태의 문단을 만드는 방법을 알아보겠습니다. 이에 앞서, 각 문단은 리턴 키를 두 번 눌러 구분하며, 각 문단에는 들여쓰기 처리가 됨을 기억하십시오.

7.1

목록

\LaTeX 에서는 세 종류의 목록을 만들 수 있습니다. 번호 붙은 목록, 구분점 목록, 설명 목록으로, `enumerate`, `itemize`, `description` 환경을 사용해 만듭니다. 목록 환경 안에서 각 항목은 `\item`으로 구분합니다. 목록 환경 안에서 또 목록 환경을 사용하여 목록을 중첩시킬 수도 있습니다.

목록을 작성하는 환경

`enumerate` 환경은 번호 붙은 목록을 만들고, `itemize` 환경은 구분점을 사용해 항목을 구분하는 목록을 만듭니다. 번호 붙은 목록과 구분점 목록은 총 4단계까지 중첩하여 만들 수 있습니다.

```

1 \begin{enumerate}
2   \item 첫째 항목입니다.
3   \begin{enumerate}
4     \item 첫째 항목의 첫째 세부 항목입니다.
5     \item 첫째 항목의 둘째 세부 항목입니다.
6   \end{enumerate}
7   \item 둘째 항목입니다.
8   \begin{enumerate}
9     \item 둘째 항목의 첫째 세부 항목입니다.
10    \item 둘째 항목의 둘째 세부 항목입니다.
11  \end{enumerate}
12 \end{enumerate}

```

1. 첫째 항목입니다.
 - (a) 첫째 항목의 첫째 세부 항목입니다.
 - (b) 첫째 항목의 둘째 세부 항목입니다.
2. 둘째 항목입니다.
 - (a) 둘째 항목의 첫째 세부 항목입니다.
 - (b) 둘째 항목의 둘째 세부 항목입니다.

```

1 \begin{itemize}
2   \item 1단계의 항목입니다.
3   \begin{itemize}
4     \item 2단계의 항목입니다.
5     \item 2단계의 항목입니다.
6   \end{itemize}
7   \item 1단계의 항목입니다.
8 \end{itemize}

```

- 1단계의 항목입니다.
 - 2단계의 항목입니다.
 - 2단계의 항목입니다.
- 1단계의 항목입니다.

`\item`은 선택 인자를 입력받습니다. 이 인자를 지정하면 번호나 구분점 대신 인자가 항목 첫머리에 출력됩니다. 번호 붙은 목록에서 번호를 계산할 때에는 해당 항목이 무시됩니다.

한편, 각각의 항목을 키워드로 지정하고, 그에 대한 설명을 작성하는 데 사용하는 목록도 만들 수 있습니다. `description` 환경을 사용하면 되며, 키워드는 `\item`의 선택 인자로 지정합니다.

```

1 \begin{description}
2   \item[\TeX] 최고의 문서 조판 시스템
3   \item[\LaTeX] \TeX 에서 사용하는 매크로 집합의 이름
4   \item[\LaTeXe] 현재 사용하는 버전의 \LaTeX
5 \end{description}

```

TeX 최고의 문서 조판 시스템
LaTeX TeX에서 사용하는 매크로 집합의 이름
LaTeX 2ε 현재 사용하는 버전의 LaTeX

설명 목록도 여러 번 중첩해서 사용할 수 있습니다. `enumerate`와 `itemize` 환경과는 달리, `description` 환경은 최대 6번까지 중첩하여 사용할 수 있습니다.

종류가 다른 목록을 중첩해 사용할 수도 있으며, 이때 각 중첩 단계는 목록의 종류에 관계없이 합산하여 계산됩니다. 예를 들어, `enumerate` 환경을 2단계까지 중첩한 후에는 `itemize` 환경을 세 번 이상 중첩해 사용할 수 없습니다. 중첩 가능 횟수를 넘기면 컴파일 과정에서 오류 메시지가 출력됩니다.

목록 형식 바꾸기

번호 붙은 목록에서 번호가 붙는 형식을 바꾸거나, 구분점 목록에서 항목의 구분 기호를 바꾸는 등 목록의 형식을 바꿀 수 있습니다. 이때에는 `enumitem` 패키지를 사용합니다.

먼저, 전처리부에서 `enumitem` 패키지를 불러온 뒤, 형식을 바꾸려는 목록 환경에 선택 인자를 다음으로 지정합니다.

`label={〈번호의 형식 또는 구분 기호〉}`

구분점 목록에서는 위 명령어의 중괄호 안에 원하는 구분 기호를 입력하면 됩니다. 번호 붙은 목록에서는 번호의 형식을 지정하는 명령어를 입력하면 됩니다. 이때, 번호가 출력될 자리에는 `\arabic*`, `\Alph*`, `\alph*`, `\Roman*`, `\roman*` 중 하나를 입력하면 되며, 각각 항목의 번호를 아라비아 숫자, 로마자 대·소문자, 로마 숫자 대·소문자로 출력합니다.

```

1 \begin{itemize}[label={\textreferencemark}]
2   \item First Item
3   \item Second item
4 \end{itemize}

```

※ First Item
※ Second item

```

1 \begin{enumerate}[label={\textbf{\Roman*}}]
2   \item First Item
3   \item Second item
4 \end{enumerate}

```

I First Item
II Second item

문서 전체의 목록 환경에 똑같은 설정을 지정하려면 전처리부에서 `\setlist` 제어 문자열을 사용합니다.

`\setlist[⟨list type⟩]{⟨options⟩}`

⟨list type⟩: 설정을 지정할 목록의 종류

⟨options⟩: 목록에 지정할 설정

목록에 지정할 설정을 필수 인자로 지정하면 되며, 구분점 목록, 번호 붙은 목록을 구분하거나 중첩된 횟수에 따라 서식을 다르게 지정하려면 선택 인자를 사용할 수 있습니다. 예를 들어,

`\setlist[enumerate,1]{label={\arabic*}}`

를 전처리부에 입력했다면, 1단계 enumerate 환경으로 조판한 목록에서의 각 항목은 번호가 '[1], [2], …'로 출력됩니다.

```

1 \begin{enumerate}
2   \item 첫 번째 항목
3   \item 두 번째 항목
4   \item 세 번째 항목
5 \end{enumerate}

```

[1] 첫 번째 항목
[2] 두 번째 항목
[3] 세 번째 항목

이외에도 enumitem 패키지는 목록의 여백을 조정하거나, 상호 참조 시의 표시 형식을 변경하는 등 많은 옵션을 제공합니다. 더 자세한 내용은 패키지 안내서를 참조하십시오.

7.2

요약문, 운문과 인용문

문서를 작성하는 데 도움이 되는 요약문·운문·인용문 작성 환경을 소개하겠습니다.

요약문

요약문은 `abstract` 환경 안에 작성합니다. 이 환경은 `article` 및 `report` 클래스에서만 사용할 수 있고, `book` 클래스에서는 사용할 수 없습니다. 별도의 인자 없이 다음과 같이 입력하면 됩니다.

```
1 \begin{abstract}
2   This document explains how to use
   \LaTeX\ to produce documents. ...
3 \end{abstract}
```

Abstract

This document explains how to use \LaTeX to produce documents. It explains the commands related to math as well as document formatting.

위 예시에서 보이는 것과 같이, 양쪽에 여백이 추가되고 위쪽에 ‘Abstract’라는 제목이 달리며, 기본 글꼴보다 작은 크기로 요약문이 작성됩니다. `kotex` 패키지를 `hangul` 옵션과 함께 불러오면 이 문구가 ‘요약’으로 바뀝니다. ‘Abstract’라는 문구는 `\abstractname`에 저장되어 있으며, 이를 수정하여 다른 내용을 제목으로 표시하도록 할 수 있습니다. 예를 들어, ‘초록’이 표시되도록 하려면 `abstract` 환경을 사용하기 전에 다음을 입력합니다.

```
\renewcommand\abstractname{초록}
```

운문

시나 노래 가사 등 운문을 작성할 때에는 `verse` 환경을 사용합니다. 각 행은 줄바꿈을 할 때처럼 `\\`로 구분하며, 각 연은 문단을 나눌 때처럼 리턴 키를 두 번 눌러 구분합니다.

```
1 다음은 「애국가」 1절과 2절의 가사이다.
2 \begin{verse}
3   동해 물과 백두산이 마르고 닳도록 \\
4   ... \\
5   대한 사람 대한으로 길이 보전하세
6
7   남산 위에 저 소나무 철갑을 두른 듯 \\
8   ... \\
9   대한 사람 대한으로 길이 보전하세
10 \end{verse}
```

다음은 「애국가」 1절과 2절의 가사이다.

동해 물과 백두산이 마르고 닳도록
하느님이 보우하사 우리나라 만세
무궁화 삼천리 화려 강산
대한 사람 대한으로 길이 보전하세

남산 위에 저 소나무 철갑을 두른 듯
바람 서리 불변함은 우리 기상일세
무궁화 삼천리 화려 강산
대한 사람 대한으로 길이 보전하세

글줄이 한 행의 내용보다 짧아 행 안에서 줄바꿈이 일어나는 경우, 두 번째 줄부터는 왼쪽에 여백이 추가됩니다.

```

1 다음은 「애국가」 3절과 4절의 가사이다.
2 \begin{verse}
3 가을 하늘 공활한데 높고 구름 없이 \\
4 밝은 달은 우리 가슴 일편단심일세 \\
5 무궁화 삼천리 화려 강산 \\
6 대한 사람 대한으로 길이 보전하세
7
8 이 기상과 이 맘으로 충성을 다하여 \\
9 괴로우나 즐거우나 나라 사랑하세 \\
10 무궁화 삼천리 화려강산 \\
11 대한 사람 대한으로 길이 보전하세
12 \end{verse}

```

다음은 「애국가」 3절과 4절의 가사이다.

가을 하늘 공활한데 높고 구름
없이
밝은 달은 우리 가슴 일편단심
일세
무궁화 삼천리 화려 강산
대한 사람 대한으로 길이 보전
하세
이 기상과 이 맘으로 충성을
다하여
괴로우나 즐거우나 나라 사랑
하세
무궁화 삼천리 화려 강산
대한 사람 대한으로 길이 보전
하세

인용문

인용문은 quotation 및 quote 환경을 사용해 작성합니다. 전자는 각 문단 첫 줄을 들여쓰기 처리하며, 후자는 들여쓰기 처리를 하지 않습니다.

```

1 \begin{quotation}
2 소년은 개울가에서 소녀를 보자 곧 윤 초시네 증손녀
  (曾孫女) 딸이라는 걸 알 수 있었다. ... 서울서는
  이런 개울물을 보지 못하기나 한 듯이.
3
4 벌써 며칠째 소녀는, 학교에서 돌아오는 길에 물장난이
  었다. ... 오늘은 징검다리 한가운데 앉아서 하고 있
  다.
5 % 황순원, '소나기'
6 \end{quotation}

```

소년은 개울가에서 소녀를 보자 곧
윤 초시네 증손녀(曾孫女) 딸이라는
걸 알 수 있었다. 소녀는 개울물에다
손을 잠그고 물장난을 하고 있는 것이
다. 서울서는 이런 개울물을 보지 못
하기나 한 듯이.

벌써 며칠째 소녀는, 학교에서 돌
아오는 길에 물장난이었다. 그런데, 어
제까지 개울 기슭에서 하더니, 오늘은
징검다리 한가운데 앉아서 하고 있다.

```

1 페르마는 자신의 저서에 이런 글귀를 남겼다.
2 \begin{quote}
3   나는 이 문제에 대한 ... 여기에 적지 않는다.
4 \end{quote}
5 여기에서 '이 문제'는 페르마의 마지막 정리를 가리킨다.

```

페르마는 자신의 저서에 이런 글귀를 남겼다.

나는 이 문제에 대한 놀라운 증명을 알고 있다. 여백이 부족하여 여기에 적지 않는다.

여기에서 '이 문제'는 페르마의 마지막 정리를 가리킨다.

7.3

문단 정렬 바꾸기

TeX은 기본적으로 글줄을 양쪽 여백에 맞추어 조판합니다. 이때 `flushleft`, `center`, `flushright` 환경을 사용하면 각각 왼쪽, 가운데, 오른쪽에 맞추어 글줄이 정렬됩니다.

```

1 양쪽에 맞추어 정렬한 문단입니다. 글줄의 양 끝이 페이지의 왼쪽 여백과 오른쪽 여백에 맞추어 조판됩니다.
2 \begin{flushleft}
3   왼쪽에 맞추어 정렬한 문단입니다. 글줄의 왼쪽 끝은 페이지의 왼쪽 여백에 맞지만, 오른쪽 끝은 맞지 않아 울퉁불퉁하게 조판됩니다.
4 \end{flushleft}
5 \begin{center}
6   가운데에 맞추어 정렬한 문단입니다. 글줄의 가운데가 페이지의 가운데에 맞도록 조판됩니다.
7 \end{center}
8 \begin{flushright}
9   오른쪽에 맞추어 정렬한 문단입니다. 글줄의 오른쪽 끝은 페이지의 오른쪽 여백에 맞지만, 왼쪽 끝은 맞지 않아 울퉁불퉁하게 조판됩니다.
10 \end{flushright}

```

양쪽에 맞추어 정렬한 문단입니다. 글줄의 양 끝이 페이지의 왼쪽 여백과 오른쪽 여백에 맞추어 조판됩니다.

왼쪽에 맞추어 정렬한 문단입니다. 글줄의 왼쪽 끝은 페이지의 왼쪽 여백에 맞지만, 오른쪽 끝은 맞지 않아 울퉁불퉁하게 조판됩니다.

가운데에 맞추어 정렬한 문단입니다. 글줄의 가운데가 페이지의 가운데에 맞도록 조판됩니다.

오른쪽에 맞추어 정렬한 문단입니다. 글줄의 오른쪽 끝은 페이지의 오른쪽 여백에 맞지만, 왼쪽 끝은 맞지 않아 울퉁불퉁하게 조판됩니다.

환경 대신 `\raggedright`, `\centering`, `\raggedleft` 제어 문자열을 사용할 수도 있습니다. 이 제어 문자열은 선언형 제어 문자열이며, 한 문단 안에서 이 제어 문자열이 여러 번 등장하면 가장 마지막의 제어 문자열을 따라 문단이 정렬됩니다. 다음 예시에서 볼 수 있듯, 환경을 사용할 때와는 다르게 문단의 위아래에 공백이 삽입되지 않습니다.

- 1 양쪽에 맞추어 정렬한 문단입니다. 글줄의 양 끝이 페이지의 왼쪽 여백과 오른쪽 여백에 맞추어 조판됩니다. `\par`
- 2 `\raggedright` 왼쪽에 맞추어 정렬한 문단입니다. 글줄의 왼쪽 끝은 페이지의 왼쪽 여백에 맞지만, 오른쪽 끝은 맞지 않아 울퉁불퉁하게 조판됩니다. `\par`
- 3 `\centering` 가운데에 맞추어 정렬한 문단입니다. 글줄의 가운데가 페이지의 가운데에 맞도록 조판됩니다. `\par`
- 4 `\raggedleft` 오른쪽에 맞추어 정렬한 문단입니다. 글줄의 오른쪽 끝은 페이지의 오른쪽 여백에 맞지만, 왼쪽 끝은 맞지 않아 울퉁불퉁하게 조판됩니다.

양쪽에 맞추어 정렬한 문단입니다. 글줄의 양 끝이 페이지의 왼쪽 여백과 오른쪽 여백에 맞추어 조판됩니다.

왼쪽에 맞추어 정렬한 문단입니다. 글줄의 왼쪽 끝은 페이지의 왼쪽 여백에 맞지만, 오른쪽 끝은 맞지 않아 울퉁불퉁하게 조판됩니다.

가운데에 맞추어 정렬한 문단입니다. 글줄의 가운데가 페이지의 가운데에 맞도록 조판됩니다.

오른쪽에 맞추어 정렬한 문단입니다. 글줄의 오른쪽 끝은 페이지의 오른쪽 여백에 맞지만, 왼쪽 끝은 맞지 않아 울퉁불퉁하게 조판됩니다.

ragged2e 패키지를 사용한 문단 정렬

\TeX 은 글줄 끝에 오는 긴 단어를 필요한 경우 중간에서 끊도록 합니다(영어 등의 경우 사이에 하이픈을 넣는 등의 처리도 합니다). 하지만 위의 명령어를 사용하여 정렬을 바꾸어 주면 단어 중간에서 줄바꿈이 일어나지 않습니다. 이에 따라, 글줄의 길이에 비해 긴 단어가 사용되면 정렬하지 않은 쪽 끝이 ‘지나치게’ 삐뚤빼뚤해질 수 있습니다.

ragged2e 패키지를 사용하면 문단의 정렬하지 않은 쪽 글줄이 ‘덜 삐뚤빼뚤’해집니다. 이 패키지를 불러온 후 앞의 제어 문자열이나 환경 대신 `\RaggedRight`, `\Centering`, `\RaggedLeft` 제어 문자열과 `FlushLeft`, `Center`, `FlushRight` 환경을 사용하면 됩니다. 또, 양쪽 정렬로 되돌리는 `\justifying` 제어 문자열도 사용할 수 있습니다.

- 1 양쪽에 맞추어 정렬한 문단입니다. 글줄의 양 끝이 페이지의 왼쪽 여백과 오른쪽 여백에 맞추어 조판됩니다.
- 2 `\begin{FlushLeft}`
- 3 왼쪽에 맞추어 정렬한 문단입니다. 글줄의 왼쪽 끝은 페이지의 왼쪽 여백에 맞지만, 오른쪽 끝은 맞지 않아 삐뚤빼뚤하게 조판됩니다.
- 4 `\end{FlushLeft}`
- 5 `\begin{Center}`
- 6 가운데에 맞추어 정렬한 문단입니다. 글줄의 가운데가 페이지의 가운데에 맞도록 조판됩니다.
- 7 `\end{Center}`
- 8 `\begin{FlushRight}`
- 9 오른쪽에 맞추어 정렬한 문단입니다. 글줄의 오른쪽 끝은 페이지의 오른쪽 여백에 맞지만, 왼쪽 끝은 맞지 않아 삐뚤빼뚤하게 조판됩니다.
- 10 `\end{FlushRight}`

양쪽에 맞추어 정렬한 문단입니다. 글줄의 양 끝이 페이지의 왼쪽 여백과 오른쪽 여백에 맞추어 조판됩니다.

왼쪽에 맞추어 정렬한 문단입니다. 글줄의 왼쪽 끝은 페이지의 왼쪽 여백에 맞지만, 오른쪽 끝은 맞지 않아 삐뚤빼뚤하게 조판됩니다.

가운데에 맞추어 정렬한 문단입니다. 글줄의 가운데가 페이지의 가운데에 맞도록 조판됩니다.

오른쪽에 맞추어 정렬한 문단입니다. 글줄의 오른쪽 끝은 페이지의 오른쪽 여백에 맞지만, 왼쪽 끝은 맞지 않아 삐뚤빼뚤하게 조판됩니다.

제 8 장

서식 지정하기

이번 장에서는 기본적인 서식을 지정하는 방법을 알아보겠습니다. 글꼴의 종류나 크기, 문단의 정렬 방법을 바꾸는 방법을 다룹니다. 마지막 절에서는 입력한 코드를 \LaTeX 문법으로 해석하지 않고 그대로 조판하는 방법을 알아봅니다.

서식을 지정할 때에는 ‘이곳에서부터 해당 서식을 사용함’을 선언하는 제어 문자열을 사용할 수도 있고, ‘여기부터 여기까지 서식을 지정’하도록 하는 제어 문자열(또는 환경)을 사용할 수도 있습니다. 전자를 ‘서식 선언형’, 후자를 ‘범위 지정형’ 제어 문자열(또는 환경)이라고 부르겠습니다. 서식 선언형 제어 문자열의 유효 범위는 해당 제어 문자열이 입력된 영역으로 한정되고, 범위 지정형 제어 문자열은 인자로 지정한 내용에만 해당 서식이 지정됩니다.

8.1

특정 부분 강조하기

특정 부분을 강조할 때에는 `\em`(서식 선언형) 또는 `\emph`(범위 지정형) 제어 문자열을 사용합니다. 강조 서식이 지정되면 글자가 이탤릭체로 조판됩니다.

The word `\em contenido` means ‘happy’ in Spanish. The word `\emph{triste}` means ‘sad’ in Spanish.

The word *contento* means ‘happy’ in Spanish. The word *triste* means ‘sad’ in Spanish.

강조 서식이 지정된 상태에서 강조 서식을 한 번 더 적용하면 다시 정체로 조판됩니다.

`\em` If you want to emphasize the text that is `\emph{already being emphasized}`, then it will become upright.

If you want to emphasize the text that is already being emphasized, then it will become upright.

밑줄을 그어 내용을 강조할 수도 있습니다. 이때에는 `\underline` 제어 문자열을 사용하면 됩니다.

밑줄을 그으려는 문자열을 인자로 지정하십시오. 이때 밑줄을 그은 문자열은 줄바꿈 처리가 되지 않으므로, 긴 내용을 인자로 지정하거나 글줄 끝에 밑줄이 오지 않도록 주의해야 합니다.

ketex 패키지를 불러온 경우 `\dotemph` 제어 문자열을 사용할 수 있으며, 이는 인자로 지정한 문자열을 드러냄표를 사용해 강조합니다.

한국어 글에서는 주로 `\underline{밑줄}`이나 `\dotemph{드러냄표}`를 사용해 내용을 강조한다.

한국어 글에서는 주로 밑줄이나 드러냄표를 사용해 내용을 강조한다.

예제 8.1 아래 문장을 `\em` 또는 `\emph`를 사용해 조판하여라.

한국어 문장을 기울임꼴을 사용하여 강조하는 것은 옳지 않다.

또, 위의 문장을 기울임꼴 대신 밑줄과 드러냄표를 사용해 조판하여라.

8.2

글꼴 바꾸기

강조 서식 외에도, 볼드체나 이탤릭체, 산세리프체를 사용하는 등 글꼴의 종류를 바꿀 수 있으며, 큰 글자나 작은 글자를 사용하는 등 글꼴의 크기를 바꿀 수도 있습니다.

글꼴 종류 바꾸기

TeX 2ε에서 글꼴 종류를 지정할 때에는 타입페이스, 웨이트, 스타일(style)의 세 가지 속성에 따라 변경할 수 있습니다.

- 타입페이스(typeface) 글꼴의 전체적인 모양을 가리킵니다. TeX에서는 세리프(serif), 산세리프(sans-serif), 고정폭(monospaced)의 세 가지 타입페이스를 사용할 수 있습니다.
- 웨이트(weight) 각 낱자를 이루는 획의 두께를 가리킵니다. TeX의 기본 글꼴은 두 가지, 보통(medium)과 볼드(bold)의 웨이트를 제공합니다.
- 스타일(style) 한 타입페이스에서의 글꼴의 모양을 가리킵니다. TeX의 기본 글꼴은 ‘정체(upright)’, ‘이탤릭체(italic)’, ‘기울임꼴(slanted)’, ‘대문자와 작은 대문자(caps and small caps)’¹⁾의 네 가지 스타일을 지정할 수 있습니다.

각 타입페이스, 웨이트, 스타일을 지정하는 제어 문자열과 함께 글꼴을 적용한 예시를 표 8.1에 나타냈습니다. `\textnormal`이나 `\normalfont`를 입력했을 때 사용되는 타입페이스·웨이트·스타일의 기본값은 세리프 글꼴, 보통 굵기, 정체입니다.

¹⁾대문자와 작은 대문자 스타일은 대문자의 크기를 줄여서 소문자를 표현하는 방식을 뜻합니다. 로마자 팬그램 ‘A Quick Brown Fox Jumps Over the Lazy Dog’에 대문자와 작은 대문자 스타일을 적용하면 아래의 결과를 얻습니다.

A QUICK BROWN FOX JUMPS OVER THE LAZY DOG.

표 8.1: 글꼴 종류, 굵기, 스타일

종류	범위 지정형 제어 문자열	서식 선언형 제어 문자열	예시
기본 글꼴	<code>\textnormal</code>	<code>\normalfont</code>	기본 글꼴입니다. This is the default font.
세리프	<code>\textrm</code>	<code>\rmfamily</code>	세리프 글꼴입니다. This is roman family.
산세리프	<code>\textsf</code>	<code>\sffamily</code>	산세리프 글꼴입니다. This is sans-serif family.
고정폭	<code>\texttt</code>	<code>\ttfamily</code>	고정폭 글꼴입니다. This is monospaced family.
보통	<code>\textmd</code>	<code>\mdseries</code>	중간 굵기 글꼴입니다. This is medium weight.
굵게	<code>\textbf</code>	<code>\bfseries</code>	굵은 글꼴입니다. This is bold weight.
정체	<code>\textup</code>	<code>\upshape</code>	Sample text in Upright style.
이탤릭체	<code>\textit</code>	<code>\itshape</code>	<i>Sample text in Italic style.</i>
기울임꼴	<code>\textsl</code>	<code>\slshape</code>	<i>Sample text in Slanted style.</i>
작은 대문자	<code>\textsc</code>	<code>\scshape</code>	SAMPLE TEXT IN SMALL-CAPS STYLE.

필요에 따라 `\textbf{글꼴}`을 바꾸면
`{\sffamily\itshape` 강조 효과}를 낼 수 있다.

필요에 따라 **글꼴**을 바꾸면 **강조 효과**를 낼 수 있다.

You can `\textbf{emphasize}` text by changing
`{\sffamily\itshape` fonts}.

You can **emphasize** text by changing *fonts*.

글꼴 설정에 따라 특정 속성의 조합을 사용하지 못할 수도 있습니다. 예를 들어, \LaTeX 기본 설정에서는 산세리프 타입페이스의 볼드 이탤릭체를 사용할 수 없습니다. 이때에는 사용 가능한 다른 글꼴로 대체되어 표시됩니다. 컴파일 과정에서

LaTeX Font Warning: Some font shapes were not available, defaults substituted.

등의 메시지가 출력되면 지정한 글꼴 속성이 제대로 적용되었는지 검토하십시오.

이전 버전 명령어에 관한 조언

[참고](#) `\it`, `\rm`, `\bf`, `\sf`, `\tt`, `\sl`, `\sc`를 모르는 분은 이 부분을 넘어가셔도 됩니다.

\LaTeX 표준 클래스에서는 `\it`, `\rm`, `\bf`, `\sf`, `\tt`, `\sl`, `\sc` 등의 제어 문자열을 사용해도 글꼴을 바꿀 수 있습니다. 이들은 \LaTeX 의 이전 버전인 \LaTeX 2.09나 플레인 \TeX 에서 사용하는 글꼴 변경 명령어입니다.

현재의 \LaTeX 2_ε에서는 공식적으로 이들 명령어가 지원되지 않으므로, 패키지나 클래스에 따라 전혀 다른 동작을 할 수도 있습니다. \LaTeX 표준 클래스에서는 하위 호환성을 위해 이 제어 문자열을 제공할 뿐입니다. 또, 타입페이스, 웨이트, 스타일의 세 가지 속성을 교차해 지정할 수 없습니다. 즉,

`\bf\sf`를 입력하면 볼드 산세리프체가 아니라 보통 굵기 산세리프체로 글자가 조판됩니다. 따라서 표 8.1의 제어 문자열을 사용해 글꼴을 변경하는 것이 바람직합니다.

글꼴 크기 바꾸기

LaTeX에서는 총 10단계로 글꼴 크기를 지정할 수 있습니다. 글꼴 크기는 문서의 클래스 옵션으로 지정한 기본 글꼴 크기에 대해 상대적으로 정해집니다. 글꼴 크기를 지정하는 제어 문자열과 환경을 표 8.2에 정리하였습니다.

표 8.2: 글꼴 크기를 바꾸는 명령어

크기	서식 선언형 제어 문자열	범위 지정형 환경	예시
4단계 작게	<code>\tiny</code>	tiny 환경	4단계 작게
3단계 작게	<code>\scriptsize</code>	scriptsize 환경	3단계 작게
2단계 작게	<code>\footnotesize</code>	footnotesize 환경	2단계 작게
1단계 작게	<code>\small</code>	small 환경	1단계 작게
기본 크기	<code>\normalsize</code>	normalsize 환경	기본 크기
1단계 크게	<code>\large</code>	large 환경	1단계 크게
2단계 크게	<code>\Large</code>	Large 환경	2단계 크게
3단계 크게	<code>\LARGE</code>	LARGE 환경	3단계 크게
4단계 크게	<code>\huge</code>	huge 환경	4단계 크게
5단계 크게	<code>\Huge</code>	Huge 환경	5단계 크게

글꼴을 `\begin{huge}크게\end{huge}` 바꿀 수도, `{\scriptsize 작게}` 바꿀 수도 있다.

글꼴을 크게 바꿀 수도, 작게 바꿀 수도 있다.

예제 8.2 표 8.1과 표 8.2의 제어 문자열을 사용하여 아래의 문장을 조판해 보자.

기억하세요! 더 많은 수의 글꼴을 한 문서 안에서 사용할수록, 문서는 더 읽기 편해지고 더 화려해집니다.

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

이 문장은 [5]의 6.2.3절에서 인용하였다.

문서의 기본 글꼴 크기에 따라 각 명령어가 설정하는 실제 글꼴 크기는 표 8.3에서 확인할 수 있습니다. 다만, 문서를 작성할 때에는 글꼴의 실제 크기가 아닌 상대적적인 크기 차이를 기준으로 글꼴 크기를 결정하는 것이 좋습니다.

표 8.3: 기본 글꼴 크기에 따라 명령어가 설정하는 글꼴 크기

기본 크기	-4	-3	-2	-1	0	+1	+2	+3	+4	+5
10 pt	5 pt	7 pt	8 pt	9 pt	10 pt	12 pt	14.4 pt	17.28 pt	20.74 pt	24.88 pt
11 pt	6 pt	8 pt	9 pt	10 pt	10.95 pt	12 pt	14.4 pt	17.28 pt	20.74 pt	24.88 pt
12 pt	6 pt	8 pt	10 pt	10.95 pt	12 pt	14.4 pt	17.28 pt	20.74 pt	24.88 pt	24.88 pt

위 첨자와 아래 첨자

흔한 경우는 아니지만, 수식이 아닌 일반적인 글을 작성할 때에도 위 첨자나 아래 첨자를 사용하기도 합니다. 대표적인 예시는 영어로 날짜를 적을 때의 서수 표기입니다. (문서를 작성하는 사람의 선호도에 따라 첨자를 사용하지 않기도 합니다.)

Today is 23rd day of April, 2023.

Today is 23rd day of April, 2023.

위 첨자는 `\textsuperscript`, 아래 첨자는 `\textsubscript`를 사용해 입력합니다. 수식 입력 모드에서의 문법인 `^`와 `_`를 사용하면 컴파일 오류가 발생하므로 주의하시기 바랍니다.

8.3

입력한 내용을 그대로 조판하기

프로그램의 소스 코드를 \LaTeX 으로 조판하려면 역슬래시나 앰퍼샌드, 중괄호 등을 표 3.1의 제어 문자열을 사용하여 입력해야 합니다. 이는 매우 귀찮은 일이므로, 입력한 문자열을 \LaTeX 문법으로 해석하지 않고 문자 그대로 조판하도록 하는 방법을 사용하는 것이 편리합니다.

verbatim 환경

여러 줄에 걸쳐 작성하는 코드는 verbatim 환경을 사용해 조판할 수 있습니다. 환경을 시작하는 명령어 `\begin{verbatim}`과 환경을 끝내는 명령어 `\end{verbatim}`을 입력하고 그 사이에 조판하려는 코드를 입력하면 됩니다.

```
1 \begin{verbatim}
2 #include <stdio.h>
3 void main() {
4     printf("Hello, world!");
5     return;
6 }
7 \end{verbatim}
```

```
#include <stdio.h>
void main() {
    printf("Hello, world!");
    return;
}
```

위 예시에서 확인할 수 있듯이, verbatim 환경에 입력한 코드는 프로그래밍 관련 글의 관행에

따라 고정폭 글꼴로 조판됩니다.

입력한 공백 문자를 명시적으로 표시해야 한다면 별표 붙은 `verbatim*` 환경을 사용할 수 있습니다. 이 환경에서는 공백 문자가 기호 `_`로 표시됩니다.

```
1 \begin{verbatim*}
2 print("Hello, world!")
3 \end{verbatim*}
```

```
print("Hello, _world!")
```

\verb 제어 문자열

그대로 조판하려는 내용이 짧은 경우에는 `\verb` 또는 `\verb*` 제어 문자열을 사용할 수 있습니다. 이 제어 문자열을 사용하면 행중수식처럼 코드가 글줄 안에 섞여서 표시됩니다.

```
\verb<delim><text><delim>
\verb*<delim><text><delim>
```

<delim>: 영역을 지정하기 위한 경계 문자. 로마자 52자와 별표(*), 공백 문자를 제외한 임의의 문자 1개를 사용할 수 있다.

<text>: 그대로 조판할 내용

예를 들어, ‘\TeX’을 조판하려면 `\verb|\TeX|`를 입력하면 되고, ‘\verb|\TeX|’를 조판하려면 `\verb+\verb|\TeX|+`를 입력하면 됩니다. (여기에서 사용한 |나 + 대신 다른 문자를 사용해도 됩니다.)

`\verb` 대신 `\verb*`를 사용하면 공백 문자가 기호 `_`로 바뀌어 명시적으로 표시됩니다. 예를 들어, `\verb|Hello, world!|`는 ‘Hello, world!’를 조판하지만 `\verb*|Hello, world!|`는 ‘Hello, _world!’를 조판합니다.

제 9 장

정의, 정리, 증명 작성하기

이번 장에서는 \LaTeX 에서 정의, 정리, 증명 등 수학적 문단을 일정한 형식에 맞춰 작성하는 방법을 알아보겠습니다. 기본적인 과정은 `\newtheorem` 제어 문자열을 사용해 정의를 작성하기 위한 환경, 정리를 작성하기 위한 환경, 증명을 작성하기 위한 환경을 만들고, 이들 환경을 사용해 문단을 작성하는 것입니다.

9.1

`\newtheorem`으로 환경 정의하기

`\newtheorem`은 정리형 환경(수학 전공 도서에서 흔히 볼 수 있는 ‘Definition’, ‘Theorem’, ‘Proof’ 등의 문단을 \LaTeX 으로 조판하기 위한 환경)을 정의하는 제어 문자열입니다. `amsthm` 패키지는 이 제어 문자열의 기능을 확장하는 역할을 하며, 이 책에서는 `amsthm` 패키지를 사용한다고 가정하고 설명하겠습니다.

`\newtheorem` 제어 문자열은 아래와 같이 사용합니다.

```
\newtheorem{<env. name>}[<ref. counter>]{<heading>}[<parent counter>]
\newtheorem*{<env. name>}{<heading>}
```

`<env. name>`: 정의하려는 환경의 이름

`<ref. counter>`: 번호를 공유할 환경의 이름

`<heading>`: 이 환경을 사용할 때 문단 첫머리에 출력할 내용

`<parent counter>`: 번호를 종속시킬 상위 카운터

이렇게 하면 이름이 `<env. name>`인 환경이 정의되며, 이 환경을 사용해 문단을 조판할 때에는 첫머리에 `<heading>`에 입력된 내용이 번호와 함께 출력됩니다. 별표 붙은 `\newtheorem*` 제어 문자열을 사용해 환경을 정의하면 해당 환경을 사용해 문단을 조판할 때 번호가 조판되지 않습니다.

예를 들어, 보조 정리(lemma)를 작성하기 위해 `lemma` 환경과 번호 없이 공리(axiom)을 작성하기 위한 `axiom` 환경을 정의하려면 아래와 같이 입력합니다.

```
1 \newtheorem{lemma}{Lemma}
2 \newtheorem*{axiom}{Axiom}
```

이렇게 하면 lemma 환경과 axiom 환경이 정의되며, lemma 환경을 사용해 문단을 작성하면 그 문단의 첫머리에 ‘Lemma 1.’, ‘Lemma 2.’, ……가 자동으로 출력됩니다. axiom 환경을 사용해 문단을 작성하면 문단의 첫머리에 번호 없이 ‘Axiom.’이 출력됩니다.

`<env. name>`에 입력하는 환경의 이름은 다음 규칙을 만족해야 합니다.

- 다른 이미 정의된 제어 문자열의 이름과 겹치면 안 된다.
- 다른 이미 정의된 환경의 이름과 겹치면 안 된다.
- ‘the’나 ‘end’로 시작하면 안 된다.

만약 문제가 되는 환경 이름을 사용했다면 컴파일 과정에서 오류 메시지가 출력됩니다. 이 경우 환경의 이름을 적절한 것으로 바꾸십시오.

`\newtheorem`으로 정의한 환경은 아래와 같이 사용합니다.

```
\begin{<env. name>}[<subtitle>]
```

`<env. name>`: 정의한 환경의 이름

`<subtitle>`: 문단의 번호 뒤에 붙일 부제

```
\end{<env. name>}
```

선택 인자를 지정하면 문단의 이름과 번호 뒤에 부제가 작성됩니다.

예를 들어, lemma와 axiom 환경을 앞과 같이 정의했다면 이들을 아래와 같이 사용할 수 있습니다.

```
1 \begin{lemma}[Zorn]
2   모든 연쇄가 상계를 가지도록 하는 순서집합은 극대원
   을 가진다.
3 \end{lemma}
4 \begin{axiom}[Choice]
5    $\emptyset \notin X$ 인 임의의 집합
    $X$ 의 선택 함수가 존재한다.
6 \end{axiom}
```

Lemma 1 (Zorn). 모든 연쇄가 상계를 가지도록 하는 순서집합은 극대원을 가진다.

Axiom (Choice). $\emptyset \notin X$ 인 임의의 집합 X 의 선택 함수가 존재한다.

9.2

환경의 번호 다루기

문단의 번호가 다르게 나타나도록 할 수도 있습니다.

다른 번호의 하위 번호로 만들기

환경의 번호를 문서의 장 또는 절 등의 하위 번호로 만들어 장 또는 절 번호가 바뀔 때마다 환경의 번호가 초기화되도록 하려면 `<parent counter>`에 해당 번호의 이름을 입력하면 됩니다. 예를 들어, theorem 환경의 번호를 장 번호의 하위 번호로 만들고자 한다면 전처리부에

```
\newtheorem{theorem}{Theorem}[chapter]
```


를 입력하면 됩니다. 이후 theorem 환경을 사용하면 아래 예시와 같이 문단의 번호가 장 번호와 함께 표시됩니다.

```
1 \begin{theorem}[Zermelo]
2 임의의 집합은 정렬 가능하다.
3 \end{theorem}
```

Theorem 9.1 (Zermelo). 임의의 집합은 정렬 가능하다.

다른 환경의 번호와 연동시키기

한 정리형 환경의 번호를 (미리 정의된) 다른 정리형 환경의 번호와 연동시키고자 한다면 `\ref{counter}`에 해당 환경의 이름을 입력하면 됩니다. 예를 들어, 위에서 정의한 theorem 환경과 연동되는 번호를 가지는 proposition 환경은 아래와 같이 정의합니다.

```
\newtheorem{proposition}[theorem]{Proposition}
```

이렇게 정의한 proposition 환경을 사용하면 아래와 같은 결과를 얻습니다.

```
1 \begin{proposition}[Hausdorff]
2 임의의 순서집합은 극대 연쇄를 가진다.
3 \end{proposition}
```

Proposition 9.2 (Hausdorff). 임의의 순서집합은 극대 연쇄를 가진다.

반대로, theorem 환경도 proposition 환경과 번호를 공유하므로, 이후 theorem 환경을 사용하면 9.3이 문단 번호로 출력됩니다.

```
1 \begin{theorem}
2 집합족  $\mathcal{X}$ 의 원소들의 곱집합은  $\mathcal{X}$ 의 선택 함수들의 집합으로 간주할 수 있다.
3 \end{theorem}
```

Theorem 9.3. 집합족 X 의 원소들의 곱집합은 X 의 선택 함수들의 집합으로 간주할 수 있다.

번호가 제목 앞에 표시되도록 하기

전처리부에 `\swapnumbers`를 입력하면 이 이후에 `\newtheorem`으로 정의되는 정리형 환경의 번호와 제목의 순서를 바꿉니다. 예를 들어, corollary 환경을 정의할 때

```
1 \swapnumbers
2 \newtheorem{corollary}[theorem]{Corollary}
```

라고 입력하면 corollary 환경을 사용할 때 아래의 결과를 얻습니다.

```
1 \begin{corollary}
2 공집합이 아닌 집합들의 곱집합은 공집합이 아니다.
3 \end{corollary}
```

9.4 Corollary. 공집합이 아닌 집합들의 곱집합은 공집합이 아니다.

9.3

문단 스타일 바꾸기

책에 따라, 용어를 정의하는 문단은 정체로, 정리는 이탤릭체나 기울임꼴로 조판하는 등 문단의 스타일에 구분을 두기도 합니다. amsthm 패키지는 다음 세 가지 기본 스타일을 제공합니다. 기본값으로는 plain 스타일이 사용됩니다.

- plain 상하 여백 있음, 제목 볼드체, 내용 이탤릭체
- definition 상하 여백 있음, 제목 볼드체, 내용 정체
- remark 상하 여백 없음, 제목 이탤릭체, 내용 정체

plain 스타일은 주로 정리, 보조정리, 따름정리, 명제 등 뒤에 증명이 따르는 문단을 작성할 때 사용하고, definition 스타일은 이름대로 정의나 표기법 등을 작성할 때 사용합니다. remark는 참고 사항이나 메모 등에 사용합니다.

`\theoremstyle` 제어 문자열을 사용하면 `\newtheorem`으로 정의되는 환경의 스타일을 바꿀 수 있습니다.

```
\theoremstyle{<style name>}
```

<style name>: 사용할 스타일의 이름

`\theoremstyle`은 인자로 지정된 스타일을 이후에 정의하는 정리형 환경에 적용합니다. 예를 들어, 다음과 같이 환경을 정의하면 definition 환경은 definition 스타일로, remark 환경은 remark 스타일로 조판됩니다.

```
1 \theoremstyle{definition}
2 \newtheorem{definition}[theorem]{Definition}
3 \theoremstyle{remark}
4 \newtheorem*{remark}{Remark}
```

```
1 \begin{definition}[선택 함수]
2   집합  $(X)$ 의 선택 함수는  $(\forall x \in X$ 
    $\ ; f(x) \in x)$ 를 만족시키는 함수  $(f \colon$ 
    $X \rightarrow \bigcup X)$ 이다.
3 \end{definition}
4 \begin{remark}
5   만약  $(\emptyset \in X)$ 이면  $(X)$ 의 선택 함수가 존재할 수 없다.
6 \end{remark}
```

Definition 9.5 (선택 함수). 집합 X 의 선택 함수는 $\forall x \in X \ f(x) \in x$ 를 만족시키는 함수 $f: X \rightarrow \bigcup X$ 이다.

Remark. 만약 $\emptyset \in X$ 이면 X 의 선택 함수가 존재할 수 없다.

`\newtheoremstyle` 제어 문자열을 사용하면 새로운 스타일을 정의할 수도 있습니다. 이 방법에 대해서는 amsthm 패키지의 안내서를 참조하십시오.

9.4

증명 작성하기

amsthm 패키지는 `\qed` 제어 문자열과 proof 환경을 제공합니다. `\qed`는 글줄의 오른쪽 끝에 속이 빈 상자 기호를 생성합니다. 증명이 완료되었음을 표시하기 위해 이 제어 문자열을 사용할 수 있습니다.

... 따라서 주어진 정리가 성립한다. `\qed`

... 따라서 주어진 정리가 성립한다.

□

proof 환경은 정리의 증명을 작성하기 위한 환경입니다. 문단의 첫머리에 ‘Proof’를 이탤릭체로 조판하고, 마지막에 증명 완료 기호 □를 생성합니다. 이때, kotex 패키지를 불러온 경우 ‘Proof’ 대신 ‘증명’이 출력됩니다. 증명 문단의 제목은 `\proofname` 제어 문자열에 저장되어 있으며, 이를 재정의하면 문서 전체에서 증명 문단의 제목을 변경할 수 있습니다.

```
1 \begin{proof}
2   ... 따라서 주어진 정리가 성립한다.
3 \end{proof}
```

증명. ... 따라서 주어진 정리가 성립한다.

□

특정 증명의 제목을 변경하고자 한다면 환경에 선택 인자를 추가하면 됩니다. 인자로 입력된 내용이 이탤릭체로 문단 첫 부분에 출력됩니다.

```
1 \begin{proof}[Proof of Lemma 1]
2   ... Thus we proved the lemma.
3 \end{proof}
```

Proof of Lemma 1. ... Thus we proved the lemma.

□

proof 환경 안의 내용이 특정 환경(목록 작성 환경이나 quote 등)이나 별행수식으로 끝나는 경우, 아래와 같이 증명 끝 기호가 환경 또는 수식 아래에 출력됩니다.

```
1 \begin{proof}
2   ... Hence we showed that
3   \[ |G:H| = \frac{|G|}{|H|} \quad
   \text{where } |G| < \infty, H \leq G. \]
4 \end{proof}
```

Proof. ... Hence we showed that

$$|G:H| = \frac{|G|}{|H|} \quad \text{where } |G| < \infty, H \leq G.$$

□

이러한 경우, `\qedhere` 제어 문자열을 입력하여 증명 완료 기호가 출력되는 위치를 수정할 수 있습니다.

```

1 \begin{proof}
2   ... Hence we showed that
3   \[ |G:H| = \frac{|G|}{|H|} \quad
   \text{where } |G| < \infty, H \leq G.
   \qedhere
4 \end{proof}

```

Proof. ... Hence we showed that

$$|G:H| = \frac{|G|}{|H|} \quad \text{where } |G| < \infty, H \leq G. \quad \square$$

증명 끝 기호는 `\qedsymbol` 제어 문자열에 저장되어 있습니다. 이를 재정의하면 문단 끝에 다른 기호가 출력되도록 할 수 있습니다.

```
\renewcommand\qedsymbol{\(\blacksquare\)}
```

를 입력하면 증명 끝 기호가 검은색 상자(■)로 바뀌며,

```
\renewcommand\qedsymbol{}
```

를 입력하면 증명 끝 기호가 출력되지 않습니다.

제 III 편

수식 작성하기

제 III 편에서는 수식을 작성하는 문법을 배웁니다. 분수나 첨자와 같이 기본적인 수식 요소들에서 시작해, 다양한 기호를 어떻게 입력하고 행렬과 같은 복잡한 수식 요소는 어떻게 넣는지, 또 여러 수식을 어떻게 배치하는지 알아볼 것입니다.

제10장은 수식을 넣는 방법과, 기본적인 수식 요소를 입력하는 문법을 알아봅니다. 분수, 근호, 첨자와 연산자, 함수를 입력하는 제어 문자열을 소개합니다. 또, 수많은 수학 기호를 입력하는 방법도 설명합니다. 이후 글꼴을 바꾸거나 장식 기호를 넣는 방법도 설명합니다.

제11장은 큰 수식을 둘러싸는 괄호나 행렬 등 복잡한 문법의 수식을 입력하는 방법을 설명합니다. 이 장에서는 다양한 환경을 소개합니다.

제12장은 별행수식을 배치하고 번호를 붙이는 방법을 설명합니다. 수식의 번호가 자동으로 계산되도록 할 수도 있고, 자신이 원하는 기호를 직접 붙일 수도 있습니다. 한편, 여러 수식을 보기 좋게 정렬하고 배치하는 방법도 설명합니다.

제13장은 tikz-cd 패키지를 사용해 가환 도표를 그리는 방법을 알아봅니다. 대수학과 범주론에서 많이 사용하는 가환 도표를 순서에 따라 그려 나가 봅시다.

제 10 장

수식 입력하기

이번 장에서는 \LaTeX 에서 수식을 입력하는 방법과 함께 수식 입력 모드의 특징을 소개하고, 수식을 이루는 요소 중 비교적 간단한 것, 즉 분수와 근호, 첨자를 입력하는 방법을 설명합니다. 또, 각종 기호와 장식적 요소를 입력하고 수식 글꼴을 바꾸는 제어 문자열을 설명합니다.

10.1

수식 입력 모드

먼저, \LaTeX 에서 수식을 입력하기 위한 ‘수식 입력 모드’에 대해 알아보겠습니다. \LaTeX 은 텍스트 모드와 수식 입력 모드 사이를 전환하며 문서를 조판합니다. 소스 파일의 컴파일을 시작할 때에는 텍스트 모드에서 조판이 시작되고, 여기에서는 \LaTeX 이 수식을 조판하기 위한 처리 과정이 진행되지 않습니다. 텍스트 모드에서 코드를 문서로 조판하는 도중 \backslash (나 \backslash 를 만나면 수식 입력 모드로 전환됩니다. 수식 입력 모드에서는 수식을 작성하기 위한 명령어를 \LaTeX 이 이해할 수 있게 되며, 수식에 입력한 기호와 각종 수식 구성 요소 사이의 간격을 올바르게 조정하는 과정을 거칩니다. 이후 \backslash 나 \backslash 가 등장하면 수식 입력 모드에서 다시 텍스트 모드로 전환됩니다. 이러한 과정을 통해 수식이 올바르게 조판된 문서를 얻게 됩니다.

\LaTeX 으로 조판하는 수식은 행중수식과 별행수식으로 구분됩니다. 행중수식은 글줄에 섞여 나타나는 비교적 짧은 수식으로, $\backslash(\dots\backslash)$ 명령어 또는 `math` 환경을 사용해 입력합니다. 별행수식은 본문과 분리된 별도의 글줄에 작성되는 수식으로, $\backslash[\dots\backslash]$ 명령어 또는 `displaymath` 환경을 사용해 입력합니다. (행중수식과 별행수식의 예시는 제3장에 이미 제시했습니다.) 한편, 별행수식에는 다른 수식을 참조하기 위해 번호를 붙일 수도 있습니다. 이때에는 `equation` 환경을 사용합니다.

```
1 다음 식 \eqref{euler}\을 오일러 항등식이라고 부른다.
2 \begin{equation}
3   e^{i\pi}+1 = 0 \label{euler}
4 \end{equation}
```

다음 식 (1)을 오일러 항등식이라고 부른다.

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

이 환경을 사용하면 수식의 번호가 자동으로 계산되어 오른쪽 끝에 조판됩니다. 위 코드에서 확인할 수 있듯, `\label` 제어 문자열과 `\eqref` 제어 문자열을 사용하면 수식을 번호를 사용해 편리하게 참조할 수도 있습니다. 수식에 번호를 붙이는 방법은 제12장에서, 수식을 번호로 참조하는 방법은 제17장에서 자세히 알아봅니다.

수식 입력 모드의 특징

이제, 수식 입력 모드의 특징을 몇 가지 알아보겠습니다. 먼저, 수식 입력 모드에서는 기호 사이의 간격이 자동으로 조절되고, 코드에 입력된 공백 문자는 무시됩니다. 즉, 소스 파일에 공백 문자를 입력하더라도 수식의 기호 사이가 벌어지지 않습니다. 예를 들어, 소스 파일에 `\(a + b = c\)`를 입력했다든 `\(a+b=c\)`를 입력했다든, 컴파일하면 같은 결과 $a + b = c$ 를 얻습니다. 코드를 알아보기 좋게 적절히 공백 문자를 입력해도 되고, 소스 파일의 용량을 절약하기 위해 명령어를 최대한 붙여 써도 됩니다.

만약 기호 사이의 간격을 직접 조정해야 한다면 표 10.1의 제어 문자열을 사용할 수 있습니다. 예를 들어, 적분식 $\int f(x) dx$ 에서 피적분함수 $f(x)$ 와 적분변수 dx 사이에는 `\,`를 입력해 약간의 간격을 넣는 것이 좋습니다.

표 10.1: 기호 사이의 간격을 조정하는 제어 문자열

제어 문자열	간격의 폭	제어 문자열	간격의 폭
<code>\thinspace</code> 또는 <code>\,</code>	3 mu	<code>\negthinspace</code> 또는 <code>\!</code>	-3 mu
<code>\medspace</code> 또는 <code>;</code>	4 mu	<code>\negmedspace</code>	-4 mu
<code>\thickspace</code> 또는 <code>\:</code> 또는 <code>\></code>	5 mu	<code>\negthickspace</code>	-5 mu

1 mu(math unit)는 대략 대문자 M의 폭(1 em)의 1/18을 나타냅니다.

```
1 Compare \(\int f(x) dx\) % bad
2 and \(\int f(x) \,dx\). % good
```

Compare $\int f(x)dx$ and $\int f(x) dx$.

한편, 별행수식에서 한 줄에 여러 수식을 나열할 때에는 `\quad`나 `\qquad`를 사용해 각 수식을 구분할 수 있습니다. `\quad`는 1 em, `\qquad`는 2 em의 간격을 넣습니다. 예를 들어, 다음과 같이 사용할 수 있습니다.

```
\[ u_t = ku_{tt}, \qquad u(x,0) = \phi(x) ]
```

$$u_t = ku_{tt}, \quad u(x, 0) = \phi(x)$$

두 번째로, 수식 입력 모드 안에서는 문단을 나눌 수 없습니다. 만약 수식 입력 모드에 빈 줄이 입력되면 컴파일 과정에서 오류가 발생합니다. 별행수식을 입력할 때에는 `\\`로 줄을 바꿀 수도 없으며, 여러 줄에 걸쳐 별행수식을 입력할 때에는 제12장에서 설명하는 환경을 사용해야 합니다.


```

1 \[ f(z) \cdot \mathrm{Ind}_{\gamma}(z)
2   = \frac{1}{2\pi i} \int_{\gamma} \frac{f(\xi)}{\xi - z} d\xi
3   \]
4 여기서 \(\mathrm{Ind}_{\gamma}(z)\)는 폐경로
   \(\gamma\)의 \(z\) 주위에서의 감은 수를 나타낸다.

```

$$f(z) \cdot \mathrm{Ind}_{\gamma}(z) = \frac{1}{2\pi i} \int_{\gamma} \frac{f(\xi)}{\xi - z} d\xi$$

여기에서 $\mathrm{Ind}_{\gamma}(z)$ 는 폐경로 γ 의 z 주위에서의 감은 수를 나타낸다.

세 번째로, 행중수식은 글줄 위아래의 공간이 좁으므로, 몇 가지 기호가 별행수식에서보다 더 작게 표시됩니다. 대표적으로, 분수, 연산자와 그 제한 범위, 적분 기호가 다음과 같이 다르게 표시됩니다.

- 연산자의 위아래에 있던 첨자는 $\lim_{n \rightarrow \infty}$ 처럼 옆으로 옮겨지고,
- 분수와 이항 계수, \int, \cap, \sum 등의 적분 기호와 n 항 연산자가 작아집니다.

다음 예시에서 두 수식의 차이를 확인할 수 있습니다. 여기에서 `\mathtest`에는 다음의 코드가 저장되어 있습니다.

```

1 \lim_{n \rightarrow \infty} \frac{1}{n}, \quad \bigcap_{i=1}^{\infty} A_i, \quad \sum_{n=-\infty}^{\infty} \hat{f}(n)e^{inx}, \quad \int_a^b f(x) dx
2 \sum_{n=-\infty}^{\infty} \hat{f}(n)e^{inx}, \quad \int_a^b f(x) dx

```

```

1 \centering
2 \(\mathtest\)
3 \[ \mathtest \]

```

$$\lim_{n \rightarrow \infty} \frac{1}{n}, \quad \bigcap_{i=1}^{\infty} A_i, \quad \sum_{n=-\infty}^{\infty} \hat{f}(n)e^{inx}, \quad \int_a^b f(x) dx$$

$$\lim_{n \rightarrow \infty} \frac{1}{n}, \quad \bigcap_{i=1}^{\infty} A_i, \quad \sum_{n=-\infty}^{\infty} \hat{f}(n)e^{inx}, \quad \int_a^b f(x) dx$$

행중수식 입력 모드에서 `\displaystyle` 제어 문자열을 사용하면 수식의 크기를 별행수식처럼 바꿀 수 있습니다. 반대로, 별행수식 입력 모드에서 `\textstyle` 제어 문자열을 사용하면 수식의 크기를 행중수식처럼 바꿀 수 있습니다.

```

1 \centering
2 \(\displaystyle \mathtest\)
3 \[ \textstyle \mathtest \]

```

$$\lim_{n \rightarrow \infty} \frac{1}{n}, \quad \bigcap_{i=1}^{\infty} A_i, \quad \sum_{n=-\infty}^{\infty} \hat{f}(n)e^{inx}, \quad \int_a^b f(x) dx$$

$$\lim_{n \rightarrow \infty} \frac{1}{n}, \quad \bigcap_{i=1}^{\infty} A_i, \quad \sum_{n=-\infty}^{\infty} \hat{f}(n)e^{inx}, \quad \int_a^b f(x) dx$$

`\displaystyle`, `\textstyle` 외에도 첨자처럼 조판하는 `\scriptstyle` 제어 문자열과, 이중 첨자처럼 조판하는 `\scriptscriptstyle` 제어 문자열도 사용할 수 있습니다.

마지막으로, 수식 입력 도중 수식이 아닌 일반 텍스트를 입력해야 한다면 `amsmath` 패키지의 `\text` 제어 문자열을 사용합니다. 인자로 텍스트 모드에서 조판할 내용을 입력하면 됩니다.

```

1 \(X = \{x \mid \text{\(x\) is a prime}\}\)
2 \[f(x)=\cos x \quad \text{and} \quad g(x)=\sin x\]

```

$$X = \{x \mid x \text{ is a prime}\}$$

$$f(x) = \cos x \quad \text{and} \quad g(x) = \sin x$$

수식 조판에 사용되는 패키지

수식을 조판할 때에는 \LaTeX 의 기본 기능뿐 아니라 `amsmath`와 `amssymb` 패키지 등 여러 패키지를 함께 사용합니다. 첫 번째 패키지는 수식을 작성할 때 사용하는 많은 편의 기능을 제공하며, 두 번째 패키지는 많은 기호와 몇 가지 추가 글꼴을 제공합니다. `mathtools` 패키지도 자주 사용되며, 이는 `amsmath` 패키지의 기능을 확장하고 더 미려한 수식을 조판할 수 있도록 합니다. `mathtools` 패키지는 `amsmath` 패키지를 불러오므로, 다음 코드를 전처리부에 입력하면 세 패키지의 기능을 모두 사용할 수 있습니다.

```
\usepackage{mathtools,amssymb}
```

달러 기호에 관한 조언

\LaTeX 이 아닌 플레인 \TeX 에서는 달러 기호 `$`로 수식 모드와 텍스트 모드를 전환합니다. `$...$`은 행중 수식을, `$$...$$`은 별행수식을 조판하는 코드입니다. \LaTeX 에서는 달러 기호를 사용하는 것보다는 앞에서 설명한 방법으로 수식을 조판하는 것이 바람직합니다. `amsmath` 등의 패키지에서는 수식이 더 보기 좋게 조판되도록 간격을 조정하는 등의 작업을 하는데, 플레인 \TeX 의 코드를 사용하면 이 사항이 적용되지 않기 때문입니다. 또, 시작 부분과 끝 부분의 기호가 구분되어 있으므로 코드 상에서 어디부터 어디까지가 수식 입력 모드인지 쉽게 확인할 수 있습니다.

10.2

기본 문법

이번 절에서는 분수와 첨자, 근호 등 기본적인 수식 구성 요소를 입력하는 방법을 설명합니다.

분수와 이항 계수

분수를 입력할 때에는 `\frac` 제어 문자열을 사용합니다. 분자와 분모를 각각 `\frac`의 인자로 지정하면 됩니다.

```
\frac{⟨numerator⟩}{⟨denominator⟩}
```

⟨numerator⟩: 분자

⟨denominator⟩: 분모

예를 들어, `\(\frac{1}{\pi}\)`를 입력하면 $\frac{1}{\pi}$ 이 조판됩니다.

예제 10.1 $\lim_{h \rightarrow 0}$ 은 `\lim_{h \to 0}`을 입력해 조판할 수 있다. 이를 사용하여

$$\frac{df}{dx}(c) = \lim_{h \rightarrow 0} \frac{f(c+h) - f(c)}{h}$$

를 조판하여라.

특별한 설정을 하지 않으면 행중수식의 분수는 별행수식의 분수보다 작게 조판됩니다. 수식의 종류에 관계없이 분수의 크기를 일정하게 조판하려면 `\frac` 대신 `\dfrac`이나 `\tfrac` 제어 문자열을

사용할 수 있습니다. 전자는 별행수식에서의 모습처럼 큰 분수를, 후자는 행중수식에서의 모습처럼 작은 분수를 조판합니다. `\dfrac`과 `\tfrac`은 `amsmath` 패키지를 불러와야 사용할 수 있습니다.

```
\[ f(x) = \frac{34}{\int_1^x g(t) + \tfrac{12}{\sin t} dt }
```

$$f(x) = \frac{3}{4} \int_1^x g(t) + \frac{1}{2} \sin t \, dt$$

한편, 분수의 분자 또는 분모로 분수를 중첩하여 입력하면 글자가 작아지므로, 연분수를 입력할 때에는 `\frac`을 사용하면 보기 좋지 않습니다. `\frac` 대신 `\cfrac`을 사용하십시오.

```
1 \sqrt{2}
2 = 1 + \cfrac{1}{2 + \cfrac{1}{2 +
   \cfrac{1}{2 + \cdots}}}
3 = 1 + \frac{1}{2 + \frac{1}{2 + \frac{1}{2
   + \cdots}}}
```

$$\sqrt{2} = 1 + \frac{1}{2 + \frac{1}{2 + \frac{1}{2 + \cdots}}}$$

예제 10.2 위 예시를 참고하여 아래 등식을 조판하여라. (힌트: ϕ 는 `\phi`로 조판한다.)

$$\phi = \frac{1 + \sqrt{5}}{2} = \frac{1}{1 + \frac{1}{1 + \cdots}}$$

이항 계수는 `amsmath` 패키지의 `\binom` 제어 문자열을 사용해 `\frac`과 비슷한 방법으로 입력할 수 있습니다.

`\binom{⟨n⟩}{⟨r⟩}`

⟨n⟩: 위쪽에 들어갈 식
⟨r⟩: 아래쪽에 들어갈 식

예를 들어, $\binom{n}{r}$ 는 `\(\binom{n}{r}\)`를 입력해 조판합니다.

예제 10.3 `\binom`을 사용하여

$$\binom{n}{r} = \binom{n-1}{r-1} + \binom{n-1}{r}$$

을 조판하여라.

`\tfrac`, `\dfrac`과 마찬가지로, 이항 계수도 `\dbinom`과 `\tbinom` 제어 문자열을 사용해 수식 입력 모드와 관계없이 같은 크기의 결과를 얻을 수 있습니다. 두 제어 문자열도 `amsmath` 패키지를 불러와야 사용할 수 있습니다.

첨자

위 첨자와 아래 첨자는 각각 ^와 _로 입력합니다. 제어 문자열의 인자를 지정할 때와 마찬가지로, 문자 하나나 제어 문자열 하나를 인자로 지정할 때에는 중괄호를 생략할 수 있습니다.

$\wedge\{\langle superscript \rangle\}$	$_ \{\langle subscript \rangle\}$
$\langle superscript \rangle$: 위 첨자로 입력할 내용	$\langle subscript \rangle$: 아래 첨자로 입력할 내용

한 문자 뒤에 위 첨자와 아래 첨자를 동시에 입력할 수도 있으며, 이때 위 첨자와 아래 첨자 문법을 입력한 순서와 조판 결과는 무관합니다. 한편, 위 첨자를 조판할 때에는 캐럿 기호 ^ 대신 \sp를, 아래 첨자를 조판할 때에는 밑줄 문자 _ 대신 \sb를 사용할 수도 있습니다.

```
1 Let \(\Delta(i,j) = \Delta_i^j\). \\
2 Let \(\Delta(i,j) = \Delta_{\sb{i}}\sp{j}\).
```

```
Let  $\Delta(i,j) = \delta_i^j$ .
Let  $\Delta(i,j) = \delta_i^j$ .
```

첨자 명령어 앞의 수식을 중괄호로 묶으면 TeX은 이를 하나의 ‘덩어리’로 인식하여 조판합니다. 첨자를 여러 번 사용하는 경우 첨자가 조판되는 위치가 달라질 수 있습니다. 아래 예시를 참고하십시오.

```
1 Compare
2 \[ x_a^b, \quad x_a^a, \quad (x^1)^2, \quad (x_1)_2, \\
\quad (x_1)_2^2 \]
3 with
4 \[ \{x_a\}^b, \quad \{x_a\}_b, \quad \{x^1\}^2, \quad \{(x_1)\}_2. \]
```

Compare

$$x_a^b, \quad x_a^a, \quad (x^1)^2, \quad (x_1)_2$$

with

$$x_a^b, \quad x_a^a, \quad (x^1)^2, \quad (x_1)_2.$$

같은 종류의 첨자를 중첩해서 사용할 때에는 중괄호를 사용해야 합니다. 예를 들어, $\{a_{k_j}\}$ 와 2^{3^4} 를 조판하려면 $\{ \{ a_{k_j} \} \}$, $\{ 2^{\{ 3^4 \}} \}$ 를 입력해야 하며, 중괄호를 생략하면 컴파일 과정에서 오류가 발생합니다.

예제 10.4 수열 $\{u_n\}$ 은 피보나치 수열을 가리킨다고 하자. 아래의 식을 조판하여라.

$$u_0 = u_1 = 1, \quad u_{n+2} = u_n + u_{n+1}$$

프라임 기호를 입력할 때에는 l을 출력하는 제어 문자열 \prime을 위 첨자로 입력하거나, 간단히 따옴표 '를 입력하면 됩니다. 즉, ‘ $f'(x)$ ’는 $\{ f^{\prime}(x) \}$ 나 $\{ f'(x) \}$ 로 조판할 수 있습니다.

예제 10.5 프라임 기호에 유의하여 다음 수식을 조판하여라. 극한 연산자를 조판하는 방법은 예제 10.1을 참조하여라.

$$f''(x) = \lim_{y \rightarrow x} \frac{f'(y) - f'(x)}{y - x} = \lim_{h \rightarrow 0} \frac{f'(x+h) - f'(x)}{h}.$$

근호

거듭제곱근은 `\sqrt` 제어 문자열을 사용해 입력할 수 있습니다.

`\sqrt[⟨radexp⟩]{⟨radicand⟩}`

⟨radexp⟩: 근지수

⟨radicand⟩: 피제곱근수

예를 들어, `\sqrt[n]{x}`를 입력하고 컴파일하면 $\sqrt[n]{x}$ 가 조판됩니다. 선택 인자를 생략하여 `\sqrt{x}`만을 입력하고 컴파일하면 \sqrt{x} 가 조판됩니다.

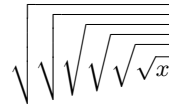
예제 10.6 x 에 대한 이차방정식 $ax^2 + bx + c = 0$ (단, $a \neq 0$)의 근의 공식

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

를 조판하여라. (힌트: \pm 은 `\pm`으로 입력할 수 있다.)

근호 안에 큰 수식이 입력되면 아래 예시에서 확인할 수 있듯 근호도 그에 맞춰 크기가 변합니다.

`\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{x}}}}}}`



예제 10.7 큰 괄호는 `\left`(와 `\right`)로 조판한다. 다음 수식을 작성하여라.

$$L = \int_a^b \sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2 + \left(\frac{dz}{dt}\right)^2} dt$$

플레인 TeX의 문법에 관한 조언

참고 `\over`나 `\choose`, `\root`를 모르는 분은 이 부분을 넘어가셔도 됩니다.

\LaTeX 이 아닌 플레인 TeX에서는 분수를 입력할 때 `\over`를, 이항 계수를 입력할 때 `\choose`를, 임의의 근지수에 대한 근호를 입력할 때 `\root`와 `\of`를 사용합니다. 이들은 일반적인 \LaTeX 의 제어 문자열과 인자를 지정하는 방법이 달라 오류가 발생하기 쉽습니다. 플레인 TeX의 문법을 알고 있더라도 \LaTeX 으로 문서를 작성할 때에는 `\frac`, `\binom`, `\sqrt`를 사용하시기 바랍니다.

- 1 Use `\frac{a}{b}` instead of `(a \over b)`. `\frac`
- 2 Use `\binom{n}{r}` instead of `(n \choose r)`. `\binom`
- 3 Use `\sqrt[n]{x}` instead of `(\root n \of x)`.

Use $\frac{a}{b}$ instead of $\frac{a}{b}$.
Use $\binom{n}{r}$ instead of $\binom{n}{r}$.
Use $\sqrt[n]{x}$ instead of $\sqrt[n]{x}$.

10.3

기호 입력하기

이번 절에서는 수식을 입력할 때 사용하는 각종 기호를 입력하는 제어 문자열을 소개하고, 몇 가지 주의사항을 설명합니다. \LaTeX 은 수많은 기호를 입력하는 제어 문자열을 기본적으로 제공하며, `amssymb` 패키지와 `mathtools` 패키지를 불러오면 더 다양한 기호를 입력할 수 있습니다. 여기에 정리된 기호 일람표는 [6]을 따라 분류하여, (1) 이항 연산자, (2) 가변 크기 연산자 및 로마자로 적는 연산자와 함수, (3) 이항 관계 기호, (4) 그리스 및 히브리 문자와 글자 모양 기호, (5) 괄호, (6) 문장 부호, (7) 기타 기호의 순서로 정리되어 있습니다.

이항 연산자

‘이항 연산자(binary operator)’는 $+$ 나 \times , \cup 처럼 두 수학적 대상 사이의 연산을 나타내는 기호입니다. \LaTeX 에서 기본 제공하는 이항 연산자는 표 10.2에, `amssymb` 패키지에서 추가로 제공하는 이항 연산자는 표 10.3에 정리하였습니다. 세 기호 $+$, $-$, $*$ 는 키보드의 대응하는 키 $+$, $-$, $*$ 를 눌러 입력합니다.

표 10.2: 이항 연산자

\amalg	\cup	\oplus	\times
$*$	\dagger	\oslash	\triangleleft
\bigcirc	\ddagger	\otimes	\triangleright
\bigtriangledown	\diamond	\pm	\leqslant
\bigtriangleup	\div	\rhd ¹	\geqslant
\bullet	\lhd ¹	\setminus	\uplus
\cap	\mp	\sqcap	\vee ²
\cdot	\odot	\sqcup	\wedge ²
\circ	\ominus	\star	\wr

¹ `latexsym` 또는 `amssymb` 패키지를 불러와야 사용할 수 있습니다.

² \vee 와 \wedge 는 각각 `\lor`와 `\land`로도 입력할 수 있습니다.

표 10.3: `amssymb` 패키지의 이항 연산자

$\bar{\wedge}$	\circledcirc	\intercal
\boxdot	\circledR	\leftthreetimes
\boxminus	\Cup	\ltimes
\boxplus	\curlyvee	\rightthreetimes
\boxtimes	\curlywedge	\rtimes
\Cap	\divideontimes	\smallsetminus
\centerdot	\dotplus	\veebar
\circledast	\doublebarwedge	

가변 크기 연산자와 로마자로 적는 연산자 및 함수

‘가변 크기 연산자(variable-sized operator)’는 \sum 이나 \prod 등의 n 항 연산자와 \int 등의 적분 기호로 이루어집니다. 이때 n 항 연산자는 임의의 수의 수학적 대상 사이의 연산을 나타내는 기호를 뜻합니다. 이들은 행중수식과 별행수식에서 조판했을 때의 크기가 다릅니다. \LaTeX 에서 기본 제공하는 가변 크기 연산자는 표 10.4에, `amsmath` 패키지에서 추가로 제공하는 가변 크기 연산자는 표 10.5에 정리하였습니다.

표 10.4: 가변 크기 연산자

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

표 10.5: `amsmath` 패키지에서 제공하는 가변 크기 연산자

\iint	<code>\iint</code>	\iiint	<code>\iiint</code>	\iiint	<code>\iiint</code>	$\int \dots \int$	<code>\int \dots \int</code>
---------	--------------------	----------	---------------------	----------	---------------------	-------------------	------------------------------

지수함수나 로그함수, 삼각함수, 극한 연산자 등은 로마자 정체로 적습니다. 이때에는 표 10.6의 제어 문자열을 사용해서 입력합니다. `amsmath` 패키지를 불러오면 표 10.7의 연산자를 추가로 입력할 수 있습니다.

표 10.6: 로마자 정체로 쓰는 연산자와 함수

<code>arccos</code>	<code>\arccos</code>	<code>csc</code>	<code>\csc</code>	<code>ker</code>	<code>\ker</code>	<code>min</code>	<code>\min¹</code>
<code>arcsin</code>	<code>\arcsin</code>	<code>deg</code>	<code>\deg</code>	<code>lg</code>	<code>\lg</code>	<code>Pr</code>	<code>\Pr¹</code>
<code>arctan</code>	<code>\arctan</code>	<code>det</code>	<code>\det¹</code>	<code>lim</code>	<code>\lim¹</code>	<code>sec</code>	<code>\sec</code>
<code>arg</code>	<code>\arg</code>	<code>dim</code>	<code>\dim</code>	<code>lim inf</code>	<code>\liminf¹</code>	<code>sin</code>	<code>\sin</code>
<code>cos</code>	<code>\cos</code>	<code>exp</code>	<code>\exp</code>	<code>lim sup</code>	<code>\limsup¹</code>	<code>sinh</code>	<code>\sinh</code>
<code>cosh</code>	<code>\cosh</code>	<code>gcd</code>	<code>\gcd¹</code>	<code>ln</code>	<code>\ln</code>	<code>sup</code>	<code>\sup¹</code>
<code>cot</code>	<code>\cot</code>	<code>hom</code>	<code>\hom</code>	<code>log</code>	<code>\log</code>	<code>tan</code>	<code>\tan</code>
<code>coth</code>	<code>\coth</code>	<code>inf</code>	<code>\inf¹</code>	<code>max</code>	<code>\max¹</code>	<code>tanh</code>	<code>\tanh</code>

¹ 이들은 연산자로 취급되어, 별행수식 입력 모드에서의 첨자는 기호의 위아래에 조판됩니다.

표 10.7: `amsmath` 패키지의 로마자 정체로 쓰는 연산자

<code>inj lim</code>	<code>\injlim</code>	\varinjlim	<code>\varinjlim</code>	\varinjlim	<code>\varinjlim</code>
<code>proj lim</code>	<code>\projlim</code>	\varprojlim	<code>\varprojlim</code>	\varprojlim	<code>\varprojlim</code>

연산자의 계산 범위는 첨자 문법을 사용해 입력합니다. 위 첨자와 아래 첨자를 사용하면 이들이 올바른 위치에 조판됩니다. n 항 연산자나 로마자 정체로 적는 연산자는 기호의 위아래에 조판되고, 적분 연산자나 로마자 정체로 적는 함수는 기호의 오른쪽 위와 오른쪽 아래에 조판됩니다. 적분식을 작성할 때에는 피적분함수와 적분변수 사이에 \backslash 를 입력해 주는 것이 좋습니다.

```
1 \[ \sum_{k=1}^n a_k = a_1+\cdots+a_n \]
2 \[ \int_A^B f(z) \, dz = F(B) - F(A) \]
3 \[ \lim_{x \rightarrow 0} \frac{\sin x}{x} = 1 \]
```

$$\sum_{k=1}^n a_k = a_1 + \cdots + a_n$$

$$\int_A^B f(z) dz = F(B) - F(A)$$

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

예제 10.8 로그함수는 $\backslash \log$ 로 입력하고, 큰 괄호는 $\backslash biggl$ (와 $\backslash biggr$)로 입력한다. 이를 참고하여 다음 식

$$\log \left(\prod_{j=1}^k a_j \right) = \sum_{j=1}^k \log a_j$$

를 행중수식과 별행수식으로 각각 조판하고, 모양이 어떻게 다른지 비교하여라.

예제 10.9 아래의 수식을 조판하여라.

$$\int_a^c f(x) dx + \int_c^b f(x) dx = \int_a^b f(x) dx$$

앞과 마찬가지로, 행중수식과 별행수식에서 모양이 어떻게 달라지는지 비교하여라.

예제 10.10 다음 수식을 조판하여라. (힌트: ∞ 는 $\backslash infty$ 로 입력한다.)

$$\limsup_{n \rightarrow \infty} x_n = \lim_{m \rightarrow \infty} \sup_{n \geq m} x_n, \quad \liminf_{n \rightarrow \infty} x_n = \lim_{m \rightarrow \infty} \inf_{n \geq m} x_n$$

이번에도, 행중수식과 별행수식에서 수식이 어떻게 다르게 조판되는지 비교하여라.

n 항 연산자에는 각각에 대응하는 같은 모양의 이항 연산자가 있습니다. 이항 연산자는 n 항 연산자에 비해 더 작으므로, 두 대상 사이의 연산을 나타내는 경우가 아니라면 이 표의 기호를 사용하는 것이 좋습니다. 예를 들어, $A \cup B$ 에서는 이항 연산자인 $\backslash cup$ 을 사용하는 것이 맞고, $\bigcup_{i=1}^{\infty} A_i$ 에서는 n 항 연산자인 $\backslash bigcup$ 을 사용하는 것이 맞습니다. 특히, 별행수식에서는 둘의 조판 결과가 두드러지게 달라집니다.

$$V = \bigcup_{i=1}^n V_i \quad \text{vs} \quad V = \bigcup_{i=1}^n V_i \quad \text{vs} \quad V = \bigcup_{i=1}^n V_i$$

첫 번째 식은 이항 연산자 $\backslash cup$ 을, 두 번째와 세 번째 식은 n 항 연산자 $\backslash bigcup$ 을 사용해 조판하였습니다. 이때 두 번째 결과는 행중수식에서의 모습, 세 번째 결과는 별행수식에서의 모습입니다. 상황에 맞는 기호를 사용하여 올바른 수식을 작성하시기 바랍니다.

예제 10.11 기호의 크기에 유의하여 다음을 조판하여라. (힌트: \in 은 `\in`으로, \cdots 는 `\cdots`로 입력한다.)

군 G 의 각 부분군 G_α 가 주어져 있을 때, G 의 원소 x 가 G_α 의 원소 x_α 의 유한합으로 유일하게 결정되면, 이를 $G = \bigoplus G_\alpha$ 로 나타낸다. 인덱스 α 의 집합이 J 로 주어져 있으면

$$G = \bigoplus_{\alpha \in J} G_\alpha$$

라고 적는다. 만약 인덱스 α 의 집합이 유한하면 $G = G_1 \oplus \cdots \oplus G_n$ 으로 쓸 수 있다.

법 연산자 `mod`는 `\bmod`와 `\pmod`로 입력합니다. 전자는 이항 연산자처럼 사용되며, 후자는 인자를 하나 입력받아 이를 괄호 안에 조판합니다. 한편, `amsmath` 패키지는 `\pmod`의 변형인 `\mod`와 `\pod` 제어 문자열을 정의합니다. `\pmod`와 마찬가지로, `\mod`와 `\pod`도 인자를 하나 입력받습니다. 각 제어 문자열을 사용해 법 연산자를 조판한 예시는 아래를 참조하십시오.

```
1 \[ 5 \bmod 2 \equiv 1 \]
2 \[ 5 \equiv 1 \pmod{2} \]
3 \[ 5 \equiv 1 \mod{2} \]
4 \[ \qquad 5 \equiv 1 \pod{2} \]
```

$$\begin{aligned} 5 \bmod 2 &\equiv 1 \\ 5 &\equiv 1 \pmod{2} \\ 5 &\equiv 1 \mod 2 \\ 5 &\equiv 1 \pod{2} \end{aligned}$$

예제 10.12 법 연산자에 유의하여 다음을 조판하여라. (힌트: \equiv 는 `\equiv`로 입력한다.)

소수 p 에 대해 $0 < i < p$ 이면 $\binom{p}{i} \equiv 0 \pmod{p}$ 이므로, 표수가 p 의 거듭제곱인 체의 다항식환에서는

$$x^p - a^p = (x - a)^p$$

가 성립한다.

연산자와 적분 기호, 함수를 조판하는 방법은 다음 장의 11.1절에서 더 자세히 다룹니다. 특히, 표 10.6와 표 10.7에 없는 연산자나 함수를 입력하는 방법이나 연산자의 계산 범위를 조판하는 더 자세한 방법을 설명하고 있습니다. 기본 정의되어 있지 않은 연산자를 조판해야 하거나 제한 범위를 여러 줄로 입력해야 하는 등, 더 다양한 식을 조판해야 한다면 해당 부분을 참조하시기 바랍니다.

이항 관계 기호

‘이항 관계 기호(binary relation)’는 기호 앞뒤에 있는 두 수학적 대상 사이의 관계를 나타냅니다. \LaTeX 에서 기본 제공하는 관계 기호는 표 10.8에, `amssymb` 패키지에서 추가로 제공하는 관계 기호는 표 10.9에 정리하였습니다. 등호와 쌍점은 각각 키보드의 대응하는 키 `=`, `:`를 눌러 입력하면 됩니다. 한편, \TeX 은 이 표의 `\mid`를 입력해 조판한 세로줄과 키보드의 `|` 키를 눌러 조판한 세로줄을 다르게 취급합니다. 자세한 내용은 83쪽의 설명을 참조하십시오.

표 10.8: 이항 관계 기호

\approx	<code>\approx</code>	\equiv	<code>\equiv</code>	\perp	<code>\perp</code>	\smile	<code>\smile</code>
\asymp	<code>\asymp</code>	\frown	<code>\frown</code>	\prec	<code>\prec</code>	\succ	<code>\succ</code>
\bowtie	<code>\bowtie</code>	\Join^1	<code>\Join^1</code>	\preceq	<code>\preceq</code>	\succeq	<code>\succeq</code>
\cong	<code>\cong</code>	\mid	<code>\mid</code>	\propto	<code>\propto</code>	\vdash	<code>\vdash</code>
\dashv	<code>\dashv</code>	\models	<code>\models</code>	\sim	<code>\sim</code>		
\doteq	<code>\doteq</code>	\parallel	<code>\parallel</code>	\simeq	<code>\simeq</code>		

¹ latexsym 또는 amssymb 패키지를 불러와야 사용할 수 있습니다.

표 10.9: amssymb 패키지의 이항 관계 기호

\approx	<code>\approxeq</code>	\curlyeqprec	<code>\curlyeqprec</code>	\prec	<code>\prec</code>	\therefore	<code>\therefore</code>
\backsimeq	<code>\backsimeq</code>	\curlyeqsucc	<code>\curlyeqsucc</code>	\risingdotseq	<code>\risingdotseq</code>	\thickapprox	<code>\thickapprox</code>
\backsim	<code>\backsim</code>	\doteqdot	<code>\doteqdot</code>	\shortmid	<code>\shortmid</code>	\thicksim	<code>\thicksim</code>
\backsimeq	<code>\backsimeq</code>	\eqcirc	<code>\eqcirc</code>	\shortparallel	<code>\shortparallel</code>	\varpropto	<code>\varpropto</code>
\because	<code>\because</code>	\fallingdotseq	<code>\fallingdotseq</code>	\smallfrown	<code>\smallfrown</code>	\Vdash	<code>\Vdash</code>
\between	<code>\between</code>	\multimap	<code>\multimap</code>	\smallsmile	<code>\smallsmile</code>	\Vdash	<code>\Vdash</code>
\Bumpeq	<code>\Bumpeq</code>	\pitchfork	<code>\pitchfork</code>	\succapprox	<code>\succapprox</code>	\Vdash	<code>\Vdash</code>
\bumpeq	<code>\bumpeq</code>	\precapprox	<code>\precapprox</code>	\succcurlyeq	<code>\succcurlyeq</code>		
\circeq	<code>\circeq</code>	\preccurlyeq	<code>\preccurlyeq</code>	\succsim	<code>\succsim</code>		

일부 기호는 앞에 `\not`을 붙이면 부정 기호가 됩니다. 예를 들어, \equiv 는 `\equiv`를 입력해 얻을 수 있으며, 이 앞에 `\not`을 붙여 `\not\equiv`를 입력하면 $\not\equiv$ 를 얻습니다. 부정 기호 중 일부는 amssymb 패키지에서 제공하기도 하며, 이때에는 원래 기호를 입력하는 제어 문자열 앞에 'n'을 입력하면 됩니다. 예를 들어, \cong 는 `\cong`를 입력해 얻을 수 있으며, `\ncong`를 입력하면 $\not\cong$ 를 얻습니다. amssymb 패키지에서 제공하는 부정 이항 관계 기호를 표 10.10에 정리하였습니다.

표 10.10: amssymb 패키지의 부정 이항 관계 기호

\ncong	<code>\ncong</code>	\nshortmid	<code>\nshortmid</code>	\nvDash	<code>\nvDash</code>	\succapprox	<code>\succapprox</code>
\nmid	<code>\nmid</code>	\nshortparallel	<code>\nshortparallel</code>	\nvdash	<code>\nvdash</code>	\succsim	<code>\succsim</code>
\nparallel	<code>\nparallel</code>	\nsim	<code>\nsim</code>	\nVDash	<code>\nVDash</code>		
\nprec	<code>\nprec</code>	\nsucc	<code>\nsucc</code>	\precapprox	<code>\precapprox</code>		
\npreceq	<code>\npreceq</code>	\nsucceq	<code>\nsucceq</code>	\precnsim	<code>\precnsim</code>		

수식 글꼴의 쌍점은 등호 등 다른 기호에 비해 아래쪽으로 치우친 형태로 디자인되어 있어, `\coloneq`를 입력하고 컴파일 하면 보기 좋지 않은 결과를 얻습니다. `\coloneq` 대신 mathtools 패키지에서 제공하는 `\coloneq` 제어 문자열을 입력하면 쌍점의 위치가 조정되어 더 보기 좋은 기호를 조판할 수 있습니다. 아래 예시를 비교해 보십시오.

$$g := f|_A \text{ (:=)} \quad \text{vs} \quad g := f|_A \text{ (\coloneq)}$$

mathtools 패키지에서 제공하는 모든 기호의 목록은 표 10.11에서 확인할 수 있습니다. 이 표에 있는 `\vcentcolon`은 세로 위치가 조정된 쌍점을 조판합니다. 키보드의 ⋮ 키를 눌러 입력한 쌍점보다 더 위쪽에 조판되므로, $=$, $-$, \approx , \sim 등의 기호와 더 잘 어울립니다.

표 10.11: mathtools 패키지의 이항 관계 기호

$\approx::$	<code>\Approxcolon</code>	\coloneq	<code>\colondash</code>	\dashcolon	<code>\Dashcolon</code>	$\sim::$	<code>\Simcolon</code>
\approx	<code>\approxcolon</code>	\coloneq	<code>\Coloneq</code>	\dashcolon	<code>\dashcolon</code>	\sim	<code>\simcolon</code>
\colonapprox	<code>\Colonapprox</code>	\coloneq	<code>\coloneq</code>	\dblcolon	<code>\dblcolon</code>	\vcentcolon	<code>\vcentcolon</code>
\colonapprox	<code>\colonapprox</code>	\colonsim	<code>\Colonsim</code>	\Eqcolon	<code>\Eqcolon</code>		
\colondash	<code>\Colondash</code>	\colonsim	<code>\colonsim</code>	\eqcolon	<code>\eqcolon</code>		

몇 가지 종류의 이항 관계 기호는 비슷한 것끼리 묶어 따로 정리하였습니다. 여기에는 집합 관계 기호, 부등호, 삼각형 모양 관계 기호, 화살표가 있습니다.

집합 관계 기호 집합 사이의 포함 관계는 표 10.12의 기호로 입력합니다. amssymb 패키지를 불러오면 표 10.13의 기호도 사용할 수 있습니다.

표 10.12: 집합 관계 기호

\sqsubset	<code>\sqsubset¹</code>	\sqsubset	<code>\sqsubset¹</code>	\subset	<code>\subset</code>	\supset	<code>\supset</code>
\sqsubseteq	<code>\sqsubseteq</code>	\sqsupseteq	<code>\sqsupseteq</code>	\subseteq	<code>\subseteq</code>	\supseteq	<code>\supseteq</code>

¹ latexsym 또는 amssymb 패키지를 불러와야 사용할 수 있습니다.

표 10.13: amssymb 패키지의 집합 관계 기호

\nsupseteq	<code>\nsupseteq</code>	\Subset	<code>\Subset</code>	\Supset	<code>\Supset</code>	\varsubsetneq	<code>\varsubsetneq</code>
\nsupseteq	<code>\nsupseteq</code>	\subseteq	<code>\subseteq</code>	\supseteq	<code>\supseteq</code>	\varsubsetneqq	<code>\varsubsetneqq</code>
\nsupseteq	<code>\nsupseteq</code>	\subsetneq	<code>\subsetneq</code>	\supsetneq	<code>\supsetneq</code>	\varsupsetneq	<code>\varsupsetneq</code>
\nsupseteq	<code>\nsupseteq</code>	\subsetneqq	<code>\subsetneqq</code>	\supsetneqq	<code>\supsetneqq</code>	\varsupsetneqq	<code>\varsupsetneqq</code>

부등호 부등호는 표 10.14의 제어 문자열로 입력합니다. 기호 $<$, $>$ 는 키보드의 대응하는 키 $<$, $>$ 를 눌러 입력합니다. amssymb 패키지는 표 10.15의 기호를 제공합니다.

표 10.14: 부등호

\geq	<code>\geq¹</code>	\gg	<code>\gg</code>	\leq	<code>\leq¹</code>	\ll	<code>\ll</code>	\neq	<code>\neq¹</code>
--------	-------------------------------	-------	------------------	--------	-------------------------------	-------	------------------	--------	-------------------------------

¹ \geq , \leq , \neq 는 각각 `\ge`, `\le`, `\ne`로도 입력할 수 있습니다.

표 10.15: amssymb 패키지의 부등호

\gg	<code>\eqslantgtr</code>	\gt	<code>\gtrdot</code>	\lesseqgtr	\ncong	<code>\ngeq</code>
\leqslant	<code>\eqslantless</code>	\gtrless	<code>\gtreqless</code>	\lesseqqgtr	\ncong	<code>\ngeqq</code>
\geq	<code>\geqq</code>	\gtrless	<code>\gtreqqless</code>	\lessgtr	\ncong	<code>\ngeqslant</code>
\gtrsim	<code>\geqslant</code>	\gtrless	<code>\gtrless</code>	\lessssim	\ncong	<code>\ngtr</code>
\ggg	<code>\ggg</code>	\gtrsim	<code>\gtrsim</code>	\lll	\ncong	<code>\nleq</code>
\approx	<code>\napprox</code>	\gtrsim	<code>\gtrsim</code>	\lnapprox	\ncong	<code>\nleqq</code>
\gneq	<code>\gneq</code>	\leqq	<code>\leqq</code>	\lneq	\ncong	<code>\nleqslant</code>
\gneqq	<code>\gneqq</code>	\leqslant	<code>\leqslant</code>	\lneqq	\ncong	<code>\nless</code>
\gnsim	<code>\gnsim</code>	\lessapprox	<code>\lessapprox</code>	\lnsim		
\gtrapprox	<code>\gtrapprox</code>	\lessdot	<code>\lessdot</code>	\lvertneqq		

삼각형 모양 관계 기호 다음 표 10.16에는 관계 기호 중 삼각형이 포함된 기호들을 정리하였습니다. 여기의 기호는 amssymb 패키지를 불러와야 사용할 수 있습니다.

표 10.16: amssymb 패키지의 삼각형 모양 관계 기호

\blacktriangleleft	<code>\blacktriangleleft</code>	\blacktriangleright	<code>\blacktriangleright</code>	\trianglelefteq	<code>\trianglelefteq</code>
\blacktriangleright	<code>\blacktriangleright</code>	\blacktrianglelefteq	<code>\blacktrianglelefteq</code>	\vartriangleleft	<code>\vartriangleleft</code>
\ntriangleleft	<code>\ntriangleleft</code>	\ntrianglelefteq	<code>\ntrianglelefteq</code>	\vartriangleright	<code>\vartriangleright</code>
\ntrianglelefteq	<code>\ntrianglelefteq</code>	\trianglelefteq	<code>\trianglelefteq</code>		

이 표의 기호 중 몇 가지는 앞의 표 10.2와 같은 모양의 기호를 조판합니다.

\triangleleft	<code>\lhd</code>	vs	<code>\vartriangleleft</code>	\trianglelefteq	<code>\unlhd</code>	vs	<code>\trianglelefteq</code>
\triangleleft	<code>\rhd</code>	vs	<code>\vartriangleright</code>	\trianglerighteq	<code>\unrhd</code>	vs	<code>\trianglerighteq</code>

각 쌍에서 전자는 이항 연산자, 후자는 이항 관계 기호로 취급되어 기호 앞뒤의 공간이 다르게 조판됩니다. 차이가 미묘하긴 하지만 상황에 알맞은 기호를 입력하는 것이 좋습니다.

$$A \trianglelefteq B \quad (\text{\unlhd}) \quad \text{vs} \quad A \trianglelefteq B \quad (\text{\trianglelefteq})$$

위 예시에서 차이를 비교해 보십시오. 예리한 분이라면 오른쪽 식에서의 \trianglelefteq 기호의 앞뒤 공간이 미세하게 더 넓다는 것을 알아차릴 수 있을 것입니다.

화살표와 작살표 화살표와 작살표(harpoon)는 각각 표 10.17과 표 10.18의 제어 문자열로 입력합니다. 여기에서 작살표는 화살표와 비슷하게 생겼지만, 빗살이 한쪽에만 있는 기호를 가리킵니다.

명제 사이의 함의 관계 및 동치 관계를 나타낼 때에는 `\impliedby`, `\implies`, `\iff`를 사용할 수 있습니다. 이들은 이중선 긴 화살표, 즉 \Leftarrow , \Rightarrow , \Longleftrightarrow 를 조판하지만 화살표 앞뒤의 공간이 표 10.17의 제어 문자열을 사용해 조판한 화살표보다 더 넓습니다.

$$ab = ac \Longleftrightarrow b = c \quad (\text{\Longleftarrow})$$

$$\text{vs} \quad ab = ac \iff b = c \quad (\text{\iff})$$

표 10.17: 화살표

\Downarrow <code>\Downarrow</code>	\Longleftarrow <code>\Longleftarrow</code>	\nwarrow <code>\nwarrow</code>
\downarrow <code>\downarrow</code>	\longleftarrow <code>\longleftarrow</code>	\Rightarrow <code>\Rightarrow</code>
\hookleftarrow <code>\hookleftarrow</code>	\Longleftrightarrow <code>\Longleftrightarrow</code>	\rightarrow <code>\rightarrow</code> ²
\hookrightarrow <code>\hookrightarrow</code>	\longleftrightarrow <code>\longleftrightarrow</code>	\searrow <code>\searrow</code>
\leadsto ¹ <code>\leadsto</code>	\mapsto <code>\mapsto</code>	\swarrow <code>\swarrow</code>
\Leftarrow <code>\Leftarrow</code>	\Longrightarrow <code>\Longrightarrow</code>	\Uparrow <code>\Uparrow</code>
\leftarrow <code>\leftarrow</code> ²	\longrightarrow <code>\longrightarrow</code>	\uparrow <code>\uparrow</code>
\Leftrightarrow <code>\Leftrightarrow</code>	\mapsto <code>\mapsto</code>	\Updownarrow <code>\Updownarrow</code>
\leftrightsquigarrow <code>\leftrightsquigarrow</code>	\nearrow <code>\nearrow</code>	\updownarrow <code>\updownarrow</code>

¹ latexsym 패키지 또는 amssymb 패키지를 불러와야 사용할 수 있습니다.

² \rightarrow 는 `\to`로도, \leftarrow 는 `\gets`로도 입력할 수 있습니다.

표 10.18: 작살표

\leftharpoonup <code>\leftharpoonup</code>	\rightharpoonup <code>\rightharpoonup</code>	\rightleftharpoons <code>\rightleftharpoons</code>
\leftharpoonup <code>\leftharpoonup</code>	\rightharpoonup <code>\rightharpoonup</code>	

amssymb 패키지를 불러오면 표 10.19, 표 10.20, 표 10.21의 기호도 입력할 수 있습니다.

표 10.19: amssymb 패키지의 화살표

\circlearrowleft <code>\circlearrowleft</code>	\leftrightsquigarrow <code>\leftrightsquigarrow</code>	\rightleftarrows <code>\rightleftarrows</code>
\circlearrowright <code>\circlearrowright</code>	\leftrightsquigarrow <code>\leftrightsquigarrow</code>	\rightrightarrows <code>\rightrightarrows</code>
\curvearrowleft <code>\curvearrowleft</code>	\Lleftarrow <code>\Lleftarrow</code>	\rightsquigarrow <code>\rightsquigarrow</code>
\curvearrowright <code>\curvearrowright</code>	\Lrightarrow <code>\Lrightarrow</code>	\Rrightarrow <code>\Rrightarrow</code>
\dashleftarrow <code>\dashleftarrow</code>	\looparrowleft <code>\looparrowleft</code>	\twoheadleftarrow <code>\twoheadleftarrow</code>
\dashrightarrow <code>\dashrightarrow</code>	\looparrowright <code>\looparrowright</code>	\twoheadrightarrow <code>\twoheadrightarrow</code>
\downdownarrows <code>\downdownarrows</code>	\Lsh <code>\Lsh</code>	\upuparrows <code>\upuparrows</code>
\leftarrowtail <code>\leftarrowtail</code>	\rightarrowtail <code>\rightarrowtail</code>	

표 10.20: amssymb 패키지의 부정 화살표

\nLeftarrow <code>\nLeftarrow</code>	\nLeftrightarrow <code>\nLeftrightarrow</code>	\nRightarrow <code>\nRightarrow</code>
\nleftarrow <code>\nleftarrow</code>	\nleftrightarrow <code>\nleftrightarrow</code>	\nrightarrow <code>\nrightarrow</code>

표 10.21: amssymb 패키지의 작살표

\downharpoonleft <code>\downharpoonleft</code>	\leftrightharpoons <code>\leftrightharpoons</code>	\upharpoonright <code>\upharpoonright</code>
\downharpoonright <code>\downharpoonright</code>	\upharpoonleft <code>\upharpoonleft</code>	

한편, 화살표 위 또는 아래에 다른 기호나 식을 입력할 수도 있습니다. 이 방법은 88쪽을 읽어 보시기 바랍니다. 또, 화살표가 많이 사용되는 가환 도표를 그리고 싶다면 제13장을 참조하시기 바랍니다.

그리스 문자, 히브리 문자와 글자 모양 기호

그리스 문자는 표 10.22의 제어 문자열로 입력합니다. 간단히 문자의 이름을 제어 문자열의 이름으로 하면 됩니다. 대문자는 이름의 첫 글자를 대문자로 입력하면 됩니다. 라틴 문자와 모양이 같은 문자(대문자 알파, 베타, …, 카이와 소문자 오미크론)는 대응하는 라틴 문자를 입력하면 됩니다.

표 10.22: 그리스 문자

α \alpha	θ \theta	ξ \xi	υ \upsilon
β \beta	ϑ \vartheta	π \pi	ϕ \phi
γ \gamma	ι \iota	ϖ \varpi	φ \varphi
δ \delta	κ \kappa	ρ \rho	χ \chi
ϵ \epsilon	\varkappa \varkappa ¹	ϱ \varrho	ψ \psi
ε \varepsilon	λ \lambda	σ \sigma	ω \omega
ζ \zeta	μ \mu	\digamma \digamma ¹	
η \eta	ν \nu	τ \tau	
Γ \Gamma	Λ \Lambda	Σ \Sigma	Ψ \Psi
Δ \Delta	Ξ \Xi	Υ \Upsilon	Ω \Omega
Θ \Theta	Π \Pi	Φ \Phi	

¹ amssymb 패키지를 불러와야 합니다.

한편, 그리스 문자 대문자를 이탤릭체로 입력하려면 amsmath 패키지를 불러온 후 역슬래시 뒤에 ‘var’를 붙이십시오. 예를 들어, Γ 는 \varGamma로 입력합니다. 다음 표 10.23에 이 제어 문자열들을 정리하였습니다.

표 10.23: amsmath 패키지의 이탤릭 그리스 대문자

Γ \varGamma	Λ \varLambda	Σ \varSigma	Ψ \varPsi
Δ \varDelta	Ξ \varXi	Υ \varUpsilon	Ω \varOmega
Θ \varTheta	Π \varPi	Φ \varPhi	

히브리 문자는 표 10.24의 제어 문자열로 입력합니다. \aleph를 제외하고는 amssymb 패키지를 불러와야 입력할 수 있습니다.

표 10.24: 히브리 문자

\aleph \aleph	\beth \beth	\gimel \gimel	\daleth \daleth
-----------------	---------------	-----------------	-------------------

글자 모양 기호(letter-like symbol)는 이름대로 로마자의 모습을 닮은 기호를 뜻합니다. 이들은 표 10.25와 표 10.26의 제어 문자열로 입력합니다.

표 10.25: 글자 모양 기호

\perp <code>\bot</code>	\forall <code>\forall</code>	\imath <code>\imath</code>	\ni ¹ <code>\ni</code>	\top <code>\top</code>
ℓ <code>\ell</code>	\hbar <code>\hbar</code>	\in <code>\in</code>	∂ <code>\partial</code>	\wp <code>\wp</code>
\exists <code>\exists</code>	\Im <code>\Im</code>	\jmath <code>\jmath</code>	\Re <code>\Re</code>	

¹ `\owns`로도 입력할 수 있습니다.

표 10.26: amssymb 패키지의 글자 모양 기호

\Bbbk <code>\Bbbk</code>	\textcircled{S} <code>\circledS</code>	\Finv <code>\Finv</code>	\hslash <code>\hslash</code>
\textcircled{R} <code>\circledR</code>	\complement <code>\complement</code>	\Game <code>\Game</code>	\nexists <code>\nexists</code>

괄호

괄호는 표 10.27의 제어 문자열을 사용하여 입력합니다. 이 표에 있는 기호들은 특정 제어 문자열을 사용해 크기를 조절할 수 있습니다. (위아래 방향 화살표가 이 표에 있는 것도 크기를 조절할 수 있기 때문입니다.) 이 방법은 다음 장의 11.2절에서 설명합니다.

표 10.27: 괄호

$($ ¹ <code>(</code>	$)$ ¹ <code>)</code>	$/$ <code>/</code>	\backslash <code>\backslash</code>
$[$ [또는 <code>\lbrack</code>	$]$] 또는 <code>\rbrack</code>	$ $ 또는 <code>\vert</code>	\parallel 또는 <code>\Vert</code>
$\{$ { 또는 <code>\lbrace</code>	$\}$ } 또는 <code>\rbrace</code>	\uparrow <code>\uparrow</code>	\Uparrow <code>\Uparrow</code>
\lfloor <code>\lfloor</code>	\rfloor <code>\rfloor</code>	\downarrow <code>\downarrow</code>	\Downarrow <code>\Downarrow</code>
\lceil <code>\lceil</code>	\rceil <code>\rceil</code>	\updownarrow <code>\updownarrow</code>	\Updownarrow <code>\Updownarrow</code>
\langle <code>\langle</code>	\rangle <code>\rangle</code>		

¹ `mathtools` 패키지를 불러오면 `\lparen`, `\rparen`으로도 입력할 수 있습니다.

여기에서 소개하는 세로줄과 겹세로줄은 앞에서 설명했던 기호와 모양이 같습니다. 하지만 조판 과정에서 기호 앞뒤의 간격이 미세하게 달라지므로 구분하여 입력해야 합니다. 앞쪽에서 소개한 `\mid`는 이항 관계 기호로써, 여기의 `|` 및 `\vert`는 괄호로써 사용해야 합니다. 마찬가지로, `\parallel`은 이항 관계 기호로써, `||` 및 `\Vert`는 괄호로써 사용해야 합니다. 예를 들어, 다음 예시에서 세로줄 앞뒤의 공간을 비교해 보십시오.

$$6 \mid 24, \{x \mid x > 0\} \quad (\text{\code{\mid}}) \quad \text{vs} \quad P(A|B), |f(x)|, \psi|_F \quad (\text{\code{\vert}})$$

세로줄을 관계 기호로써 사용한 전자와 이외의 의미로 사용한 후자를 비교해 보시기 바랍니다. 겹세로줄을 사용한 다음 예시도 확인하십시오.


표 10.29: 문장 부호

\cdot <code>\cdotp</code>	$:$ <code>\colon</code>	\cdot <code>\ldotp</code>	\vdots <code>\vdots</code>
\cdots <code>\cdots</code>	\ddots <code>\ddots</code>	\dots <code>\ldots</code> ¹	


¹ `\dots`나 `\mathellipsis`로도 입력할 수 있습니다.


쉼표, 마침표, 가운데점, 쌍점은 조판할 수 있는 명령어가 여럿 있습니다. 여러 명령어가 같은 모양의 기호를 조판하더라도 앞뒤 식과의 간격이 다르기 때문에 상황에 맞는 명령어를 사용해야 합니다.

123,456 (<code>{,}</code> , 일반 기호)	vs	123,456 (<code>,</code> , 문장 부호)
123.456 (<code>.</code> , 일반 기호)	vs	123.456 (<code>\ldotp</code> , 문장 부호)
123 · 456 (<code>\cdot</code> , 연산자)	vs	123·456 (<code>\cdotp</code> , 문장 부호)
123 : 456 (<code>:</code> , 관계 기호)	vs	123:456 (<code>\colon</code> , 문장 부호)

쉼표는 키보드의  키를 눌러 바로 입력할 수도, 중괄호로 감싸 `{,}`로 입력할 수도 있습니다. 수학적 대상을 나열하거나 좌표를 적는 등, 일반적인 경우에 사용하는 문장 부호로써의 쉼표는 그냥 `,`로 입력하면 됩니다. 하지만, 천 단위 구분 기호로써 쉼표를 사용할 때에는 쉼표를 중괄호로 감싸 쉼표와 쉼표 뒤의 숫자 사이의 공백을 없애 주어야 합니다.

$$\{100, 200, 300\} \quad \text{vs} \quad \mathbf{a} = (0, 0, 0) \quad \text{vs} \quad \$1,200$$

마침표와 가운데점은 수식에서는 문장 부호로써 사용할 일이 거의 없을 것입니다. 마침표는 키보드의  키를 눌러 입력하면 되고, 가운데점은 `\cdot`로 입력하면 됩니다. 하지만 이 둘을 문장 부호로써 입력해야 한다면 `\ldotp`, `\cdotp`를 사용하십시오.

쌍점은 키보드의  키를 눌러 입력할 수도, `\colon`로 입력할 수도 있습니다. `:`은 비를 나타낼 때나 수학적 대상 사이의 관계를 나타낼 때에 입력하고, `\colon`은 쌍점을 문장 부호로써 사용할 때, 즉 함수의 정의역과 공역을 표시하거나 집합을 정의할 때 입력합니다.

$$16 : 9 \quad \text{vs} \quad [\mathbb{Q}(x) : \mathbb{Q}] = 2 \quad \text{vs} \quad \mathbb{Z}_+ = \{x \in \mathbb{Z} : x > 0\} \quad \text{vs} \quad f : X \rightarrow Y$$

한편, `\cdots`와 `\ldots`는 세로 위치가 다른 줄임표를 출력합니다. 세로 가운데에 줄임표를 출력하는 `\cdots`는 연산자 사이나 곱셈 기호를 생략한 항 사이에, 세로 아래쪽에 줄임표를 출력하는 `\ldots`는 쉼표 사이 또는 나열된 기호 중 일부를 생략했을 때 사용합니다.

$$\sum_{i=1}^n a_i = a_1 + a_2 + \cdots + a_n \quad \text{vs} \quad \prod_{i=1}^n a_i = a_1 a_2 \cdots a_n$$

$$\text{vs} \quad A = \{a_1, a_2, \dots, a_n\} \quad \text{vs} \quad x = 0.a_1 a_2 \dots a_n$$

기타 기호

위의 어느 곳에도 해당하지 않지만 \LaTeX 또는 `amssymb` 패키지가 제공하는 나머지 기호를 표 10.30과 표 10.31에 정리하였습니다. 앞서 설명했듯, 프라임 기호를 출력하는 `\prime`은 위 첨자로 입력해야 흔히 사용하는 모습을 얻게 됩니다.

표 10.30: 기타 기호

\angle <code>\angle</code>	\diamond <code>\Diamond¹</code>	∞ <code>\infty</code>	\prime <code>\prime</code>
\backslash <code>\backslashash</code>	\diamond <code>\diamondsuit</code>	\mho <code>\mho¹</code>	\spadesuit <code>\spadesuit</code>
\square <code>\Box¹</code>	\emptyset <code>\emptyset</code>	∇ <code>\nabla</code>	\surd <code>\surd</code>
\clubsuit <code>\clubsuit</code>	\heartsuit <code>\heartsuit</code>	\neg <code>\neg²</code>	\triangle <code>\triangle</code>

¹ latexsym 또는 amssymb 패키지를 불러와야 합니다.

² \lnot으로도 입력할 수 있습니다.

표 10.31: amssymb 패키지의 기타 기호

\backprime <code>\backprime</code>	\blacktriangledown <code>\blacktriangledown</code>	\measuredangle <code>\measuredangle</code>	\varnothing <code>\varnothing</code>
\bigstar <code>\bigstar</code>	\diagdown <code>\diagdown</code>	\mho <code>\mho</code>	\vartriangle <code>\vartriangle</code>
\blacklozenge <code>\blacklozenge</code>	\diagup <code>\diagup</code>	\sphericalangle <code>\sphericalangle</code>	
\blacksquare <code>\blacksquare</code>	\eth <code>\eth</code>	\square <code>\square</code>	
\blacktriangle <code>\blacktriangle</code>	\lozenge <code>\lozenge</code>	\triangledown <code>\triangledown</code>	

더 다양한 기호를 입력해야 한다면 [6]을 참조할 수 있습니다. 기호를 입력하는 패키지를 불러올 때에는 수식의 로마자 글꼴과 어울리는지 확인하시기 바랍니다.

10.4

장식 기호

이번 장에서는 다양한 장식 기호를 입력하는 방법을 설명하겠습니다. 문자 위에 가로줄이나 물결표 등을 그리는 방법, 화살표 위에 식을 입력하는 방법, 연산자나 관계 기호 위에 문자를 입력하는 방법을 알 수 있습니다.

악센트

\acute{a} , \bar{a} , \breve{a} , \check{a} 에서 a 위에 입력된 기호를 ‘악센트’라고 부릅니다. 수식에서 악센트는 표 10.32의 제어 문자열을 사용해 입력합니다. 악센트를 붙일 문자를 인자로 지정하면 됩니다. i 나 j 위에 악센트를 달 때는 `\imath`와 `\jmath`를 대신 사용하는 것이 좋습니다. 위쪽의 점이 사라져(i, j 대신 \imath, \jmath 가 조판됨) 더 자연스러운 결과를 얻을 수 있습니다.

표 10.32: 수식 악센트

\acute{a} <code>\acute{a}</code>	\dddot{a} <code>\dddot{a}¹</code>	\grave{a} <code>\grave{a}</code>	\vec{a} <code>\vec{a}</code>
\bar{a} <code>\bar{a}</code>	\ddot{a} <code>\ddot{a}¹</code>	\hat{a} <code>\hat{a}</code>	
\breve{a} <code>\breve{a}</code>	\dot{a} <code>\dot{a}</code>	\mathring{a} <code>\mathring{a}</code>	
\check{a} <code>\check{a}</code>	\dot{a} <code>\dot{a}</code>	\tilde{a} <code>\tilde{a}</code>	

¹ amsmath 패키지를 불러와야 사용할 수 있습니다.

`\[\hat{\alpha}([f]) = [\bar{\alpha}] * [f] * [\alpha] \]`

$$\hat{\alpha}([f]) = [\bar{\alpha}] * [f] * [\alpha]$$

예제 10.15 아래 수식을 조판하여라.

$$\begin{aligned} \text{(a)} \quad & \vec{B} = \frac{\mu_0}{4\pi} \int \frac{Id\vec{l} \times \hat{r}}{r^2} \\ \text{(b)} \quad & (2x\hat{i} + 3x^2\hat{j} - 7\hat{k}) \cdot (3x\hat{i} - 2\hat{j}) = 0 \end{aligned}$$

표 10.32의 장식 기호는 글자 하나만을 인자로 지정해야 합니다. \overrightarrow{AB} 를 얻기 위해 `\(\vec{AB}\)`를 입력하면 \overrightarrow{AB} 처럼 기호가 문자열 전체의 폭에 맞게 늘어나지 않습니다. 이때에는 표 10.33의 기호를 사용합니다. 이 표의 장식 기호는 인자로 지정한 문자열의 길이에 맞춰 늘어납니다. 예를 들어, \overrightarrow{AB} 는 `\(\overrightarrow{AB}\)`로 입력합니다.

표 10.33: 가변 길이 악센트

\widetilde{abcd}	<code>\widetilde{abcd}</code>	\widehat{abcd}	<code>\widehat{abcd}</code>
\overleftarrow{abcd}	<code>\overleftarrow{abcd}</code>	\overrightarrow{abcd}	<code>\overrightarrow{abcd}</code>
\overline{abcd}	<code>\overline{abcd}</code>	\underline{abcd}	<code>\underline{abcd}</code>
\overbrace{abcd}^1	<code>\overbrace{abcd}^1</code>	\underbrace{abcd}_1	<code>\underbrace{abcd}_1</code>
$\overleftrightarrow{abcd}$	<code>\overleftrightarrow{abcd}</code>	$\underleftrightarrow{abcd}$	<code>\underleftrightarrow{abcd}</code>
$\overleftrightharpoonup{abcd}^2$	<code>\overleftrightharpoonup{abcd}^2</code>	$\underleftrightharpoonup{abcd}^2$	<code>\underleftrightharpoonup{abcd}^2</code>
$\overleftarrow{\hspace{0.5em}}{abcd}^2$	<code>\overleftarrow{\hspace{0.5em}}{abcd}^2</code>	$\overrightarrow{\hspace{0.5em}}{abcd}^2$	<code>\overrightarrow{\hspace{0.5em}}{abcd}^2</code>
$\overbracket{abcd}^{1,3}$	<code>\overbracket{abcd}^{1,3}</code>	$\underbracket{abcd}_{1,3}$	<code>\underbracket{abcd}_{1,3}</code>

¹ 첨자 문법을 사용해 괄호 위나 아래에 내용을 입력할 수 있습니다.

² `amsmath` 패키지를 불러와야 사용할 수 있습니다.

³ `mathtools` 패키지를 불러와야 사용할 수 있습니다.

예제 10.16 아래 수식을 조판하여라.

$$\begin{aligned} \text{(a)} \quad & \overrightarrow{AB} + \overrightarrow{BC} = \overrightarrow{AC}, \quad \overrightarrow{AC} - \overrightarrow{AB} = \overrightarrow{BC} \\ \text{(b)} \quad & \underbrace{a + a + \cdots + a}_{n \text{ times}} = na, \quad \underbrace{a \times a \times \cdots \times a}_{n \text{ times}} = a^n \end{aligned}$$

화살표 위아래에 문자 입력하기

연산을 적용하기 전후의 변화를 화살표를 사용해 나타낼 때, 화살표 위에 적용한 연산을 적어 두면 유용한 경우가 있습니다. 이처럼 화살표 위나 아래에 문자를 입력하기 위한 제어 문자열을 표 10.34에 정리하였습니다.

표 10.34: 화살표 위아래에 문자를 입력하기 위한 제어 문자열

$\stackrel{abc}{\leftarrow}{def}$	<code>\xleftarrow[def]{abc}</code> ¹	$\stackrel{abc}{\rightarrow}$	<code>\xrightarrow[def]{abc}</code> ¹
\hookleftarrow{abc}	<code>\xhookleftarrow[def]{abc}</code> ²	\leftrightharpoons{abc}	<code>\xleftrightharpoons[def]{abc}</code> ²
\hookrightarrow{abc}	<code>\xhookrightarrow[def]{abc}</code> ²	\mapsto{abc}	<code>\xmapsto[def]{abc}</code> ²
\Leftrightarrow{abc}	<code>\xLeftrightarrow[def]{abc}</code> ²	\Rrightarrow{abc}	<code>\xRrightarrow[def]{abc}</code> ²
\leftharpoonup{abc}	<code>\xleftharpoonup[def]{abc}</code> ²	\rightharpoonup{abc}	<code>\xrightarrow[def]{abc}</code> ²
\leftrightharpoonup{abc}	<code>\xleftrightharpoonup[def]{abc}</code> ²	\rightharpoonup{abc}	<code>\xrightarrow[def]{abc}</code> ²
\leftrightharpoonup{abc}	<code>\xleftrightharpoonup[def]{abc}</code> ²	\leftrightharpoonup{abc}	<code>\xleftrightharpoonup[def]{abc}</code> ²
\leftrightharpoonup{abc}	<code>\xleftrightharpoonup[def]{abc}</code> ²	\leftrightharpoonup{abc}	<code>\xleftrightharpoonup[def]{abc}</code> ²
\leftrightharpoonup{abc}	<code>\xleftrightharpoonup[def]{abc}</code> ²	\leftrightharpoonup{abc}	<code>\xleftrightharpoonup[def]{abc}</code> ²
\leftrightharpoonup{abc}	<code>\xleftrightharpoonup[def]{abc}</code> ²	\leftrightharpoonup{abc}	<code>\xleftrightharpoonup[def]{abc}</code> ²

¹ amsmath 패키지를 불러와야 합니다.² mathtools 패키지를 불러와야 합니다.

예제 10.17 표 10.34의 제어 문자열을 사용하여 $F \xrightarrow{\varphi} K$ 를 조판하여라.

기호 위에 문자 입력하기

임의의 기호 위에도 문자를 입력할 수 있습니다. 이때에는 `\stackrel` 및 `\stackbin` 제어 문자열을 사용합니다. 전자는 이항 관계 기호가, 후자는 이항 연산자가 아래에 올 때 사용합니다.

```
\stackrel{<text>}{<symbol>}
\stackbin{<text>}{<symbol>}
```

<text>: 기호 위쪽에 적을 내용

<symbol>: 입력하려는 기호

예를 들어, 등호 위에 'def'라는 문자열을 입력하려면

```
\stackrel{\text{def}}{=}
```

라고 입력합니다. <text>에 입력한 내용은 수식 입력 모드에서 조판되므로, 텍스트 모드에서 조판하려면 `\text`를 사용해야 합니다.

```
\[ \nabla f \stackrel{\text{def}}{=} \left( \frac{\partial}{\partial x_1} f, \dots, \frac{\partial}{\partial x_n} f \right) \]
```

$$\nabla f \stackrel{\text{def}}{=} \left(\frac{\partial}{\partial x_1} f, \dots, \frac{\partial}{\partial x_n} f \right)$$

10.5

수식 글꼴

벡터를 나타낼 때는 \mathbf{u} , \mathbf{v} 처럼 볼드체를 사용하곤 합니다. 또, 수 집합을 나타낼 때는 \mathbb{R} , \mathbb{N} 처럼 칠판 볼드체(blackboard bold)를 사용합니다. 집합의 모임을 나타낼 때에는 흘림체를 사용해 \mathcal{B} , \mathcal{S} , \mathcal{T} 와 같이 나타내기도 합니다.

수식 입력 모드에서 사용할 수 있는 글꼴을 표 10.35에 나타내었습니다. 이 표에 나타낸 제어 문자열은 모두 범위 지정형 제어 문자열이며, 로마자 대·소문자, 숫자, 그리스 문자 대문자에만 유효합니다.

표 10.35: 수식 입력 모드에서 사용할 수 있는 글꼴

글꼴	제어 문자열	예시
기본		$abcdABCD1234\Gamma\Delta\Theta\Lambda$
이탤릭체	<code>\mathit</code>	$abcdABCD1234\Gamma\Delta\Theta\Lambda$
수식 이탤릭체	<code>\mathnormal</code>	$abcdABCD1234\Gamma\Delta\Theta\Lambda$
정체	<code>\mathrm</code>	$abcdABCD1234\Gamma\Delta\Theta\Lambda$
볼드체	<code>\mathbf</code>	$\mathbf{abcdABCD1234\Gamma\Delta\Theta\Lambda}$
산세리프체	<code>\mathsf</code>	$abcdABCD1234\Gamma\Delta\Theta\Lambda$
고정폭 서체	<code>\mathtt</code>	$abcdABCD1234\Gamma\Delta\Theta\Lambda$
흘림체	<code>\mathcal</code>	$\mathcal{A}\mathcal{B}\mathcal{C}\mathcal{D}\mathcal{E}\mathcal{F}\mathcal{G}\mathcal{H}\mathcal{I}\mathcal{J}^1$
독일 활자체	<code>\mathfrak</code>	$\mathfrak{a}\mathfrak{b}\mathfrak{c}\mathfrak{d}1234^{2,3}$
칠판 볼드체	<code>\mathbb</code>	$\mathbb{A}\mathbb{B}\mathbb{C}\mathbb{D}\mathbb{E}\mathbb{F}\mathbb{G}\mathbb{H}\mathbb{I}\mathbb{J}^{1,3}$

¹ 로마자 대문자에만 사용할 수 있습니다.

² 그리스 문자 대문자에 사용할 수 없습니다.

³ `amssymb` 또는 `amsfonts` 패키지를 불러와야 사용할 수 있습니다.

위의 표에서 설명한 제어 문자열을 사용하면 \mathbf{u} 는 `\(\mathbf{u}\)`로, \mathbb{R} 는 `\(\mathbb{R}\)`로, \mathcal{T} 는 `\(\mathcal{T}\)`로 입력할 수 있습니다.

예제 10.18 경우에 따라서는 무한소를 나타내는 로마자 d 를 정체로 조판하기도 한다. 글꼴에 주의하면서 다음 수식을 조판하여라.

$$\frac{d}{dx} \int_a^x f(t) dt = f(x)$$

한 가지 주의할 점으로, 연산자나 함수의 이름을 입력할 때에는 `\mathrm`이 아닌, 다음 장에서 설명하는 `\operatorname` 또는 `\DeclareMathOperator` 제어 문자열을 사용해야 합니다. 이름 앞뒤에 오는 수식의 요소들이 올바른 간격으로 배치되어야 하기 때문입니다. 다음 예시를 참고하십시오.

`\[2 \mathrm{rank} A \quad \text{vs} \quad 2 \operatorname{rank} A]`

$2\mathrm{rank}A$ vs $2\operatorname{rank}A$

예제 10.19 글꼴에 주의하면서 다음 수식을 조판하여라.

$$\text{tr}(\mathbf{u}\mathbf{v}^T) = \mathbf{u} \cdot \mathbf{v}$$

수식 볼드체로 조판하기

앞서 수식 문자를 볼드체로 조판하기 위한 제어 문자열로 `\mathbf`를 소개하였습니다. 하지만 이는 제한된 범위의 문자에만 유효하고, 그리스 문자 소문자나 수학 기호 등에는 전혀 영향을 주지 않습니다. 또 로마자는 이탤릭체 볼드체가 아닌 정체 볼드체로 조판됩니다. 다른 기호의 볼드체를 사용하거나 로마자의 이탤릭 볼드체가 필요하다면 `amsmath` 패키지에서 제공하는 `\boldsymbol` 제어 문자열을 사용할 수 있습니다.

하지만, `amssymb` 패키지에서 제공하는 기호나 가변 크기 연산자는 `\boldsymbol`의 영향을 받지 않습니다. 이때에는 ‘최후의 수단’으로 `\pmb` 제어 문자열을 사용할 수 있습니다. 이 제어 문자열은 인자로 지정된 수식을 여러 번 겹쳐서 조판함으로써 볼드체의 효과를 냅니다. `\pmb`를 사용한 조판 결과는 미려하지 않으므로, 꼭 필요한 경우에만 사용해야 합니다.

기본 글꼴을 사용해 기호를 조판한 결과와 `\mathbf`, `\boldsymbol`, `\pmb` 제어 문자열을 사용해 볼드체를 조판한 결과를 아래에서 비교해 보십시오. 여기에서 사용한 `\mathtest`는 다음 코드를 줄인 것입니다.

```
abc ~ ABC ~ \Gamma\Delta\Theta ~ \varGamma\varDelta\varTheta ~
\alpha\beta\gamma ~ {\to}{\cap}{\oplus} ~
{\approx}{\nsubseteq}{\triangleleft} ~ \sum\prod\int
```

```
1 \[ \mathtest ]
2 \[ \mathbf{\mathtest} ]
3 \[ \boldsymbol{\mathtest} ]
4 \[ \pmb{\mathtest} ]
```

$abc \ ABC \ \Gamma\Delta\Theta \ \varGamma\varDelta\varTheta \ \alpha\beta\gamma \ \rightarrow\cap\oplus \ \approx\nsubseteq\triangleleft \ \sum\prod\int$
 $abc \ \mathbf{ABC} \ \mathbf{\Gamma\Delta\Theta} \ \mathbf{\varGamma\varDelta\varTheta} \ \mathbf{\alpha\beta\gamma} \ \mathbf{\rightarrow\cap\oplus} \ \mathbf{\approx\nsubseteq\triangleleft} \ \mathbf{\sum\prod\int}$
 $abc \ \boldsymbol{ABC} \ \boldsymbol{\Gamma\Delta\Theta} \ \boldsymbol{\varGamma\varDelta\varTheta} \ \boldsymbol{\alpha\beta\gamma} \ \boldsymbol{\rightarrow\cap\oplus} \ \boldsymbol{\approx\nsubseteq\triangleleft} \ \boldsymbol{\sum\prod\int}$
 $abc \ \pmb{ABC} \ \pmb{\Gamma\Delta\Theta} \ \pmb{\varGamma\varDelta\varTheta} \ \pmb{\alpha\beta\gamma} \ \pmb{\rightarrow\cap\oplus} \ \pmb{\approx\nsubseteq\triangleleft} \ \pmb{\sum\prod\int}$

이전 버전 명령어에 관한 조언

참고 `\it`, `\mit`, `\rm`, `\bf`, `\sf`, `\tt`, `\cal`을 모르는 분은 이 부분을 넘어가셔도 됩니다.

\LaTeX 표준 클래스에서는 `\it`, `\mit`, `\rm`, `\bf`, `\sf`, `\tt`, `\cal` 등의 제어 문자열을 사용해도 수식 글꼴을 바꿀 수 있습니다. 이들은 \LaTeX 의 이전 버전인 $\text{\LaTeX} 2.09$ 나 \TeX 에서 사용하는 글꼴 변경 명령어입니다. 앞서 8.2절에서 설명했던 것과 같은 이유로, 이 제어 문자열 대신 표 10.35의 제어 문자열을 사용해 글꼴을 변경하는 것이 바람직합니다.

제 11 장

복잡한 수식 문법

이번 장에서는 앞에서 알아본 기본 문법에 더하여, 더 복잡한 수식 작성 문법을 알아보겠습니다. 이들은 주로 행중수식보다는 별행수식을 입력할 때 사용됩니다.

11.1

연산자, 적분 기호와 함수

앞서 알아보았던 연산자, 적분 기호와 로마자 정체로 적는 함수를 입력하는 데 사용할 수 있는 몇 가지 명령어를 추가로 알아보겠습니다.

로마자자로 적는 연산자와 함수 직접 입력하기

표 10.6에 없는 연산자와 함수는 amsmath 패키지의 `\operatorname` 또는 `\operatorname*` 제어 문자열을 사용해 입력합니다. 띄어쓰기를 해야 한다면 공백 문자 대신 `\,`를 입력하십시오.

```
\operatorname{<name>}
\operatorname*{<name>}
```

`<name>`: 연산자나 함수의 이름

별표 붙은 것과 붙지 않은 것의 차이는 별행수식에서 첨자를 조판하는 위치입니다. 별표 없는 것은 기호의 오른쪽 위와 오른쪽 아래에, 별표 붙은 것은 기호의 위아래에 첨자가 조판됩니다.

```
1 \[ \operatorname{artanh}^2 x \quad \operatorname{lcm}_{i \in I} a_i \]
```

$$\operatorname{artanh}^2 x \quad \operatorname{lcm}_{i \in I} a_i$$

특정 연산자나 함수를 자주 입력한다면 amsmath 패키지가 제공하는 `\DeclareMathOperator`와 `\DeclareMathOperator*` 제어 문자열을 사용해 연산자나 함수를 입력하는 제어 문자열을 정의할 수도 있습니다. 별표 없는 것과 별표 붙은 것의 차이는 `\operatorname`의 경우와 똑같습니다.

```
\DeclareMathOperator{<csname>}{<name>}
\DeclareMathOperator*{<csname>}{<name>}
```

<csname>: 연산자나 함수를 입력하는 데 사용할 제어 문자열

<name>: 연산자나 함수의 이름

예를 들어, 행렬의 계수를 뜻하는 함수 rank를 조판하기 위한 제어 문자열을 \rank로 정의하려면 전처리부에

```
\DeclareMathOperator{\rank}{rank}
```

를 입력합니다. 이렇게 정의한 \rank는 다음과 같이 사용할 수 있습니다.

The rank of a matrix $\backslash(A\backslash)$ is denoted as $\backslash(\backslashrank A\backslash)$.

The rank of a matrix A is denoted as rank A .

\DeclareMathOperator를 사용해 제어 문자열을 정의할 때, 제어 문자열의 이름이 'end'로 시작하거나 이미 정의된 제어 문자열의 이름이 사용되면 안 됩니다. 이 경우에는 컴파일 과정에서 오류가 발생합니다.

첨자의 위치

표 10.4 ~ 10.7의 제어 문자열이나 \operatorname* 제어 문자열을 사용해 입력한 연산자, 혹은 \DeclareMathOperator*로 정의한 제어 문자열로 입력한 기호 뒤에 입력한 첨자를 조판할 위치를 기호의 위아래와 오른쪽 위 및 오른쪽 아래 중 선택할 수 있습니다.

연산자 뒤에 \limits를 입력한 뒤 첨자를 입력하면 이는 연산자의 위아래에 조판됩니다. 반대로, 연산자 뒤에 \nolimits를 입력한 뒤 첨자를 입력하면 이는 연산자의 오른쪽 위와 오른쪽 아래에 조판됩니다.

```
\[ \int\limits_{\partial D} f(x) \, dx =
\int_0^1 f(\alpha(z)) \, dz \]
```

$$\int_{\partial D} f(x) dx = \int_0^1 f(\alpha(z)) dz$$

```
1 \[ E = \sum' E_n \quad \% wrong
2 E = \sum\nolimits' E_n \] \% correct
```

$$E = \sum^{\prime} E_n \quad E = \sum' E_n$$

amsmath 패키지를 불러올 때 sumlimits, nosumlimits, intllimits, nointlimits, namelimits, nonamelimits 옵션을 지정하면 연산자와 적분 기호의 첨자 조판 위치를 일괄적으로 변경할 수 있습니다. 'sum'이 포함된 옵션은 n 항 연산자를, 'int'가 포함된 옵션은 적분 기호를, 'name'이 포함된 옵션은 로마자로 조판하는 연산자의 설정을 변경하며, 앞에 'no'가 붙은 옵션은 첨자를 기호의

오른쪽 위와 오른쪽 아래에 조판하도록 하고, 앞에 'no'가 붙지 않은 옵션은 첨자를 기호의 위아래에 조판하도록 합니다. 기본값은 `sumlimits`, `nointlimits`, `namelimits`입니다.

연산자의 계산 범위 여러 줄로 입력하기

아래의 식과 같이, 연산자의 계산 범위를 여러 줄에 걸쳐 입력해야 하는 경우도 있습니다.

$$f(x) = \prod_{\substack{1 \leq i, j \leq n \\ i \neq j}} f_{ij}(x)$$

이런 경우에는 `amsmath` 패키지에서 제공하는 `\substack` 제어 문자열을 사용하면 됩니다. 입력하려는 내용을 `\substack`의 인자로 지정합니다. 줄을 나눌 때에는 `\\`를 입력합니다. 즉, 위의 수식은

```
\[ f(x) = \prod_{\substack{1 \leq i, j \leq n \\ i \neq j}} f_{ij}(x) \]
```

를 입력하여 조판할 수 있습니다.

예제 11.1 다음 수식을 조판하여라.

$$\int_{-\infty}^{\infty} f(x) dx = \lim_{\substack{a \rightarrow -\infty \\ b \rightarrow \infty}} \int_a^b f(x) dx$$

11.2

괄호 크기 조절하기

행중수식을 입력할 때나 괄호 사이에 작은 수식을 입력할 때에는 괄호의 크기를 신경쓰지 않아도 되지만, 별행수식에서 분수나 대형 연산자 등 크기가 큰 기호들이 포함된다면 괄호의 크기를 그에 맞게 조정해 주어야 합니다. 괄호의 크기는 안에 입력된 수식의 내용에 따라 자동으로 조절할 수도 있고, 제어 문자열을 사용해 수동으로 조절할 수도 있습니다.

괄호의 크기 자동으로 조절하기

먼저, 괄호의 크기를 자동으로 조정하려면 `\left`와 `\right` 제어 문자열을 사용합니다.

```
\left<left delim.> ... \right<right delim.>
```

`<left delim.>`: 왼쪽 괄호

`<right delim.>`: 오른쪽 괄호

`<left delim.>`과 `<right delim.>`에는 표 10.27의 기호를 입력합니다. `\left`와 `\right`는 반드시 짝을 맞춰 사용해야 하지만, 괄호의 종류는 짝을 맞추어 사용하지 않아도 됩니다.

```
\[ \sum_{k=1}^n k^3 = \left[ \frac{n(n+1)}{2} \right]^2 \]
```

$$\sum_{k=1}^n k^3 = \left[\frac{n(n+1)}{2} \right]^2$$

양쪽 중 어느 한쪽 괄호를 생략하려면, 표에 있는 기호 대신 마침표(.)를 입력하면 됩니다.

$$\left[\left. \frac{df}{dx} \right|_{x=1} = \lim_{x \rightarrow 1} \frac{f(x)-f(1)}{x-1} \right]$$

$$\left. \frac{df}{dx} \right|_{x=1} = \lim_{x \rightarrow 1} \frac{f(x) - f(1)}{x - 1}$$

복잡한 조건부 확률 식이나 분수 등을 입력할 때, 왼쪽과 오른쪽 괄호뿐 아니라 가운데에 입력하는 구분선의 높이도 동일하게 맞추어야 하는 경우가 있습니다. 이때에는 `\left`, `\right`의 사이에 `\middle`을 사용할 수 있습니다.

$$\backslash\mathbf{left}\langle\mathit{left\ delim.}\rangle\ \dots\ \backslash\mathbf{middle}\langle\mathit{middle\ delim.}\rangle\ \dots\ \backslash\mathbf{right}\langle\mathit{right\ delim.}\rangle$$

$\langle left\ delim.\rangle$: 왼쪽 괄호

⟨*middle delim.*⟩: 가운데에 올 구분선

<right delim.>: 오른쪽 괄호

왼쪽 또는 오른쪽 괄호와 마찬가지로, *(middle delim.)*에도 표 10.27에 있는 기호를 입력해야 합니다. 여기에 입력한 구분선은 *(left delim.)*과 *(right delim.)*과 같은 높이로 크기가 조정됩니다. 필요한 경우에는 한 쌍의 `\left`와 `\right` 사이에 `\middle`을 여러 번 사용할 수도 있습니다.

$$\left[P! \left(x_2 \leq x_2^n \mid x_1 \leq x_1^n \right) \right]$$

$$P(X_2 \leq x_2^{-1} | X_1 \leq x_1^{-1})$$

$$\left[\frac{dy}{dx} = \left. \frac{dy}{dt} \right/ \frac{dx}{dt} \right]$$

$$\frac{dy}{dx} = \frac{dy}{dt} / \frac{dx}{dt}$$

괄호의 크기 수동으로 조절하기

괄호의 크기를 수동으로 조절할 수도 있습니다. 여는 괄호의 앞에 `\bigl`, `\Bigl`, `\biggl`, `\Biggl`을, 닫는 괄호의 앞에 `\bigr`, `\Bigr`, `\biggr`, `\Biggr`를 입력하면 됩니다. 괄호의 상대적 크기를 표 11.1에서 확인하십시오.

예를 들어 괄호 안에 n 항 연산자나 적분 기호 등이 있을 때, `\left`와 `\right`를 사용하면 괄호가 지나치게 커집니다. 연산자보다 괄호가 조금만 크게 표시되도록 괄호의 크기를 수동으로 조절하는 것이 좋습니다.

$$\left[\sum_{i=1}^n \left| x_i \right| \right]^{\frac{1}{p}} \quad \text{vs} \quad \left| \sum_{i=1}^n x_i \right|^{\frac{1}{p}}$$

$$\left[\sum_{i=1}^n |x_i|^p \right]^{1/p} \quad \text{VS} \quad \left[\sum_{i=1}^n |x_i|^p \right]^{1/p}$$

표 11.1: 수동으로 조절한 괄호의 크기

괄호의 크기	제어 문자열	조판한 모습
기본 크기	사용하지 않음	$\langle a \rangle \quad \left(\frac{a}{b} \right) \quad \left\{ a + b\alpha : a, b \in \mathbb{Z}; \alpha = \frac{1 + \sqrt{-19}}{2} \right\} \quad \left[\sum_{i=1}^n x_i ^p \right]^{1/p}$
1단계 크게	<code>\bigl, \bigr</code>	$\langle a \rangle \quad \left(\frac{a}{b} \right) \quad \left\{ a + b\alpha : a, b \in \mathbb{Z}; \alpha = \frac{1 + \sqrt{-19}}{2} \right\} \quad \left[\sum_{i=1}^n x_i ^p \right]^{1/p}$
2단계 크게	<code>\Bigl, \Bigr</code>	$\langle a \rangle \quad \left(\frac{a}{b} \right) \quad \left\{ a + b\alpha : a, b \in \mathbb{Z}; \alpha = \frac{1 + \sqrt{-19}}{2} \right\} \quad \left[\sum_{i=1}^n x_i ^p \right]^{1/p}$
3단계 크게	<code>\biggl, \biggr</code>	$\langle a \rangle \quad \left(\frac{a}{b} \right) \quad \left\{ a + b\alpha : a, b \in \mathbb{Z}; \alpha = \frac{1 + \sqrt{-19}}{2} \right\} \quad \left[\sum_{i=1}^n x_i ^p \right]^{1/p}$
4단계 크게	<code>\Biggl, \Biggr</code>	$\langle a \rangle \quad \left(\frac{a}{b} \right) \quad \left\{ a + b\alpha : a, b \in \mathbb{Z}; \alpha = \frac{1 + \sqrt{-19}}{2} \right\} \quad \left[\sum_{i=1}^n x_i ^p \right]^{1/p}$
자동 조절	<code>\left, \right</code>	$\langle a \rangle \quad \left(\frac{a}{b} \right) \quad \left\{ a + b\alpha : a, b \in \mathbb{Z}; \alpha = \frac{1 + \sqrt{-19}}{2} \right\} \quad \left[\sum_{i=1}^n x_i ^p \right]^{1/p}$

열거나 닫는 괄호가 아니라 가운데에 있는 괄호 혹은 구분선의 크기를 조절할 때에는 `\bigm`, `\Bigm`, `\biggm`, `\Biggm` 또는 `\big`, `\Big`, `\bigg`, `\Bigg`를 사용합니다. 처음 네 제어 문자열을 사용해 조판한 기호는 관계 기호로 취급되어 기호 좌우의 간격이 넓게 조판되고, 뒤쪽 네 제어 문자열을 사용해 조판한 기호는 연산자로 취급되어 기호 좌우에 간격이 생기지 않습니다.

```
\[ A = \biggl\{ x \times \biggm| \frac{x+1}{5} \in \mathbb{Z} \biggr\}
```

$$A = \left\{ x \mid \frac{x+1}{5} \in \mathbb{Z} \right\}$$

```
\[ \frac{dy}{dx} = \frac{dy}{dt} \bigg/ \frac{dx}{dt}
```

$$\frac{dy}{dx} = \frac{dy}{dt} \bigg/ \frac{dx}{dt}$$

앞서 설명한 `\left`, `\right`와 다르게, 괄호의 크기를 수동으로 조절하는 제어 문자열은 짝을 맞추어 사용하지 않아도 됩니다.

11.3

행렬 작성하기

이번 절에서는 행렬을 작성하는 방법을 알아보겠습니다. 행렬을 작성할 때에는 `amsmath` 패키지에서 제공하는 행렬 작성 환경을 사용합니다. 행렬을 감싸는 괄호에 따라 표 11.2의 6가지 행렬 환경을 사용할 수 있습니다. 이번 절에서는 이 표의 환경을 사용해 행렬을 작성하는 방법을 알아보겠습니다.

먼저, 원하는 괄호를 사용하는 행렬 환경을 선택하고, 그 환경 안에 행렬의 각 성분을 입력합니다. 행렬의 성분은 한 행 안의 성분들을 먼저 입력하고 다음 행으로 넘어가는, 행 우선(row-major) 방식

표 11.2: amsmath 패키지의 행렬 작성 환경

환경 이름	예시	환경 이름	예시	환경 이름	예시
matrix	$\begin{matrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{matrix}$	bmatrix	$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$	vmatrix	$\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix}$
pmatrix	$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$	Bmatrix	$\begin{Bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{Bmatrix}$	Vmatrix	$\begin{Vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{Vmatrix}$

으로 입력합니다. 즉, 표 11.2와 같은 결과를 얻으려면 a_{11} , a_{12} , a_{21} , a_{22} 의 순서로 성분을 입력해야 합니다. 이때 한 행 안에서 열의 구분은 &로, 행의 구분은 \\로 합니다.

```
1 \begin{bmatrix}
2   a_{11} & a_{12} \\
3   a_{21} & a_{22} \end{bmatrix}
```

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

표에 없는 다른 괄호를 사용하는 행렬을 작성하려면 matrix 환경을 사용하고, 행렬 앞뒤에 \left와 \right를 사용해 큰 괄호를 입력해 주면 됩니다.

```
1 \left\langle \begin{matrix}
2   a_{11} & a_{12} \\
3   a_{21} & a_{22} \end{matrix} \right\rangle
```

$$\left\langle \begin{matrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{matrix} \right\rangle$$

예제 11.2 아래 식을 조판하여라.

$$\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11}a_{22} - a_{12}a_{21}$$

하나의 행렬 환경은 최대 10×10 행렬까지만 조판할 수 있습니다.¹

mathtools 패키지의 추가 환경

amsmath 패키지에서 제공하는 행렬 환경에서는 각 성분이 가운데에 맞춰 조판됩니다. 성분을 왼쪽 또는 오른쪽에 맞춰 조판하고 싶다면 mathtools 패키지를 불러온 뒤, 별표 붙은 행렬 환경을 사용하고 l, c, r 중 원하는 것을 선택 인자로 지정하면 됩니다. 옵션에 따라 성분이 왼쪽, 가운데, 오른쪽에 맞추어 조판됩니다. 선택 인자를 생략하면 기본값인 c가 사용됩니다.

¹하나의 행렬 환경으로 조판할 수 있는 행렬의 최대 크기는 MaxMatrixCols 카운터에 저장되어 있고, 이 카운터의 기본값은 10입니다. 이 값을 변경하여 행렬의 최대 크기를 변경할 수 있습니다. 카운터에 대한 설명은 22.1절을 참조하십시오.

```

1 \begin{bmatrix*}[r]
2   1 & 2 & 0 & 3 & 0 \\ 1 & 2 & -1 & -1 & 0 \\
3   0 & 0 & 1 & 4 & 0 \\ 2 & 4 & 1 & 10 & 1 \\
4   0 & 0 & 0 & 0 & 1
5 \end{bmatrix*}

```

$$\begin{bmatrix} 1 & 2 & 0 & 3 & 0 \\ 1 & 2 & -1 & -1 & 0 \\ 0 & 0 & 1 & 4 & 0 \\ 2 & 4 & 1 & 10 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

mathtools 패키지는 smallmatrix 환경도 제공합니다. 행렬 환경은 행중수식에서도 크게 조판되어 보기 좋지 않지만, smallmatrix 환경을 사용하면 행중수식에서 더 보기에 나은 행렬을 조판할 수 있습니다. 환경 이름의 앞에 p, b, B, v, V를 입력해 행렬 앞뒤에 괄호를 붙일 수도, 환경 이름의 뒤에 *를 붙여 정렬 방식을 변경할 수도 있습니다.

```

1 \((ad-bc \neq 0)\)일 때, 행렬
2 \((A = \begin{bmatrix} a & b \\ c & d \end{bmatrix})\)의 역행렬은
3 \((A^{-1} = \frac{1}{ad-bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix})\)이다.

```

$ad - bc \neq 0$ 일 때, 행렬 $A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ 의 역행렬은 $A^{-1} = \frac{1}{ad-bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$ 이다.

smallmatrix 환경으로 조판할 수 있는 행렬의 크기에는 제한이 없지만, 행중수식에서는 3×3 보다 큰 행렬은 조판하지 않는 것이 좋습니다.

11.4

여러 식 묶어서 작성하기

여러 식을 세로로 나열하고 왼쪽에 중괄호를 사용하여 묶을 때에는 amsmath 패키지에서 제공하는 cases 환경을 사용합니다. 예를 들어, 조각적으로 정의된(piecewise-defined) 함수나 연립방정식을 작성할 때 사용할 수 있습니다.

먼저, 각각의 식은 \\를 입력해 구분합니다. 각 식의 설명을 오른쪽에 적으려면 행렬 조판 환경처럼 &를 입력하면 됩니다. cases 환경 안의 식은 행중수식 스타일로 조판되므로, 복잡한 식을 별행수식 스타일로 조판하고 싶다면 \displaystyle 제어 문자열을 입력해야 합니다. 대신, 다음 페이지에서 설명하는 dcases 환경을 사용할 수도 있습니다.

```

1 \begin{displaymath}
2   \begin{cases}
3     a_1x + b_1y = c_1 \\ a_2x + b_2y = c_2
4   \end{cases}
5 \end{displaymath}

```

$$\begin{cases} a_1x + b_1y = c_1 \\ a_2x + b_2y = c_2 \end{cases}$$

```

1 \begin{displaymath}
2 |x| =
3 \begin{cases}
4 x, & \& x \geq 0 \\
5 -x, & \& x < 0
6 \end{cases}
7 \end{displaymath}

```

$$|x| = \begin{cases} x, & x \geq 0 \\ -x, & x < 0 \end{cases}$$

예제 11.3 다음 수식을 조판하여라. (힌트: 둘째 줄에서는 두 함수 u 와 u_t 사이에 \sim 를 입력해 두 식을 구분하였다.)

$$\begin{cases} u_{tt} - c^2 \Delta u = 0, & t > 0, \mathbf{x} \in \mathbb{R}^3 \\ u(\mathbf{x}, 0) = \phi(\mathbf{x}), u_t(\mathbf{x}, 0) = \psi(\mathbf{x}), & \mathbf{x} \in \mathbb{R}^3 \\ u(0, t) = 0, & t > 0 \end{cases}$$

mathtools 패키지의 추가 환경

mathtools 패키지를 불러오면 cases 환경 외에

cases* dcases dcases* rcases rcases* drcases drcases*

환경을 추가로 사용할 수 있습니다. 먼저, dcases 환경 안에서는 `\displaystyle` 제어 문자열을 따로 입력하지 않아도 수식을 별행수식 스타일로 조판합니다. rcases 환경은 식을 감싸는 중괄호가 오른쪽에 조판되도록 합니다. drcases는 dcases와 rcases를 합친 환경입니다. 마지막으로, 별표 붙은 환경은 앰퍼샌드 &의 오른쪽 부분을 텍스트 모드에서 조판합니다. 다음 표 11.3의 예시를 참조하십시오. 같은 코드를 환경만 달리하여 조판한 것입니다.

표 11.3: mathtools 패키지의 cases 환경

환경 이름	예시	환경 이름	예시
cases	$\begin{cases} \frac{abc}{def} & \text{cond. 1} \\ \sum_{i=1}^n a_i & \text{cond. 2} \end{cases}$	cases*	$\begin{cases} \frac{abc}{def} & \text{cond. 1} \\ \sum_{i=1}^n a_i & \text{cond. 2} \end{cases}$
dcases	$\begin{cases} \frac{abc}{def} & \text{cond. 1} \\ \sum_{i=1}^n a_i & \text{cond. 2} \end{cases}$	dcases*	$\begin{cases} \frac{abc}{def} & \text{cond. 1} \\ \sum_{i=1}^n a_i & \text{cond. 2} \end{cases}$
rcases	$\left. \begin{array}{ll} \frac{abc}{def} & \text{cond. 1} \\ \sum_{i=1}^n a_i & \text{cond. 2} \end{array} \right\}$	rcases*	$\left. \begin{array}{ll} \frac{abc}{def} & \text{cond. 1} \\ \sum_{i=1}^n a_i & \text{cond. 2} \end{array} \right\}$
drcases	$\left. \begin{array}{ll} \frac{abc}{def} & \text{cond. 1} \\ \sum_{i=1}^n a_i & \text{cond. 2} \end{array} \right\}$	drcases*	$\left. \begin{array}{ll} \frac{abc}{def} & \text{cond. 1} \\ \sum_{i=1}^n a_i & \text{cond. 2} \end{array} \right\}$

```

1 \begin{displaymath}
2   f(x) =
3   \begin{dcases*}
4     0, & \text{if } (x \in \mathbb{R} \setminus \mathbb{Q}) \\
5     \frac{1}{q}, & \text{if } (x = p/q \in \mathbb{Q} \text{ and } (\gcd(p,q) = 1)) \\
6   \end{dcases*}
7 \end{displaymath}

```

$$f(x) = \begin{cases} 0, & \text{if } x \in \mathbb{R} \setminus \mathbb{Q} \\ \frac{1}{q}, & \text{if } x = p/q \in \mathbb{Q} \text{ and } \gcd(p, q) = 1 \end{cases}$$

예제 11.4 아래 수식을 조판하여라. (힌트: 별표 붙은 환경을 사용하면 간편하다.)

$$(-1)^n = \begin{cases} 1, & n \text{이 짝수일 때} \\ -1, & n \text{이 홀수일 때} \end{cases}$$

11.5

바둑판 모양으로 식 배열하기

바둑판 모양으로 여러 식을 배열해야 하는 경우도 있습니다. 이때에는 `array` 환경을 사용합니다. `array` 환경은 아래와 같이 사용합니다.

```
\begin{array}[\langle position \rangle]{\langle column spec \rangle}
```

`\langle position \rangle`: 앞뒤 수식과의 세로 배치

`\langle column spec \rangle`: 열의 개수 및 정렬 방법, 구분 기호, 서식 등

```
\end{array}
```

`\langle column spec \rangle`에는 열의 개수와 각 열의 정렬 방식을 인자로 지정해 주어야 합니다. `l`, `c`, `r`, `p{\langle width \rangle}`를 조합해서 입력하며, 처음 세 가지 명령어는 각각 왼쪽 정렬된 열, 가운데 정렬된 열, 오른쪽 정렬된 열을 생성함을 뜻합니다. 마지막 `p{\langle width \rangle}`는 폭이 `\langle width \rangle`인 열을 생성하며, 이 열의 내용은 양쪽으로 정렬함을 뜻합니다. `p`열에 입력한 내용은 텍스트 모드로 조판됩니다.

다음 예시에서는 `cccc`를 인자로 지정해 가운데로 정렬된 열을 4개 생성하였습니다. 3개의 열을 가진 `array` 환경을 만들 때, 첫째 열은 왼쪽으로, 둘째 열은 오른쪽으로, 셋째 열은 가운데로 정렬하려면 환경의 인자를 `lrc`로 입력하면 됩니다.

```

1 \begin{array}{cccc}
2   (a_1, b_1) & (a_2, b_1) & (a_3, b_1) & \dots \\
3   (a_1, b_2) & (a_2, b_2) & (a_3, b_2) & \dots \\
4   (a_1, b_3) & (a_2, b_3) & (a_3, b_3) & \dots \\
5   \vdots & \vdots & \vdots & \\
6 \end{array}

```

$$\begin{array}{cccc} (a_1, b_1) & (a_2, b_1) & (a_3, b_1) & \dots \\ (a_1, b_2) & (a_2, b_2) & (a_3, b_2) & \dots \\ (a_1, b_3) & (a_2, b_3) & (a_3, b_3) & \dots \\ \vdots & \vdots & \vdots & \end{array}$$

예제 11.5 총 5개의 열을 가진 array 환경을 사용해 수식을 조판하려 한다. 제1열은 오른쪽으로 정렬하고, 제2~4열은 가운데로, 마지막 열은 오른쪽으로 정렬하려면 환경의 인자를 어떻게 지정해야 하는가?

환경의 열의 개수와 정렬 방식을 지정하였다면, 앞에서 알아보았던 matrix 환경과 같은 방법으로 수식을 array 환경 안에 입력합니다. 각각의 식은 행 우선 방식으로 입력하며, 한 행 안에서의 열의 구분은 &로, 행과 행의 구분은 \\로 합니다. 위의 예시를 참고하십시오.

구분선 넣기

array 환경에서는 가로 또는 세로로 구분선을 넣을 수도 있습니다. 세로선은 환경의 인자를 지정할 때 열 지정자(l, c, r, p{width})와 열 지정자 사이에 |를 입력하면 되고, 가로선은 한 행의 작성이 끝나고 \\ 뒤에 \hline을 입력하면 됩니다. 현재 행의 모든 열에 걸쳐 가로선을 그리지 않고, 일부 열에만 가로선을 그리려면 \cline 대신 \cline{i-j}을 사용합니다. {i}열부터 {j}열까지만 가로선이 그려집니다.

```
1 \left( \begin{array}{c|c}
2 A & B \\ \hline C & D
3 \end{array} \right)
```

$$\left(\begin{array}{c|c} A & B \\ \hline C & D \end{array} \right)$$

예제 11.6 array 환경을 사용하여, 아래의 식을 작성하여라.

$$\left[\begin{array}{c|c} A & B \\ 0 & C \end{array} \right]^n = \left[\begin{array}{c|c} A_n & B^n \\ 0 & C_n \end{array} \right]$$

예제 11.7 array 환경 안에 행렬을 중첩하여 작성할 수도 있다. 아래 식을 작성하여라.

$$M = \left(\begin{array}{cccc|c} a & b & c & d & \\ e & f & g & h & B \\ i & j & k & l & \\ \hline & & C & & D \end{array} \right)$$

(힌트: 먼저 array 환경으로 2×2 행렬을 그리고, 이 행렬의 (1,1) 성분에서 matrix 환경을 사용하면 된다.)

이번 절에서는 array 환경을 사용하는 데 필수적인 내용만을 다루었습니다. 환경의 선택 인자인 <position>을 지정하는 방법이나, <column spec>에 지정할 수 있는 다른 인자들, \multicolumn 제어 문자열의 사용 방법은 제14장의 tabular 환경에 대한 설명을 참고하시기 바랍니다. 표를 작성하기 위한 tabular 환경은 array 환경과 사용하는 방법이 비슷하므로, 이번 절의 내용을 잘 알아두면 표를 작성하는 문법을 수월하게 익힐 수 있을 것입니다.

제 12 장

별행수식 입력하기

이번 장에서는 제10장과 제11장에서 알아보았던 수식 문법에 더하여, 별행수식만의 특별한 기능들을 알아보겠습니다. `amsmath` 패키지에서 제공하는 다양한 환경과 제어 문자열을 사용하여, 수식에 번호를 붙이거나 여러 줄에 걸쳐 수식을 작성하는 방법을 다룹니다.

12.1

수식에 번호 붙이기

이번 절에서는 수식에 번호를 붙이는 방법을 알아보겠습니다. \TeX 의 자동 번호 계산 기능을 사용하여 순서에 맞도록 번호가 자동으로 입력되도록 할 수도, 원하는 번호나 기호를 수동으로 붙일 수도 있습니다.

자동으로 번호 붙이기

번호 붙인 별행수식을 작성하는 기본 환경은 `equation`입니다.

```
1 \begin{equation}
2   f(z) \cdot \mathrm{Ind}_{\gamma}(z)
3   = \frac{1}{2\pi i} \int_{\gamma} \frac{f(\xi)}{\xi - z} d\xi
4 \end{equation}
```

$$f(z) \cdot \mathrm{Ind}_{\gamma}(z) = \frac{1}{2\pi i} \int_{\gamma} \frac{f(\xi)}{\xi - z} d\xi \quad (12.1)$$

이 환경을 사용하면 수식의 번호가 자동으로 조판됩니다. `book` 및 `report` 클래스의 경우 위와 같이 장 번호에 종속되어 (<장 번호>.<수식 번호>)의 형식으로 붙습니다. `article` 클래스에서는 장절 번호의 영향을 받지 않고 (<수식 번호>)의 형태로 붙습니다.

문서의 전처리부에 `\numberwithin{equation}{section}`을 추가하면 수식 번호가 절 번호에 종속되도록 변경할 수 있습니다. 이는 \TeX 의 ‘카운터’라는 기능을 사용하는 것으로, 수식 번호의 형식에 관한 자세한 내용은 22.1절을 참조하십시오.

원하는 기호 수동으로 붙이기

수식 번호가 자동으로 계산되도록 하는 대신 원하는 기호를 직접 입력할 수도 있습니다. amsmath 패키지는 `\tag` 제어 문자열을 제공합니다. 이는 `equation`, `eqnarray`, `align` 등 별행수식을 입력하는 환경에서 수식에 기호를 수동으로 붙일 수 있도록 합니다.

```
1 \begin{equation}
2 f(z) \cdot \mathrm{Ind}_\gamma(z)
3 = \frac{1}{2\pi i} \int_\gamma \frac{f(\xi)}{\xi - z} d\xi \tag{*}
4 \end{equation}
```

$$f(z) \cdot \mathrm{Ind}_\gamma(z) = \frac{1}{2\pi i} \int_\gamma \frac{f(\xi)}{\xi - z} d\xi \quad (*)$$

`\tag`의 별표 붙은 버전인 `\tag*`도 같은 방식으로 작동하지만, 기호의 앞뒤에 괄호를 추가하지 않습니다.

수식 번호의 좌우 위치 변경

수식 번호는 수식의 오른쪽에 붙습니다. 제4장에서 설명한 문서 클래스 옵션 중 `leqno`를 사용하면 문서 전체의 수식 번호가 왼쪽 여백에 붙도록 할 수 있습니다. 즉, 소스 파일의 맨 첫 번째 줄에

```
\documentclass[leqno]{book}
```

를 입력하면 다음의 결과를 얻습니다.

```
1 \begin{equation}
2 \sum_{n=1}^{\infty} \frac{1}{n} = \infty
3 \end{equation}
4 \begin{equation}
5 \sum_{n=1}^{\infty} \frac{1}{n^2} =
6 \frac{\pi^2}{6}
7 \end{equation}
```

$$(12.2) \quad \sum_{n=1}^{\infty} \frac{1}{n} = \infty$$

$$(12.3) \quad \sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6}$$

예제 12.1 이 절의 내용을 사용하여 `article` 클래스 문서에서 다음 수식을 구현하여라. 마지막 줄을 제외한 모든 수식 번호가 자동 부여되어야 한다.

$$\zeta(1) = \sum_{n=1}^{\infty} \frac{1}{n} = \infty \quad (1)$$

$$\zeta(2) = \sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6} \quad (2)$$

$$\sum_{n=1}^{\infty} n = \sum_{n=1}^{\infty} \frac{1}{n^{-1}} = \zeta(-1) = -\frac{1}{12} ? \quad (?)$$

예제 12.2 이 절의 내용을 사용하여 article 클래스 문서에서 다음 수식을 구현하여야. 모든 수식 번호가 자동 부여되어야 한다. (힌트: \hat{a} 는 `\hat{a}`로, \check{a} 는 `\check{a}`로 입력한다.)

$$\hat{f}(\xi) = \int_{\mathbb{R}^n} f(x) e^{-2\pi i \xi \cdot x} dx \quad (1)$$

$$f(x) = \int_{\mathbb{R}^n} \hat{f}(\xi) e^{2\pi i \xi \cdot x} d\xi \quad (2)$$

$$\hat{\hat{f}} = \check{\check{f}} = f \quad (3)$$

여러 줄 수식의 번호 붙이기

다음 절에서는 수식을 여러 줄에 걸쳐 작성하는 방법을 설명합니다. 각 환경은 별표 붙은 것과 별표 붙지 않은 것이 쌍으로 존재합니다(예: `align` 환경과 `align*` 환경). 별표 붙은 것은 각 행에서 수식 번호를 조판하지 않고, 별표가 붙지 않은 것은 각 행에 수식 번호를 조판합니다. 별표 붙은 환경에서는 `\tag` 또는 `\tag*` 제어 문자열을 사용하여 수식 번호를 직접 붙일 수 있고, 별표 붙지 않은 환경에서는 원하는 행에 `\nonumber`를 입력해 번호가 그 행에서만 번호가 출력되지 않도록 할 수 있습니다.

한편, `amsmath` 패키지에서 제공하는 `subequations` 환경을 사용하면 다음과 같이 정렬된 수식의 수식 번호를 하나의 수식 번호의 하위 번호로 부여할 수 있습니다.

```
1 \begin{subequations}
2   \begin{align}
3     A(Z(y-x^2)) &= k[x,y]/(y-x^2) \\
4     &\cong k[x,x^2] \\
5     &= k[x]
6   \end{align}
7 \end{subequations}
```

$$A(Z(y-x^2)) = k[x,y]/(y-x^2) \quad (12.4a)$$

$$\cong k[x,x^2] \quad (12.4b)$$

$$= k[x] \quad (12.4c)$$

12.2

여러 줄에 걸쳐 수식 작성하기

여러 줄에 걸쳐 수식을 작성할 때에는 정렬이 필요한지 필요하지 않은지 고려해야 합니다. 단순히 여러 개의 식을 하나의 환경에서 작성하려 하거나, 하나의 식이 길어 여러 줄에 나누어 작성할 때에는 정렬이 필요하지 않습니다. 한편, 등호나 부등호를 기준으로 식이 변화하는 과정을 나타내는 등 특정 지점을 기준으로 여러 수식을 맞춰 조판할 때에는 정렬이 필요합니다.

정렬이 필요하지 않은 수식

하나의 긴 수식을 여러 줄에 나누어 조판할 때에는 `amsmath` 패키지의 `multline` 환경을 사용합니다. 각 줄은 `\\`로 구분하며, 첫 줄은 왼쪽 여백에 가깝게, 마지막 줄은 오른쪽 여백에 가깝게, 나머지

줄은 가운데에 맞추어 조판됩니다.

```
1 \begin{multline}
2 X_{m,n}(x) = A_0 + \sum_{m=1}^{\infty} A_m
   \cos \left( \frac{m\pi}{l} x \right) \phi(x) \\
   + \sum_{n=1}^{\infty} B_n \sin \left( \frac{n\pi}{l} x \right) \psi(x)
3 \end{multline}
```

$$X_{m,n}(x) = A_0 + \sum_{m=1}^{\infty} A_m \cos \left(\frac{m\pi x}{l} \right) \phi(x) + \sum_{n=1}^{\infty} B_n \sin \left(\frac{n\pi x}{l} \right) \psi(x) \quad (12.5)$$

이 환경에 번호를 붙이는 경우, 환경에 입력된 수식 전체가 하나로 취급되어, 마지막 줄에만 번호가 붙습니다. 문서 클래스 옵션으로 leqno를 지정한 경우에는 첫 줄 왼쪽에 번호가 붙습니다.

여러 별행수식을 정렬 없이 이어서 조판하려는 경우에는 `\[...\]`를 연달아 사용하지 않고, gather 환경을 사용합니다. 환경 안에서는 마찬가지로 `\\`를 입력해 줄을 바꿉니다.

```
1 \begin{gather}
2 \bigcap_i A_i^c = \left( \bigcup_i A_i \right)^c \\
3 \bigcup_i A_i^c = \left( \bigcap_i A_i \right)^c
4 \end{gather}
```

$$\bigcap_i A_i^c = \left(\bigcup_i A_i \right)^c \quad (12.6)$$

$$\bigcup_i A_i^c = \left(\bigcap_i A_i \right)^c \quad (12.7)$$

정렬이 필요한 수식

앞선 예시에서는 두 수식이 같은 꼴이었으므로 따로 정렬을 하지 않아도 줄이 맞습니다. 그러나 수식의 길이가 다른 경우 등호나 부등호를 기준으로 정렬되어 있지 않다면 가독성을 해치게 됩니다. 특히, 다음과 같이 논리적 단계를 거쳐 전개되는 수식의 경우 정렬을 해 주는 것이 좋습니다.

```
1 \begin{gather*}
2 \int f'g \, dx = \int [(fg)' - fg'] \, dx \\
   \\
3 = fg - \int fg' \, dx
4 \end{gather*}
```

$$\begin{aligned} \int f'g \, dx &= \int [(fg)' - fg'] \, dx \\ &= fg - \int fg' \, dx \end{aligned}$$

이때에는 align 환경을 사용해 수식을 정렬해 줄 수 있습니다. 정렬 지점을 등호나 부등호의 왼쪽에 `&`를 입력해 지정해 주면 됩니다.

```

1 \begin{align*}
2 \int f'g \, dx &= \int [(fg)' - fg'] \\
3 &\quad \int f'g \, dx \\
4 &= fg - \int fg' \, dx
5 \end{align*}

```

$$\begin{aligned}\int f'g \, dx &= \int [(fg)' - fg'] \, dx \\ &= fg - \int fg' \, dx\end{aligned}$$

한편, 수식 환경 안에서 사용하는 `aligned` 환경도 있습니다(이 환경에는 별표 붙은 버전이 따로 없습니다). 이는 별행수식의 일부를 정렬하는 데 사용합니다. 이 환경 자체는 수식 번호를 붙이지 않지만, 이 환경을 포함하고 있는 바깥 환경에서는 `aligned` 환경에 입력한 수식을 하나의 ‘덩어리’로 취급하므로, 수식 번호도 하나만 붙입니다. 예를 들어, 아래와 같이 사용할 수 있습니다.

```

1 \begin{displaymath}
2 \begin{aligned}
3 \bigcap_i A_i^c &= \left( \bigcup_i A_i \right)^c \\
4 \bigcup_i A_i^c &= \left( \bigcap_i A_i \right)^c \\
5 \end{aligned}
6 \tag{De Morgan's Laws}
7 \end{displaymath}

```

$$\begin{aligned}\bigcap_i A_i^c &= \left(\bigcup_i A_i \right)^c \\ \bigcup_i A_i^c &= \left(\bigcap_i A_i \right)^c\end{aligned} \quad (\text{De Morgan's Laws})$$

예제 12.3 `align` 환경을 이용하여 다음 수식을 조판하여라.

$$\begin{aligned}\int f'g \, dx &= \int [(fg)' - fg'] \, dx && (\because \text{Product Rule}) \\ &= fg - \int fg' \, dx && (\because \text{FTC})\end{aligned}$$

예제 12.4 `equation` 환경과 `aligned` 환경을 이용하여 다음 수식을 조판하여라. (수식 번호가 두 행의 중간에 위치한다. 수식 번호는 `\tag`를 이용하여 입력하여라.)

$$\begin{aligned}A(Z(xy-1)) &= k[x, y]/(xy-1) \\ &\cong k[x, x^{-1}]\end{aligned} \quad (*)$$

여러 수식을 정렬하는 도중 일반 텍스트를 입력한다면 `amsmath` 패키지의 `\intertext` 제어 문자열을 사용할 수 있습니다. 만약 이 제어 문자열을 사용했을 때의 위아래 공백이 넓으면 `mathtools` 패키지의 `\shortintertext` 제어 문자열을 사용하세요.

```

1 By Taylor's series we have
2 \begin{align*}
3 \cos x &= 1 - \frac{1}{2!}x^2 + \\
&\frac{1}{4!}x^4 - \cdots \\
4 \text{\shortintertext{and}} \\
5 \sin x &= x - \frac{1}{3!}x^3 + \\
&\frac{1}{5!}x^5 - \cdots. \\
6 \text{\intertext{Therefore}} \\
7 \cos x + i \sin x &= 1 + ix + \\
&\frac{1}{2!}(ix)^2 + \\
&\frac{1}{3!}(ix)^3 + \cdots \\
8 &= e^{ix}. \\
9 \end{align*}

```

By Taylor's series we have

$$\cos x = 1 - \frac{1}{2!}x^2 + \frac{1}{4!}x^4 - \cdots$$

and

$$\sin x = x - \frac{1}{3!}x^3 + \frac{1}{5!}x^5 - \cdots$$

Therefore

$$\begin{aligned} \cos x + i \sin x &= 1 + ix + \frac{1}{2!}(ix)^2 + \frac{1}{3!}(ix)^3 + \cdots \\ &= e^{ix}. \end{aligned}$$

eqnarray 환경에 관한 조언

참고 | eqnarray 환경을 모르시는 분은 이 부분을 넘어가셔도 됩니다.

LaTeX에서는 여러 줄에 걸친 정렬된 수식을 작성하는 환경으로 eqnarray를 제공합니다. 이 환경에서는 앰퍼샌드 두 개를 사용해 각 행의 식을 정렬하고, \\로 행을 구분합니다.

```

1 \begin{eqnarray*}
2 \bigcap_i A_i^c &= & \biggl( \bigcup_i A_i \biggr)^c \\
&& \\
3 \bigcup_i A_i^c &= & \biggl( \bigcap_i A_i \biggr)^c \\
4 \end{eqnarray*}

```

$$\begin{aligned} \bigcap_i A_i^c &= \left(\bigcup_i A_i \right)^c \\ \bigcup_i A_i^c &= \left(\bigcap_i A_i \right)^c \end{aligned}$$

하지만, eqnarray 환경은 일반적인 수식에 비해 &가 입력된 부분의 간격을 넓게 조판하므로, 조판 결과가 보기 좋지 않습니다. eqnarray 환경 대신 앞서 설명한 align 환경을 사용하십시오.

12.3

여러 열의 수식 동시에 정렬하기

align 환경을 사용하면 여러 행, 여러 열에 입력한 수식도 정렬하여 조판할 수 있습니다. 한 행에서 수식과 수식을 구분할 때에도 &를 입력하면 됩니다.

```

1 \begin{align*}
2 \quad \bigcap_i A_i^c \ \&= \left( \bigcup_i A_i \right)^c \\
3 \ \&\text{\texttt{\textbackslash therefore}} A^c \ \cap B^c \ \&= (A \cup B)^c \\
4 \quad \bigcap_i A_i^c \ \&= \left( \bigcup_i A_i \right)^c \\
5 \ \&\text{\texttt{\textbackslash therefore}} A^c \ \cup B^c \ \&= (A \cap B)^c \\
6 \end{align*}

```

$$\bigcap_i A_i^c = \left(\bigcup_i A_i \right)^c \quad \therefore A^c \cap B^c = (A \cup B)^c$$

$$\bigcup_i A_i^c = \left(\bigcap_i A_i \right)^c \quad \therefore A^c \cup B^c = (A \cap B)^c$$

위 예시에서 짝수번째 &가 각 열을 구분하는 역할을 하며, 홀수번째 &가 각 수식 내에서의 기준점 역할을 합니다.

예제 12.5 align* 환경을 사용하여 다음을 조판하여라.

$$\begin{aligned} u_{xx} &= c^2 u_{tt} + f(x, t) & \text{(DE)} \\ u(x, 0) &= \phi(x), & u_x(x, 0) = \psi(x) & \text{(IC)} \\ u(0, t) &= g(t) & & \text{(BC)} \end{aligned}$$

이외에도, amsmath 패키지에서는 alignat과 flalign 환경을 제공합니다. alignat 환경은 각 열 사이의 간격을 조절할 수 있도록 하며, flalign 환경은 첫째 열과 마지막 열을 양 끝 여백에 맞추어 수식을 조판합니다. 필요한 경우 패키지 문서를 확인하십시오.

제 13 장

가환 도표 그리기

가환 도표(commutative diagram)는 여러 대상과 그 사이의 여러 사상들의 호환 관계를 나타내는 도표로, 주로 범주론과 대수학 및 그 응용 분야에서 많이 등장합니다. 이번 장에서는 tikz-cd 패키지에서 제공하는 tikzcd 환경을 이용하여 가환 도표를 그리는 방법을 알아보겠습니다.

13.1

tikzcd 환경

먼저, 전처리부에 `\usepackage{tikz-cd}`를 입력해 tikz-cd 패키지를 불러옵니다. 이후, 수식 작성 환경 안에서 tikzcd 환경을 사용해 가환 도표를 그립니다.¹

```
\begin{tikzcd}[\langle options \rangle]
```

\langle options \rangle: 가환 도표 전체에 지정할 옵션(들)

```
\end{tikzcd}
```

이 환경은 하나의 선택 인자를 입력받으며, 여기에는 가환 도표 전체에 적용할 옵션을 지정합니다. 여러 옵션을 지정할 수도 있으며, 이때 각 옵션은 쉼표를 사용해 구분합니다. (13.1)은 tikzcd 환경을 사용해 그린 가환 도표의 예시입니다.

¹tikzcd 환경은 텍스트 모드나 수식 입력 모드에 관계없이 사용할 수 있지만, 가환 도표는 일반적으로 수식으로 취급되므로 수식 작성 환경에서 사용하는 것이 바람직합니다.

```

1 \begin{equation}
2   \begin{tikzcd}[column sep=2cm]
3     U\arrow[r]\arrow[d,swap,"i"] &
4     X\arrow[d,"f"] \\
5     T\arrow[r]\arrow[ur,dashed] & Y
6   \end{tikzcd}
7 \end{equation}

```

이 도표에는 대상, 화살표, 라벨 등 많은 요소가 포함되어 있습니다. 이들을 각각 어떻게 입력하는지 차례대로 알아보겠습니다.

13.2

대상 배치하기

(13.1)에는 U , X , T , Y 라는 네 가지 대상이 포함되어 있습니다. 대상을 배치할 때에는 array 환경에서처럼, 한 행 안에서 &로 열을 바꾸고 \\로 행을 바꿉니다.

tikzcd 환경의 *options* 부분에 `row sep=<length>`를 입력해 행 사이의 간격을, `column sep=<length>`를 입력해 열 사이의 간격을 조정할 수 있습니다. *<length>*에는 수치를 직접 지정할 수 있으며, 간단히 `small`을 입력해 표준보다 좁게, `large`를 입력해 표준보다 넓게 할 수도 있습니다. 기본값은 `normal`입니다.

이상의 내용을 바탕으로 (13.1)의 대상들을 배치하면 아래와 같습니다.

```

1 \begin{equation}
2   \begin{tikzcd}[column sep=2cm]
3     U & X \\
4     T & Y
5   \end{tikzcd}
6 \end{equation}

```

13.3

화살표 그리기

대상을 배치했다면, 대상 사이의 관계를 나타내는 화살표를 그려야 합니다. 화살표를 그린 후, 화살표의 여러 옵션을 지정하는 방법을 알아보겠습니다.

화살표 방향 지정하기

화살표는 `\arrow` 또는 `\ar` 제어 문자열을 사용해 그립니다. 둘은 사용 방법이 완전히 같습니다.

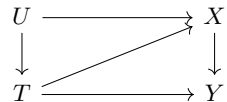
`\arrow[⟨options⟩]``\ar[⟨options⟩]``⟨options⟩`: 화살표에 지정할 옵션

`⟨options⟩`에는 화살표의 방향을 비롯해 여러 옵션을 지정하며, 각 옵션은 쉼표를 사용해 구분합니다.

화살표에 지정할 수 있는 여러 옵션 중 필수인 것은 화살표의 방향입니다. `\arrow`가 입력된 칸이 화살표의 시작점이 되며, 시작점을 기준으로 u는 위쪽, d는 아래쪽, l은 왼쪽, r는 오른쪽 칸을 가리킵니다. 인접하지 않은 칸을 가리키는 화살표를 그리고자 한다면 u, d, l, r를 조합하여 방향을 지정할 수 있습니다. 예를 들어, 오른쪽으로 1칸, 위쪽으로 1칸에 있는 대상을 가리키도록 하려면 ur를, 오른쪽으로 2칸, 아래쪽으로 1칸에 있는 대상을 가리키도록 하려면 drr를 입력하면 됩니다. 다른 옵션들은 필요하지 않다면 추가하지 않아도 되지만, 화살표의 방향이 지정되지 않거나 대상이 없는 칸을 향해 지정된다면 컴파일 과정에서 오류가 발생하므로 주의해야 합니다.

(13.1)에는 5개의 화살표가 있습니다. (13.2)에 이 5개의 화살표를 추가하면 아래와 같습니다.

```
1 \begin{equation}
2   \begin{tikzcd}[column sep=2cm]
3     U\arrow[r]\arrow[d] & X\arrow[d] \\
4     T\arrow[r]\arrow[ur] & Y
5   \end{tikzcd}
6 \end{equation}
```



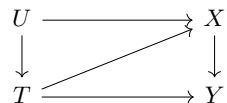
(13.3)

이와 동치로, `\arrow`나 `\ar`는 아래와 같이 방향을 인자로 지정하여 사용할 수도 있습니다.

`\arrow[⟨options⟩]{⟨direction⟩}``\ar[⟨options⟩]{⟨direction⟩}``⟨options⟩`: 화살표가 가리킬 방향을 제외한 옵션`⟨direction⟩`: 화살표가 가리킬 방향

이 문법을 사용하면 (13.3)과 동일한 도표를 아래의 코드를 입력해 얻을 수 있습니다.

```
1 \begin{displaymath}
2   \begin{tikzcd}[column sep=2cm]
3     U \arrow{d}\arrow{r} & X\arrow{d} \\
4     T \arrow{r}\arrow{ur} & Y
5   \end{tikzcd}
6 \end{displaymath}
```



방향을 포함한 축약된 화살표 입력 제어 문자열인

`\uar`, `\dar`, `\lar`, `\rar`, `\ular`, `\urar`, `\dlar`, `\drar`

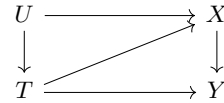
를 사용할 수도 있습니다. `\rar[⟨options⟩]`는 `\arrow[⟨options⟩]{r}`와 동치이며, 나머지 제어 문자 열도 마찬가지로 사용할 수 있습니다.

이상의 방법과 반대로, 화살표의 끝점에 `\arrow`를 입력하고 옵션으로 `from=⟨direction⟩`을 입력해 시작점을 지정할 수도 있습니다. `⟨direction⟩`에는 앞과 마찬가지로 `u`, `d`, `l`, `r`를 조합해 입력합니다. 하지만, 이러한 방법은 13.3절에서 설명하는 입체 가환 도표와 같은 특수한 상황이 아니라면 잘 사용되지 않습니다.

```

1 \begin{displaymath}
2   \begin{tikzcd}[column sep=2cm]
3     U\arrow[d]\arrow[r] & X\arrow[from=dl] \\
4     T\arrow[r] & Y\arrow[from=u]
5   \end{tikzcd}
6 \end{displaymath}

```



화살표의 머리와 꼬리 모양 바꾸기

다른 옵션을 지정하지 않았을 때의 화살표는 머리(끝점)의 모양은 화살 모양이고 꼬리(시작점)에는 특별한 모양이 없습니다. 표 13.1의 명령어를 `\arrow`나 `\ar`의 `⟨options⟩` 부분에 추가하면 화살표의 모양을 변경할 수 있습니다. 오른쪽 열의 명령어를 사용하면 화살표의 앞뒤가 바뀐 모양으로 그려 집니다.

표 13.1: tikzcd 환경의 화살표 머리 및 꼬리 옵션

명령어	모양	명령어	모양
<code>rightarrow</code>	\longrightarrow	<code>leftarrow</code>	\longleftarrow
<code>Rrightarrow</code>	\Longrightarrow	<code>Lleftarrow</code>	\Longleftarrow
<code>leftrightharrow</code>	\longleftrightarrow		
<code>Leftrightharrow</code>	\Longleftrightarrow		
<code>mapsto</code>	\mapsto	<code>mapsfrom</code>	\mapsfrom
<code>Mapsto</code>	\Mapsto	<code>Mapsfrom</code>	\Mapsfrom
<code>hookrightarrow</code>	\hookrightarrow	<code>hookleftarrow</code>	\hookleftarrow
<code>rightarrowtail</code>	\rightarrowtail	<code>leftarrowtail</code>	\leftarrowtail
<code>twoheadrightarrow</code>	\twoheadrightarrow	<code>twoheadleftarrow</code>	\twoheadleftarrow
<code>dashrightarrow</code>	\dashrightarrow	<code>dashleftarrow</code>	\dashleftarrow
<code>rightsquigarrow</code>	\rightsquigarrow	<code>leftsquigarrow</code>	\leftsquigarrow
<code>leftrightsquigarrow</code>	\leftrightsquigarrow		
<code>rightharpoonup</code>	\rightharpoonup	<code>leftharpoonup</code>	\leftharpoonup
<code>rightharpoondown</code>	\rightharpoondown	<code>leftharpoondown</code>	\leftharpoondown
<code>dash</code>	---		
<code>equal</code>	=		

이외에도, 다른 명령어를 조합해 더 다양한 모양의 화살표를 그릴 수도 있습니다. `to` head는 머리를 \rightarrow 의 머리 모양으로, `maps to`는 꼬리를 \mapsto 의 꼬리 모양으로, `hook`과 `hook'`은 꼬리를 \hookrightarrow 와 \hookleftarrow 의 꼬리 모양으로 바꾸어 주며, `tail`은 꼬리를 \rightarrow 의 꼬리 모양으로, `two heads`는 머리를 \rightarrow 의 머리 모양으로 바꾸어 줍니다. `harpoon`과 `harpoon'`은 머리를 \rightarrow 와 \rightarrow 의 머리 모양으로 바꾸어 줍니다. `dashed`나 `squiggly`를 입력하면 화살표의 양 끝에는 영향을 주지 않고 줄을 파선이나 물결 선으로 바꾸며, `no head`는 머리에, `no tail`은 꼬리에 특별한 모양이 없도록 해 줍니다. 예를 들어, $\rightarrow\text{---}\rightarrow$ 는 `\rar[tail, two heads, dashed]`를 입력해 그릴 수 있습니다.

(13.1)에서는 사선 방향 화살표에 파선을 사용하였습니다. 이를 (13.3)에 적용하면 다음의 결과를 얻습니다.

```
1 \begin{equation}
2 \begin{tikzcd}[column sep=2cm]
3   U\arrow[r]\arrow[d] & X\arrow[d] \\
4   T\arrow[r]\arrow[ur,dashed] & Y \\
5 \end{tikzcd}
6 \end{equation}
```

화살표 굽히기

경우에 따라서는 두 대상을 곧은 선이 아닌 굽은 선으로 이어야 할 때도 있습니다. 이때에는 화살표의 옵션으로 `bend left`와 `bend right`를 지정할 수 있습니다. 이때 `left`와 `right`는 출발점에서 도착점을 바라보았을 때를 기준으로 각각 왼쪽과 오른쪽으로 굽히는 것을 뜻합니다.

```
1 \begin{equation}
2 \begin{tikzcd}[column sep=2cm]
3   U\arrow[r]\arrow[d] & X\arrow[d] \\
4   T\arrow[r]\arrow[ur,dashed] & Y \\
5 \end{tikzcd}
6 \end{equation}
```

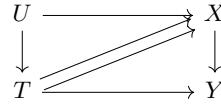
화살표의 위치 이동하기

두 대상 사이에 2개 이상의 사상이 존재함을 표시할 때, 단순히 `\arrow`를 두 번 사용하면 두 화살표가 겹쳐져 그려집니다. 이때에는 `shift left`와 `shift right` 옵션을 사용해 각 화살표의 위치를 조정해야 합니다.

```

1 \begin{displaymath}
2   \begin{tikzcd}[column sep=2cm]
3     U\arrow[r]\arrow[d] & X\arrow[d] \\
4     T\arrow[ur,shift right] & \\
5     \arrow[ur,shift left]\arrow[r] & Y
6   \end{tikzcd}
7 \end{displaymath}

```

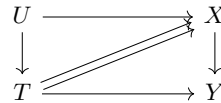


`shift left=<length>`, `shift right=<length>`와 같이 화살표를 이동할 거리를 직접 지정할 수도 있습니다.

```

1 \begin{displaymath}
2   \begin{tikzcd}[column sep=2cm]
3     U\arrow[r]\arrow[d] & X\arrow[d] \\
4     T\arrow[ur,shift right=.5mm] & \\
5     \arrow[ur,shift left=.5mm]\arrow[r] & Y
6   \end{tikzcd}
7 \end{displaymath}

```



입체 가환 도표

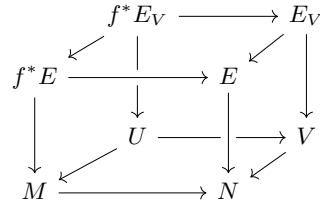
`crossing over` 옵션을 사용하면 다른 화살표와 교차하는 지점에서 교차하는 화살표가 잠시 끊어지게 만들 수 있습니다. 이는 입체 도표에서 앞쪽에 놓인 화살표를 표현하기 위해 사용될 수 있습니다. 단, 이 경우 끊어지도록 하려는(`crossing over` 옵션을 지정하는) 화살표의 코드가 더 앞쪽에 있어야 합니다.

예를 들어, 다음 코드에서는 $U \rightarrow V$ 의 화살표를 그리기 위한 코드가 $E \rightarrow N$ 의 화살표를 그리기 위한 코드보다 앞쪽에 놓여야 하므로, $E \rightarrow N$ 의 화살표는 N 이 적힌 칸에 `from=uu`를 이용하여 입력해야 합니다.

```

1 \begin{displaymath}
2   \begin{tikzcd}[column sep=small,row
   sep=small]
3     & f^*E_V\arrow{dl}\arrow{rr}\arrow{dd}
4       && E_V\arrow{dl}\arrow{dd} \\
5     f^*E\arrow{rr, crossing
   over}\arrow{dd}
6       && E \\
7     & U\arrow{dl}\arrow{rr} && V\arrow{dl}
8       \\
9     M\arrow{rr}
10      && N\arrow[from=uu, crossing over]
11 \end{tikzcd}
12 \end{displaymath}

```



13.4

화살표에 라벨 붙이기

두 대상 사이의 관계를 나타내는 화살표에는 그 관계를 나타내는 함수 등을 라벨로 붙일 수 있습니다. 라벨은 `\arrow[options]` 문법의 `options` 부분에서 큰따옴표로 감싼 코드를 옵션으로 지정해 입력합니다. 즉, 라벨이 f 인 오른쪽 방향 화살표는 `\arrow[r, "f"]`로 입력합니다. 이때, 라벨의 내용이 쉼표를 포함한다면 내용을 중괄호로 감싸야 합니다.

라벨에 추가로 지정할 옵션이 있다면 그 라벨 뒤에 (쉼표 없이) 이어서 입력합니다. 작은따옴표 (') 또는 `swap`은 라벨을 반대 방향으로 넘기도록 하며, `near start`와 `near end`는 각각 라벨을 시작점과 끝점에 가깝게 조판되도록 합니다. 즉, 라벨 f 가 아래쪽에 붙는 오른쪽 방향 화살표는 `\arrow[r, "f" swap]`로 입력합니다. 여러 옵션을 동시에 지정하려는 경우 중괄호 안에 각 옵션을 쉼표로 구분하여 입력하면 됩니다.

단, 해당 화살표의 모든 라벨을 반대 방향으로 넘기고자 한다면, `swap`을 라벨 뒤에 입력하는 대신 `\arrow[r, "f", swap]`과 같이 화살표 자체의 옵션으로 추가할 수 있습니다.

```

1 \begin{displaymath}
2   \begin{tikzcd}
3     A\arrow[r, "\phi" near start, "\psi"
4       swap, "\eta" near end] & B
5   \end{tikzcd}
6 \end{displaymath}

```

$$A \xrightarrow[\psi]{\phi, \eta} B$$

(13.1)의 화살표 중 두 개에는 i 와 f 라는 라벨이 붙어 있습니다. 여기에서 알아본 방법을 사용하여 (13.4)의 코드에서 라벨을 추가하면 아래와 같습니다.

```

1 \begin{equation}
2   \begin{tikzcd}[column sep=2cm]
3     U\arrow[r]\arrow[d,swap,"i"] &
4     X\arrow[d,"f"] \\
5     T\arrow[r]\arrow[ur,dashed] & Y
6   \end{tikzcd}
7 \end{equation}

```

$$\begin{array}{ccc}
 U & \xrightarrow{\quad} & X \\
 i \downarrow & \nearrow \text{dashed} & \downarrow f \\
 T & \xrightarrow{\quad} & Y
 \end{array} \quad (13.5)$$

예제 13.1 다음 가환 도표를 작성하여라.

$$\begin{array}{ccc}
 \pi_1(A, a_0) & \xrightarrow{i_*} & \pi_1(\mathbb{R}^n, a_0) \\
 & \searrow h_* & \downarrow \tilde{h}_* \\
 & & \pi_1(Y, y_0)
 \end{array}$$

라벨을 붙이는 다른 문법

위의 방법을 사용하면 한 라벨에 여러 옵션을 지정하려는 경우 코드가 복잡해질 수 있다는 문제점이 있습니다. 이때에는 `\arrow[options]{direction}` 문법을 사용할 수 있습니다. 각 라벨은 `[label options]{label}`의 문법을 사용해 입력하며, `label`에는 라벨의 내용을 (따옴표 없이) 작성하고, `label options`에는 그 라벨에 지정할 옵션을 작성합니다. 이를 `\arrow[options]{direction}` 뒤에 이어 붙이면 됩니다.

```

1 \begin{displaymath}
2   \begin{tikzcd}
3     A\arrow[dashed]{r}[near
4       start]{\phi}[swap]{\psi}[near
5       end]{\eta} & B
6   \end{tikzcd}
7 \end{displaymath}

```

$$A \overset{\phi}{\underset{\psi}{\dashrightarrow}} B$$

`\rar[options]` 등 방향이 지정된 제어 문자열은 `\arrow[options]{r}`와 동치이므로 마찬가지로 라벨을 뒤에 덧붙일 수 있습니다.

```

1 \begin{displaymath}
2   \begin{tikzcd}
3     A\rar[dashed]{r}[near
4       start]{\phi}[swap]{\psi}[near
5       end]{\eta} & B
6   \end{tikzcd}
7 \end{displaymath}

```

$$A \overset{\phi}{\underset{\psi}{\dashrightarrow}} B$$

제 IV 편

표와 그림 다루기

제 IV 편에서는 \LaTeX 문서에 표와 그림을 넣는 방법을 알아봅니다. 표를 작성할 때에는 `tabular` 환경을 사용하며, 그림을 삽입할 때에는 `graphicx` 패키지의 `\includegraphics` 제어 문자열을 사용합니다. 마지막 장에서는 표와 그림을 배치하는 방법을 알아봅니다.

제14장에서는 표에 대해 다룹니다. 표를 작성하는 기본적인 문법에서 비롯해, 몇 가지 패키지를 통해 더 보기 좋은 표를 작성하는 방법을 알아봅니다.

제15장에서는 그림에 대해 다룹니다. 그림을 삽입하고, 삽입한 그림의 크기를 조정하는 등 지정할 수 있는 여러 옵션을 설명합니다.

제16장에서는 표와 그림을 배치하는 방법을 배웁니다. \LaTeX 에서 이들은 기본적으로 문자와 같이 취급되지만, '유동 개체'로 취급하여, 본문과 별개로 배치할 수도 있습니다. 또, 표와 그림에 캡션을 다는 방법도 설명합니다.

제 14 장

표 작성하기

이번 장에서는 tabular 환경과 array 환경을 사용해 표를 작성하는 방법을 알아봅니다. 또, 다양한 인자를 조정하여 표의 모습을 변경하고, 패키지를 사용해 표를 꾸미는 방법을 알아봅니다.

두 환경은 환경 안에 입력된 내용이 조판되는 모드만 제외하면 같은 동작을 하고, 두 환경을 사용하는 문법도 같습니다. 따라서 두 환경을 사용하는 방법을 이번 장에서 함께 설명하겠습니다.

14.1

기본적인 표 그리기

다음은 tabular 환경과 array 환경을 사용한 가장 간단한 예시입니다.

```

1 \begin{center}
2   \begin{tabular}{|l|c|r|}
3     \hline
4     왼쪽 & 가운데 & 오른쪽 \\ \hline
5     정렬된 & 정렬된 & 정렬된 \\ \hline
6     1열 & 2열 & 3열 \\ \hline
7   \end{tabular}
8 \end{center}

```

왼쪽	가운데	오른쪽
정렬된	정렬된	정렬된
1열	2열	3열

```

1 \begin{displaymath}
2   \begin{array}{|l|c|r|}
3     \hline
4     left & center & left \\ \hline
5     aligned & aligned & aligned \\ \hline
6     column & column & column \\ \hline
7   \end{array}
8 \end{displaymath}

```

<i>left</i>	<i>center</i>	<i>left</i>
<i>aligned</i>	<i>aligned</i>	<i>aligned</i>
<i>column</i>	<i>column</i>	<i>column</i>

이 예시에서 볼 수 있듯, 두 환경은 인자를 하나 입력받습니다. l, c, r를 입력한 개수만큼 열이 생성됩니다. 이때 l로 생성된 열은 왼쪽에, c로 생성된 열은 가운데에, r로 생성된 열은 오른쪽에 맞추어 내용이 정렬됩니다. 각 열을 생성하는 문자의 좌우에 세로선 |을 입력하면 표에 세로 구분선이 그려집니다.

환경 안에는 표에 작성할 내용을 입력합니다. 한 행의 내용을 모두 작성한 뒤 다음 행으로 넘어가는 순서로 입력하며, 행과 행의 구분은 \\로, 한 행 안에서의 열 구분은 &로 합니다. 행과 행 사이의 구분선은 \\ 뒤에 \hline을 입력해 그릴 수 있습니다.

위의 예시를 작성하기 위한 코드를 시각적으로 나타내면 그림 14.1과 같습니다.

```
\begin{tabular}
{ | l | c | r | }
\hline
왼쪽 & 가운데 & 오른쪽 \\
\hline
정렬된 & 정렬된 & 정렬된 \\
\hline
1열 & 2열 & 3열 \\
\hline
\end{tabular}
```

그림 14.1: 표를 작성하는 코드의 시각화

앞 페이지의 예시에서 확인할 수 있듯, tabular 환경과 array 환경은 동일한 문법으로 사용됩니다. 조판 결과에서는 약간의 차이가 있는데, tabular 환경은 표의 내용을 텍스트 모드에서, array 환경은 표의 내용을 수식 입력 모드에서 조판합니다. array 환경은 수식 입력 모드에서 사용해야 한다는 점도 주의하시기 바랍니다.

예제 14.1 tabular 환경을 사용하여 다음 표를 생성하여라. center 환경 안에서 표를 작성하면 그려진 표가 페이지의 가운데에 배치될 것이다.

과목	중간고사	기말고사
선형대수학	79	93
해석학 I	83	88

예제 14.2 array 환경을 사용하여 다음 표를 작성하여라.

e_1	e_2	$e_1 \wedge e_2$	$e_1 \vee e_2$	$e_1 \rightarrow e_2$
T	T	T	T	T
T	F	F	T	F
F	T	F	T	T
F	F	F	F	T

l, c, r로 생성된 열은 입력된 내용을 줄바꿈 없이 한 줄로 조판합니다. 즉, 조판할 내용이 길면 페이지의 여백을 침범한 채로 내용이 조판됩니다. 만약 칸에 긴 내용을 입력하고 줄바꿈이 일어나도록

하려면 열을 생성하는 인자로 $p\langle width \rangle$ 를 사용하면 됩니다. $\langle width \rangle$ 에는 해당 열의 폭을 지정하면 되며,¹ 이 폭에 맞추어 글줄의 길이가 결정되고 줄바꿈 처리가 진행됩니다. 이때, array 환경을 사용하더라도 p 인자로 생성한 열의 내용은 텍스트 모드에서 조판되므로 주의하시기 바랍니다.

```

1 \begin{center}
2   \begin{tabular}{c|p{4cm}}
3     \hline
4     1행 & \texttt{p} 인자로 생성한 열에서는 긴
        내용이 여러 줄로 조판됩니다. \\ \hline
5     2행 & 표 안에는 지나치게 긴 내용을 입력하지 마십
        시오. \\ \hline
6   \end{tabular}
7 \end{center}

```

1행	p 인자로 생성한 열에서는 긴 내용이 여러 줄로 조판됩니다.
2행	표 안에는 지나치게 긴 내용을 입력하지 마십시오.

예제 14.3 다음 표를 작성하여라. 마지막 열의 폭은 7cm로 하여라.

이름	기호	설명
논리합	\vee	두 명제 중 하나 이상이 참이면 참을 반환하고, 둘이 모두 거짓이면 거짓을 반환한다.
논리곱	\wedge	두 명제가 모두 참이면 참을 반환하고, 하나 이상이 거짓이면 거짓을 반환한다.
부정	\neg	명제가 참이면 거짓을 반환하고, 거짓이면 참을 반환한다.

14.2

환경의 세부 문법

앞 절의 내용만으로도 기본적인 표를 작성하는 데에는 충분할 것입니다. 하지만 더 다양한 표를 더 편리하게 작성하려면 환경을 사용하는 더 자세한 방법을 알아야 합니다. 이번 절에서는 tabular 및 array 환경을 사용하는 세부 문법을 설명하겠습니다. 한편, array 패키지 [4]를 불러오면 L^AT_EX의 기본 표 작성 기능을 확장할 수 있습니다. 이 내용도 이번 절에서 함께 알아보겠습니다.

tabular 환경과 array 환경은 다음과 같이 두 가지 인자를 입력받습니다.

`\begin{tabular}[\langle position \rangle]{\langle spec \rangle}`

$\langle position \rangle$: 앞뒤 텍스트와의 세로 정렬

$\langle spec \rangle$: 열의 개수와 각 열의 속성

`\end{tabular}`

`\begin{array}[\langle position \rangle]{\langle spec \rangle}`

$\langle position \rangle$: 앞뒤 수식과의 세로 정렬

$\langle spec \rangle$: 열의 개수와 각 열의 속성

`\end{array}`

필수 인자인 $\langle spec \rangle$ 에는 표 14.1의 인자를 조합하여 입력하며, 선택 인자인 $\langle position \rangle$ 에는 t, c, b

¹길이를 지정하는 방법은 22.2절을 참조하십시오.

중 하나를 입력합니다. 선택 인자를 생략하면 기본값인 `c`가 사용됩니다. 이제 각 인자를 지정하는 방법을 종류별로 나누어 설명하겠습니다.

표 14.1: `table` 및 `array` 환경의 열 속성 인자

인자	역할
<code>l</code>	왼쪽 정렬된 한 줄짜리 열을 생성합니다.
<code>c</code>	가운데 정렬된 한 줄짜리 열을 생성합니다.
<code>r</code>	오른쪽 정렬된 한 줄짜리 열을 생성합니다.
<code>p{<width>}</code>	폭이 <code><width></code> 인 위쪽 정렬된 문단형 열을 생성합니다.
<code>m{<width>}¹</code>	폭이 <code><width></code> 인 가운데 정렬된 문단형 열을 생성합니다.
<code>b{<width>}¹</code>	폭이 <code><width></code> 인 아래쪽 정렬된 문단형 열을 생성합니다.
<code> </code>	열과 열 사이에 구분선을 입력합니다.
<code>@{<text>}</code>	열과 열 사이의 간격을 삭제하고 해당 위치에 <code><text></code> 의 내용을 조판합니다.
<code>!{<text>}¹</code>	열과 열 사이의 간격을 삭제하지 않고 해당 위치에 <code><text></code> 의 내용을 조판합니다.
<code>>{<format>}¹</code>	오른쪽에 있는 열의 첫머리에 <code><format></code> 에 입력한 내용을 삽입합니다.
<code><{<format>}¹</code>	왼쪽에 있는 열의 끝부분에 <code><format></code> 에 입력한 내용을 삽입합니다.
<code>*{<num.>}{<spec.>}</code>	<code><spec.></code> 에 입력된 내용을 <code><num.></code> 번 반복하여 열을 생성합니다.

¹ `array` 패키지를 불러와야 사용할 수 있습니다.

열의 개수와 속성 지정하기

열을 생성할 때에는 `l`, `c`, `r`, `p{<width>}`, `m{<width>}`, `b{<width>}`를 사용할 수 있습니다(`m`과 `b`는 `array` 패키지를 불러와야 사용할 수 있습니다). `l`, `c`, `r`는 표의 내용을 줄바꿈 없이 한 줄로 조판하는 열을 생성하며, 해당 열에 입력된 내용은 각각 왼쪽, 가운데, 오른쪽에 맞춰 정렬됩니다. 그림 14.2를 참고하십시오.

더미 텍스트 더미 텍스트	더미 텍스트 더미 텍스트	더미 텍스트 더미 텍스트
<code>l</code> 로 생성한 열은	<code>c</code> 로 생성한 열은	<code>r</code> 로 생성한 열은
왼쪽 정렬됩니다.	가운데 정렬됩니다.	오른쪽 정렬됩니다.
더미 텍스트 더미 텍스트	더미 텍스트 더미 텍스트	더미 텍스트 더미 텍스트

그림 14.2: `l`, `c`, `r` 인자로 생성한 열

`p`, `m`, `b`는 표의 내용을 지정한 폭 `<width>`에 맞추어 내용을 조판하는 열을 생성합니다. 입력된 내용이 `<width>`보다 길다면 줄바꿈 처리가 일어나고, 글줄의 양 끝이 가지런해지도록 양쪽 정렬 처리됩니다. 각 인자는 같은 행 다른 열의 내용이 이 열의 위쪽, 가운데, 아래쪽에 맞춰 조판되도록 합니다. 그림 14.3을 참고하십시오.

더미 텍스트 더미 텍스트	p로 생성한 열은 같은 행의 다른 열이 위쪽에 정렬되도록 합니다. 더미 텍스트 더미 텍스트 더미 텍스트 더미 텍스트 더미 텍스트 더미 텍스트 더미 텍스트 더미 텍스트
더미 텍스트 더미 텍스트	m으로 생성한 열은 같은 행의 다른 열이 가운데에 정렬되도록 합니다. 더미 텍스트 더미 텍스트 더미 텍스트 더미 텍스트 더미 텍스트 더미 텍스트 더미 텍스트 더미 텍스트
더미 텍스트 더미 텍스트	b로 생성한 열은 같은 행의 다른 열이 아래쪽에 정렬되도록 합니다. 더미 텍스트 더미 텍스트 더미 텍스트 더미 텍스트 더미 텍스트 더미 텍스트 더미 텍스트 더미 텍스트

그림 14.3: p, m, b 인자로 생성한 열

열과 열의 구분

열의 좌우에는 ‘약간의 공백’이 삽입되어 시각적으로 인접한 열과 구분됩니다. `<spec.>` 인자에 세로줄 |을 입력하면 좌우 두 열 사이에 세로 구분선을 그릴 수 있습니다. 이 문자를 여러 번 입력해 다중 세로선을 그릴 수도 있습니다.

`@{<text>}`를 인자로 사용하면 열과 열 사이의 ‘약간의 공백’을 없애고 `<text>`에 입력된 내용을 넣을 수 있습니다. 예를 들어, `@{}`를 입력하면 열과 열 사이의 공백을 삭제할 수 있고, `@{\>}`를 입력하면 열과 열 사이에 5 mu의 작은 공백만 삽입되도록 할 수 있습니다.

```

1 \begin{displaymath}
2 \left\{ \begin{array}{l}
3 \quad \{ @{} r@{\;} c@{\;} r@{\>} c@{\>} l@{} \}
4 \quad 2x_1 \& + \& 3x_2 \& = \& 4 \& \\
5 \quad -3x_1 \& - \& x_2 \& = \& -1
6 \end{array} \right.
7 \end{displaymath}

```

$$\begin{cases} 2x_1 + 3x_2 = 4 \\ -3x_1 - x_2 = -1 \end{cases}$$

이 예시는 열 사이의 간격을 조정하여 연립방정식을 조판한 예시입니다. 여기에서 간격을 조정하지 않고 수식을 조판하면 아래의 결과를 얻습니다.

```

1 \begin{displaymath}
2 \left\{ \begin{array}{l} \rcrc{l}
3 \quad \dots
4 \end{array} \right.
5 \end{displaymath}

```

$$\begin{cases} 2x_1 + 3x_2 = 4 \\ -3x_1 - x_2 = -1 \end{cases}$$

이 기능을 응용하면 열의 내용을 소숫점을 기준으로 정렬할 수도 있습니다. 소숫점을 기준으로 해당 열을 두 개의 열로 나누어 조판하고, 이 두 열 사이에 소숫점이 입력되도록 하는 것입니다. 표의 첫 행은 뒤에서 설명하는 `\multicolumn` 제어 문자열을 사용해 두 열을 합쳐 작성하면 됩니다.

```

1 \begin{center}
2   \begin{tabular}{c|r@{.}l}
3     정보 & \multicolumn{2}{c}{측정값} \\ \hline
4     신장 & 180&3\,cm \\
5     체중 & 76&23\,kg \\
6   \end{tabular}
7 \end{center}

```

정보	측정값
신장	180.3 cm
체중	76.23 kg

array 패키지를 불러오면 @ 대신 !를 사용할 수도 있습니다. 이 인자를 사용하면 @ 인자와 같은 동작을 하되, 열과 열 사이의 기본 간격이 삭제되지 않습니다. 예를 들어, 열과 열을 세로선 대신 쌍점을 사용해 구분하려 할 때 이 인자를 사용할 수 있습니다.

```

1 \begin{center}
2   \begin{tabular}{c|c!{:}c}
3     \hline
4     분야 & A국 & B국 \\ \hline
5     득점 & 3 & 1 \\
6     유효슈팅 & 10 & 8 \\
7     파울 & 2 & 4 \\ \hline
8   \end{tabular}
9 \end{center}

```

분야	A국	B국
득점	3	1
유효슈팅	10	8
파울	2	4

열 서식 지정하기

한 열의 모든 칸의 첫 부분이나 끝 부분에 같은 코드를 넣을 일이 있다면 array 패키지의 $\langle format \rangle$ 과 $\langle format \rangle$ 인자를 사용할 수 있습니다. $\langle format \rangle$ 를 입력하면 이 인자 오른쪽에 있는 열의 첫머리에 $\langle format \rangle$ 의 코드를 삽입한 채로 내용이 조판되고, 반대로 $\langle format \rangle$ 를 입력하면 이 인자 왼쪽에 있는 열의 끝부분에 $\langle format \rangle$ 의 코드를 삽입한 채로 내용이 조판됩니다. 예를 들어,

```
>\sffamily c | l r<{원}>\{<{r}<{\%}\}
```

를 $\langle spec. \rangle$ 인자로 입력한 경우 제1열은 산세리프체로 내용이 조판되며, 제3열은 각 칸의 끝에 ‘원’이 추가된 채로 조판되고, 제4열은 수식 입력 모드에서 조판되며 끝에 ‘%’가 붙습니다.

```

1 \begin{center}
2   \begin{tabular}
3     {>\sffamily c|lr<{원}>\{<{r}<{\%}\}}
4     \hline
5     A1 & 사과 & 3000 & 3.1 \\
6     A2 & 배 & 1500 & -2.8 \\
7     B1 & 볼펜 & 1800 & 1.3 \\ \hline
8   \end{tabular}
9 \end{center}

```

A1	사과	3000원	3.1%
A2	배	1500원	-2.8%
B1	볼펜	1800원	1.3%

서식이 지정된 열의 특정 칸에는 이 서식이 적용되지 않도록 하려면 다음 절에서 설명하는 `\multicolumn` 제어 문자열을 사용할 수 있습니다. 예를 들어, 위 예시에서 제1행을 다음과 같이 추가할 수 있습니다.

```
1 ...
2 \hline
3 코드 & \sffamily 이름
4 & \multicolumn{1}{c}{\sffamily 가격}
5 & \multicolumn{1}{c}{\sffamily 증가율} \\
6 \hline
7 ...
```

코드	이름	가격	증가율
A1	사과	3000원	3.1%
A2	배	1500원	-2.8%
B1	볼펜	1800원	1.3%

여러 열 반복 생성하기

같은 속성의 열을 반복해서 생성해야 한다면 `*{<num.>}{<spec.>}`을 인자로 지정할 수 있습니다. 이 인자는 `<spec.>`을 `<num.>`번 반복 입력한 것과 같은 효과를 냅니다. 예를 들어, 열과 열 사이에 구분선이 있는 7열짜리 표를 그린다면 인자로

```
|c|c|c|c|c|c|c|
```

를 지정할 수도 있지만, 간단히 `*{7}{c|}`를 인자로 지정해도 같은 결과를 얻습니다.

```
1 KAIST 수리과학과에서 개설하는 주요 과목은 다음과 같다.
2 \begin{center}
3 \begin{tabular}{|*{7}{c|}}
4 \hline
5 과목명 & 과목코드 & 과목 구분 & 개설 학기 & 연습반/퀴즈 유무 & 핵심과목 여부 & 학점 \\ \hline
6 미적분학 I & MAS101 & 기초필수 & 봄, 가을 & 있음 & 아니오 & 3 \\
7 ...
8 르베그적분론 & MAS441 & 전공선택 & 가을 & 없음 & 아니오 & 3 \\ \hline
9 \end{tabular}
10 \end{center}
```

KAIST 수리과학과에서 개설하는 주요 과목은 다음과 같다.

과목명	과목코드	과목 구분	개설 학기	연습반/퀴즈 유무	핵심과목 여부	학점
미적분학 I	MAS101	기초필수	봄, 가을	있음	아니오	3
선형대수학개론	MAS109	기초선택	봄, 가을	있음	아니오	3
선형대수학	MAS212	전공선택	봄, 가을	없음	예	3
해석학 I	MAS241	전공선택	봄	있음	예	4
현대대수학 I	MAS311	전공선택	봄	있음	예	4
미분기하학개론	MAS321	전공선택	가을	있음	예	4
위상수학	MAS331	전공선택	봄	있음	예	4
르베그적분론	MAS441	전공선택	가을	없음	아니오	3

* 인자의 *<spec.>* 부분에는 표 14.1의 인자 중 * 자기 자신을 제외한 모든 것을 넣을 수 있으므로, 다음과 같이 사용할 수도 있습니다.

```

1 \begin{displaymath}
2   \begin{array}{*{3}{r@{\;}c@{\;}}r@{\;}c@{\;}l}
3     x_1 & - & 2x_2 & + & x_3 & 2x_4 & = & 1 \\
4     x_1 & + & x_2 & - & x_3 & + & x_4 & = & 2 \\
5     x_1 & + & 7x_2 & - & 5x_3 & - & x_4 & = & 3 \\
6   \end{array}
7 \end{displaymath}

```

$$\begin{array}{rcl}
 x_1 - 2x_2 + x_3 - 2x_4 & = & 1 \\
 x_1 + x_2 - x_3 + x_4 & = & 2 \\
 x_1 + 7x_2 - 5x_3 - x_4 & = & 3
 \end{array}$$

앞뒤 내용과의 세로 정렬

tabular 및 array를 사용해 작성한 표는 기본적으로 문자와 똑같이 취급됩니다. 따라서 별도의 문단에 표를 작성한 것이 아니라면 표의 앞뒤에 텍스트나 수식이 올 수도 있습니다. 이때 표의 위쪽, 가운데, 아래쪽 중 어디에 맞춰 글줄이 오도록 할지를 *<position>* 인자가 결정합니다. 기본값인 c는 표의 가운데에 글줄이 오도록 하며, t와 b는 각각 표의 위쪽과 아래쪽에 맞춰 글줄이 오도록 합니다. 그림 14.4를 참고하십시오.

더미 텍스트	더미 텍스트	선택 인자로 t를 지정하면 표의 위쪽에 맞춰 앞뒤의 내용이 배치됩니다.	더미 텍스트	더미 텍스트
더미 텍스트	더미 텍스트	선택 인자로 c를 지정하면 표의 가운데에 맞춰 앞뒤의 내용이 배치됩니다.	더미 텍스트	더미 텍스트
더미 텍스트	더미 텍스트	선택 인자로 b를 지정하면 표의 아래쪽에 맞춰 앞뒤의 내용이 배치됩니다.	더미 텍스트	더미 텍스트

그림 14.4: 표 앞뒤 내용과의 세로 정렬

예를 들어, 여러 표를 한 줄에 나란히 배치할 때 이 인자를 유용하게 사용할 수 있습니다. 높이가 다른 표를 배치할 때 첫 행, 가운데, 마지막 행 중 어디에 맞춰서 배치할지 결정할 수 있습니다.

```

1 \begin{center}
2   \begin{tabular}[t]{c}{cc}
3     ...
4   \end{tabular}\quad
5   \begin{tabular}[t]{c}{cc}
6     ...
7   \end{tabular}
8 \end{center}

```

퀴즈	점수	평균	퀴즈	점수	평균
1	17.5	16.3	1	13.5	16.6
2	13.5	15.1	2	16.0	15.1
3	20.0	12.7	3	19.5	12.8
			4	11.0	14.4

14.3

칸 합치고 구분선 그리기

이번 절에서는 tabular 및 array 환경 안에서 표의 칸을 합치고 가로 방향 구분선을 그리는 방법을 설명하겠습니다.

여러 열 합치기

`\multicolumn` 제어 문자열을 사용하면 한 행 안의 여러 칸을 합칠 수 있습니다.

`\multicolumn{<num.>}{<spec.>}{<text>}`

`<num.>`: 합칠 열의 개수

`<spec.>`: 합친 칸의 열 속성

`<text>`: 합친 칸에 작성할 내용

`<spec.>`에는 열 생성 인자(`l`, `c`, `r`, `p`, `m`, `b` 등)를 하나만 입력하고, 필요한 경우 `|`나 `@`, `>`, `<`를 적절히 사용하면 됩니다. 합친 칸 안에 작성할 내용은 `<text>` 인자로 지정하십시오. `\multicolumn` 제어 문자열을 사용한 칸에 이 명령어의 인자 외에 다른 내용이 있다면 컴파일 과정에서 오류가 발생합니다.

```

1 \begin{center}
2   \begin{tabular}{|c|c|c|c|}
3     \hline
4     A1 & B1 & C1 & D1 \\ \hline
5     A2 & \multicolumn{3}{c}{B2:D2} \\ \hline
6     \multicolumn{2}{|c|}{A3:B3} & C3 & D3 \\ \hline
7     A4 & B4 & C4 & D4 \\ \hline
8   \end{tabular}
9 \end{center}

```

A1	B1	C1	D1
A2	B2:D2		
A3:B3		C3	D3
A4	B4	C4	D4

`\multicolumn`을 사용하면 합칠 대상이 되는 열들에 지정된 서식(`>`, `<`로 지정한 것)이 무시되고, 각 열의 오른쪽에 지정된 구분 기호(`|`, `@`, `!` 등)가 무시됩니다. 합칠 칸에 제1열이 포함되어 있다면 표 가장 왼쪽에 지정된 구분 기호도 무시됩니다. 따라서 `<spec.>` 인자를 지정할 때 구분선이 사라지지 않도록, 간격이 변하지 않도록 주의해 주어야 합니다.

예를 들어, 위 예시에서 B2, C2, D2 칸을 합칠 때에는 B2, C2, D2 칸의 오른쪽에 있는 구분선이 사라지고, B2 칸의 왼쪽에 있는 구분선은 사라지지 않습니다. 이때 칸 오른쪽의 구분선이 사라지지 않도록 하려면 `<spec.>` 인자로 `c|`를 입력하여 구분선이 그려지도록 하면 됩니다. 반면, A3, B3 칸을 합칠 때에는 A3 칸의 왼쪽과 오른쪽, B3 칸의 오른쪽에 있는 구분선이 모두 사라집니다. 따라서 `<spec.>` 인자로 `|c|`를 입력하여 양쪽의 구분선을 다시 그려 주었습니다.

구분선을 특별히 지정하지 않은 경우와의 차이를 비교하기 위해, 두 `<spec.>` 인자에 입력한 `|`를 모두 삭제한 후 조판한 결과를 다음 페이지에 제시합니다.

```

1 \begin{center}
2   \begin{tabular}{|c|c|c|c|}
3     \hline
4     A1 & B1 & C1 & D1 \\ \hline
5     A2 & \multicolumn{3}{c}{B2:D2} \\ \hline
6     \multicolumn{2}{c}{A3:B3} & C3 & D3 \\ \hline
7     A4 & B4 & C4 & D4 \\ \hline
8   \end{tabular}
9 \end{center}

```

A1	B1	C1	D1
A2	B2:D2		
A3:B3		C3	D3
A4	B4	C4	D4

예제 14.4 다음 표를 작성하여라. 제1행과 제2행 사이의 구분선은 `\hline` 대신 `\cline{2-5}`를 입력하여 1열의 첫 두 칸을 가로지르지 않도록 할 수 있다.

	전설 모음		후설 모음	
	평순 모음	원순 모음	평순 모음	원순 모음
고모음	ㅣ	ㄱ	ㅡ	ㅈ
중모음	ㅓ	ㅗ	ㅑ	ㅜ
저모음	ㅕ		ㅓ	

여러 행 합치기

여러 행에 걸친 칸을 합치려면 먼저 `multirow` 패키지를 불러와야 합니다. 이후 `\multirow` 제어 문자열을 사용하면 됩니다.

`\multirow[⟨position⟩]{⟨num.⟩}[⟨width⟩]{⟨text⟩}`

⟨position⟩: 합친 칸의 내용을 정렬할 위치

⟨num.⟩: 합칠 행의 개수

⟨width⟩: 합친 칸에서 내용을 조판할 글줄의 폭

⟨text⟩: 합친 칸에 작성할 내용

⟨num.⟩ 인자에는 양수를 입력할 수도, 음수를 입력할 수도 있습니다. 양수를 입력하면 제어 문자열이 입력된 칸을 기준으로 아래 방향으로 칸을 합치며, 음수를 입력하면 제어 문자열이 입력된 칸을 기준으로 위 방향으로 칸을 합칩니다.

⟨width⟩ 인자에는 길이 값을 입력할 수도 있고, 이외에 별표 *와 등호 =를 입력할 수도 있습니다. *를 입력하면 ⟨text⟩에 입력한 내용이 한 줄로 조판되도록 합친 칸의 폭을 \textwidth 이 자동으로 계산합니다. 합치려는 칸들이 p, m 또는 b 인자로 생성된 열이라면, 이 열의 폭을 그대로 사용하고자 할 때 =를 사용할 수 있습니다.

```

1 \begin{center}
2   \begin{tabular}{|c|c|c|c|}
3     \hline
4     A1 & \multirow{3}{*}{B1:B3} & C1 & D1 \\
5     A2 & & C2 & D2 \\
6     A3 & & C3:C4 & D3 \\
7     A4 & B4 & & D4 \\
8   \end{tabular}
9 \end{center}

```

A1	B1:B3	C1	D1
A2		C2	D2
A3		C3:C4	D3
A4	B4		D4

여러 행에 걸쳐 칸을 합친 경우에는 위 예시의 B2, B3, C4 칸처럼 칸을 빈 상태로 두어야 합니다. 또, 가로 구분선을 그릴 때 합친 칸을 가로지르는 구분선이 자동으로 지워지지 않으므로 `\hline` 대신 뒤쪽에서 설명하는 `\cline`을 사용해 그려 주어야 합니다. `\cline`을 사용하지 않고 `\hline`을 사용하면 아래의 결과를 얻습니다.

```

1 \begin{center}
2   \begin{tabular}{|c|c|c|c|}
3     \hline
4     A1 & \multirow{3}{*}{B1:B3} & C1 & D1 \\
5     A2 & & C2 & D2 \\
6     A3 & & C3:C4 & D3 \\
7     A4 & B4 & & D4 \\
8   \end{tabular}
9 \end{center}

```

A1	B1:B3	C1	D1
A2		C2	D2
A3		C3:C4	D3
A4	B4		D4

예제 14.5 아래 표를 그려라. 제1열을 가로지르지 않는 가로 구분선은 다음 페이지의 `\cline`에 대한 설명을 참고하여 그려라.

		양순음	치조음	경구개음	연구개음	후음
파열음	평음	ㅂ	ㄷ		ㄱ	
	경음	ㅃ	ㄸ		ㄲ	
	격음	ㅍ	ㅌ		ㅋ	
파찰음	평음			ㅈ		
	경음			ㅊ		
	격음			ㅉ		
마찰음	평음		ㅅ			ㅎ
	경음		ㅆ			
비음		ㅁ	ㄴ		ㅇ	
유음			ㄹ			

선택 인자인 $\langle position \rangle$ 에는 t, c, b 중 하나를 입력합니다. 이 인자는 tabular 및 array 환경의 $\langle position \rangle$ 인자와 같은 역할을 합니다. 기본값은 c로, 합친 칸의 내용을 칸의 세로 가운데에 조판하도록 하며, t 또는 b를 인자로 지정하면 내용이 합친 칸의 위쪽이나 아래쪽에 맞춰 조판됩니다.

```

1 \begin{center}
2   \begin{tabular}{|c|c|c|c|c|}
3     \hline
4     \multirow[t]{3}{*}{위쪽} & B1 & & D1 & \\
7     \multirow[c]{3}{*}{가운데} & B2 & & D2 & \\
6     \multirow[b]{3}{*}{아래쪽} & B3 & & D3 & \\
5     \cline{2-2}\cline{4-4}
7   & B2 & & D2 & \\
6   & B3 & & D3 & \\
7   \end{tabular}
8 \end{center}

```

위쪽	B1	가운데	D1	아래쪽
	B2		D2	
	B3		D3	

가로 방향 구분선 그리기

첫 번째 절에서 설명했듯, 가로 방향 구분선은 `\hline` 제어 문자열을 사용해 그립니다. 이 제어 문자열을 여러 번 연달아 입력하면 다중 가로선을 그릴 수도 있습니다.

한 행의 일부 열에만 가로선을 그리려면 `\hline` 대신 `\cline` 제어 문자열을 사용할 수 있습니다.

`\cline{<range>}`

$\langle range \rangle$: 가로줄을 그릴 열의 범위

인자 $\langle range \rangle$ 에는 가로줄을 그릴 열의 범위를 $\langle i \rangle$ - $\langle j \rangle$ 의 형식으로 입력하면 됩니다. 첫 열의 번호는 1입니다. 중간중간 끊어져 있는 가로선을 그리려면 이 제어 문자열을 여러 번 사용하면 됩니다. 예를 들어, 위 예시의 `\cline{2-2}\cline{4-4}`는 제2열과 제4열에만 가로선을 그리고, 나머지 열에는 가로선을 그리지 않도록 합니다.

14.4

표 전체 폭 조정하기

tabular 환경을 사용해 작성한 표는 전체 폭이 표의 내용에 따라 변합니다. 표 전체의 폭을 고정하고, 이에 맞추어 열과 열 사이의 공간을 늘이려면 tabular* 환경을 사용할 수 있습니다.

`\begin{tabular*}{<width>}[<position>]{<spec.>}`

$\langle width \rangle$: 표 전체의 폭

$\langle position \rangle$: 앞뒤 텍스트와의 세로 정렬

$\langle spec. \rangle$: 열의 개수와 각 열의 속성

`\end{tabular*}`

예를 들어, $\langle width \rangle$ 인자로 $\backslash linewidth$ 를 지정하면 글줄의 길이와 일치하는 폭의 표가 만들어집니다. 이 인자를 지정한 후, $\langle spec \rangle$ 인자를 일반적인 tabular 환경처럼 지정한 뒤 첫 열의 왼쪽에 $!\{\backslash vline\extracolsep{\backslash fill}\}$ 을 입력하면 됩니다. 이 명령어는 각 열 사이의 빈 공간을 균등하게 넓히는 역할을 해 주어 표 전체의 폭이 $\langle width \rangle$ 에 지정된 길이에 맞게 해 줍니다. 첫 열의 왼쪽에 세로선을 그리고 싶다면, 앞의 명령어 대신 아래 예시처럼 첫 열의 왼쪽에 $!\{\backslash vline\extracolsep{\backslash fill}\}$ 을 입력하면 됩니다.

```
1 \begin{tabular*}{\linewidth}
  {!\{\backslash vline\extracolsep{\backslash fill}\}lcr|}
2 \hline
3 왼쪽 & 가운데 & 오른쪽 \\\ \hline
4 정렬된 & 정렬된 & 정렬된 \\\ \hline
5 1열 & 2열 & 3열 \\\ \hline
6 \end{tabular*}
```

왼쪽	가운데	오른쪽
정렬된	정렬된	정렬된
1열	2열	3열

하지만, tabular* 환경은 두 가지 단점이 있습니다. 첫째는 세로 구분선이 열과 열 사이의 정가운데에 그려지지 않는다는 점입니다. 아래 예시를 참고하시기 바랍니다.

```
1 \begin{tabular*}{\linewidth}
  {!\{\backslash vline\extracolsep{\backslash fill}\}l|c|r|}
2 \hline
3 왼쪽 & 가운데 & 오른쪽 \\\ \hline
4 정렬된 & 정렬된 & 정렬된 \\\ \hline
5 1열 & 2열 & 3열 \\\ \hline
6 \end{tabular*}
```

왼쪽	가운데	오른쪽
정렬된	정렬된	정렬된
1열	2열	3열

둘째는 각 열에서 내용이 작성되는 공간의 폭이 표 전체의 폭에 맞추어 자동으로 결정되도록 할 수 없다는 점입니다. 각 열의 폭은 표 전체의 폭과 독립적으로 결정되고, 남은 공간은 공백으로 채워집니다. 이때에는 tabular* 환경 대신 tabularx 패키지에서 제공하는 tabularx 환경을 사용할 수 있습니다. 이 환경은 tabular* 환경과 같은 인자를 입력받습니다.

```
\begin{tabularx}{\langle width \rangle}{\langle position \rangle}{\langle spec \rangle}
```

$\langle width \rangle$: 표 전체의 폭

$\langle position \rangle$: 앞뒤 텍스트와의 세로 정렬

$\langle spec \rangle$: 열의 수, 정렬 방법 및 구분선

```
\end{tabularx}
```

여기에서 $\langle spec \rangle$ 인자를 지정할 때에는 l, c, r나 p 대신 X로도 열을 생성할 수 있습니다. X 인자로 생성된 열의 폭은 $\langle width \rangle$ 인자에 따라 자동으로 계산됩니다. 다음 예시를 참고하시기 바랍니다.

```

1 \begin{tabularx}{\linewidth}{|c|X|X|}
2 \hline
3 1행 & \Lcmd{X} 인자로 생성된 각 열은 & 1행 \\
4 \hline
5 2행 & 나머지 열을 조판하고 남은 공간을 & 2행 \\
6 \hline
7 3행 & 완전히 채우도록 균등하게 분배됩니다. & 3행 \\
8 \hline
9 \end{tabularx}

```

1행	X 인자로 생성된 각 열은	1행
2행	나머지 열을 조판 하고 남은 공간을	2행
3행	완전히 채우도록 균등하게 분배됩 니다.	3행

c 인자로 생성된 열은 각 칸의 내용에 맞추어 폭이 결정되었지만, X 인자로 생성된 두 열은 표의 전체 폭에서 다른 열을 조판하고 남은 공간을 완전히 채우도록 폭이 결정되었음을 확인할 수 있습니다. 각 열의 폭은 균등하게 분배됩니다.

제 15 장

그림 삽입하기

LaTeX 문서에 그림을 삽입할 때에는 먼저 전처리부에서 graphicx 패키지를 불러와야 합니다. 이후, 그림을 삽입하려는 곳에서 `\includegraphics` 제어 문자열을 사용합니다.

```
\includegraphics[<options>]{<filename>}
```

<options>: 그림을 삽입할 때 지정할 옵션

<filename>: 삽입할 그림 파일의 이름

문서에 넣으려는 그림 파일을 소스 파일과 같은 폴더에 놓고, `\includegraphics`의 인자로 파일의 이름을 입력하면 됩니다. 하위 폴더에 있는 그림을 넣으려면 ‘<폴더 이름>/<파일 이름>’을 인자로 입력하면 됩니다. 이때 삽입하려는 그림은 EPS, PDF, PNG, 또는 JPEG 형식이어야 합니다.

- 1 `\LaTeX` 문서에는 아래와 같이 그림을 삽입할 수 있습니다. `\par`
- 2 `\medskip`
- 3 `\includegraphics{Pictures/sample.pdf}`

LaTeX 문서에는 아래와 같이 그림을 삽입할 수 있습니다.



문서에 삽입한 그림 파일은 소스 파일을 컴파일 할 때마다 필요합니다. 문서를 수정할 경우를 대비하여, 그림 파일이 이동·삭제되지 않도록 주의하십시오.

그림 삽입 옵션

`\includegraphics`의 선택 인자를 통해 그림을 삽입할 때 크기를 바꾸거나 그림을 회전하는 등 여러 옵션을 지정할 수 있습니다. graphicx 패키지의 안내서에는 사용할 수 있는 모든 옵션이 설명되어 있지만, 여기에서는 자주 사용되는 다섯 개만을 설명하겠습니다.

옵션을 지정할 때에는 ‘<key>=<value>’의 형식을 사용하며, 이때 <key>에는 표 15.1의 항목을 입력

표 15.1: 그림을 삽입할 때의 주요 옵션

<i><key></i>	설명
width	그림을 삽입할 때의 폭을 지정합니다.
height	그림을 삽입할 때의 높이를 지정합니다.
scale	그림을 확대·축소할 배율을 지정합니다.
angle	그림을 반시계 방향으로 회전할 각도를 지정합니다.
page	여러 페이지가 있는 PDF 문서를 삽입할 경우, 표시할 페이지를 지정합니다.

하고, *<value>*에는 적절한 길이나 수치를 입력합니다. 여러 옵션을 지정할 때에는 각 옵션을 쉼표로 구분해 주면 됩니다. 예를 들어, 다음 코드는 sample.pdf 문서의 제2페이지를 폭 5 cm로 삽입합니다.

```
\includegraphics[page=2, width=5cm]{sample.pdf}
```

길이를 지정하는 자세한 방법은 22.2절을 참조하시기 바랍니다.

그림을 삽입할 때 폭과 높이 중 하나만을 지정하면 그림의 종횡비를 유지한 채로 크기가 조정되며, 두 값을 모두 지정하면 원래의 종횡비가 무시됩니다. 그림의 크기를 지정하는 경우 참고하십시오.

그림의 배치

그림 파일은 문자와 똑같이 취급됩니다. 그림을 삽입하는 명령어 앞뒤에 별도의 문단 나눔이 없다면 글줄에 섞여 삽입됩니다. 그림을 글줄과 구분하고, 가운데에 맞추어 배치하려면 앞서 알아본 center 환경 안에 그림을 삽입하면 됩니다. 한편, 여러 그림을 한 줄에 배치하려 한다면 각 그림을 한 문단 안에 삽입하면 됩니다. 그림의 높이를 일치시키면 더 보기 좋은 문서를 만들 수 있습니다. 그림 사이의 간격은 \enspace, \quad, \qquad나 \hspace를 사용해 조정할 수 있습니다.

```
1 \verb|center| 환경을 사용하면 그림이 가운데에 배치
  됩니다.
2 \begin{center}
3   \includegraph-
4     ics[height=1cm]{sample1.eps}%
5   \quad
6   \includegraph-
7     ics[height=1cm]{sample2.eps}%
8   \quad
9   \includegraph-
10    ics[height=1cm]{sample3.eps}
11 \end{center}
```

center 환경을 사용하면 그림이 가운데에 배치됩니다.



한편, figure 환경을 사용하면 글줄과 별개로 페이지의 위쪽이나 아래쪽에 그림을 배치하거나 캡션을 달 수 있습니다. 이에 대해서는 제16장을 참조하십시오.

제 16 장

표와 그림 배치하고 캡션 달기

표와 그림은 기본적으로 문자와 같이 배치됩니다. 따라서 특별한 처리를 하지 않으면 글줄에 섞여서 배치됩니다. center 등의 환경을 사용하면 표나 그림이 글줄의 가운데에 오도록 할 수 있습니다.

L^AT_EX에서는 표와 그림을 유동 개체(float)로 처리하여, 글줄과 별개로 페이지의 위쪽이나 아래쪽에 배치하는 기능을 제공합니다. 또, 이 기능을 사용하면 표와 그림에 캡션을 작성할 수도 있습니다. 이번 장에서는 유동 개체와 캡션에 대해 알아봅니다.

16.1

유동 개체로써 표와 그림 배치

표와 그림은 세로로 긴 경우가 많아 배치할 때 주의해야 합니다. 특별한 처리를 하지 않으면 현재 페이지에 개체를 배치할 공간이 부족할 때 이 개체를 다음 페이지로 넘기게 되는데, 이때 앞 페이지에 빈 공간이 많이 생기기 때문입니다. 이들을 유동 개체로 취급하면 이 빈 공간에 표나 그림 다음에 올 내용을 앞 페이지에 채워 조판할 수 있습니다. 간단히 말해, 아래아한글의 ‘자리 차지’, Word의 ‘위/아래’ 방식으로 개체를 배치한 것과 비슷한 결과를 얻습니다.

그림은 figure 환경 안에 작성하면 유동 개체로 처리됩니다.

```
\begin{figure}[\langle position \rangle]
```

\langle position \rangle: 유동 개체의 배치를 허용할 곳

```
\end{figure}
```

마찬가지로, 표는 table 환경 안에 같은 방법으로 작성하면 됩니다. 이들 개체는 (1) 코드 순서 상 바로 그 위치나 (2) 페이지의 위쪽, (3) 페이지의 아래쪽에 적절히 배치되며, 필요한 경우에는 (4) 개체들만을 조판하는 별도의 페이지를 만들기도 합니다(그림 16.1을 참고하십시오).

위에서 설명한 네 가지 위치 중 배치를 허용할 곳을 직접 지정해 줄 수도 있습니다. 환경의 *\langle position \rangle* 선택 인자에 h, t, b, p 중 배치를 허용할 곳을 입력하면 됩니다. 각 문자의 의미는 표 16.1를 참조하

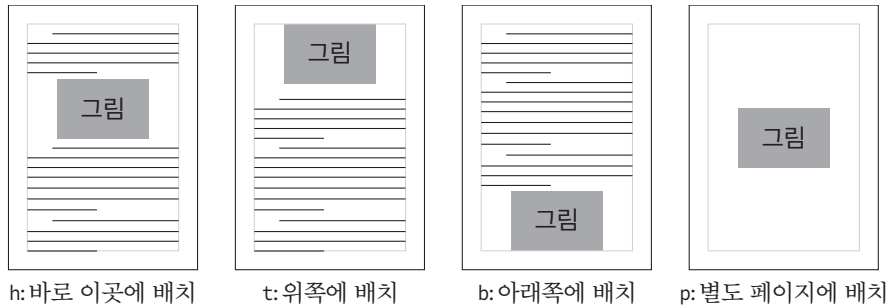


그림 16.1: 유동 개체의 위치 지정 인자에 따른 배치

십시오. 먼저 입력한 인자의 방식으로 배치를 시도하며, 공간이 부족한 경우 다음 인자의 방식으로 배치를 시도합니다. 예를 들어, 기본값인 `tb`는 ‘페이지의 위쪽(t)에 배치를 시도하며, 공간이 부족한 경우 페이지의 아래쪽(b)에 배치를 시도하고, 이때에도 공간이 부족하다면 개체를 별도의 페이지(p)에 배치함’을 뜻합니다.

\LaTeX 은 개체를 배치하는 데 여러 내부 인자¹를 사용하므로, 원하는 대로 개체가 배치되지 않을 수도 있습니다. 이때에는 느낌표 `!`를 인자의 맨 앞에 붙여 보십시오. 내부 인자를 일부 무시하므로, 의도하는 위치에 개체가 조판될 가능성이 높아집니다. (다만 항상 가능한 것은 아닙니다.)

표 16.1: 유동 개체의 위치 지정 인자

인자	설명
<code>h</code>	개체를 현재 위치에 배치합니다.
<code>t</code>	개체를 현재 페이지의 위쪽에 맞추어 배치합니다.
<code>b</code>	개체를 현재 페이지의 아래쪽에 맞추어 배치합니다.
<code>p</code>	개체를 별도의 페이지에 배치합니다. 이 페이지에는 유동 개체만 배치됩니다.
<code>!</code>	개체를 배치하기 위해 \LaTeX 이 사용하는 내부 인자를 일부 무시합니다.

그림과 표를 유동 개체로 처리하고 위치를 지정하는 예시로, 그림 16.1과 표 16.1을 배치하는 데 사용한 코드의 일부를 아래에 제시합니다.

```

1 \begin{figure}[!tb]
2   \centering
3   \includegraphics{...}
4   \caption{유동 개체의 위치 지정 인자에 따른 배치}
5 \end{figure}
6
7 위에서 설명한 네 가지 ... 가능성이 높아집니다. {\small(다만 ... 아닙니다.)}
8
9 \begin{table}[!htp]
```

¹사용하는 내부 인자로는 페이지의 일부분에 개체를 배치할 때 개체가 차지할 수 있는 공간의 최대 크기, 페이지 전체에 개체만을 배치할 때 개체가 차지하는 공간의 최소 크기, 한 페이지에 배치할 개체의 최대 수 등이 있습니다. 자세한 내용은 latexref.xyz/Floats.html을 참조하십시오.

```

10 \caption{유동 개체의 위치 지정 인자}
11 \begin{tabularx}{\linewidth}{c|X}
12 ...
13 \end{tabularx}
14 \end{table}

```

유동 개체와 페이지 나눔

유동 개체를 배치하면서 `\newpage`를 사용해 페이지를 나눈 경우, 유동 개체가 `\newpage`를 입력해 만들어진 다음 페이지에 배치될 수도 있습니다. 만약 유동 개체의 배치를 모두 끝내고 새로운 페이지를 만들려면 `\newpage` 대신 `\clearpage`를 사용하면 됩니다. 또, 양면 편집 문서에서는 `\cleardoublepage`를 사용할 수도 있습니다. 새로 만들어진 페이지가 홀수 페이지가 되도록, 필요한 경우에는 새로운 페이지를 하나 더 추가합니다.

16.2

캡션 작성

table 또는 figure 환경 안에 표와 그림을 작성하였다면, 캡션을 다는 것은 간단합니다. 환경 안에서 `\caption` 제어 문자열을 입력하고 캡션의 내용을 인자로 지정하면 됩니다.

```
\caption[<short caption>]{<caption>}
```

<short caption>: 표·그림 목차에서 사용할 캡션

<caption>: 캡션의 내용

이 제어 문자열은 `\chapter`나 `\section`과 마찬가지로 선택 인자를 입력받습니다. 표 목차나 그림 목차에는 다른 내용을 사용하려면 그 내용을 선택 인자로 지정하면 됩니다. 컴파일 과정을 거치면 표 또는 그림 번호와 캡션이 함께 조판됩니다. 한편, 캡션에 번호를 붙이지 않고 싶다면 대신 별표 붙은 `\caption*`을 사용하면 됩니다.

캡션 번호 앞에는 ‘Figure’, ‘Table’ 또는 ‘그림’, ‘표’가 출력되며, 각각 `\figurename`, `\tablename` 제어 문자열에 저장되어 있습니다. 원하는 경우 이들을 재정의하여 표시되는 문자열을 바꿀 수 있습니다. 또, 캡션이나 번호의 서식을 바꾸려면 caption 패키지를 사용할 수 있습니다. 또, figure 및 table 환경 안에서 그림이나 표에 하위 번호를 붙이려는 경우에는 subcaption 패키지를 사용할 수 있습니다. 자세한 사용법은 각 패키지의 안내서를 참조하십시오.

제 V 편

상호 참조 및 특수 기능

제 V 편에서는 참조·인용과 관련된 기능을 배웁니다. 많은 책에서 찾을 수 있는 목차, 참고 문헌 목록, 색인을 만드는 방법도 알아봅니다.

제17장에서는 다른 표나 그림, 장과 절을 참조하는 방법을 설명합니다. 또, `hyperref` 패키지를 사용해 PDF 파일에 하이퍼링크를 넣는 방법도 설명합니다.

제18장은 목차를 생성하고, 항목을 추가하는 방법을 설명합니다.

제19장은 문헌을 인용하고, 참고 문헌 목록을 생성하는 방법을 설명합니다. 이 과정에서 `BiBTeX` 을 사용하게 됩니다.

제20장에서는 색인을 만드는 방법을 다룹니다. 소스 파일을 작성한 뒤 `MakeIndex`나 `xindy`를 실행해 색인을 만드는 방법을 알아봅니다.

제 17 장

상호 참조 및 링크

이번 장에서는 번호 붙은 대상들을 상호 참조(cross referencing)하고, PDF 파일에 책갈피와 하이퍼링크를 삽입하는 방법을 알아봅니다. \LaTeX 문서에서의 번호(장절 번호, 수식 번호, 그림·표 번호 등)들은 소스 파일에서 수동으로 입력하는 것이 아니라, 컴파일 과정에서 \TeX 의 카운터 기능을 사용하여 자동으로 결정됩니다. 따라서, 이들을 참조할 때에는 참조할 대상에 이름을 붙이고, 그 이름을 사용하면 컴파일 과정에서 올바른 번호가 자동으로 조판됩니다.

17.1

참조하기

\LaTeX 에서는 `\chapter`나 `\section` 등으로 생성한 장절 표제, table 환경에 삽입한 표, figure 환경에 삽입한 그림 등 번호를 붙인 대상을 참조할 수 있습니다. equation 등 수식 작성 환경의 수식이나 enumerate 환경의 각 `\item`, 심지어는 amsthm 패키지를 사용해 만든 정리류 환경까지도 번호만 붙어 있다면 참조할 수 있습니다.

참조할 때에는 먼저 참조할 대상에 `\label` 제어 문자열을 사용하여 가상의 이름을 붙입니다. 로마자, 숫자, 문장 부호로 구성된 임의의 문자열을 `\label`의 인자로 지정하면 됩니다. 장절 표제를 참조하려면 해당 명령어 바로 뒤에 `\label`을 입력하고, 표나 그림을 참조하려면 해당 환경 안의 `\caption` 뒤에 `\label`을 입력하면 됩니다. 같은 방법으로, 수식을 참조하려면 해당 수식 환경에, 번호 목록의 항목을 참조하려면 해당 `\item` 뒤에 `\label`을 입력하십시오.

참조할 대상에 이름을 붙였다면, 그 대상을 참조하려는 곳에서 `\ref` 제어 문자열을 사용하면 됩니다. 참조할 대상의 이름을 `\ref`의 인자로 입력하십시오. 컴파일 과정을 거치면 해당 대상의 번호가 출력됩니다. 수식을 참조한다면 `\eqref`를 사용하여 수식 번호에 괄호를 자동으로 씌울 수 있습니다. 또, `\pageref`를 사용하면 해당 `\label`이 위치한 페이지 번호가 출력됩니다.

대상의 이름을 지정할 때, 참조 대상의 종류에 따라 접두어를 지정해 주면 그 식별자를 쉽게 구분할 수 있습니다. 가령, 장절 표제에 대해서는 `ch`나 `sec` 등을, 그림과 표, 수식에는 각각 `fig`, `table`, `eq` 등을 사용할 수 있습니다. 다음 예시에서도 접두어를 사용했음을 확인할 수 있습니다.

```

1 \section{장절 표제 참조 예시}
  \label{sec:something}
2 이번 절을 참조할 때에는 \pageref{sec:something} 쪽
  의 제 \ref{sec:something} 절을 참조하여라.'라고 작성
  하면 된다.
3
4 \subsection{다른 절 참조하기}
  \label{sec:another}

```

2 장절 표제 참조 예시

이번 절을 참조할 때에는 '5쪽의 제2절을 참조하여라.'라고 작성하면 된다.

2.1 다른 절 참조하기

예제 17.1 위의 예시에서 `sec:another` 키워드에 저장된 절 번호와 그 페이지 번호를 참조하려면 어떤 코드를 작성해야 하는가?

컴파일 과정에서의 주의 사항

`\label`을 사용해 이름을 정의한 대상은 `.aux` 확장자를 가지는 보조 파일에 저장됩니다. 첫 번째 컴파일 과정에서 각 대상의 이름과 번호, 페이지 번호가 보조 파일에 저장되고, 두 번째 컴파일 과정에서 이 정보를 바탕으로 올바른 참조 결과가 조판됩니다. 따라서, `\label`을 사용해 새로운 참조 대상을 만들었다면 컴파일 과정을 두 번 거쳐야 올바른 결과를 얻을 수 있습니다. 한편, Overleaf나 다른 `latexmk`를 사용하는 편집기에서는 컴파일 한 번 만으로도 올바르게 번호가 조판되도록 합니다. 이 경우에는 컴파일 횟수를 신경쓰지 않아도 됩니다.

컴파일 과정을 처음 거친 후 생성된 PDF 파일을 보면 `\pageref`나 `\ref`가 사용된 부분에서 올바른 번호가 출력되지 않고 '??'로 출력되어 있을 것입니다. 또, 경고 메시지로

```

Label `sec:something' on page 5 undefined on input line 20.
Label(s) may have changed. Rerun to get cross-references right.
There were undefined references.

```

가 출력될 것입니다. 컴파일 과정을 한 번 더 거치면 `TeX`이 보조 파일에 저장된 내용을 읽어들이어, 절 번호와 페이지 번호를 올바르게 작성한 PDF 파일을 생성해 냅니다.

앞의 세 경고 메시지 중 첫 번째와 세 번째 것은 다른 상황에서도 볼 수 있습니다. `\label`을 사용해 정의하지 않은 식별자를 `\ref`나 `\pageref`에서 사용한 경우에도 이 경고 메시지가 출력되며, `\ref`나 `\pageref`가 사용된 위치에는 마찬가지로 '??'가 출력됩니다.

`\label`을 사용해 이름을 정의할 때에는 당연히 다른 대상과 겹치지 않도록 해야 합니다. 같은 이름을 여러 번 사용하면 경고 메시지로

```

Label `<이름>' multiply defined.
There were multiply-defined references.

```

가 출력됩니다. `\ref`나 `\pageref`가 사용된 위치에는 가장 마지막으로 `\label`이 등장했던 곳을 기준으로 대상 번호와 페이지 번호가 조판됩니다.

한국어 문서를 위한 자동 조사 기능

한국어의 조사는 ‘이/가’, ‘을/를’, ‘로/으로’ 등과 같이 앞에 오는 글자의 종성 유무에 따라 달라집니다. 참조 기능을 사용하면 `\ref`가 어떤 숫자를 출력할지 알 수 없기 때문에, 이 뒤의 조사를 올바르게 입력하기 위해서는 계산된 숫자를 확인한 뒤 조사를 수정해야 할 수도 있습니다.

kotex 패키지는 이러한 불편을 없애기 위해 ‘자동 조사’ 기능을 제공합니다. 앞 글자에 따라 변화하는 조사의 앞에 역슬래시를 붙이면 되며, 이 앞에 오는 문자를 읽는 방법에 따라 올바른 조사가 자동으로 선택됩니다.

`\은` `\는` `\이` `\가` `\을` `\를` `\와` `\과` `\로` `\으로` `\라` `\이라`

를 자동 조사 제어 문자열로 사용할 수 있습니다. 예를 들어,

그림 `\ref{fig:sample}``\을` 참조하여라.

를 입력했을 때, `\ref{fig:sample}`이 3을 출력한다면 ‘그림 3을 참조하여라.’가 조판되고, 4를 출력한다면 ‘그림 4를 참조하여라.’가 조판됩니다.

17.2

하이퍼링크

PDF 문서에 하이퍼링크를 삽입하면 문서를 더 편리하게 읽을 수 있습니다. 목차에서 제목을 클릭하면 해당 위치로 바로 이동하고, 상호 참조된 대상의 번호를 클릭하면 그 대상이 있는 곳으로 바로 이동할 수 있기 때문입니다. 간단히 `hyperref` 패키지를 불러오면 이 기능이 자동으로 적용됩니다(이 패키지는 `kotex`를 제외한 다른 패키지보다 나중에 불러와야 합니다).

이번 절에서는 이렇게 자동으로 삽입되는 하이퍼링크 외에, 문서 작성자가 수동으로 링크를 삽입하는 방법을 설명합니다. 크게 문서 내의 특정 지점으로 이동하는 링크와 문서 외부로 이동하는 링크로 구분할 수 있습니다.

문서 내의 링크

`\hypertarget`과 `\hyperlink` 제어 문자열을 사용합니다. `\label`과 `\ref`와 비슷하게 사용하되, 하이퍼링크를 지정할 대상 문자열을 두 번째 인자로 입력해야 합니다.

<code>\hypertarget{<name>}{<text>}</code>	<code>\hyperlink{<name>}{<text>}</code>
<code><name></code> : 이 위치의 이름 <code><text></code> : 이동할 대상의 내용	<code><name></code> : 이동할 대상의 이름 <code><text></code> : 링크를 삽입할 내용

예를 들어, 문서의 한 곳에는 `\hypertarget{target}{Alpha}`를 입력하고, 다른 한 곳에는 `\hyperlink{target}{Beta}`를 입력한 후 컴파일하여 얻은 문서에서, ‘Beta’를 클릭하면 ‘Alpha’가 입력된 곳으로 이동합니다.

문서 외부로의 링크

`\url` 및 `\href` 제어 문자열을 사용하면 문서 외부로의 링크를 삽입할 수 있습니다.

<code>\url{<url>}</code>	<code>\href{<url>}{<text>}</code>
<code><url></code> : 이동할 대상의 URL	<code><url></code> : 이동할 대상의 URL <code><text></code> : 링크를 삽입할 내용

`\url` 제어 문자열을 사용하면 `<url>`에 입력된 내용이 고정폭 서체로 조판되며, 해당 주소로 이동하는 링크가 삽입됩니다. 예를 들어, `\url{https://www.kaist.ac.kr}`을 입력한 후 컴파일하면 ‘https://www.kaist.ac.kr’이 조판되며, 이 URL을 클릭하면 해당 웹사이트로 이동합니다.

`\href` 제어 문자열을 사용하면 `<text>`에 입력된 내용이 조판되며, `<url>`의 주소로 이동하는 링크가 여기에 삽입됩니다. 예를 들어,

```
\href{https://www.google.co.kr}{이 문자열}
```

을 입력하고 컴파일 과정을 거치면 ‘이 문자열’이 조판되며, 이를 클릭하면 https://www.google.co.kr로 이동합니다.

링크 표시 방법

hyperref 패키지는 문서에 삽입된 링크의 서식을 변경하는 옵션을 제공합니다. 기본적으로 링크가 삽입된 모든 문자열에는 색이 있는 사각형 상자가 그려짐으로써 이 문자열에 링크가 설정되어 있음을 표시합니다. 이 상자는 프린터를 사용해 종이에 문서를 출력할 때에는 표시되지 않습니다.

만약 이 상자가 표시되지 않도록 하려면 hyperref 패키지를 불러올 때 `colorlinks=true`를 옵션으로 지정하면 됩니다. 링크가 삽입된 문자열에 상자가 그려지는 대신, 글자 자체의 색이 변경됩니다. 문자열의 색은 크게 세 가지로 구분됩니다. 문서 내의 링크는 빨간색으로, 인용한 문헌 표시는 녹색으로, 웹 사이트 URL은 자홍색으로 표시되며, 이는

```
colorlinks=true, linkcolor=red, citecolor=green, urlcolor=magenta
```

를 hyperref 패키지를 불러올 때 옵션으로 지정하고, 원하는 부분의 색 이름을 바꿈으로써 변경할 수 있습니다. 링크에 특별한 색을 지정하지 않으려면 black이라고 입력하면 됩니다. 색 이름을 지정하는 방법에 대해서는 제28장을 참조하십시오.

제 18장

목차

이번 장에서는 \LaTeX 에서 목차를 만들고 수정하는 방법을 알아보겠습니다.

18.1

목차 만들기

\LaTeX 에서 목차를 생성하는 것은 간단합니다. 목차를 생성하려는 곳에 `\tableofcontents` 제어 문자열을 입력하면 끝입니다. 그림 목차나 표 목차도 같은 방법으로, `\listoffigures` 또는 `\listoftables` 제어 문자열을 입력하면 됩니다.

이들 제어 문자열이 입력되면, \TeX 은 컴파일 과정에서 소스 파일에 입력된 `\part`, `\chapter`, `\section` 등 장절 표제를 작성하는 제어 문자열이나, `\caption` 제어 문자열에 입력된 정보를 바탕으로 목차를 자동으로 생성합니다.

목차·표 목차·그림 목차는 보조 파일을 거쳐 생성됩니다. 매 컴파일 과정에서 \TeX 은 목차 내용이 저장된 `.toc`, `.lot`, `.lof` 파일을 업데이트하며, 또 이 파일에 저장되어 있는 내용을 바탕으로 목차를 생성합니다. 따라서 올바른 목차를 얻기 위해서는 컴파일 과정을 두 번 거쳐야 합니다. 이 과정을 그림으로 나타내면 아래와 같습니다.

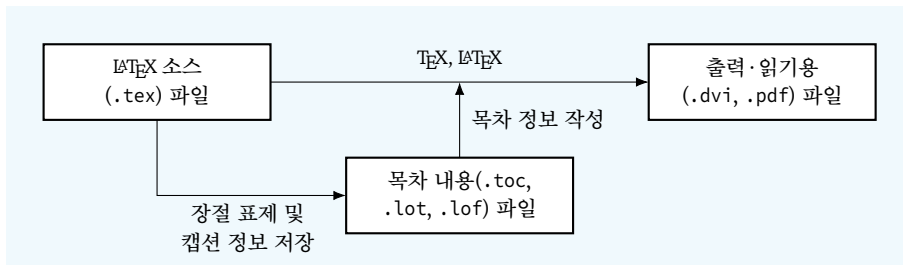


그림 18.1: \LaTeX 문서의 목차·표 목차·그림 목차 작성 과정

18.2

목차 내용 수정하기

이번 절에서는 목차의 여러 내용을 수정하는 방법을 알아보겠습니다. 여기에서 설명하는 내용 외에 목차의 각 항목을 조판하는 서식 등을 수정하려면 titletoc 패키지 등을 사용할 수 있습니다.

목차 제목 바꾸기

LaTeX에서 목차, 표 목차, 그림 목차를 생성하면 각각 ‘Contents’, ‘List of Tables’, ‘List of Figures’를 제목으로 한 목록이 출력됩니다. 한국어 문서를 작성하기 위해 kotex 패키지를 hangul 옵션과 함께 불러오면 ‘차례’, ‘표 차례’, ‘그림 차례’로 바뀌며, hanja 옵션과 함께 불러오면 ‘目次’, ‘表 目次’, ‘그림 目次’로 바뀝니다.

한편, 목차의 제목을 원하는 내용으로 바꿀 수 있습니다. 각각의 제목은 `\contentsname`, `\listtablename`, `\listfigurename` 제어 문자열에 저장되어 있으므로, 이 제어 문자열을 재정의해 주면 됩니다. 전 처리부에 (단, kotex 패키지를 사용한다면 이 패키지를 불러온 후에) 아래 코드를 입력하면 해당 내용으로 목차가 바뀝니다. `\title`에 원하는 제목을 입력하십시오.

```
1 \renewcommand\contentsname{\title}
2 \renewcommand\listtablename{\title}
3 \renewcommand\listfigurename{\title}
```

목차에 표시할 내용의 수준 바꾸기

article 클래스의 문서에서는 `\subsubsection`으로 입력한 내용까지, report 또는 book 클래스의 문서에서는 `\subsection`으로 입력한 내용까지 목차에 표시됩니다. 목차에 표시되는 내용의 수준을 변경하려면

```
\setcounter{tocdepth}{\level}
```

을 `\tableofcontents`의 앞에 입력하십시오. `\level`에는 표 6.1에서 원하는 수준의 표제에 대응하는 숫자를 입력합니다. 예를 들어, `\setcounter{tocdepth}{1}`을 입력하고 컴파일하면 `\section`에 입력한 내용까지만 목차에 작성됩니다.

목차에 항목 추가하기

`\chapter`나 `\section` 등의 제어 문자열 또는 `\caption`을 사용하지 않고도, 목차에 항목을 수동으로 추가할 수 있습니다. 이때에는 `\addcontentsline`을 사용합니다.

```
\addcontentsline{\extension}{\unit}{\text}
```

`\extension`: 항목을 추가할 목차 내용 파일의 확장자

`\unit`: 추가하려는 항목의 종류

`\text`: 추가하려는 내용

이 제어 문자열이 사용되는 대표적인 경우는 ‘차례’, ‘표 차례’, ‘그림 차례’를 목차에 추가하려 할 때와 번호 없는 항목을 목차에 추가하려 할 때입니다. 두 경우 모두 목차에 항목이 추가되지 않으므로, 수동으로 넣어 주어야 합니다.

〈*extension*〉에는 목차 파일의 확장자, 즉 toc(목차), lot(표 목차), lof(그림 목차) 중 하나를 입력하면 됩니다. 〈*unit*〉에는 추가하려는 항목의 종류를 입력합니다. 〈*extension*〉에 toc를 입력했다면 〈*unit*〉에는 장절 표제 수준의 이름, 즉 chapter, section 등을 입력하면 되며, 〈*extension*〉이 lot이면 〈*unit*〉에는 table을, lof이면 figure를 입력하면 됩니다. 마지막 인자 〈*text*〉에는 목차에 추가할 내용을 입력하면 됩니다.

예를 들어, \tableofcontents 앞에

```
\addcontentsline{toc}{section}{\contentsname}
```

을 입력하면 목차에 ‘차례’ 항목이 추가됩니다.¹ 또, 본문에서 \section*{연습문제}를 입력했을 때 목차에도 이 항목이 나타나게 하려면 이 명령어 뒤에

```
\addcontentsline{toc}{section}{연습문제}
```

를 입력하면 됩니다.

¹페이지 번호가 잘못 조판되는 경우, 문서 클래스가 book이거나 클래스 옵션으로 openright를 지정하였다면 \addcontentsline 앞에 \cleardoublepage를 추가하십시오. 또, 문서의 클래스가 report이거나 openany를 클래스 옵션으로 지정하였다면 \addcontentsline의 앞에 \clearpage를 추가하십시오.

제 19장

참고 문헌

\LaTeX 은 참고 문헌을 관리하는 기능도 제공합니다. \LaTeX 소스 파일과는 별개의 문헌 정보(데이터베이스) 파일을 사용하며, 이를 바탕으로 문서의 참고 문헌 목록을 생성하고, 또 문헌을 인용합니다.

문헌 목록을 생성할 때에는 글의 제목과 저자, 출판 연도 등을 정확하게 입력하고, 일정한 형식에 맞추어야 하는 경우가 많습니다. 이를 BibTeX이라는 프로그램을 사용하면 쉽게 처리할 수 있습니다. 이번 절에서는 BibTeX을 사용해 \LaTeX 에서 참고 문헌 목록을 생성하고, 문헌을 인용하는 방법을 알아봅니다.

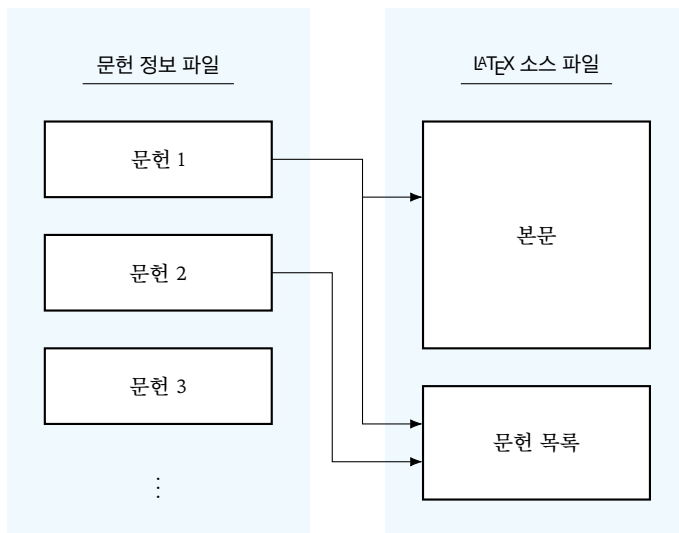
19.1

\LaTeX 에서 문헌을 관리하는 방법

\LaTeX 은 문헌 정보 파일을 사용해 문헌을 관리합니다. 이 파일의 내용은 자신이 직접 작성할 수도 있으며, 신뢰할 수 있는 출처로부터 파일을 받아서 사용할 수도 있습니다. 문헌 정보 파일 파일에는 다양한 책, 논문, 잡지, 보고서 등 다양한 종류의 문헌에 대한 각종 정보가 저장되어 있습니다. 제목과 저자, 출판 또는 작성 시기, 출판사, 학회, 관련 기관, 권 번호, 인용할 페이지 번호 등 수많은 정보가 구분되어 기록됩니다.

\LaTeX 소스 파일에서는 현재 문서에서 사용할 문헌 정보 파일을 불러오고, 문헌들을 정리하고 표시할 형식(스타일)을 지정해 줍니다. 문헌 정보 파일의 내용을 바탕으로 문헌 목록 작성과 인용 작업이 진행됩니다. 이때 소스 파일에서 인용하는 문헌의 정보만이 실제 문서 조판 과정에서 사용되므로, 공동의 문헌 정보 파일을 만들어 두고 여러 \LaTeX 문서에서 이 파일을 불러와 사용할 수 있습니다. (원한다면 본문에서 인용되지 않은 문헌들도 목록에 추가할 수 있습니다.)

즉, \LaTeX 소스 파일에서 사용되는 문헌은 그림 19.1과 같이 관리됩니다. 이러한 과정을 거쳐 소스 파일을 작성한 후 \LaTeX 과 BibTeX을 실행해 최종 문서를 얻게 됩니다. 다음 절부터는 문헌 정보 파일을 작성하는 방법과 문헌을 인용하는 방법, 그리고 컴파일 과정을 알아볼 것입니다.



문헌 1: 본문에서 인용되었고, 문헌 목록에도 등장하지 않습니다.

문헌 2: 본문에서 인용되지 않았지만, 문헌 목록에는 등장합니다.

문헌 3: 본문에서도 인용되지 않고, 문헌 목록에도 등장하지 않습니다.

그림 19.1: BibTeX을 사용해 참고 문헌을 관리하는 방법

19.2

문헌 정보 파일 작성하기

문헌 정보 파일은 .bib 확장자를 가지는 플레인 텍스트 파일입니다. 즉, LaTeX 소스 파일을 작성할 때처럼 임의의 텍스트 편집기에서 내용을 작성하고 저장하면 됩니다. 이때, BibTeX은 문헌 정보 파일에 작성된 내용을 처리할 때 올바른 스타일로 조판하기 위해 내용을 몇 가지 규칙에 따라 해석합니다. 따라서, 프로그램이 작성된 내용을 올바르게 이해할 수 있도록 내용을 작성할 때 지켜야 하는 규칙이 많습니다.

문헌 정보 파일의 형식

각 문헌의 정보는 아래 형식의 코드를 사용해 작성합니다.

```

1 @<entry type>{<key>,
2   <field> = <data>,
3   <field> = <data>,
4   ...
5   <field> = <data>
6 }
```

<entry type>에는 문헌의 종류(단행본, 논문, 웹 페이지 등)를 지정하며, <key>에는 각 문헌의 식별자를 (\label의 인자로 지정했던 것처럼) 지정합니다. 문헌의 자세한 정보는 <field>와 <data>를 사용해 작

예시 문서 19.1 mybib.bib

```

1 @book{texbook,
2   author = {Donald Ervin Knuth},
3   title = {The \TeX book},
4   publisher = {Addison-Wesley},
5   year = {1983},
6   address = {Reading, Messachusetts}
7 }
8 @book{latexdps,
9   author = {Leslie Lamport},
10  title = {\LaTeX : a document
11          preparation system},
12  publisher = {Addison-Wesley},
13  year = {1993},
14  address = {Reading, Messachusetts},
15  edition = {Second}
16 }
17 @book{tlc2e,
18   author = {Frank Mittelbach and Michel
19            Goossens},
20   title = {The \LaTeX Companion},
21   publisher = {Addison-Wesley},
22   year = {2004},
23   address = {Reading, Messachusetts},
24   edition = {Second}
25 }
26 @manual{lshort,
27   author = {Tobias Oetiker and Hubert
28            Partl and Irene Hyna and Elisabeth
29            Schlegl},
30   title = {The Not So Short Introduction
31            to \LaTeXe},
32   month = mar,
33   year = {2021},
34   note = {Accessed: October 2, 2022},
35 }

```

성합니다. *<field>*에는 문헌의 어떤 정보를 작성할 것인지(제목, 저자, 출판 연도 등)를, *<data>*에는 그 필드에 관한 구체적인 정보를 작성하면 됩니다.

사용할 수 있는 문헌의 종류는 표 19.1에서, 각 문헌에서 지정할 수 있는 필드의 이름은 표 19.2에서 확인할 수 있습니다. 필드는 필수, 선택, 무효 필드로 구분되며, 이 구분은 문헌의 종류에 따라 달라집니다. 표 19.3에서 이 정보를 확인할 수 있습니다.

- 필수 필수로 정보를 입력해야 하는 필드입니다. 이 필드에 관한 정보가 없으면 컴파일 이후 경고 메시지가 출력됩니다.
- 선택 필수로 지정할 필요는 없으나, 여기에 내용을 입력하면 문헌 목록에 추가 정보가 조판됩니다.
- 무효 컴파일 과정에서 무시되며, 문헌 목록에도 조판되지 않습니다. 문헌 정보 파일을 직접 읽는 사람에게 추가적인 정보를 주거나, 주석처럼 사용할 수 있습니다.

예를 들어, 이 책에서 인용하고 있는 문헌 중 일부의 정보는 예시 문서 19.1과 같이 작성합니다. 각 정보 사이에 쉼표를 생략하면 안 된다는 점을 주의하십시오. 파일을 저장할 때에는 확장자를 .bib로 하여 저장합니다.

정보를 입력할 때의 기본 규칙

<data> 부분에 입력하는 정보는 중괄호({ ... })나 큰따옴표(" ... ")로 감싸 주어야 합니다. 다만, year 필드와 같이 숫자만을 입력할 때에는 중괄호나 큰따옴표를 생략해도 됩니다.

정보를 작성할 때에는 원문의 표기를 최대한 살려서 입력합니다. 대·소문자 표기, 약어 표기 등을 최대한 살려서 입력해 줍니다. 다만, 축약된 내용을 풀어서 적으려는 경우, 작성자가 직접 추가한

표 19.1: 사용할 수 있는 문헌 종류

<ul style="list-style-type: none"> • article 학술지나 잡지 등에 실린 글 • book 출판 등록이 된(출판사가 존재하는) 책이나 단행본 • booklet 출판물로 만들어지고 제본이 되었지만, 출판 등록이 되지 않은 글 • conference inproceedings와 동일함 • inbook 책의 일부 — 하나의 장이나 절 또는 몇 페이지 등 • incollection 별도의 제목을 가지는 책의 일부분 • inproceedings 학회 논문집에서의 논문 	<ul style="list-style-type: none"> • manual 기술적인 문서나 사용 설명서 등 • mastersthesis 석사 학위 논문 • misc 기타 — 다른 모든 종류의 문헌에 속하지 않는 경우에 사용함 • phdthesis 박사 학위 논문 • proceedings 학회의 논문집 • techreport 학교나 기관 등에서 출판한 보고서로, 일련 번호가 있음 • unpublished 저자와 제목이 있지만 정식으로 출판되지 않은 글
---	--

표 19.2: 사용할 수 있는 문헌 정보 필드

<ul style="list-style-type: none"> • address 출판사나 기관 등이 위치한 주소 • annote 주석 — BibTeX의 표준 스타일에서는 사용되지 않지만 다른 스타일에서 주석을 사용하는 경우 이 키를 사용함 • author 저자(들)의 이름 • booktitle 인용되는 책의 제목 — 책의 일부를 인용하는 경우에 이 키를 사용하며, 책 전체를 인용할 때에는 title 키를 사용함 • chapter 장 또는 절 등의 번호 • crossref 다른 문헌의 정보를 현재 문헌에 그대로 사용하려 할 때, 해당 문헌의 참조 이름(<i><name></i>에 작성한 이름) — 현재 문헌에 추가 정보가 작성되어 있다면 참조된 문헌의 정보 대신 그 정보를 사용함 • edition 책의 판(edition) 번호 • editor 편집자(들)의 이름 • howpublished 정식으로 출판되지 않은 글의 출판된 방법 • institution 연구 보고서에서, 연구를 지원한 단체의 이름 	<ul style="list-style-type: none"> • journal 학술지의 이름 • key 저자나 편집자 정보가 없는 문헌에서, 문헌의 순서를 정렬하고 저자 이름으로 라벨을 만드는 경우 저자 이름 대신 사용할 내용 • month 글이 출판된(작성된) 날짜의 월 • note 읽는이에게 제공할 수 있는 추가 정보 — 첫 글자는 대문자로 적어야 함 • number 학술지, 잡지, 연구 보고서, 시리즈가 있는 글 등의 번호 • organization 학회를 주관했거나 매뉴얼을 출판한 단체 • pages 인용할 글의 페이지 번호나 범위 • publisher 출판사의 이름 • school 학위논문이 작성된 학교의 이름 • series 여러 권의 책이 하나의 시리즈로 출판된 경우에 그 시리즈의 이름 • title 글의 제목 • type 연구 보고서의 형식 • volume 학술지나 분권된 책의 권(volume) 번호 • year 글이 출판된(작성된) 날짜의 연도
--	--

다음 필드는 표준이 아니며, 특정 문헌 스타일에서 사용합니다.

<ul style="list-style-type: none"> • doi 글의 DOI • isbn 책의 ISBN 	<ul style="list-style-type: none"> • issn 출판물의 ISSN • url 웹 페이지의 URL
--	--

표 19.3: 문헌 종류에 따른 필드의 분류

필드 (field)	문헌 종류 (entry type)												
	article	book	booklet	inbook	incollection	inproceedings ¹	manual	mastersthesis	misc	phdthesis	proceedings	techreport	unpublished
address		선택	선택	선택	선택	선택	선택	선택	선택	선택	선택	선택	
annotate													
author ³	필수	필수	선택	필수	필수	필수	선택	필수		필수		필수	필수
booktitle					필수	필수							
chapter ⁴				필수	선택								
crossref													
edition		선택		선택	선택		선택						
editor ³				필수	선택	선택					선택		
howpublished			선택						선택				
institution												필수	
journal	필수												
key													
month	선택	선택	선택	선택	선택	선택	선택	선택	선택	선택	선택	선택	선택
note	선택	선택	선택	선택	선택	선택	선택	선택	선택	선택	선택	선택	필수
number ²	선택	선택		선택	선택	선택					선택	선택	
organization						선택	선택				선택		
pages ⁴	선택			필수	선택	선택							
publisher		필수		필수	필수	선택					선택		
school								필수		필수			
series		선택		선택	선택	선택					선택		
title	필수	필수	필수	필수	필수	필수	필수	필수	선택	필수	필수	필수	필수
type				선택	선택			선택		선택		선택	
volume ²	선택	선택		선택	선택	선택					선택		
year	필수	필수	선택	필수	필수	필수	선택	필수	선택	필수	필수	필수	선택

무효로 처리되는 필드는 빈칸으로 나타냈습니다. 이 표에 없는 필드는 기본적으로 무효로 처리되나, 문헌 표시 형식에 따라 다르게 처리될 수 있습니다.

¹ conference 문헌은 inproceedings의 내용을 참고하십시오.

² volume과 number 필드는 동시에 사용할 수 없습니다.

³ author 필드와 editor 필드가 모두 필수로 표시된 경우, 두 키 중 하나만을 지정해도 됩니다.

⁴ chapter 필드와 pages 필드가 모두 필수로 표시된 경우, 두 키 중 하나만을 지정해도 됩니다.

부분은 대괄호를 사용해야 합니다. 예를 들어, 원문에서 저자를 'D.E. Knuth'라고 표기하였다면 `author = "D.E. Knuth"`라고 적어야 하며, 이름을 축약하지 않고 표기하고자 한다면 `"D[onald] E. Knuth"`라고 입력해 줍니다.

입력된 정보는 문헌 목록을 조판할 때 BibTeX이 일관된 규칙에 따라 처리합니다. 이때 대·소문자가 변경될 수 있습니다. 만약 입력한 내용 일부분의 대·소문자가 변하지 않도록 하려면 그 부분을 중괄호로 감싸 주어야 합니다. 예를 들어, 제목이 'A research about Korean culture'인 문헌과 'A Research about Korean Culture'인 문헌의 정보는 각각 다음과 같이 작성합니다.

```
title = {A research about {Korean} culture}
title = {A Research about {Korean} Culture}
```

제어 문자열을 사용해 문자를 입력할 때에도 중괄호로 감싸 주어야 합니다. 예를 들어, 'L'Hôpital'은 `"L'H{\^o}pital"`이라고 입력해야 하며, 'The TeXbook'은 `"The {\TeX}book"`이라고 입력해야 합니다.

사람 이름을 작성할 때의 규칙

author 필드와 editor 필드에는 사람의 이름을 입력합니다. 이름을 작성할 때의 규칙은 다른 필드에 이름을 적을 때에 비해 매우 까다로우므로, 주의해서 작성해 주어야 합니다.

이름의 네 부분 BibTeX은 인명을 이름 부분(*<first>*), von 부분(*<von>*), 성 부분(*<last>*), Jr 부분(*<jr>*)으로 나누어 처리합니다.

- 이름 부분 이름(first name)과 가운데 이름(middle name)을 적습니다.
- von 부분 'von', 'de', 'van', 'van der' 등 소문자로 쓰는 접두어를 적습니다.
- 성 부분 성(last name)을 적습니다.
- Jr 부분 'Jr.', 'Sr.' 등을 적습니다. 앞에 쉼표(,)를 적어야 합니다.

서구권에서 인명을 적을 때에는 이름을 먼저 적기도 하고, 성을 적고 쉼표를 적은 뒤 그 뒤에 이름을 적기도 합니다. 따라서, 이에 따라 BibTeX이 이름을 올바르게 해석할 수 있도록 적절한 처리를 해주어야 합니다.

- 이름을 먼저 적는 경우 '*<first> <von> <last>*'의 형식으로 적습니다.
 - 소문자로 적은 부분이 있다면, 이 부분은 von 부분으로 처리됩니다. 이 부분의 앞뒤는 각각 이름 부분과 성 부분으로 처리됩니다.
 - von 부분이 없다면, 마지막 단어는 성으로, 나머지는 이름으로 처리됩니다. 성이 여러 단어인 경우에는 중괄호로 묶어 주어야 합니다.
- 성을 먼저 적는 경우 '*<von> <last>, <first>*'의 형식으로 적습니다.
 - 소문자로 적은 부분이 있다면, 이 부분은 von 부분으로 처리됩니다.
 - 쉼표를 기준으로 성 부분과 이름 부분이 구분됩니다.

- Jr 부분이 있는 경우 ‘*von*’ *last*), *jr*), *first*’의 형식으로 적습니다.
 - 소문자로 적은 부분이 있다면, 이 부분은 von 부분으로 처리됩니다.
 - 쉼표를 기준으로 성 부분과 Jr 부분, 이름 부분이 구분됩니다.

문헌 정보 파일에 인명을 작성할 때 어떤 문헌은 성을 앞에 적고 어떤 문헌은 이름을 앞에 적더라도, 문헌 목록을 조판할 때에는 지정한 스타일에 따라 일정하게 조판됩니다. 또, ‘Jr.’와 ‘Sr.’ 앞뒤의 쉼표도 자동으로 처리됩니다. 문헌 정보 파일에서는 출력 스타일을 신경쓰지 말고, 일정한 형식에 맞추어 정보를 입력하십시오.

표 19.4: BibTeX에 이름 올바르게 작성하기

이름	올바른 코드	이름 분석
Donald Ervin Knuth	Donald Ervin Knuth Knuth, Donald Ervin	Donald Ervin / · / Knuth / ·
Albert Einstein	Albert Einstein Einstein, Albert	Albert / · / Einstein / ·
Daniel Day-Lewis	Daniel Day-Lewis Day-Lewis, Daniel	Daniel / · / Day-Lewis / ·
Andrew Lloyd Webber	Andrew {Lloyd Webber} Lloyd Webber, Andrew	Andrew / · / Lloyd Webber / ·
Johannes Diderik van der Waals	Johannes Diderik van der Waals van der Waals, Johannes Diderik	Johannes Diderik / van der / Waals / ·
Ludwig van Beethoven	Ludwig van Beethoven van Beethoven, Ludwig	Ludwig / van / Beethoven / ·
Leonardo da Vinci	Leonardo da Vinci da Vinci, Leonardo	Leonardo / da / vinci / ·
Martin Luther King, Jr.	King, Jr., Martin Luther	Martin Luther / · / King / Jr.

두 가지 방법으로 입력할 수 있는 경우에는 ‘올바른 코드’ 열에 두 방법을 모두 나타냈습니다.

여러 사람의 이름을 입력하는 경우 여러 사람의 이름을 입력할 때에는 and로 구분합니다. 세 명 이상의 이름을 입력하는 경우에는 이름 사이사이마다 and를 입력해야 합니다. 예를 들어, [1]의 저자 정보는

```
author = "Frank Mittelbach and Michel Goossens and  
Johannes Braams and David Carlisle and Chris Rowley"
```

와 같이 입력할 수 있습니다. 인명에 ‘and’가 포함된다면 그 사람의 이름 전체를 중괄호로 감싸면 됩니다. 저자가 많아 일부를 생략하려 한다면 and others를 입력하면 됩니다. 문헌 표시 스타일에 따라 ‘et al.’ 등 적절한 문자열이 조판될 것입니다.

날짜를 입력할 때의 규칙

month 필드와 year 필드에는 각각 문헌이 출판된 월과 연도를 입력합니다. 월을 입력할 때에는 따옴표나 중괄호 없이 세 글자 약어(jan, feb, mar, ……)를 입력해야 하며, 연도를 입력할 때에는 네 자리 아라비아 숫자(2019, 1984 등)로 입력합니다. 예를 들어, 2021년 5월에 작성된 문헌의 작성 시기는 다음과 같이 작성합니다.

```
month = may,
year = 2021
```

판 번호

edition 필드에는 책의 판 번호를 작성합니다. 대문자로 시작하는 영어로 작성하며, ‘edition’, ‘ed.’, ‘e.’ 등의 문구는 작성하지 않습니다. 예를 들어, 어떤 책 제2판을 문헌 정보 파일에 작성한다면 edition 필드는 edition = {Second}라고 입력합니다.

다른 문헌의 정보 참조하기

다른 문헌의 정보를 가져와 현재 문헌의 빈 필드에 그대로 사용할 수도 있습니다. 이러한 경우 crossref 필드를 사용합니다. 아래 예시를 사용해 설명하겠습니다. (이 예시는 [7]에서 인용하였습니다.)

```
@INPROCEEDINGS{no-gnats,
  crossref = "gg-proceedings",
  author = "Rocky Gneisser",
  title = "No Gnats Are Taken for Granite",
  pages = "133-139"
}
@PROCEEDINGS{gg-proceedings,
  editor = "Gerald Ford and Jimmy Carter",
  title = "The Gnats and Gnus 1988 Proceedings",
  booktitle = "The Gnats and Gnus 1988 Proceedings"
}
```

식별자가 no-gnats인 문헌은 gg-proceedings의 여러 필드 중 no-gnats에 없는 필드의 정보를 상속받습니다. 이 경우에는 editor와 booktitle 필드의 정보를 상속받아 다음과 동일한 내용을 가지게 됩니다.

```
@INPROCEEDINGS{no-gnats,
  author = "Rocky Gneisser",
  editor = "Gerald Ford and Jimmy Carter",
  title = "No Gnats Are Taken for Granite",
  booktitle = "The Gnats and Gnus 1988 Proceedings",
  pages = "133-139"
}
```


이처럼, 두 문헌 사이에 여러 정보가 겹치는 경우에는 상호 참조 기능을 사용해 같은 내용을 여러 번 작성하는 것을 피할 수 있습니다.

문헌 인용 표지에 원하는 문자열 사용하기

문헌 참조 형식 중에는 저자의 이름을 사용하는 방식이 있습니다. 문헌 인용 표지에 축약된 저자의 이름과 출판 연도를 사용하여, 저자가 Donald E. Knuth이고 출판연도가 1984년이라면 '[Knu84]' 처럼 표시하는 식입니다. 저자의 이름이 지정되지 않았다면 editor 필드나 organization 필드의 정보를 대신 사용합니다.

하지만, 원하는 경우에는 key 필드를 사용해 문헌 인용 표지에 저자·편집자·단체 이름 대신 원하는 문자열을 사용하도록 할 수도 있습니다. 예컨대 organization 필드의 정보로 라벨이 생성되는 다음 상황을 생각해 봅시다. (이 예시는 [1]의 764쪽에서 인용하였습니다.)

```
organization = "The Association for Computing Machinery",
year = "1986"
```

이 문헌을 본문에서 인용하면 라벨이 '[Ass86]'으로 출력됩니다. 이때, 단체의 이름 앞쪽을 딴 'Ass'(관사 'The'는 자동으로 무시됩니다) 대신 단체의 약어 'ACM'이 출력되도록 하려면 key = "ACM"이라고 입력하면 되고, 그러면 문헌 인용 표지가 '[ACM86]'으로 바뀝니다.

19.3

L^AT_EX 소스 파일에서 문헌 정보 파일 사용하기

문헌 정보 파일을 모두 작성하였다면, 이 파일을 L^AT_EX 소스 파일에 불러온 후 문헌을 인용하고 문헌 목록을 생성할 차례입니다. 이번 절에서는 이 과정에서 사용하는 명령어를 설명합니다.

사용할 문헌 정보 파일 지정하기

맨 처음으로, L^AT_EX 문서에서 사용할 문헌 정보 파일을 지정해야 합니다. \bibliography 제어 문자열을 사용해 사용할 파일을 지정해 줍니다.

```
\bibliography{files}
```

<files>: 사용할 문헌 정보 파일의 이름. 확장자(.bib)는 제외하고 입력한다.

여러 문헌 정보 파일을 동시에 사용할 수도 있으며, 이때에는 각 파일을 쉼표로 구분합니다. 이렇게 지정한 파일에 저장돼 있는 문헌들을 인용할 수 있게 됩니다. 또, 인용된 문헌들의 목록은 이 제어 문자열이 입력된 곳에 조판됩니다. 일반적으로 참고 문헌 목록은 책의 맨 뒤, 색인 바로 앞에 작성되므로 이 제어 문자열은 소스 파일의 거의 끝부분에 입력됩니다.

인용하고 문헌 목록에 문헌 추가하기

사용할 문헌 정보 파일을 지정했다면, 문헌의 인용 표시를 넣으려는 곳에서 `\cite` 제어 문자열을 사용합니다.

```
\cite[⟨note⟩]{⟨key⟩}
```

⟨note⟩: 인용할 문헌에 관한 추가 정보

⟨key⟩: 인용하려는 문헌의 식별자

예를 들어, 식별자가 `texbook`인 문헌을 인용하려면 `\cite{texbook}`을 입력합니다.

여러 문헌을 동시에 인용할 수도 있으며, 이 경우 ⟨key⟩에 각 문헌의 식별자를 쉼표로 구분해 입력하면 됩니다. 예를 들어, 식별자가 `texbook`인 문헌과 `latexdps`인 문헌을 동시에 인용하려면 `\cite{texbook, latexdps}`를 입력합니다. 선택 인자 ⟨note⟩에 내용을 지정하면 문헌에서 인용하려는 페이지 번호 등 추가 정보를 독자에게 제공할 수 있습니다.

`\cite`를 사용해 인용한 문헌들의 자세한 정보는 앞서 설명했듯 참고 문헌 목록에 조판됩니다. 본문에서 인용하지 않은 문헌을 목록에 추가하려면 `\nocite` 제어 문자열을 사용하면 됩니다.

```
\nocite{⟨keys⟩}
```

⟨keys⟩: 목록에 추가할 문헌들의 식별자. 각 식별자는 쉼표로 구분한다.

문헌 참조 형식 지정하기

마지막으로, 문헌을 참조할 때와 목록을 조판할 때의 형식을 지정해 주어야 합니다. 이때 ‘형식’은 다음 사항을 어떤 형태로 조판할지 결정하는 규칙을 뜻합니다.

- 문헌 인용 표지로 사용하는 문자열(문헌의 번호, 저자 이름과 출판 연도, 문헌의 제목 등, 괄호 표시 여부)
- 문헌 목록에서 문헌들의 정렬 순서(본문에서 인용한 순서에 따라서, 저자 이름과 연도에 따라서, 제목에 따라서 등)
- 문헌 목록에서의 인용 표지 조판 여부
- 각 문헌의 세부 정보를 표시할 때의 서식(제목에 볼드체를 지정하는지, 저자의 이름은 성이 먼저 오는지, 학술지의 제목에 이탤릭체를 지정하는지 등)

CTAN에 등록된 문헌 참조 형식은 200여 가지가 넘으며, \LaTeX 의 표준 참조 형식으로는

`abbrv` `acm` `alpha` `apalake` `ieeetr` `plain` `siam` `unsrt`

가 있습니다. 수많은 참조 형식은 일반적으로 위 목록에서 첫 번째 항목에 따라, 크게 다음 세 가지로 구분할 수 있습니다.

- 각 문헌에 번호를 붙여 나타내는 방식 위 8개의 표준 형식 중에는 `abbrv`, `acm`, `ieeetr`, `plain`, `siam`, `unsrt`가 이 방식이며, 인용된 문헌이 단순히 ‘[1], [2], …’로 표시됩니다.

- 각 문헌을 저자와 출판 연도로 나타내는 방식 위 8개의 표준 형식 중에는 alpha, apalike가 이 방식을 사용합니다. 예를 들어, ‘[Knu84]’ 또는 ‘[Knuth, 1984]’처럼 표시됩니다.
- 짧은 제목을 표시하는 방식 문헌이 인용된 곳에서 문헌의 제목이 표시됩니다. 예를 들어, ‘*The T_EXbook* by Knuth’처럼 표시됩니다.

문헌 스타일은 전처리부에서 `\bibliographystyle` 제어 문자열을 사용해 지정합니다.

```
\bibliographystyle{<style>}
```

<style>: 사용할 문헌 참조 형식의 이름

예를 들어, 문헌을 plain 스타일을 사용해 표시하며, mybib.bib 파일을 문헌 정보 파일로 사용하려면 전처리부에는

```
\bibliographystyle{plain}
```

를 입력하고, 본문의 문헌 목록을 출력하려는 부분에서는

```
\bibliography{mybib}
```

을 입력합니다. 위의 표준 스타일 8가지 외에 다른 스타일의 목록은 bibtex.com에서 확인할 수 있습니다. 또, 각 스타일을 사용했을 때의 조판 예시도 여기에서 확인할 수 있습니다.

19.4

소스 파일 컴파일 하기

L^AT_EX 소스 파일과 문헌 정보 파일이 모두 준비되었다면, 컴파일 과정을 통해 PDF 문서를 얻을 차례입니다. 이때에는 네 단계를 거치게 됩니다. 먼저, L^AT_EX 소스 파일을 컴파일하여 보조 파일(.aux 파일)에 현재 문서에서 인용한 문헌의 정보를 업데이트 합니다. 둘째로, B_IB_TE_X을 실행하여, 문헌 정보 파일과 보조 파일을 바탕으로 문헌 목록을 조판하는 코드가 저장된 .bbl 파일을 생성합니다. 세 번째로, L^AT_EX 소스 파일을 컴파일 하여 .bbl 파일을 바탕으로 문헌 목록을 생성합니다. 마지막으로, L^AT_EX 소스 파일을 한 번 더 컴파일 하여 `\cite`를 사용한 부분에 문헌 인용 표지가 올바르게 표시되도록 합니다.

이 과정에서 사용되는 파일과 프로그램 사이의 관계를 그림 19.2에 정리하였습니다. 이때 .bst 파일은 문헌 참조 형식 파일이며, .bbl 파일은 B_IB_TE_X을 실행한 뒤 생성되는 로그 파일입니다.

Overleaf에서 작업하거나 latexmk를 기반으로 컴파일하는 몇 가지 편집기에서는 위 과정을 자동으로 처리해 줍니다. 소스 파일을 작성한 뒤 컴파일 과정을 한 번만 거쳤을 때, 참고 문헌 목록이 잘 생성되어 있다면 이러한 상황에 해당하는 것으로, 프로그램을 네 번 실행하는 과정을 거치지 않아도 됩니다. 컴파일 과정을 거친 뒤 참고 문헌 목록이 생성되지 않았다면, B_IB_TE_X을 실행한 후 L^AT_EX을 두 번 추가로 실행해 주면 됩니다.

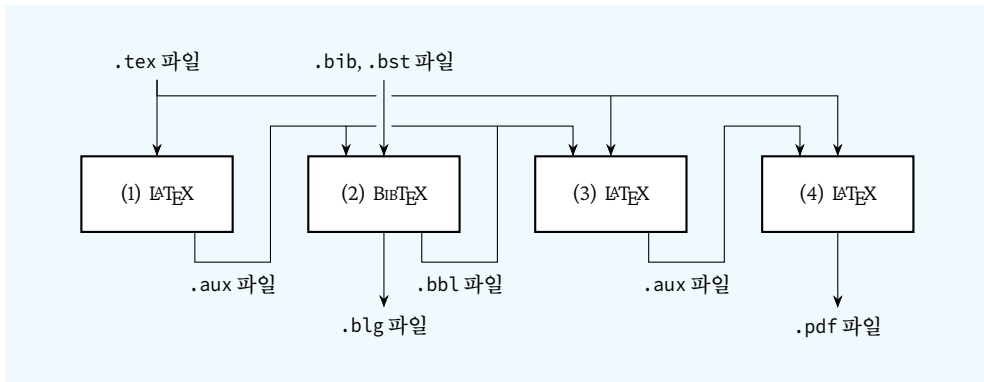


그림 19.2: BibTeX을 사용해 참고 문헌 목록을 조판하는 과정

참고 문헌 목록의 제목 바꾸기

참고 문헌 목록의 제목은 `\refname`이라는 제어 문자열에 저장되어 있습니다. 문서 클래스가 `article` 이면 ‘References’가, `report`나 `book`이면 ‘Bibliography’가 저장되어 있으며, `kotex` 패키지를 불러왔다면 옵션에 따라 ‘참고 문헌’이나 ‘著書 目錄’이 저장되어 있습니다. 만약 다른 제목으로 바꾸고 싶다면, 목차의 제목을 바꿀 때처럼 다음 코드를 전처리부에 입력하고, `\langle name \rangle` 부분에 원하는 제목을 입력하면 됩니다.

```
\renewcommand\refname{\langle name \rangle}
```

제 20 장

색인

적당한 전공책을 찾아, 책 뒷편에 있는 색인을 봅시다. 여기에는 수많은 용어들이 가나다순, 또는 알파벳순으로 정렬되어 있으며, 각 용어의 옆에는 그 용어의 정의가 있는 페이지 번호가 적혀 있습니다. 경우에 따라 특정 용어의 하위에 또 용어들이 나열되어 있기도 하고, 페이지 번호에 이탤릭체 또는 밑줄 등의 서식이 지정된 경우도 있습니다. 이번 장에서는 \LaTeX 문서에서 색인을 생성하는 방법을 알아보겠습니다.

20.1

\LaTeX 에서 색인을 생성하는 과정

\LaTeX 에서 색인을 만들 때에는 세 단계를 거칩니다. 첫째로, \LaTeX 소스 파일에서 `\index` 제어 문자열을 사용해 색인에 포함시킬 항목을 입력한 후 컴파일 합니다. 이 과정에서 색인 항목들이 저장된 `.idx` 파일이 생성됩니다. 둘째로, *MakeIndex* 또는 *xindy*라는 프로그램을 실행합니다. 이 프로그램은 `.idx` 파일에 저장된 항목을 올바른 순서로 정렬하여 `.ind` 파일을 생성합니다. 마지막으로, \LaTeX 소스 파일을 다시 한번 컴파일 하면 `.ind` 파일을 바탕으로 색인 부분이 출력 문서에 조판됩니다.

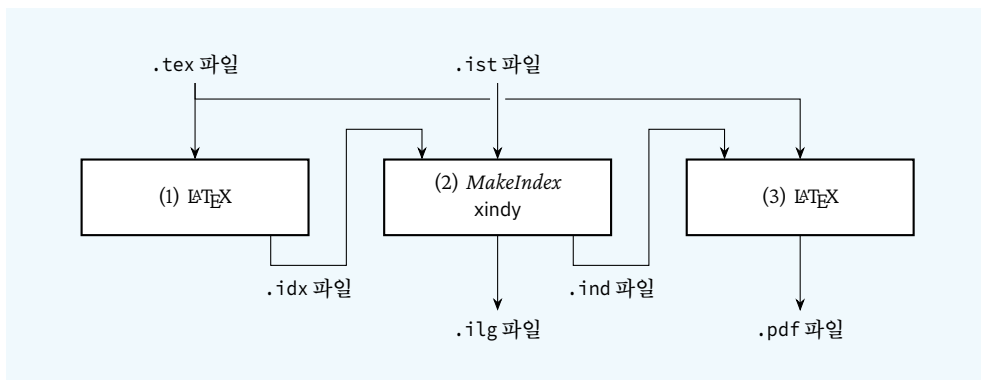


그림 20.1: \LaTeX 으로 색인을 생성하는 과정

그림 20.1에 \LaTeX 문서에서 색인을 생성하는 과정과 사용되는 파일을 나타냈습니다. 이때 `.ist` 파일은 색인을 조판하는 데 사용하는 스타일 파일이며, `.ilg` 파일은 색인 항목 정렬 프로그램을 실행한 뒤 생성되는 로그 파일입니다.

20.2

\LaTeX 소스 파일 작성하기

색인을 만들기 위해 제일 먼저, 소스 파일에 적절한 코드를 작성해야 합니다. 전처리부에서 `makeidx` 패키지를 불러오고, `\makeindex` 제어 문자열을 입력합니다. 또, 색인을 조판하고 싶은 위치에 `\printindex` 제어 문자열을 입력합니다. 일반적으로 색인은 문서의 맨 마지막에 위치하므로, 소스 파일의 코드는 대략 다음과 같은 순서로 있을 것입니다.

```
1 \documentclass{book}
2 ...
3 \usepackage{makeidx}
4 \makeindex
5 ...
6 \begin{document}
7 ...
8 \printindex
9 \end{document}
```

`\makeindex`는 \LaTeX 소스 파일 컴파일 과정에서 문서에 있는 `\index`의 인자를 `.idx` 파일에 보내도록 합니다. `\printindex`는 `.ind` 파일이 생성된 뒤, 이 파일을 불러들여 색인이 조판되도록 합니다. `.idx` 파일이 업데이트되지 않도록 하려면 `\makeindex`를, 색인이 조판되지 않도록 하려면 `\printindex`를 소스 파일에서 삭제하면 됩니다.

색인에 항목 추가하기

기본적인 코드를 입력했다면, 이제 본문에서 색인에 추가할 항목을 지정해 주어야 합니다. 색인 항목은 `\index` 제어 문자열의 인자로 지정합니다. 예를 들어, ‘군’이라는 용어를 색인에 추가하려면 간단히 `\index{군}`이라고 입력하면 됩니다. 나중에 색인이 조판되면 ‘군’이라는 용어의 옆에 `\index`가 입력된 페이지의 번호가 출력될 것이므로, 해당 용어가 등장하는 곳에 `\index`를 입력하면 됩니다. 예를 들어, 다음과 같이 사용합니다.

```
1 \begin{definition}
2   \textbf{군}\index{군}은 다음 조건을 만족하는
3   ...
4 \end{definition}
```

정의 1.1. 군은 다음 조건을 만족하는 연산 $*$ 을 갖춘 집합 G 이다. ...

같은 용어를 색인 항목에 여러 번 추가할 수도 있습니다. 이 경우 색인에서는 그 용어가 한 번만 나타나며, 해당 용어가 등장한 페이지 번호가 한꺼번에 표시됩니다.

하위 항목을 만들 때에는 `\index`의 인자를 ‘ $\langle entry \rangle!$ $\langle subentry \rangle$ ’의 형식으로 지정합니다. 하위 항목은 2단계까지 만들 수 있습니다. 예를 들어, 위 예시에 이어 ‘군’의 하위 항목으로 ‘가환군’을 추가하려면 `\index{군!가환군}`이라고 입력하면 되고, ‘동형정리’의 하위 항목 ‘군’의 하위 항목으로 ‘제1동형정리’를 추가한다면 `\index{동형정리!군!제1동형정리}`라고 입력하면 됩니다. 용어의 하위 항목을 만들 때에는 꼭 그 상위 용어가 색인 항목에 포함되어 있지 않아도 됩니다. 즉, `\index{준동형}`이 없더라도 `\index{준동형!군 준동형}`을 문제 없이 사용할 수 있습니다.

1쪽: \index{군}
 1쪽: \index{군!가환군}
 3쪽: \index{대칭군}
 4쪽: \index{교대군}
 4쪽: \index{순환군}
 4쪽: \index{생성원}
 5쪽: \index{준동형!군 준동형}
 5쪽: \index{동형!군 동형}
 6쪽: \index{부분군}
 7쪽: \index{정규 부분군}
 9쪽: \index{몫군}
 10쪽: \index{동형정리!군!제1동형정리}
 11쪽: \index{동형정리!군!제2동형정리}
 11쪽: \index{동형정리!군!제3동형정리}
 12쪽: \index{동형정리!군!제4동형정리}
 13쪽: \index{단순군}
 17쪽: \index{군의 작용}
 24쪽: \index{유한생성된 가환군의 기본정리}
 25쪽: \index{유한생성된 가환군의 기본정리
!유한 가환군의 기본정리}
 26쪽: \index{반직접곱}
 29쪽: \index{환}
 29쪽: \index{환!항등원이 있는}
 29쪽: \index{환!가환환}
 29쪽: \index{체}
 30쪽: \index{다항식환}
 30쪽: \index{준동형!환 준동형}
 30쪽: \index{동형!환 동형}
 31쪽: \index{부분환}
 32쪽: \index{아이디얼}
 32쪽: \index{아이디얼!주아이디얼}
 32쪽: \index{몫환}
 33쪽: \index{동형정리!환!제1동형정리}
 34쪽: \index{동형정리!환!제2동형정리}
 35쪽: \index{동형정리!환!제3동형정리}
 35쪽: \index{동형정리!환!제4동형정리}
 36쪽: \index{아이디얼!소아이디얼}
 36쪽: \index{아이디얼!극대아이디얼}
 41쪽: \index{가군}
 42쪽: \index{가군}
 43쪽: \index{가군}
 49쪽: \index{체}

찾아보기

ㄱ

가군, 41 - 43
 교대군, 4
 군, 1
 가환군, 1
 군의 작용, 17

ㄴ

다항식환, 30
 단순군, 13
 대칭군, 3
 동형

 군 동형, 2
 환 동형, 30

동형정리

 군

 제1동형정리, 6
 제2동형정리, 7
 제3동형정리, 7
 제4동형정리, 8

 환

 제1동형정리, 33
 제2동형정리, 34
 제3동형정리, 35
 제4동형정리, 35

ㄷ

몫군, 5
 몫환, 32

ㄹ

반직접곱, 26
 부분군, 3
 부분환, 31

ㅍ

생성원, 4
 순환군, 4

ㅎ

아이디얼, 32
 극대아이디얼, 36
 소아이디얼, 36
 주아이디얼, 32
 유한생성된 가환군의 기
 본정리, 24
 유한 가환군의 기본정
 리, 25

ㅈ

정규 부분군, 7
 준동형
 군 준동형, 2
 환 준동형, 30

ㅊ

체, 29, 49

ㅋ

환, 27
 가환환, 29
 항등원이 있는, 29

위는 색인을 생성한 예시입니다. 왼쪽과 같이 색인 항목을 지정한 후 색인이 생성되면 오른쪽의 결과를 얻게 됩니다.

색인 항목과 페이지 번호에 서식 지정하기

색인 항목을 추가할 때, 정렬 기준 문자열과 출력할 때의 모습이 다르게 나타나도록 할 수도 있습니다. 이때에는 ‘`\langle entry \rangle @ \langle format \rangle`’의 형식으로 `\index`의 인자를 지정하면 됩니다. 예를 들어, 체 K 위의 다항식환을 나타내는 기호 $K[x]$ 를 색인에 추가하되, 문자열 ‘ Kx ’를 기준으로 정렬하고 싶다면 `\index{Kx@(\(K[x]\))}`를 입력하면 됩니다. 또, ‘환’이라는 용어를 볼드체로 출력하려면 `\index{환@ \textbf{환}}`이라고 입력합니다.

한편, 용어를 색인에 추가하였을 때 페이지 번호의 모습을 다르게 표시할 수도 있습니다. 이때에는 ‘`\langle entry \rangle | \langle page number format \rangle`’의 형식으로 입력합니다. 세로선 |의 뒤에 있는 문자열을 이름으로 하는 제어 문자열이 실행됩니다. 인자를 입력받는 제어 문자열이라면 마지막 인자로 페이지 번호가 지정됩니다. 즉, 페이지 번호가 #1이라면 색인에는

```
\langle entry \rangle, \langle page number format \rangle \{#1\}
```

이 조판됩니다. 예를 들어, ‘체’라는 용어의 페이지 번호에 밑줄을 그으려면 `\index{체|underline}`이라고 입력합니다. 이는 색인 목록을 작성할 때

```
체, \underline{#1}
```

로 변환되어 입력됩니다. 또, ‘부분군’의 페이지 번호의 뒤에 ‘ff’를 붙이려면 `\ff`라는 제어 문자열을 다음과 같이 정의한 후 `\index{부분군|ff}`라고 입력하면 됩니다.

```
\newcommand*\ff[1]{#1\textit{ff}}
```

같은 용어가 세 페이지 이상 연속으로 참조되는 경우에는 페이지 번호가 자동으로 연결됩니다. 즉, 41쪽부터 43쪽까지 ‘가군’이라는 용어가 계속 참조되었다면 이 용어의 페이지 번호는 ‘41-43’이라고 출력됩니다. 한편, 각 페이지에서 용어를 참조하지 않고도 여러 페이지에서 용어가 언급된 것처럼 표시할 수도 있습니다. 언급하려는 첫 페이지에서 ‘`\langle entry \rangle | (`’를, 마지막 페이지에서 ‘`\langle entry \rangle |)`’를 입력하면 됩니다. 앞과 같은 상황이라면 41쪽에 `\index{가군| (}`을, 43쪽에 `\index{가군|)}`을 입력하면 같은 결과를 얻게 됩니다.

```
전처리부: \newcommand*\ff[1]{#1\textit{ff}}
6쪽:      \index{부분군|ff}
29쪽:     \index{환@ \textbf{환}}
29쪽:     \index{체|underline}
30쪽:     \index{다항식환}
30쪽:     \index{Kx@(\(K[x]\))}
31쪽:     \index{부분환|ff}
41쪽:     \index{가군| (}
43쪽:     \index{가군| )}
49쪽:     \index{체}
```

찾아보기

| | |
|-------------|-----------|
| | ㅂ |
| | 부분군, 6ff |
| K | 부분환, 31ff |
| $K[x]$, 30 | |
| ㄱ | 체, 29, 49 |
| 가군, 41 - 43 | |
| ㅎ | |
| ㄷ | 환, 29 |
| 다항식환, 30 | |

마지막으로, 앞서 언급했던 !, @, |를 색인 항목에 포함시키려는 경우에는 앞에 큰따옴표 "를 입력해 주면 됩니다. 큰따옴표를 출력하려면 마찬가지로 앞에 큰따옴표를 붙여 ""라고 입력합니다.

`\index`의 인자로 지정된 문자 중 "가 앞에 오는 문자들은 특수한 문법으로 해석되지 않기 때문입니다. 단, 제어 문자열 `\`"는 원래 의미(뒤에 오는 문자에 옴라우트를 붙임)로 해석됩니다.

20.3

색인 파일 생성하고 색인 조판하기

색인 항목을 모두 추가했다면, 먼저 `TeX`으로 컴파일 하여 색인 항목 파일을 생성합니다. 소스 파일의 이름이 `main.tex`이었다면 생성된 색인 항목 파일은 이름이 `main.idx`일 것입니다. 이후 `MakeIndex` 또는 `xindy`를 실행하여 색인 항목을 올바른 순서로 정렬해 줍니다. 프로그램을 실행해 `main.ind` 파일이 생성되었다면, `main.tex` 파일을 다시 한번 `TeX`으로 컴파일 해 주면 됩니다.

Overleaf에서 작업하거나 `latexmk`를 기반으로 컴파일하는 몇 가지 편집기에서는 이 과정을 자동으로 처리해 줍니다. 만약 소스 파일을 작성하고 컴파일을 한 번만 했는데 색인이 잘 생성되어 있다면 이 과정이 자동으로 처리된 것입니다. 소스 파일을 컴파일 했지만 색인이 생성되지 않았다면 색인 정렬 프로그램을 직접 실행해 주어야 합니다. 이번 절에서는 이 내용을 다루겠습니다. 따라서, 색인이 한 번의 컴파일 만으로 잘 생성된다면 이번 절을 넘어가셔도 됩니다.

색인 항목을 정렬하고 색인 부분 코드를 작성해 주는 프로그램에는 `MakeIndex`와 `xindy`의 두 가지가 있습니다. 영어 문서를 작성할 때에는 `MakeIndex`를 사용하더라도 색인을 잘 작성할 수 있지만, 프랑스어나 독일어 등 로마자를 사용하는 다른 언어에서는 `xindy`를 사용하는 것이 좋습니다. 한국어 문서에서는 두 프로그램 모두 잘 작동합니다.

MakeIndex로 색인 만들기

터미널을 열고 작업하는 파일이 있는 곳으로 이동합니다. 이후 터미널에 다음 명령을 입력합니다.

```
$ makeindex <file name>.idx
```

한국어 문서를 작성하고 있다면 다음과 같이 입력합니다.

```
$ komkindex -s kotex <file name>.idx
```

터미널에 다음 메시지가 나타났다면 `.ind` 파일이 정상적으로 작성된 것입니다. 이후 `TeX`으로 소스 파일을 다시 컴파일 하면 문서에 색인이 조판될 것입니다. 만약 오류가 발생했다는 메시지가 있으면 `\index`를 잘못 사용하지 않았나 소스 파일을 검토해 보십시오.

```
Output written in <file name>.ind.
Transcript written in <file name>.ilg.
```

xindy로 색인 만들기

마찬가지로, 터미널을 열고 작업하는 파일이 있는 곳으로 이동한 뒤, 레거시 엔진에서는

```
$ texindy -L <language> -I latex -C <code page> <file name>
```

을, 유니코드 엔진에서는

```
$ texindy -L <language> -I xelatex <file name>
```

를 입력한 후 실행하십시오. 특별한 경우가 아니라면 `<code page>`에는 utf8을 입력하면 되고, `<language>`에는 문서의 언어(예를 들어, english나 korean 등)를 입력하면 됩니다. 명령을 실행한 이후 터미널에 다음 메시지가 나타났다면 .ind 파일이 정상적으로 작성된 것입니다. 이후 \LaTeX 으로 소스 파일을 다시 컴파일 하면 문서에 색인이 조판될 것입니다. 만약 오류가 발생했다는 메시지가 있으면 `\index`를 잘못 사용하지 않았는지 소스 파일을 검토해 보십시오.

```
Writing markup... [10%] [20%] [30%] [40%] [50%] [60%] [70%] [80%] [90%]
[100%]
Markup written into file "./<file name>.ind".
```

색인 형식 바꾸기

`MakeIndex`는 스타일 파일을 사용하여, `xindy`는 모듈을 사용하여 색인의 형식을 바꿀 수 있습니다. 자세한 방법은 이 책 대신 각 프로그램의 매뉴얼 페이지나 [1]를 참조하십시오. 매뉴얼 페이지는 터미널에서 `man makeindex` 또는 `man texindy`를 입력해 읽을 수 있습니다.

색인의 제목은 `\indexname`이라는 제어 문자열에 저장되어 있습니다. 특별히 설정하지 않았다면 ‘Index’가 저장되어 있으며, `kotex` 패키지를 불러왔다면 ‘찾아보기’로 바뀝니다. `\renewcommand` 제어 문자열을 사용해 색인의 제목을 자신이 원하는 문자열로 바꿀 수 있습니다. 예를 들어 제목을 ‘색인’으로 바꾸고자 한다면 다음을 입력하면 됩니다.

```
\renewcommand\indexname{색인}
```


제 VI 편

효율적으로 L^AT_EX 문서 작성하기

제 VI 편에서는 L^AT_EX 문서를 효율적으로 작성하기 위한 방법을 설명합니다. 프로그래밍과 약간의 관련이 있는 L^AT_EX의 고급 기능을 다루므로, 이 부분의 내용은 다소 낯설 수 있습니다.

제21장에서는 매크로를 정의하는 방법을 알아봅니다. 반복적으로 사용하는 기능이나 복잡한 코드를 쉽게 사용할 수 있도록 매크로를 정의함으로써 큰 규모의 문서를 더 쉽게 작성할 수 있는 방법을 배웁니다.

제22장에서는 길이와 카운터에 대해 다룹니다. L^AT_EX 문서에서 번호를 매길 때 사용하는 내부 변수인 카운터와, 문서의 레이아웃을 지정할 때 사용하는 내부 변수인 길이 지정 제어 문자열을 다루는 방법을 알아봅니다.

제23장에서는 큰 규모의 문서를 작성할 때 파일을 분리하여 관리하는 방법을 설명합니다. 소스 파일을 여러 파일로 분리해 두면 코드를 관리하는 데 여러모로 편리합니다.

제 21 장

매크로 정의하기

이번 장에서는 \LaTeX 의 `\newcommand`, `\newenvironment` 제어 문자열을 사용해 매크로를 정의하는 방법을 알아보겠습니다.

21.1

매크로란?

매크로는 쉽게 말해 ‘함수’입니다. 함수에 인자를 대입하면 인자에 대응하는 하나의 함수값을 얻는 것처럼, 매크로를 정의하고, 그 매크로에 인자를 대입하면 매크로의 정의에 따라 일정한 결과를 출력합니다.

매크로를 정의하는 방법을 알아보기에 앞서, 먼저 매크로를 사용하는 이유를 알아보겠습니다. 예를 들어, 두 벡터의 내적을 출력하도록 하는 매크로 `\ip`를 정의해 봅시다. (자세한 문법은 뒤에서 설명할 것입니다.) 아래의 코드를 전처리부에 입력하고,

```
\newcommand{\ip}[2]{\left< #1, #2 \right>}
```

본문에 아래의 코드를 입력한 뒤 컴파일해 봅시다.

```
\(\ip{\mathbf{u}}{\mathbf{v}}\)
```

그러면, $\langle \mathbf{u}, \mathbf{v} \rangle$ 가 조판됨을 확인할 수 있습니다. 현재 문서의 다른 곳에서 \mathbf{x} 와 $\mathbf{0}$ 의 내적 $\langle \mathbf{x}, \mathbf{0} \rangle$ 을 입력해야 한다면, 또 간단히

```
\(\ip{\mathbf{x}}{\mathbf{0}}\)
```

을 입력하면 됩니다. 이처럼, 매크로를 정의하면 복잡한 문법을 간단하게 입력할 수 있습니다.

이제, \mathbf{u} 와 \mathbf{v} 의 내적을 $\langle \mathbf{u}, \mathbf{v} \rangle$ 가 아니라 $(\mathbf{u} | \mathbf{v})$ 로 표기하기로 마음을 바꿨다고 생각해 봅시다. 만약 매크로를 사용하지 않고, 내적을 표기할 때 이 문법을 일일이 입력했다면, 자신이 이 기호를 입력했던 곳들을 찾아가 일일이 수정해 주어야 합니다. 문서에서 내적 표기를 자주 사용했다면, 이를 수정하는 것조차 쉽지 않을 것입니다. 하지만, 위에서 정의한 매크로 `\ip`를 사용해 내적 표기를 정의했다면, 전처리부에 입력했던 `\ip`의 정의 부분을

```
\newcommand{\ip}[2]{\left( #1 \mid #2 \right)}
```

으로 바꾸어 주기만 하면 문서 내의 모든 내적 표기가 바뀝니다. 여기에서 매크로를 사용하는 두 번째 이유를 확인할 수 있습니다. 문서 내에서 자주 입력하는 내용이나 자주 사용하는 서식이 일관되게 나타나도록 할 수 있고, 변경할 때에도 쉽게 수정할 수 있습니다.

정리하면, \LaTeX 에서 매크로는 자주 사용하거나 복잡한 문법을 간단히 입력할 수 있도록 할 뿐 아니라, 문법의 일관성을 유지하고 쉽게 관리할 수 있도록 도와줍니다. 다음 절에서는 매크로를 정의하는 방법을 알아보겠습니다.

21.2

제어 문자열 정의하기

제어 문자열을 정의할 때는 `\newcommand` 또는 `\newcommand*` 제어 문자열을 사용합니다. 인자를 지정할 때 문단 나눔을 허용하려는 경우에는 별표 없는 것을, 그 외의 경우에는 별표 있는 것을 사용합니다. 일반적으로는 별표 있는 제어 문자열을 사용하는 것이 좋습니다.

```
\newcommand{<cname>}[<param no.>][<default arg.>]{<definition>}
\newcommand*{<cname>}[<param no.>][<default arg.>]{<definition>}
```

`<cname>`: 정의할 제어 문자열의 이름
`<param no.>`: 사용할 매개 변수의 개수
`<default arg.>`: 첫 번째 매개 변수의 기본값
`<definition>`: 제어 문자열의 정의

`<cname>`에 입력하는 제어 문자열은

- 역슬래시 + 로마자가 아닌 문자 1개
- 역슬래시 + 로마자로 이루어진 문자열

중 하나의 형태로 되어 있어야 합니다. \LaTeX 은 제어 문자열에서 로마자의 대소문자를 구분합니다.

예제 21.1 아래의 제어 문자열 이름의 후보 중, 적절한 것과 적절하지 않은 것을 구분하여라.

```
\HelloWorld      \print3times      \?              \eng2kor
\13!              \ifvalid          \the#ofchars     \N
```

이미 정의된 제어 문자열을 `<cname>`에 입력하면 컴파일 과정에서 오류가 발생합니다. 또, ‘end’나 ‘the’로 시작하는 이름의 제어 문자열은 각각 환경이나 카운터를 정의할 때 사용되므로 사용할 수 없습니다.

`<definition>`에는 제어 문자열의 정의를 입력하면 됩니다. 예시로 간단한 제어 문자열을 하나 정의해 봅시다. 가운데점을 출력하는 제어 문자열 `\textperiodcentered`가 너무 길어 간단히 `\textcdot`만을 입력해도 동일한 기호를 출력하도록 하려면, 전처리부에

```
\newcommand*{\textcdot}{\textperiodcentered}
```


를 입력하면 됩니다. 본문에서 `\textcdot`를 입력하면 컴파일 과정에서 이 제어 문자열이 그 정의인 `\textperiodcentered`로 치환되어 동일한 결과를 얻게 됩니다.

예제 21.2 `amssymb` 패키지를 불러온 후, `\mathbb{R}`와 `\mathbb{N}`을 수식 입력 모드에서 입력하면 각각 \mathbb{R} 와 \mathbb{N} 을 얻는다. 간단히 `\R`와 `\N`만을 입력해도 동일한 결과를 얻을 수 있도록 매크로를 정의하여라.

1차원 공간에서의 동차 파동 방정식 $u_{tt} - c^2 u_{xx} = 0$ 을 출력하는 제어 문자열 `\waveeqn`을 정의하려면

```
\newcommand*\waveeqn{u_{tt} - c^2 u_{xx} = 0}
```

를 입력하면 됩니다. 본문에서 `\waveeqn`를 입력하면 위 제어 문자열이 출력됩니다.

예제 21.3 이번에는 1차원 공간에서의 동차 확산 방정식

$$u_t - k u_{xx} = 0$$

를 출력하도록 하는 제어 문자열 `\diffeqn`를 정의해 보자.

인자를 입력받는 제어 문자열

지정하는 값에 따라 다른 결과를 만들어내는 매크로를 정의할 수도 있습니다. 매크로를 정의할 때, 입력받은 값을 저장하는 변수를 ‘매개 변수(parameter)’라고 합니다. 또, 정의한 매크로를 사용할 때 각각의 매개 변수에 지정한 값은 ‘인자(argument)’라고 합니다.

앞의 `\newcommand*`의 문법에서, 선택 인자인 $\langle param\ no. \rangle$ 에는 사용할 매개 변수의 개수를 지정합니다. 매개 변수를 사용하지 않으려면 이 인자를 생략하면 됩니다. 매개 변수는 최대 9개까지 사용할 수 있습니다.

매개 변수를 사용하도록 하였다면, 매크로를 정의할 때 매개 변수가 들어갈 자리에 #1, #2, ……를 입력하면 됩니다. 정의한 매크로를 사용할 때에는 제어 문자열을 입력하고, 그 뒤에 중괄호로 각 인자를 차례로 지정하면 됩니다.

앞서 정의한 `\waveeqn`의 정의를 수정하면서 매개 변수와 인자에 대해 알아봅시다. 우변의 함수를 나타내는 매개변수를 만들고, 여기에 지정한 인자를 우변에 출력하도록 매크로의 정의를 바꿔 봅시다. `\newcommand*`의 $\langle param\ no. \rangle$ 에 1을 입력하고, $\langle definition \rangle$ 에서 입력했던 0 대신 #1을 입력하여

```
\newcommand*\waveeqn[1]{u_{tt} - c^2 u_{xx} = #1}
```

로 수정하면 됩니다. 이후 `\waveeqn{f(x)}`를 입력하면 $u_{tt} - c^2 u_{xx} = f(x)$ 가 조판됩니다.

예제 21.4 앞 예제에서 정의했던 `\diffeqn`을 수정해, 우변의 함수를 입력받아 ‘0’ 대신 해당 식을 출력하도록 해 보자. 이후 `\diffeqn`을 사용해

$$u_t - k u_{xx} = \cos x$$

를 출력하여라.

예제 21.5 이 책에서 제어 문자열을 입력할 때에는 `\verb` 제어 문자열을 사용하지 않고, 직접 정의한 `\Lcs` 제어 문자열을 사용한다. `\Lcs{abcdefg}`를 입력하면 `\abcdefg`를 출력하는 이 제어 문자열을 직접 정의해 보아라. (힌트: 정의에는 `\texttt`와 `\textbackslash`가 사용된다.)

선택 인자가 있는 제어 문자열

선택 인자가 있는 제어 문자열을 정의할 수도 있습니다. 첫 번째 인자를 선택 인자로 만들 수 있으며, 이러한 경우에는 이 인자가 생략되었을 때 지정할 기본값을 *(default value)*에 적어 주어야 합니다.

예를 들어, 앞서 정의했던 `\waveeqn`에 첫 번째 매개 변수를 함수를 나타내는 기호로 사용하고, 생략하면 u 를 사용하도록 하며, 두 번째 매개 변수는 우변의 함수를 뜻하도록 하려면 정의를 다음과 같이 수정합니다.

```
\newcommand*{\waveeqn}[2][u]{\{{\#1}_tt - c^2 {\#1}_{xx} = {\#2}}
```

이후 `\waveeqn[v]{g(x)}`를 입력하면 $v_{tt} - c^2 v_{xx} = g(x)$ 가, `\waveeqn{h(x)}`를 입력하면 첫 번째 인자로 기본값 u 가 사용되어 $u_{tt} - c^2 u_{xx} = h(x)$ 가 출력됩니다.

예제 21.6 앞 예제에서 정의했던 `\diffeqn`을 수정해, 첫 번째 매개 변수로 상수 k 자리에 들어 갈 식을 지정하고, 두 번째 매개 변수로 우변의 함수를 지정할 수 있도록 하여라. 첫 번째 인자는 선택 인자로 만들고, 생략하면 아무 것도 출력되지 않도록 하여라. 이후 `\diffeqn`을 사용해 다음의 두 식을 조판하여라.

$$u_t - u_{xx} = e^x$$

$$u_t - 40u_{xx} = \alpha x^2 + \beta x + \gamma$$

수식 입력 모드와 텍스트 모드에서 모두 사용할 수 있는 매크로

`\frac`, `\mathbb`, `\rightarrow` 등 제어 문자열 중 일부는 수식 입력 모드에서만 사용할 수 있습니다. 따라서 이러한 제어 문자열을 새로운 매크로의 정의에 포함시키면 그 제어 문자열도 수식 모드에서만 사용할 수 있게 됩니다.

예를 들어, 미분 연산자를 간단하게 입력하기 위해

```
\newcommand*{\diff}[2]{\frac{d#1}{d#2}}
```

로 정의한 제어 문자열 `\diff`는 수식 입력 모드에서만 사용할 수 있으며, 텍스트 모드에서 $\frac{dy}{dx}$ 를 출력하려면 `\(\diff{y}{x}\)`와 같이 모드 전환 제어 문자열을 사용해 주어야 합니다. 이러한 경우 `\diff`의 정의에 `\ensuremath`를 사용해 제어 문자열을 모드 구분 없이 사용하도록 할 수 있습니다.

```
\newcommand*{\diff}[2]{\ensuremath{\frac{d#1}{d#2}}}
```

예제 21.7 앞에서 정의했던 `\R`의 정의를 수정해, `\R`를 수식 입력 모드와 텍스트 모드에서 모두 사용할 수 있도록 하여라.

기존 매크로 재정의하기

`\newcommand`나 `\newcommand*`를 사용해 매크로를 정의할 때는, 새로 정의하려는 매크로가 이미 정의되어 있으면 안 됩니다. 예를 들어, `\newcommand*{\LaTeX}{...}`을 입력하면 컴파일 과정에서 오류가 발생합니다. 기존에 정의된 매크로를 재정의하려면 `\renewcommand` 또는 `\renewcommand*` 제어 문자열을 대신 사용하면 됩니다. 사용하는 방법은 앞과 동일하지만, 반대로 재정의하려는 제어 문자열이 기존에 정의되어 있지 않았다면 오류가 발생합니다.

기존에 있던 매크로를 재정의할 때는 이 매크로의 원래의 정의는 무엇이었는지, 또 이 매크로가 다른 곳에서 쓰인 적은 없는지 주의해야 합니다. 문서를 작성하는 사람은 이 매크로를 직접 사용하지 않더라도, 다른 매크로(특히 패키지에서 정의하는 매크로)의 정의에 이 매크로가 포함되어 있을 수도 있기 때문입니다.

플레인 TeX의 명령어에 관한 조언

참고 | `\def`나 `\let`을 모르시는 분은 이 부분을 넘어가셔도 됩니다.

플레인 TeX의 `\def` 제어 문자열도 제어 문자열을 정의하는 데 쓰입니다. 이 제어 문자열은 `\newcommand`와 달리 정의하려는 제어 문자열이 이미 정의되어 있는지 확인하지 않으므로, 기존에 정의된 매크로를 재정의하는 문제를 일으키기 쉽습니다. 이러한 이유로 `\newcommand`를 `\def` 대신 사용하는 것이 바람직합니다.

한편, `\newcommand`를 사용해 매크로를 정의할 때에는 인자를 지정할 때 대괄호나 중괄호만을 사용해야 하고, `\def`의 ‘패턴 매칭’ 방식을 사용할 수 없습니다. 이때에는 `\NewDocumentCommand`를 대신 사용할 수 있습니다. (기존 매크로를 재정의할 때에는 `\RenewDocumentCommand`를 사용할 수 있습니다.) 이 제어 문자열은 L^AT_EX3의 문법을 사용하므로 이 책에서 더 설명하지 않겠습니다. 필요한 경우 `usrguide3.pdf` 문서를 참조하십시오.

플레인 TeX의 `\let`은 한 제어 문자열의 정의를 복사해 다른 제어 문자열을 정의합니다. `\let`도 `\def`와 마찬가지로 정의를 복사해 두려는 매크로가 이미 존재하는지 검사하지 않으므로 사용할 때 주의해야 합니다. 또, L^AT_EX의 많은 제어 문자열은 여러 보조 제어 문자열을 사용해 복잡하게 정의되므로, `\let`을 사용해 정의를 복사하면 복사된 정의가 제대로 작동하지 않을 수 있습니다. 매크로의 정의를 복사해야 한다면 `\NewCommandCopy` 또는 `\RenewCommandCopy`를 사용할 수 있습니다. 이 제어 문자열도 L^AT_EX3의 문법을 사용하므로, 자세한 정보는 `usrguide3.pdf`를 참조하십시오.

21.3

유용한 매크로들

자주 사용하는 기호나 문법은 매크로로 정의해 두면 문서를 작성할 때 요긴하게 사용할 수 있습니다. 긴 명령어를 축약하여 입력하기 편하게 하거나, 여러 명령어들의 조합으로 이루어진 명령어를 정의하는 두 가지 방식으로 주로 사용됩니다. 아래에는 정의해 두면 편리한 매크로를 모아 두었습니다.

수 집합

자연수 집합 \mathbb{N} , 정수 집합 \mathbb{Z} , 실수 집합 \mathbb{R} 등은 반복적으로 사용되는 경우가 많습니다. 장피에르 세르의 표기법에 따라 수 집합을 \mathbb{N} 대신 \mathbb{N} 을 사용하여 나타내기도 합니다.

```
1 \newcommand*\N{\mathbb N}
2 \newcommand*\Z{\mathbb Z}
3 \newcommand*\Q{\mathbb Q}
4 \newcommand*\R{\mathbb R}
5 \newcommand*\C{\mathbb C}
```

연산자와 함수

특정 연산자는 값을 나타낼 때 기준이 되는 대상을 아래 첨자로 나타내기도 합니다. 차원을 나타내는 \dim 이 그 예로, X 의 \mathbb{R} -벡터 공간으로서의 차원을 $\dim_{\mathbb{R}} X$ 로 표기하는 식입니다. 특정 스칼라 체를 자주 사용한다면 이를 제어 문자열로 정의해 간단하게 입력할 수 있습니다.

```
1 \newcommand*\dimr{\dim_{\R}}
2 \DeclareMathOperator{\codim}{codim}
3 \newcommand*\codimr{\codim_{\R}}
```

```
1 \begin{gather*}
2 \quad \dim X, \quad \dimr X \quad \\
3 \quad \codim X, \quad \codimr X \\
4 \end{gather*}
```

| | |
|-------------|-------------------------|
| $\dim X,$ | $\dim_{\mathbb{R}} X$ |
| $\codim X,$ | $\codim_{\mathbb{R}} X$ |

해석학 기호

변수가 무한대로 갈 때의 극한, 도함수와 편도함수, $\mathbb{R} = (-\infty, \infty)$ 나 $T = (-\pi, \pi)$ 에서의 적분, $e^{i\pi x}$ 꼴의 지수함수가 자주 사용됩니다.

```
1 \newcommand*\limn{\lim_{n\ra\infty}}
2 \newcommand*\limN{\lim_{N\ra\infty}}
3 \newcommand*\limm{\lim_{m\ra\infty}}
4 \newcommand*\limi{\lim_{i\ra\infty}}
5 \newcommand*\limj{\lim_{j\ra\infty}}
6 \newcommand*\limsupn{\limsup_{n\ra\infty}}
7 \newcommand*\liminfn{\liminf_{n\ra\infty}}
8
9 \newcommand*\dif[2]{\frac{d\#1}{d\#2}}
10 \newcommand*\dl[3]{\dif{\#1}{\#2}\bigg|_{\#3}}
11 \newcommand*\pa{\partial}
12 \newcommand*\bpa{\bar{\partial}}
13 \newcommand*\pdif[2]{\frac{\pa\#1}{\pa\#2}}
14 \newcommand*\pdl[3]{\pdif{\#1}{\#2}\bigg|_{\#3}}
```

```

15
16 \newcommand*\intr{\int_{-\infty}^{\infty}}
17 \newcommand*\intrp{\int_0^{\infty}}
18 \newcommand*\intt{\int_{-\pi}^{\pi}}
19
20 \newcommand*\eitx{e^{2\pi itx}}
21 \newcommand*\emitx{e^{-2\pi itx}}

```

이렇게 정의한 제어 문자열은 아래와 같이 사용할 수 있습니다.

```

1 \Lambda \limn f_n = \limn \Lambda f_n
2 \nabla f = \pdif fx\hat{\mathrm x} + \pdif
  fy\hat{\mathrm y}
3 df = \pdl fxy\,dx + \pdl fyx\,dy
4 \intt f(\theta) \,d\theta
5 \hat f(t) = \intr f(x)\emitx \,dx

```

$$\begin{aligned}
 \Lambda \lim_{n \rightarrow \infty} f_n &= \lim_{n \rightarrow \infty} \Lambda f_n \\
 \nabla f &= \frac{\partial f}{\partial x} \hat{x} + \frac{\partial f}{\partial y} \hat{y} \\
 df &= \frac{\partial f}{\partial x} \Big|_y dx + \frac{\partial f}{\partial y} \Big|_x dy \\
 &\quad \int_{-\pi}^{\pi} f(\theta) d\theta \\
 \hat{f}(t) &= \int_{-\infty}^{\infty} f(x) e^{-2\pi itx} dx
 \end{aligned}$$

대형 연산자

합과 곱 등을 나타내는 대형 연산자는 특정 범위에서 자주 사용됩니다.

```

1 \newcommand*\sumn{\sum_{n=1}^{\infty}}
2 \newcommand*\sumi{\sum_{i=1}^n}
3 \newcommand*\sumz{\sum_{-\infty}^{\infty}}
4 \newcommand*\prodn{\prod_{n=1}^{\infty}}
5 \newcommand*\prodi{\prod_{i=1}^n}
6 \newcommand*\cupn{\bigcup_{n=1}^{\infty}}
7 \newcommand*\cupi{\bigcup_{i=1}^n}
8 \newcommand*\capn{\bigcap_{n=1}^{\infty}}
9 \newcommand*\capi{\bigcap_{i=1}^n}

```

이렇게 정의한 제어 문자열은 아래와 같이 사용할 수 있습니다.

```

1 \begin{displaymath}
2 \quad \sumn \frac{1}{n^2} = \frac{\pi^2}{6}
3 \quad A = \bigoplus_{i \in I} A_i
4 \end{displaymath}

```

$$\sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6} \quad A = \bigoplus_{i \in I} A_i$$

21.4

환경 정의하기

새로운 환경을 정의할 때에는 `\newenvironment` 및 `\newenvironment*` 제어 문자열을 사용합니다.

```
\newenvironment{<envname>}[<param no.>][<dflt arg.>]{<defn start>}{<defn end>}
\newenvironment*{<envname>}[<param no.>][<dflt arg.>]{<defn start>}{<defn end>}
```

`<envname>`: 정의할 환경의 이름
`<param no.>`: 사용할 매개 변수의 개수
`<dflt arg.>`: 첫 번째 매개 변수의 기본값
`<defn start>`: 환경 시작 부분에서의 정의
`<defn end>`: 환경 끝 부분에서의 정의

환경의 정의에 두 개의 인자를 사용하는 것을 제외하면, `\newcommand`와 동일한 문법으로 사용합니다. 환경을 정의했다면 `\begin{<envname>}`은 `<defn start>`에 입력한 내용으로, `\end{<envname>}`은 `<defn end>`에 입력한 내용으로 치환됩니다. 이때 `<defn end>` 부분에는 매개 변수 #1~#9를 사용할 수 없다는 점을 주의하십시오.

`\newcommand*`와 마찬가지로, `\newenvironment*`를 사용해 정의한 환경을 사용할 때에는 인자에 문단 나눔을 포함할 수 없습니다. 환경 안의 내용에는 문단 나눔을 입력할 수 있습니다.

예를 들어, 다음과 같이 `subjectintrobox` 환경을 정의할 수 있습니다.

```
1 \newenvironment*{subjectintrobox}[3]
2   {\begin{tabular}{@{}p{\linewidth}@{}} \hline \textbf{#1} {\footnotesize
3     #2} \hfill #3 \\
4   {\end{tabular}}}
```

이렇게 정의한 환경은 다음과 같이 사용합니다.

```
1 \begin{subjectintrobox}{선형대수
2   학}{MAS212}{3:0:3}
3 \end{subjectintrobox}\par
4 \medskip
5 \begin{subjectintrobox}{해석학
6   I}{MAS241}{3:2:4}
7 \end{subjectintrobox}
```

| | |
|--|-------|
| 선형대수학 MAS212 | 3:0:3 |
| 벡터 공간과 선형 변환을 바탕으로 하여, 선형대수학과 관련된 다양한 내용을 다룬다. | |
| 해석학 I MAS241 | 3:1:4 |
| 실수 공간의 완비성을 다루고, 이에 따라 실수 열의 극한, 급수의 수렴성, 실수 값 함수들의 극한과 미분, 적분 등을 다룬다. | |

한편, 이미 존재하는 환경을 다시 정의하려면 `\renewenvironment` 또는 `\renewenvironment*`를 동일한 방식으로 사용하면 됩니다. `\renewcommand`를 사용할 때와 마찬가지로, 자신이 정의를 수정하려는 환경이 다른 곳에서 사용되고 있지는 않은지 주의하십시오.

제 22 장

카운터와 길이 정의하기

TeX에서는 매크로 외에도 다른 용도의 제어 문자열을 정의할 수 있습니다. 이번 장에서는 카운터와 길이를 정의하고 사용하는 방법을 알아보겠습니다.

22.1

카운터

‘카운터(counter)’란, 프로그래밍에 비유하자면 TeX에서 사용하는 정수형 변수로, 번호를 기록할 때 사용합니다. 페이지 번호와 장절 번호에서부터, 그림 번호, 표 번호, 수식 번호까지 모두 TeX의 카운터 기능을 사용합니다. 심지어 enumerate 환경에서 번호 붙은 목록을 작성할 때의 항목의 번호도 카운터를 사용해 조판합니다.

프로그래밍할 때 변수를 선언하고 초기화하는 것처럼, TeX에서 카운터를 사용할 때에도 카운터를 선언한 후 초기화해야 합니다. 필요에 따라 해당 카운터에 저장된 값을 사용할 수도, 변경할 수도 있습니다.

카운터에 값 저장하기

가장 먼저, 새로운 카운터를 선언하려면 `\newcounter` 제어 문자열을 사용합니다.

```
\newcounter{<name>}[<parent>]
```

`<name>`: 선언하려는 카운터의 이름

`<parent>`: 현재 카운터를 종속시킬 카운터

`\newcounter`를 사용하면 `<name>`을 이름으로 하는 카운터가 만들어지며, 이 카운터에 저장된 값은 0으로 초기화됩니다. 카운터의 이름은 아래의 조건을 만족해야 합니다.

- 로마자 52자(A-Z, a-z)만으로 구성되어야 합니다. 로마자의 대소문자는 구분합니다.
- 이미 존재하지 않는 이름이어야 합니다.

선택 인자인 `<parent>`를 지정하면 이 카운터를 `<parent>` 카운터에 종속시킬 수 있습니다. 카운터의 종속에 대해서는 뒤쪽에서 다룹니다.

카운터에 저장된 값은 `\setcounter`나 `\addtocounter` 제어 문자열을 사용해 바꿀 수 있습니다.

`\setcounter{<name>}{<value>}`

`<name>`: 값을 바꾸려는 카운터의 이름

`<value>`: 새로 저장할 값

`\addtocounter{<name>}{<value>}`

`<name>`: 값을 바꾸려는 카운터의 이름

`<value>`: 원래의 값에 더할 값

한편, 카운터를 사용해 번호를 매길 때에는 값을 1씩 증가시키는 일이 많습니다. 이때에는 위의 제어 문자열 대신 `\stepcounter` 또는 `\refstepcounter` 제어 문자열을 사용합니다. `\refstepcounter`를 사용한 경우에는 `\label`과 `\ref` 제어 문자열을 사용해 해당 카운터의 값을 사용할 수 있습니다.

앞서 설명한 제어 문자열을 다음과 같이 사용할 수 있습니다. (여기에서 사용한 `\themycounter`는 `mycounter`에 저장된 값을 보여주는 제어 문자열입니다. 이에 대해서는 뒤쪽에서 자세히 설명합니다.)

```
1 \newcounter{mycounter}%
2 mycounter 카운터를 새로 만들었습니다. 초기에 저장된
   값은 \themycounter 입니다. \par
3 \setcounter{mycounter}{12}%
4 mycounter 카운터의 값을 \themycounter\로 변경하
   였습니다. \par
5 \addtocounter{mycounter}{3}%
6 이번에는 mycounter 카운터에 3을 더해 값을
   \themycounter\로 변경하였습니다. \par
7 \stepcounter{mycounter}%
8 mycounter 카운터의 값을 1 증가시켰습니다. 현재
   \themycounter\이 저장되어 있습니다. \par
9 \refstepcounter{mycounter}\label{mylabel}%
10 \verb|\refstepcounter|를 사용하면 카운터에 저장
   된 값에 상호 참조 기능을 사용할 수 있습니다.
   mycounter를 참조하면 \ref{mylabel}을 얻습니다.
```

mycounter 카운터를 새로 만들었습니다. 초기에 저장된 값은 0입니다.

mycounter 카운터의 값을 12로 변경하였습니다. 이번에는 mycounter 카운터에 3을 더해 값을 15로 변경하였습니다.

mycounter 카운터의 값을 1 증가시켰습니다. 현재 16이 저장되어 있습니다.

`\refstepcounter`를 사용하면 카운터에 저장된 값에 상호 참조 기능을 사용할 수 있습니다. mycounter를 참조하면 17을 얻습니다.

카운터에 저장된 값 사용하기

이제 카운터에 저장한 값을 사용하는 방법을 알아보시다. 카운터에 저장된 값을 문서에 조판하여 우리가 눈으로 확인할 수 있도록 하기도 하고, 값을 다른 명령어의 인자 등으로 사용하도록 할 수도 있습니다.

먼저, `\arabic`, `\Roman`, `\roman`, `\Alph`, `\alph`, `\fnsymbol` 제어 문자열은 카운터에 저장된 값을

각각 아라비아 숫자, 로마 숫자 대·소문자, 로마자 대·소문자, 기호로 출력합니다. 인자로 값을 출력하려는 카운터의 이름을 입력하면 됩니다.

```

1 \setcounter{mycounter}{3}
2 \arabic{mycounter},\qqquad \Roman{mycounter},\qqquad
3 \roman{mycounter},\qqquad \Alph{mycounter},\qqquad
4 \alph{mycounter},\qqquad \fnsymbol{mycounter} \par
5 \stepcounter{mycounter}
6 \arabic{mycounter},\qqquad \Roman{mycounter},\qqquad
7 \roman{mycounter},\qqquad \Alph{mycounter},\qqquad
8 \alph{mycounter},\qqquad \fnsymbol{mycounter}

```

| | | | | | |
|----|------|------|----|----|---|
| 3, | III, | iii, | C, | c, | ‡ |
| 4, | IV, | iv, | D, | d, | § |

\fnsymbol은 외국 문서에서 주석을 달 때 종종 사용합니다. 1부터 9까지의 번호가 아래 표의 대응하는 기호로 표시됩니다. 10 이상의 수는 표시되지 않습니다.

표 22.1: \fnsymbol의 번호에 대응하는 기호

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|----|----|----|
| * | † | ‡ | § | ¶ | | ** | †† | ‡‡ |

kotex 패키지를 불러온 상태에서는 카운터 번호를 출력할 때 한국어 문서에서 자주 사용하는 원문자(①, ②, …… , ㉠, ㉡, ……), 괄호 문자((a), (b), ……), 자음자나 음절자(ㄱ, ㄴ, …… , 가, 나, ……) 등을 사용할 수 있습니다. 사용할 수 있는 카운터 형식과 출력 예시를 표 22.2에 나타냈습니다.

표 22.2: kotex 패키지에서 제공하는 카운터 출력 형식

| 카운터 형식 | 출력 예시 | | | | 카운터 형식 | 출력 예시 | | | |
|--------|-------|-----|-----|-----|-----------|-------|-----|-----|-----|
| \jaso | ㄱ | ㄴ | ㄷ | ㄹ | \gana | 가 | 나 | 다 | 라 |
| \ojaso | ㉠ | ㉡ | ㉢ | ㉣ | \ogana | ㉠ | ㉡ | ㉢ | ㉣ |
| \pjaso | (ㄱ) | (ㄴ) | (ㄷ) | (ㄹ) | \pgana | (가) | (나) | (다) | (라) |
| \onum | ① | ② | ③ | ④ | \oeng | ㉠ | ㉡ | ㉢ | ㉣ |
| \pnum | (1) | (2) | (3) | (4) | \peng | (a) | (b) | (c) | (d) |
| \hnum | 하나 | 둘 | 셋 | 넷 | \hroman | i | ii | iii | iv |
| \Hnum | 첫째 | 둘째 | 셋째 | 넷째 | \hRoman | I | II | III | IV |
| \hNum | 일 | 이 | 삼 | 사 | \hanjanum | 一 | 二 | 三 | 四 |

여기의 제어 문자열도 앞과 마찬가지로, 출력하려는 카운터의 이름을 인자로 지정하면 됩니다.

```

1 \setcounter{mycounter}{9}
2 \jaso{mycounter},\qqquad \gana{mycounter},\qqquad
3 \hnum{mycounter},\qqquad \hNum{mycounter}

```

스, 자, 아홉, 구

카운터의 번호를 출력할 때 기본 형식을 지정하고, 이 형식에 따라 번호가 일정하게 출력되도록 할 수도 있습니다. 이름이 $\langle name \rangle$ 인 카운터는 $\backslash the\langle name \rangle$ 이라는 제어 문자열로 출력하며, 형식의 기본값은 $\backslash arabic\{\langle name \rangle\}$ 입니다. 형식을 $\backslash renewcommand*$ 를 사용해 바꾸어 줄 수도 있습니다.

```

1 \setcounter{mycounter}{12}
2 \newcounter{subcounter}\setcounter{subcounter}
  {15}
3 \themycounter\quad\thesubcounter \par
4 \renewcommand*\{\themycounter}
  {\Roman{mycounter}}
5 형식을 바꾸어 출력한 mycounter: \themycounter \par
6 \renewcommand*\{\thesubcounter}
  {\themycounter.\arabic{subcounter}}
7 형식을 바꾸어 출력한 subcounter: \thesubcounter

```

12 15

형식을 바꾸어 출력한 mycounter: XII

형식을 바꾸어 출력한 subcounter: XII.15

한편, 정수를 인자로 입력받는 제어 문자열이나 환경에서 카운터의 값을 사용하려면 $\backslash value$ 제어 문자열을 사용합니다. 제어 문자열의 인자로 카운터의 이름을 입력하면 됩니다. 예를 들어, subcounter에 저장된 값을 mycounter 카운터에도 저장한다면 다음과 같이 입력합니다.

```

1 \setcounter{mycounter}{\value{subcounter}}

```

LaTeX의 기본 카운터

앞에서 설명하였듯, LaTeX은 몇 가지 카운터를 기본적으로 정의하여 사용합니다. 이를 표 22.3에 정리하였습니다.

표에 정리된 카운터 외에도, 불러온 패키지에 따라 다른 카운터가 추가로 정의될 수도 있습니다.

카운터의 종속

한 카운터가 다른 카운터에 종속되도록 하거나, 이미 종속된 카운터를 더 이상 종속되지 않도록 할 수도 있습니다. 카운터 A가 다른 카운터 B에 ‘종속’되면, 카운터 B의 값이 변경되었을 때 카운터 A의 값이 0으로 초기화됩니다. 카운터 사이에 상하 관계를 만들 때 이 기능을 사용합니다. 예를 들어, subsection 카운터는 section 카운터에, section 카운터는 chapter 카운터에 종속되어 있습니다. 또, enumii 카운터는 enumi 카운터에 종속되어 있습니다.

앞서 설명하였듯 $\backslash newcounter$ 제어 문자열을 사용해 정의할 때 선택 인자 $\langle parent \rangle$ 를 지정하면, 새로 만들어지는 $\langle name \rangle$ 카운터가 $\langle parent \rangle$ 카운터에 종속됩니다. $\backslash counterwithin$ 제어 문자열과 $\backslash counterwithout$ 제어 문자열을 사용해 종속 관계를 설정하거나 해제할 수도 있습니다.

$\backslash counterwithin\{\langle name \rangle\}\{\langle parent \rangle\}$

$\langle name \rangle$: $\langle parent \rangle$ 카운터에 종속시킬 하위 카운터의 이름

$\langle parent \rangle$: 상위 카운터의 이름

표 22.3: L^AT_EX에서 기본적으로 정의한 카운터

| 카운터 이름 | 설명 |
|---------------|----------------------------------|
| part | 편(part)의 번호를 저장합니다. |
| chapter | 장(chapter)의 번호를 저장합니다. |
| section | 절(section)의 번호를 저장합니다. |
| subsection | 하위 절(subsection)의 번호를 저장합니다. |
| subsubsection | 최하위 절(subsubsection)의 번호를 저장합니다. |
| paragraph | 문단(paragraph)의 번호를 저장합니다. |
| subparagraph | 하위 문단(subparagraph)의 번호를 저장합니다. |
| figure | 그림의 번호를 저장합니다. |
| table | 표의 번호를 저장합니다. |
| equation | 수식의 번호를 저장합니다. |
| page | 페이지 번호를 저장합니다. |
| footnote | 각주 번호를 저장합니다. |
| mpfootnote | minipage 환경 내에서의 각주 번호를 저장합니다. |
| enumi | 번호 붙은 목록에서 1단계 번호를 저장합니다. |
| enumii | 번호 붙은 목록에서 2단계 번호를 저장합니다. |
| enumiii | 번호 붙은 목록에서 3단계 번호를 저장합니다. |
| enumiv | 번호 붙은 목록에서 4단계 번호를 저장합니다. |

`\counterwithout{<name>}{<parent>}`

`<name>`: 더 이상 종속되지 않도록 할 카운터의 이름

`<parent>`: `<name>` 카운터가 종속되었던 상위 카운터의 이름

위 두 제어 문자열을 사용하면 `\the<name>`이 재정의됩니다. `\counterwithin`을 사용한 경우에는 `\the<parent>.\arabic{<name>}`으로, `\counterwithout`을 사용한 경우에는 `\alph{<name>}`으로 재설정됩니다. 각 제어 문자열의 별표 붙은 버전인 `\counterwithin*`과 `\counterwithout*`을 사용하면 카운터의 종속 관계는 설정·해제되지만, `\the<name>`은 재정의되지 않습니다.

예를 들어, article 클래스에서 equation 카운터는 다른 카운터에 종속되어 있지 않습니다. 이를 절 번호에 종속시키려면 다음과 같이 입력합니다.

```
\counterwithin{equation}{section}
```

한편, report 또는 book 클래스에서는 equation 카운터가 장 번호에 종속되어 있습니다. 이때의 종속 관계를 해제하려면 다음과 같이 입력합니다.

```
\counterwithout{equation}{chapter}
```

calc 패키지를 사용한 연산

calc 패키지를 사용하면 카운터에 값을 저장할 때 덧셈, 뺄셈, 곱셈, 나눗셈의 산술 연산을 사용할 수 있습니다. 각각 +, -, *, / 기호를 사용하며, 계산 결과가 실수인 경우 버림하여 정수 결과가 저장됩니다. 예를 들어, subcounter에 3이 저장되어 있을 때 다음 코드는 mycounter 카운터에 10을 저장합니다.

```
\setcounter{mycounter}{\value{subcounter}*2+4}
```

calc 패키지를 불러오면 두 개의 인자를 입력받는 \maxof, \minof 제어 문자열을 사용할 수도 있습니다. 각각의 이름에서 알 수 있듯 두 값 중 최댓값, 최솟값을 반환합니다.

22.2

길이

TeX은 제어 문자열을 길이를 저장하는 변수로써도 사용합니다. 이에 대해 알아보기에 앞서, TeX에서 길이를 지정하는 문법을 이번 절에서 먼저 설명하겠습니다.

TeX으로 문서를 작성하다 보면 길이를 지정할 일이 생깁니다. 표의 폭을 지정하거나, 구분선의 길이를 지정하는 등의 상황에서 길이를 지정해야 합니다. TeX에서 길이는 다음과 같이 지정합니다.

길이 → $\langle sign \rangle \langle numbers \rangle \langle unit \rangle$

$\langle sign \rangle$: +와 - 중 하나를 부호로 지정. +이면 생략할 수 있음

$\langle numbers \rangle$: 부호 없는 정수나 (소수로 나타낸) 실수

$\langle unit \rangle$: 길이의 단위

즉, 1cm, -1.25in, +35pt와 같이 입력하면 됩니다. TeX에서 사용할 수 있는 길이 단위는 표 22.4에서 확인할 수 있습니다.

한편, 제어 문자열에 저장되어 있는 길이를 하나의 단위처럼 사용할 수도 있습니다. 이때에는 다음과 같이 입력합니다.

길이 → $\langle sign \rangle \langle factor \rangle \langle csname \rangle$

$\langle sign \rangle$: +와 - 중 하나를 부호로 지정. +이면 생략할 수 있음

$\langle factor \rangle$: $\langle csname \rangle$ 에 저장된 값에 곱할 값

$\langle csname \rangle$: 길이가 저장된 제어 문자열

예를 들어, 문단 사이의 간격이 저장된 \parskip의 2배의 길이는 2\parskip으로, -1.35배는 -1.35\parskip으로 입력하면 됩니다.

유동적 길이

경우에 따라 ‘유동적 길이’를 사용하기도 합니다. 유동적 길이를 사용하면 지정된 길이를 기준으로, 문서를 조판할 때의 상황(상하좌우의 여유 공간 등)에 따라 어느 정도 늘이거나 줄인 길이를 대신하여

표 22.4: TeX에서 사용할 수 있는 길이의 단위

| 단위 | 이름 | 설명 |
|----|------------------------|---------------------|
| pt | point | |
| pc | pica | 1 pc = 12 pt |
| in | inch | 1 in = 72.27 pt |
| bp | big point | 72 bp = 1 in |
| cm | centimeter | 2.54 cm = 1 in |
| mm | millimeter | 10 mm = 1 cm |
| dd | didot point | 1157 dd = 1238 pt |
| cc | cicero | 1 cc = 12 dd |
| sp | scaled point | 65536 sp = 1 pt |
| ex | x-height | 대략 현재 글꼴의 소문자 x의 높이 |
| em | | 대략 현재 글꼴의 대문자 M의 폭 |
| mu | math unit ¹ | 18 mu = 1 em |

¹ mu는 수식 입력 모드에서 특수한 명령어와 함께 사용됩니다.

사용할 수 있습니다. 유동적 길이는 길이를 지정하는 모든 상황에서 사용할 수 있는 것은 아닙니다.

유동적 길이 → $\langle \text{dimen.} \rangle$ plus $\langle \text{stretchability} \rangle$ minus $\langle \text{shrinkability} \rangle$

$\langle \text{dimen.} \rangle$: 기준 길이. 공간이 남거나 좁지 않은 경우에는 이 값을 사용한다.

$\langle \text{stretchability} \rangle$: 길이를 늘이는 것을 허용할 최댓값. 공간이 남는 경우 여기에 저장된 길이만큼을 늘일 수 있다.

$\langle \text{shrinkability} \rangle$: 길이를 줄이는 것을 허용할 최댓값. 공간이 부족한 경우 여기에 저장된 길이만큼을 줄일 수 있다.

필요에 따라 plus $\langle \text{stretchability} \rangle$ 와 minus $\langle \text{shrinkability} \rangle$ 중 하나 이상을 생략할 수 있습니다. 두 값을 모두 생략하면 유동적 길이를 사용할 수 있는 환경에서 반드시 고정된 길이를 사용하도록 할 수 있습니다.

예를 들어, 문단 사이의 간격을 지정하는 `\parskip`의 기본값은 0 pt plus 1 pt입니다. 페이지에 여러 문단을 배치할 때, 본문 조판 영역에 여유 공간이 생기면 문단 사이사이에 최대 1 pt의 간격을 추가하는 식으로 페이지가 조판됩니다.

22.3

길이 변수 제어 문자열

앞 절에서 TeX의 길이 문법을 알아보았습니다. 이번 절에서는 제어 문자열을 길이 변수로써 사용하는 문법을 알아보겠습니다. 첫째로, 새로운 길이 변수를 선언할 때에는 `\newlength` 제어 문자열을 사용합니다.

```
\newlength{<cname>}
```

<cname>: 새로 선언할 제어 문자열의 이름

\newlength를 사용하면 <cname>에 입력된 제어 문자열이 길이를 저장하는 변수로써 선언되며, 그 길이는 0pt로 초기화됩니다. 제어 문자열의 이름은 매크로나 카운터를 정의할 때와 같은 조건을 만족해야 합니다. 길이가 저장된 제어 문자열은 \setlength와 \addtolength를 사용해 값을 변경할 수 있습니다.

```
\setlength{<cname>}{<dimen.>}
```

<cname>: 값을 바꾸려는 제어 문자열

<dimen.>: 새로 저장할 길이

```
\addtolength{<cname>}{<dimen.>}
```

<cname>: 값을 바꾸려는 제어 문자열

<dimen.>: 원래의 값에 더할 길이

\setlength를 사용하면 <dimen.>에 입력된 값을 <cname>에 저장합니다. 한편, \addtolength를 사용하면 <dimen.>에 지정된 값을 <cname>이 원래 가지고 있던 값에 더합니다.

예를 들어, \mylength를 길이 변수 제어 문자열로써 선언하려면 아래와 같이 입력합니다.

```
\newlength\mylength
```

\mylength에는 길이 0pt가 저장되어 있습니다. \mylength에 2in의 길이가 저장되도록 하려면

```
\setlength\mylength{2in}
```

을 입력하면 됩니다. 또, 이 상태에서

```
\addtolength\mylength{3in}
```

를 입력하면 \mylength에 3in의 길이가 더해져 5in가 저장됩니다.

길이 변수 제어 문자열에 저장된 값을 문서에 조판하려면 \the를 해당 제어 문자열 앞에 붙여 줍니다. 예를 들어, 위에서 정의했던 \mylength를 사용하여 \the\mylength를 입력하면 361.34999pt가 조판됩니다(길이는 pt 단위로 변환되어 출력됩니다).

calc 패키지를 사용한 연산

카운터와 마찬가지로, calc 패키지를 사용하면 제어 문자열에 길이를 저장할 때 덧셈, 뺄셈, 곱셈, 나눗셈의 산술 연산을 사용할 수 있습니다. 곱셈과 나눗셈을 할 때에는 곱하거나 나눌 수보다 길이가 더 앞쪽에 와야 하며, 곱하거나 나누는 수가 정수가 아니라면 \real 제어 문자열의 인자로 지정해 주어야 합니다. 즉, \mylength의 1.6배는 \mylength*1.6이라고 입력해야 합니다. \maxof와 \minof 제어 문자열도 동일하게 사용할 수 있습니다.

또, `\widthof` 제어 문자열을 사용할 수 있습니다. 이 제어 문자열은 인자를 한 개 입력받으며, 여기에 입력된 문자열을 조판했을 때의 폭을 측정해 반환합니다. `calc` 패키지의 더 다양한 기능은 패키지 안내서를 참조하시기 바랍니다.

제 23 장

문서 파일 분리하기

큰 규모의 문서를 작성할 때는, 하나의 문서 파일에 모든 내용을 작성하는 것보다, 여러 개의 파일에 나누어 내용을 작성하고 이를 합쳐서 컴파일하는 것이 좋습니다. 한 파일에 코드를 모두 입력한 경우에는 문서의 특정 부분을 어디에 작성했는지 스크롤해서 확인해야 하는 반면, 파일이 분리되어 있다면 해당 파일을 찾아 열어 주면 되기 때문입니다. 이번 장에서는 \LaTeX 에서 소스 파일을 분리해 문서를 작성하는 방법을 알아보겠습니다.

\LaTeX 제어 문자열

파일을 분리해 문서를 작성하는 가장 간단한 방법은 \LaTeX 을 사용하는 것입니다. \LaTeX 은 하나의 인자를 입력받으며, 여기에 파일의 경로와 이름을 입력하면 그 파일의 내용이 현재 파일의 이 위치에 입력됩니다. 경로나 파일 이름에 공백 문자가 포함되어 있다면 해당 파일은 큰따옴표로 감싸 주어야 합니다.

예를 들어, 현재 작업 중인 폴더에 `main.tex` 파일과 `equation.tex` 파일이 있고, 두 파일의 내용이 아래와 같다고 가정해 봅시다.

main.tex:

```
1 \documentclass[a4paper]{article}
2 \usepackage{mathtools,amssymb}
3 \begin{document}
4 The time-independent Schrödinger
  equation is given as follows:
5 \input{equation}
6 \end{document}
```

equation.tex:

```
1 \begin{equation}
2 \frac{1}{2m} \left( i\hbar \frac{d}{dx} \right)^2 \psi + U\psi =
  \frac{1}{2m} (p)^2 \psi + U\psi = E\psi
3 \end{equation}
```

이후 `main.tex` 파일을 컴파일하면 아래의 결과를 얻게 됩니다.

The time-independent Schrödinger equation is given as follows:

$$\frac{1}{2m} \left(i\hbar \frac{d}{dx} \right)^2 \psi + U\psi = \frac{1}{2m} (p)^2 \psi + U\psi = E\psi \quad (1)$$

주의할 점으로, `\input`하는 파일에는 다음의 세 가지 코드를 사용하면 안 됩니다.

```
\documentclass      \begin{document}      \end{document}
```

`\input`은 `\input`되는 소스 파일에 중첩하여 사용할 수도 있습니다. 예를 들어, 아래와 같이 `main.tex` 파일을 비롯해 여러 파일을 분리해 작성하고, 각 장의 내용을 담은 소스 파일은 `main.tex` 파일이 저장된 폴더 하위의 `chapters` 폴더에, 각 절의 내용을 담은 소스 파일은 `sections` 폴더에 분리해 둘 수도 있습니다. 전처리부에 작성할 내용이 많은 경우 `preamble.tex` 파일을 따로 작성해서 포함시킬 수도 있습니다.

`main.tex:`

```
1 \documentclass[a4paper]{book}
2 \input{preamble}
3 \title{Sample Document}
4 \author{Anonymous}
5
6 \begin{document}
7 \frontmatter
8 \maketitle
9 \input{chapters/preface}
10 \tableofcontents
11
12 \mainmatter
13 \input{chapters/chapter1}
14 \input{chapters/chapter2}
15 \appendix
16 \input{chapters/appendix}
17
18 \end{document}
```

`chapters 폴더 > chapter1.tex:`

```
1 \chapter{The First Chapter}
2 \input{sections/section1-1}
3 \input{sections/section1-2}
4 \input{sections/section1-3}
```

`chapters 폴더 > chapter2.tex:`

```
1 \chapter{The Second Chapter}
2 \input{sections/section2-1}
3 \input{sections/section2-2}
```

`chapters 폴더 > appendix.tex:`

```
1 \chapter{The Appendix}
2 \input{sections/sectionA-1}
3 \input{sections/sectionA-2}
4 \input{sections/sectionA-3}
```

`\include` 제어 문자열

`\input` 대신 `\include` 제어 문자열을 사용할 수도 있습니다. `\include`를 사용하는 경우에는 코드를 삽입하기 전후에 페이지가 나뉘며, `\include`는 중첩하여 사용할 수 없습니다(`\include`되는 파일 안에서 `\input`은 사용할 수 있습니다). 따라서, book이나 report 문서에서 하나의 장을 하나의 파일에 작성하고, 이를 `\include`하는 방식으로 사용할 수 있습니다. `\input`과 마찬가지로 `\include`되는 파일에 `\document`나 `\begin{document}`, `\end{document}`를 입력하면 안 되며, `\include`는 전처리부에서 사용할 수 없습니다.

전처리부에서 `\includeonly` 제어 문자열을 사용하면 `\include`한 파일 중 일부만을 실제 컴파일에 사용하도록 할 수 있습니다. 이때 상호 참조 정보 등은 유지되어, 삽입되지 않은 파일에 작성된 대상을 참조하더라도 페이지 번호나 대상의 번호는 올바르게 표시됩니다. `\includeonly` 제어 문자열의 인자로 컴파일에 사용할 파일들의 이름을 쉼표로 구분해 입력하면 됩니다.

예를 들어, 위의 예시에서 일부 명령어를 수정하여 다음과 같이 사용할 수 있습니다.

```

1 \documentclass[a4paper]{book}
2 \input{preamble}
3 \includeonly{chapters/chapter2,chapters/appendix}
4 \title{Sample Document}
5 \author{Anonymous}
6
7 \begin{document}
8 \frontmatter
9 \maketitle
10 \include{chapters/preface}
11 \tableofcontents
12
13 \mainmatter
14 \include{chapters/chapter1}
15 \include{chapters/chapter2}
16 \appendix
17 \include{chapters/appendix}
18
19 \end{document}

```

위 문서를 컴파일하면 제2장과 부록 부분만 실제 컴파일에 사용되며, 서문과 제1장은 컴파일 되지 않습니다. 제3행을 주석 처리하면 모든 파일이 포함된 채로 컴파일 됩니다. `\input`으로 불러온 파일인 `preamble.tex`은 영향을 `\includeonly`의 영향을 받지 않으며, 항상 포함된 채로 컴파일 됩니다.

제 VII 편

문서 사용자 지정하기

제 VII 편에서는 문서의 서식을 원하는 대로 바꾸는 방법을 설명합니다. 문서를 사용자화할 수 있는 범위는 매우 넓기 때문에 이 책에서 모든 부분을 다루지는 않고, 자주 사용하는 기능 위주로 설명하였습니다.

제24장에서는 문서의 레이아웃, 즉 여백 설정과 다단 편집에 관해 다룹니다. 다양한 길이 변수에 저장된 값을 바꿈으로써 여백의 폭과 단의 수를 원하는 대로 바꿀 수 있습니다. 문단의 형태를 바꾸는 방법도 설명합니다.

제25장에서는 공백을 삽입하고 구분선을 넣는 방법을 알아봅니다. 이들을 적절히 사용하면 문서의 내용을 더 쉽게 알아볼 수 있습니다.

제27장에서는 문서의 글꼴을 변경하는 방법을 다룹니다. 특정 글꼴로 설정해 주는 패키지를 사용하거나, fontspec 패키지를 통해 TrueType 및 OpenType 글꼴을 지정하는 방법을 알아봅니다.

제28장에서는 문서에 다양한 색을 넣는 방법을 알아봅니다. 글자와 수식의 색을 바꾸고, 색깔 있는 글상자를 그리는 방법을 알아봅니다.

제 24 장

레이아웃 편집하기

이번 장에서는 문서의 레이아웃을 바꾸는 방법을 알아보겠습니다. \LaTeX 에서 문서 조판에 사용하는 인자를 바꾸거나 기본 명령어를 사용할 수도 있고, 외부 패키지를 사용할 수도 있습니다.

24.1

페이지 설정 바꾸기

\LaTeX 은 많은 명령어를 사용해 페이지의 레이아웃을 설정합니다. 종이의 크기, 여백의 크기, 머리글과 바닥글을 조판하기 위한 공간 등이 모두 명령어를 통해 결정됩니다.

먼저, 페이지의 크기는 `\paperwidth`와 `\paperheight`로 지정합니다. 일반적인 경우에는 문서의 클래스 옵션으로 지정하면 되지만, 옵션으로 지정할 수 없는 크기를 사용할 때에는 이 명령어에 길이를 직접 지정해 주면 됩니다.

둘째로, 본문 조판 영역의 크기는 `\textwidth`와 `\textheight`로 지정합니다. 이때 본문 조판 영역은 상하좌우 여백을 제외하고, 실제로 문자가 조판될 수 있는 부분의 크기를 의미합니다.

셋째로, 왼쪽과 위쪽 여백은 `\oddsidemargin`과 `\topmargin`으로 지정합니다. 왼쪽 여백은 종이의 왼쪽 끝에서부터 본문 조판 영역 왼쪽 끝까지의 간격을, 위쪽 여백은 종이의 위쪽 끝에서부터 머리글 조판 영역 위쪽 끝까지의 간격을 가리킵니다. 이 제어 문자열에 저장된 값에서 1인치씩을 뺀 값이 실제 여백이 됩니다. 오른쪽 여백과 아래쪽 여백은 이상의 제어 문자열을 바탕으로 계산됩니다. 한편, 문서를 양면 편집하는 경우에는 `\evensidemargin`으로 짝수 페이지의 왼쪽 여백을 따로 지정할 수 있습니다.

머리글과 바닥글에 관련된 명령어에는 `\headheight`, `\headsep`, `\footskip`이 있습니다. 각각 머리글을 조판하는 영역의 높이, 머리글의 베이스라인과 본문 조판 영역 위쪽 사이의 간격, 바닥글의 베이스라인과 본문 조판 영역 아래쪽 사이의 간격을 가리킵니다.

이상에서 설명한, 각 제어 문자열이 페이지 레이아웃에서 나타내는 길이를 그림 24.1에 나타냈습니다. 각 제어 문자열이 어디부터 어디까지의 길이를 나타내는지 그림을 통해 더 명확하게 확인하십시오.

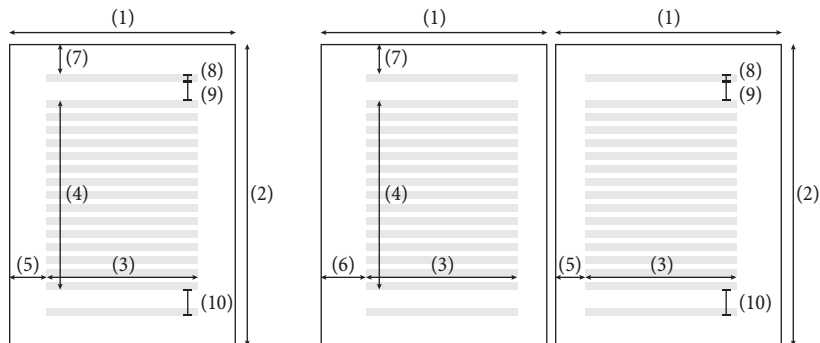


그림 24.1: 페이지 레이아웃. 왼쪽 그림은 단면 편집 시, 오른쪽 그림은 양면 편집 시의 레이아웃이다.

(1) `\paperwidth` (2) `\paperheight` (3) `\textwidth` (4) `\textheight` (5) `\oddsidemargin+1inch`
 (6) `\evensidemargin+1inch` (7) `\topmargin+1inch` (8) `\headheight` (9) `\headsep` (10) `\footskip`

전처리부에서 각각의 제어 문자열에 저장된 길이 값을 `\setlength` 및 `\addtolength` 제어 문자열을 사용하여 원하는 값으로 바꾼 뒤 컴파일하면 해당 레이아웃이 적용된 문서가 생성됩니다. 예를 들어, 이 책에서 사용한 설정은 다음과 같습니다.

```
1 \setlength\paperwidth{188mm}
2 \setlength\paperheight{257mm}
3 \setlength\textwidth{13cm}
4 \setlength\textheight{21cm}
5 \setlength\oddsidemargin{2.3cm}
6 \addtolength\oddsidemargin{-1in}
7 \setlength\evensidemargin{3.5cm}
8 \addtolength\evensidemargin{-1in}
9 \setlength\topmargin{1.5cm}
10 \addtolength\topmargin{-1in}
11 \setlength\headheight{5mm}
12 \setlength\headsep{7mm}
```

`geometry` 패키지를 사용하면 길이를 더 편리하게 지정하고, 여러 옵션을 사용할 수도 있습니다. 필요한 경우 해당 패키지 안내서를 참조하십시오.

24.2

여러 단으로 문서 작성하기

경우에 따라서는 글줄이 두 개의 단을 이루는 형태로 문서를 작성하기도 합니다. \LaTeX 에서는 클래스 옵션 `onecolumn`과 `twocolumn`을 통해 1단 및 2단 편집을 지원합니다. 한편, 한 문서 안에서 단의 수를 변경할 수도 있습니다. 이는 `\onecolumn` 및 `\twocolumn` 제어 문자열을 사용하면 됩니다. `\twocolumn`을 입력하면 해당 위치에서 새로운 페이지가 시작되며, 그 페이지에서 2단 편집이 시작됩니다. 반대로, `\onecolumn`을 입력하면 해당 위치에서 새로운 페이지가 시작되고, 그 페이지에서 1단 편집이 시작됩니다.


```

1 \documentclass{article}
2 \usepackage[hangul]{kotex}
3 \begin{document}
4   \twocolumn
5   문서 클래스 옵션에 ... 조판됩니다. \par
6   문서를 2단으로 조판하면 ... 사용하십시오.
7 \end{document}

```

문서 클래스 옵션에 `twocolumn`이 없더라도, `\twocolumn` 제어 문자열을 사용하면 본문이 2단으로 조판됩니다.

문서를 2단으로 조판하면 글줄의 길이가

짧아지므로, 경우에 따라 조판 결과가 보기에 좋지 않을 수 있습니다. 꼭 필요한 경우에만 2단 조판 기능을 사용하십시오.

2단 편집 시에는 여러 인자를 통해 문서의 레이아웃을 변경할 수 있습니다. 먼저, 단 사이의 간격은 `\columnsep`에 저장되어 있습니다. 또, 각 단의 폭은 `\columnwidth`에 저장되어 있습니다. 따라서 두 값을 조정하면 그에 맞게 글줄의 길이가 조정됩니다. 한편, 단 사이에 구분선을 넣으려면 `\columnseprule`의 값을 조정하십시오. 이 제어 문자열은 구분선의 두께를 지정하며, 초깃값은 0 pt 입니다(즉 구분선은 기본적으로 출력되지 않습니다). 일반적으로 0.4 pt가 적합합니다.

```

1 \documentclass{article}
2 \usepackage[hangul]{kotex}
3 \setlength\columnsep{1cm}
4 \setlength\columnseprule{.4pt}
5 \begin{document}
6   \twocolumn
7   문서 클래스 옵션에 ... 조판됩니다. \par
8   문서를 2단으로 조판하면 ... 사용하십시오.
9 \end{document}

```

문서 클래스 옵션에 `twocolumn`이 없더라도, `\twocolumn` 제어 문자열을 사용하면 본문이 2단으로 조판됩니다.

문서를 2단으로 조

판하면 글줄의 길이가 짧아지므로, 경우에 따라 조판 결과가 보기에 좋지 않을 수 있습니다. 꼭 필요한 경우에만 2단 조판 기능을 사용하십시오.

한쪽 단을 완전히 채우지 않고 다음 단으로 넘어가 내용을 입력할 때에는 아래 예시처럼 `\newpage`를 사용합니다.

```

1 \twocolumn
2 이번 단은 첫 번째 단입니다. \verb|\newpage|를 입력해 단을 나눌 수 있습니다. \par
3 \newpage
4 이번 단은 두 번째 단입니다.

```

이번 단은 첫 번째 단입니다. `\newpage`를 입력해 단을 나눌 수 있습니다.

이번 단은 두 번째 단입니다.

한편, `\clearpage`를 사용하면 현재 단에 관계없이 새로운 페이지가 시작됩니다.

다단 편집 시의 유동 개체

다단 편집을 하는 경우, 표나 그림 등은 한 단 안에 넣기 힘든 경우도 있습니다. 이러한 경우에는 개체가 여러 단에 걸쳐 배치되도록 하는 것이 좋습니다. 제16장에서 알아보았던 `figure` 및 `table` 환경은 개체를 한 단 안에 조판합니다. 이때 이들의 별표 붙은 버전인 `figure*` 및 `table*` 환경을 사용하면 개체들이 전체 단에 걸쳐 배치됩니다. 이들 환경에도 똑같은 선택 인자를 지정함으로써 배치를 허용할 위치를 지정해 둘 수 있습니다.

```

1 \twocolumn
2 \begin{figure*}
3   \centering
4   \includegraphics{pictures/sample.eps}
5   \caption{Sample figure inserted using
6     \texttt{figure*} environment.}
7 \end{figure*}
8 \centering
9 \includegraphics{pictures/sample.eps}
10 \caption{Sample figure inserted using
11   \texttt{figure} environment.}
12 \end{figure}
13 \lipsum[1][1-7] % inserts dummy text

```



Figure 1: Sample figure inserted using `figure*` environment.



Figure 2: Sample figure inserted using `figure` environment.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae,

felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo.

이 환경 안에서 다음 단으로 넘어가려는 경우에는 `\newpage` 대신 `\newcolumn` 제어 문자열을 사용합니다.

multicol 패키지로 다단 만들기

한 페이지 안에서 문서의 다단의 수를 바꾸고자 한다면 `multicol` 패키지가 제공하는 `multicols` 환경을 사용할 수 있습니다. 이 환경을 사용하면 단의 수를 자유롭게 설정할 수도 있습니다. 만들려는 단의 수를 환경의 인자로 지정하면 됩니다.

```

1 \documentclass{article}
2 \usepackage{multicol}
3 \begin{document}
4 환경 밖에 내용을 작성하면 1단으로 본문이 조판됩니다.
5 \begin{multicols}{3}
6 이 예시에서는 \verb|multicols| 환경을 사용해 ...
7 \end{multicols}
8 환경 전후로 단 수는 바뀌지만 페이지가 나뉘지는 않습니
  다.
9 \end{document}

```

환경 밖에 내용을 작성하면 1단으로 본문이 조판됩니다.

이 예시 있는 문서를 단으로 만들
에 서 는 작성하였습 어졌음을 확
multicols 니다. 3을 인 인할 수 있습
환경을 사 자로 지정하 니다.
용해 다단이 여 문서가 3

환경 전후로 단 수는 바뀌지만 페이지가 나뉘지는 않습니다.

24.3

문단의 모양 바꾸기

여러 변수를 사용하면 문단의 모양도 바꿀 수 있습니다. 문단의 모양을 결정하는 데에는 첫 줄의 들여쓰기 간격을 조절하는 `\parindent`, 줄 간격을 조정하는 `\baselineskip`과 `\baselinestretch`, 문단과 문단 사이의 간격을 조정하는 `\parskip`이 영향을 미칩니다.

문단의 들여쓰기

모든 문단의 첫 번째 줄은 `\parindent`에 저장된 길이만큼 들여쓰기 처리가 됩니다. 단, `\chapter`나 `\section` 등으로 만든 장절 표제 다음의 첫 번째 문단은 첫 줄을 들여 쓰지 않습니다. 문서의 특정 문단을 들여 쓰도록 하거나 들여 쓰지 않도록 하려면 문단의 첫 머리에 `\indent`나 `\noindent`를 입력하면 됩니다. 모든 문단의 첫 줄을 들여 쓰지 않으려면 다음 코드를 입력하여 `\parindent`에 저장된 값을 0pt로 바꾸면 됩니다.

```
\setlength\parindent{0pt}
```

`\parindent`에 저장된 기본값은 글꼴 크기 10 pt 문서에서 15 pt, 글꼴 크기 11 pt 문서에서 17pt, 글꼴 크기 12 pt 문서에서 1.5 em입니다. 2단 편집을 하는 경우에는 기본 글꼴 크기에 관계없이 1em으로 설정됩니다.

줄 간격과 문단 사이의 간격

글줄과 글줄 사이의 간격은 `\baselineskip`에 저장된 값만큼 벌어집니다. 글꼴의 크기에 따라 줄 간격이 다르게 설정되며, 한 문단 안에서 글꼴이 여러 번 바뀐 경우 맨 마지막에 사용된 글꼴의 크기에 맞춰 줄 간격이 결정됩니다. `\baselinestretch`에는 줄 간격에 곱해 줄 배수가 저장되어 있습니다. 기본값은 1이며, kotex 패키지를 `hangul` 또는 `hanja` 옵션과 함께 불러오면 이 값이 1.3333으로 바뀝니다. 이는 아래아한글에서 줄 간격을 160%로 설정한 것과 같은 효과를 냅니다.

줄 간격을 바꿀 때에는 이들 제어 문자열에 저장된 값을 바꾸기보다는, `\linespread` 제어 문자열을 사용하는 것이 좋습니다. `\baselinestretch`에 지정하고 싶은 값을 `\linespread`의 인자로

지정하면니다. 예를 들어, `\linespread{1.3}`은 줄 간격을 글꼴 크기의 1.5배, `\linespread{1.6}`은 2배로 설정해 줍니다.

`\linespread`를 전처리부에 입력하였다면 문서 전반에 걸쳐 효과가 나타나며, 문서의 일부분에만 영향을 주기 위해 document 환경 안에서 이 제어 문자열을 사용한 경우에는 뒤에 `\selectfont`를 덧붙여 주어야 합니다. `\selectfont`가 없다면 효과가 올바르게 나타나지 않을 수 있기 때문입니다.

한편, 문단과 문단 사이의 간격은 `\parskip` 제어 문자열로 조정합니다. 여기에는 유동적 길이가 쓰일 수 있으며, 기본값은 0 pt plus 1 pt입니다.

24.4

오버풀과 언더풀

TeX을 사용해 문서를 작성한 후 컴파일 과정을 거치면 다음 경고 메시지를 자주 볼 것입니다.

```
Overfull \hbox ...
Overfull \vbox ...
Underfull \hbox ...
Underfull \vbox ...
```

TeX이 문서를 작성할 때 줄 바꿈이나 페이지 나눔을 실패한 경우 이러한 메시지가 출력됩니다. ‘Overfull’로 시작하는 메시지는 본문이 여백까지 침범해 조판되었음을 뜻하며, ‘Underfull’로 시작하는 메시지는 단어와 단어 사이, 글줄과 글줄 사이 등이 지나치게 벌어졌음을 뜻합니다. 두 상황을 각각 ‘오버풀’과 ‘언더풀’이라고 부릅니다. 이 경고 메시지가 발생한 경우에는 해당 부분의 내용을 조금씩 수정하여 경고가 사라지도록 해야 합니다.

| | |
|---|---|
| <p>본문의 내용을 작성할 때 <code>\verb</code>를 남용하면 오버풀이나 언더풀이 발생할 가능성이 높아진다. <code>\verb</code>를 사용해 입력한 내용의 중간에서는 줄바꿈이 일어날 수 없기 때문이다…….</p> | <p>본문의 내용을 작성할 때 <code>\verb</code>를 남용하면 오버풀이나 언더풀이 발생할 가능성이 높아진다. <code>\verb</code>를 사용해 입력한 내용의 중간에서는 줄바꿈이 일어날 수 없기 때문이다…….</p> |
|---|---|

(a) 오버풀의 예시

(b) 언더풀의 예시

그림 24.2: 오버풀과 언더풀의 예시

그림 24.2에 오버풀과 언더풀의 예시를 나타냈습니다. 본래의 글줄의 길이를 위쪽에 가로선으로 나타냈습니다. 오버풀이 발생한 왼쪽 그림에서는 첫째 줄과 둘째 줄에서 글줄의 길이가 넘어갔음을 확인할 수 있고, 언더풀이 발생한 오른쪽 그림에서는 글줄의 길이는 맞춰졌지만 단어 사이의 간격이 지나치게 넓어졌음을 확인할 수 있습니다.

제 25 장

공백과 구분선

이번 장에서는 다양한 공백과 구분선을 삽입하는 방법을 설명하겠습니다. 필요에 따라 이들을 적절히 사용하면 문서를 읽는 데 시각적인 도움을 줄 수 있습니다.

25.1

공백 삽입하기

TeX 문서에서 공백은 별도의 명령어를 사용해 입력해야 합니다. 단어와 단어 사이에서 스페이스 키를 여러 번 누르고 컴파일을 진행하더라도 그 사이의 공백은 스페이스 바를 한 번 누른 것과 동일하게 조판되며, 문단과 문단 사이에 빈 줄을 여러 번 입력하더라도 문단 사이의 간격이 넓어지지 않습니다. 따라서, 문자와 문자 사이의 간격을 넓힐 때에는 가로 방향 공백을, 글줄과 글줄, 또는 문단과 문단 사이의 간격을 넓힐 때에는 세로 방향 공백을 삽입하는 명령어를 사용해야 합니다.

가로 방향 공백

가로 방향의 공백을 입력할 때에는 `\enspace`, `\quad`, `\qquad`를 사용할 수 있습니다. 각각 0.5 em, 1 em, 2 em의 간격을 입력합니다.

- 1 0.5\,em의\enspace 공백 \
- 2 1\,em의\quad 공백 \
- 3 2\,em의\qquad 공백

0.5 em의 공백
1 em의 공백
2 em의 공백

`\hspace`를 사용하면 공백의 크기를 임의의 값으로 설정할 수 있습니다. 유동적 길이를 인자로 입력하면 되며, 제어 문자열이 입력된 곳에 해당 크기의 공백을 삽입합니다. 만약 `\hspace`가 입력된 위치에서 줄이 바뀌거나 시작된다면 이 공백은 무시됩니다. 별표 붙은 `\hspace*`를 사용하면 이러한 경우에도 공백이 무시되지 않습니다.

정답은 (`\hspace{12mm}`)이다.

정답은 ()이다.

하나의 글줄의 왼쪽 끝과 오른쪽 끝에 내용을 각각 배치하려면 `\hfil`이나 `\hfill`을 사용할 수 있습니다. 이들은 길이가 무한대인 가로 방향 공백을 삽입하므로, `\hfil`이나 `\hfill`의 좌우에 입력한 내용이 본문 조판 영역 양쪽 끝에 조판되는 효과를 얻습니다.¹

왼쪽 끝 `\hfill` 오른쪽 끝

왼쪽 끝

오른쪽 끝

세로 방향 공백

글줄과 글줄 사이, 문단과 문단 사이에 간격을 넣을 때에는 `\smallskip`이나 `\medskip` 또는 `\bigskip`을 사용할 수 있습니다. `\smallskip`은 3 pt plus 1 pt minus 1 pt의 공백을 삽입하며, `\medskip`과 `\bigskip`은 각각 `\smallskip`의 2배, 4배의 공백을 넣습니다. 문단의 중간에 이 제어 문자열이 입력되면 해당 제어 문자열이 있는 글줄과 그 위의 글줄 사이에 간격이 삽입됩니다.

- ¹ 필요한 경우에는 문단과 문단 사이에 추가 간격을 주어 두 내용이 구분됨을 시각적으로 보여 주는 것이 좋다. `\par`
- ² `\medskip`
- ³ 하지만, 불필요한 경우에 공백을 남발하는 것은 좋지 않다.

필요한 경우에는 문단과 문단 사이에 추가 간격을 주어 두 내용이 구분됨을 시각적으로 보여 주는 것이 좋다.

하지만, 불필요한 경우에 공백을 남발하는 것은 좋지 않다.

가로 방향 공백과 마찬가지로, 세로 방향 공백도 그 크기를 직접 지정할 수 있습니다. 유동적 길이를 `\vspace`의 인자로 지정하면 됩니다. 만약 `\vspace`가 입력된 곳에서 페이지가 바뀐다면 TeX 은 이 공백을 무시합니다. 이때 공백이 무시되지 않도록 하려면 별표 붙은 `\vspace*`를 사용하십시오. `\hfil`, `\hfill`과 마찬가지로 `\vfil`, `\vfill`도 사용할 수 있습니다.

수식에서의 공백

제10장에서 설명하였듯, 표 10.1의 제어 문자열을 사용하면 수식 입력 모드에서 공백을 입력할 수 있습니다. 이들은 작은 크기의 공백을 입력하므로, 수식에서 기호와 기호 사이의 간격을 미세하게 조정하는 데 사용됩니다.

수식 입력 모드에서는 앞서 설명한 `\hspace`와 `\vspace`를 사용해서 공백을 입력할 수도 있습니다. 또, `amsmath` 패키지를 불러오면 `\mspace`라는 제어 문자열을 사용할 수 있으며, `\mu` 단위를 사용한 미세한 공백을 입력할 수 있습니다(`\mspace`를 사용할 때에는 `\mu` 단위만을 사용해야 합니다). 예를 들어, `\medspace`는 `\mspace{4\mu plus 2\mu minus 2\mu}`와 같은 동작을 합니다.

¹`\hfill`의 우선 순위가 `\hfil`보다 높으므로, `\hfil`이 원하는 결과를 만들지 않는다면 `\hfill`을 사용하십시오.

[2]와 [3]은 \,와 \!를 사용해 수식 기호 사이의 간격을 수동으로 지정해야 하는 경우를 몇 가지 소개하고 있습니다. 아래의 예시를 확인하십시오.

| | | |
|-------------------------|----------------------------------|-------------------------------------|
| 적분식에서: | $\int_0^x \int_0^y f(u,v) du dv$ | 대신 $\int_0^x \int_0^y f(u,v) du dv$ |
| 근호 뒤에서: | $\sqrt{2}x$ | 대신 $\sqrt{2}x$ |
| 분수를 빗금을 사용해 적을 때: | $1/\ln x$ | 대신 $1/\ln x$ |
| 팩토리얼 뒤에 숫자나 글자가 올 때: | $\frac{10!}{6!4!}$ | 대신 $\frac{10!}{6!4!}$ |
| 글자와 첨자 사이에 넓은 간격이 생길 때: | Γ_2, Δ^2 | 대신 Γ_2, Δ^2 |
| 단위를 적을 때: | $10\text{cm} = 25.4\text{in}$ | 대신 $10\text{ cm} = 25.4\text{ in}$ |

여기에 나열되지 않은 경우에도, 필요에 따라 공백을 적절히 사용하십시오.

25.2

구분선 넣기

구분선은 \rule 제어 문자열을 사용해 입력할 수 있습니다.

`\rule[⟨raise⟩]{⟨width⟩}{⟨height⟩}`

⟨raise⟩: 구분선의 상하 위치

⟨width⟩: 구분선의 폭(가로 방향 길이)

⟨height⟩: 구분선의 높이(세로 방향 길이)

구분선은 기본적으로 베이스라인(로마자 대문자가 정렬되는 아래쪽 기준선)에 그려지며, 선택 인자인 ⟨raise⟩에 양수의 길이를 지정하면 위쪽으로, 음수의 길이를 지정하면 아래쪽으로 이동합니다. 세 인자에는 모두 고정된 길이를 지정합니다.

예를 들어, 문서에

이름: _____

와 같이 이름을 적는 난을 만들려면 `\rule[-2pt]{1in}{0.4pt}`와 같이 입력하면 됩니다. 여기에서 사용한 0.4 pt는 구분선의 두께로 자주 사용됩니다.

예제 25.1 세로로 글줄의 가운데에 구분선이 오도록 _____ 적절한 \LaTeX 코드를 작성하여라. (힌트: 베이스라인에서 x-height의 절반 정도만큼 구분선을 올리면 적당하다.)

예제 25.2 `\rule`을 잘 사용하면 임의의 사각형을 생성할 수 있다. 폭 0.5 cm, 높이 4 pt인 직사각형을 그리기 위한 \LaTeX 코드를 작성해 보자. 오른쪽에 그려진 사각형 정도의 크기이다. ■

제 26 장

다른 언어로 문서 작성하기

LaTeX으로 다른 언어의 문서를 작성할 때에는 문서의 내용을 해당 언어로 입력하는 것 외에도, 몇 가지 추가 설정을 해 주어야 합니다. 예를 들어, 해당 언어의 문자들을 올바르게 표시할 수 있도록 글꼴을 지정하고, ‘차례’나 ‘찾아보기’ 등의 상용구도 해당 언어에 맞게 바꾸어야 하며, 날짜 표시 형식이나 사소한 조판 관행 등도 해당 언어의 문서에 알맞게 변경해 주어야 합니다. 이번 장에서는 이런 내용을 다룹니다.

컴퓨터에서 언어를 처리하는 방식의 특성에 따라, 내용을 크게 두 개의 절로 구분하였습니다. 첫 번째 절은 로마자, 키릴 문자, 그리스 문자를 사용하는 언어를 처리하는 방법을 설명합니다. 두 번째 절은 한중일 문자를 처리하는 방법을 다룹니다. 이외의 언어(다른 문자 체계를 사용하거나 오른쪽에서 왼쪽으로 쓰는 언어 등)에 관해서는 다루지 않습니다.

26.1

로마자, 키릴 문자, 그리스 문자를 사용하는 언어

이들 언어로 문서를 작성할 때에는 babel 패키지 또는 polyglossia 패키지를 불러온 뒤, 문서에서 사용할 언어를 지정해 주면 됩니다(polyglossia 패키지는 XeTeX이나 LuaLaTeX 엔진에서만 사용할 수 있습니다).

영어가 아닌 언어로 문서를 작성할 때에는 (필요한 경우 fontenc 패키지를 불러와) 단순히 문자를 출력할 수 있도록 하는 것이 아니라, 그 언어에 맞는 설정을 갖춰 주어야 합니다. ‘로마자’라는 문자 체계를 공유하더라도 언어에 따라 문장부호, 줄 바꿈 패턴, 날짜 표시 형식, 사소해 보일 수 있는 조판 관행 등 여러 부분에서 차이가 있기 때문입니다.

babel 패키지를 불러오면 언어 설정을 할 수 있습니다. 예를 들어, 프랑스어 문서를 작성한다면 전처리부에

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

을 입력하면 됩니다. 한 문서에서 여러 언어를 사용할 때에도 이에 맞는 설정을 해 줄 수 있습니다. 옵션으로 지정할 수 있는 언어 및 방언을 비롯하여 더 자세한 내용은 babel 패키지의 안내서를 읽어 보시기 바랍니다.

한편, Xe_{La}TeX과 Lua_{La}TeX에서는 같은 기능을 polyglossia라는 다른 패키지를 통해 사용할 수 있습니다. 이때에는 조금 다른 명령어가 사용됩니다.

```
1 \usepackage{polyglossia}
2 \setdefaultlanguage{french}
```

26.2

한중일 언어

한국어, 중국어, 일본어를 \LaTeX 에서 처리하는 것은 유럽 언어에 비해 복잡합니다. 많은 유럽 언어는 많아야 200개 정도의 문자를 사용하지만, 한글과 한자는 문자 수가 매우 많아 일상 언어에서 사용하는 문자는 2,000자를 거뜬히 넘기 때문에입니다. 한국, 중국, 일본은 자국의 언어로 \LaTeX 을 사용하기 위한 여러 방법을 개발하였습니다.

한국어 문서

\LaTeX 소스 파일에 특별한 설정을 하지 않고 한글을 입력한 뒤 컴파일을 시도하면 한글이 출력되지 않습니다. 이는 \LaTeX 기본 글꼴에 한글이 없기 때문입니다. 레거시 엔진에서는 오류를 띄우기까지 합니다.

한글 입력은 간단히 kotex 패키지를 불러옴으로써 문제가 해결됩니다. 즉, 전처리부에

```
\usepackage{kotex}
```

를 입력하면 됩니다. 레거시 엔진을 사용한다면 한글 글꼴로 나눔명조와 나눔고딕이 지정되며, 유니코드 엔진을 사용한다면 은 명조와 은 돋움¹이 지정됩니다.¹

다음은 영어 문서의 일부에서 kotex 패키지를 사용해 한글을 출력한 예시입니다.

```
1 \documentclass{article}
2 \usepackage{kotex}
3 \begin{document}
4 This is a sample English document with
  some Hangul (한글) characters.
5 \end{document}
```

This is a sample English document with some Hangul (한글) characters.

kotex 패키지를 불러오는 많은 경우에는 한국어 문서를 작성하고자 할 것입니다. 이때에는 패키지 옵션으로 hangul을 지정하여, 한국어 문서 작성에 알맞은 설정을 해 주어야 합니다.

```
\usepackage[hangul]{kotex}
```

전처리부에 위 코드를 입력하고 한국어 문서를 작성하면 ‘Contents’, ‘Index’, ‘Bibliography’ 등의 상용구가 ‘차례’, ‘찾아보기’, ‘참고 문헌’ 등으로 바뀌고, 줄 간격도 한국어 글꼴에 어울리도록 조정됩니다. hangul 대신 hanja를 입력할 수도 있으며, 이 경우 상용구가 한글이 아닌 한자로 조판됩니다.

¹유니코드 엔진에서 한글 글꼴을 변경하는 방법은 제27장에서 설명합니다.

kotex 패키지를 사용해 한국어 문서를 작성한 예시는 제2장에 제시하였으므로, 여기에서는 제시하지 않고 넘어가겠습니다.

한편, 앞선 4장에서 설명한 oblivoir 클래스를 사용한다면 한국어 처리를 위한 별도의 패키지를 불러오지 않고도 형식이 잘 갖춰진 한국어 문서를 작성할 수 있습니다. oblivoir 클래스는 memoir 클래스와 kotex 패키지에 의존하므로, 자세한 사용법은 oblivoir 클래스 안내서, kotex 패키지 안내서, memoir 클래스 안내서를 읽어 보시기 바랍니다.

중국어 문서

중국어 문서를 작성할 때에는 ctex 패키지를 사용합니다. 이 패키지를 불러오면 엔진에 관계없이 한자를 문제없이 입력하고 조판할 수 있습니다. 또는, article, report, book 클래스를 ctexart, ctexrep, ctexbook 클래스로 바꾸어 문서를 작성할 수도 있습니다. 즉, 아래의 두 코드는 같은 결과를 생성합니다.

```
1 \documentclass[a4paper]{article}
2 \usepackage[UTF8]{ctex}
3 \begin{document}
4 中文\LaTeX 文件。
5 \end{document}
```

```
1 \documentclass[a4paper,UTF8]{ctexart}
2 \begin{document}
3 中文\LaTeX 文件。
4 \end{document}
```

한편, 다른 언어의 문서 일부에서만 중국어를 출력하고 싶다면 CJKutf8 패키지 또는 xeCJK 패키지를 사용하면 됩니다. PDF_{La}T_EX에서 CJKutf8 패키지를 불러온 경우, 중국어 부분은 CJK* 환경 안에 작성해야 합니다. 각 엔진에서의 예시는 다음과 같으며, 더 자세한 내용은 패키지 안내서를 참조하시기 바랍니다.

```
1 \documentclass{article} % LaTeX,
   PDFLaTeX
2 \usepackage{CJKutf8}
3 \begin{document}
4 \begin{CJK*}{UTF8}{gbsn}
5 中文\LaTeX 文件。
6 \end{CJK*}
7 \end{document}
```

```
1 \documentclass{article} % XeLaTeX
2 \usepackage{xeCJK}
3 \begin{document}
4 中文\LaTeX 文件。
5 \end{document}
```

일본어 문서

한자와 세로쓰기 등의 이유로, 일본어 문서를 작성할 때에는 p_{La}T_EX과 up_{La}T_EX이라는 다른 엔진을 사용해 컴파일합니다. 클래스도 article, report, book 대신 일본어 문서 작성에 적합한 jsarticle, jsreport, jsbook을 사용합니다. 또, 많은 패키지가 p_{La}T_EX과 up_{La}T_EX과 호환되지 않아서, 일본어 엔진용 패키지를 별도로 사용해야 합니다.

컴파일 과정의 번거로움만 감수하면, 오늘날에는 많은 도구가 개발되어 일본어 문서를 더 쉽게 작성할 수 있습니다. 먼저 문서의 클래스는 `jlreq`를 사용합니다. `\documentclass` 제어 문자열의 인자로 `jlreq`를 입력한 뒤, 옵션에 실제로 원하는 클래스(`article`, `report`, `book`)를 입력하면 됩니다. 또, 패키지 문제는 `plautopatch`가 해결해 줍니다. 다른 엔진에서 사용하기 위한 패키지를 불러왔을 때 자동으로 일본어용 패키지로 대체해 줍니다. 일반적인 일본어 문서의 소스 코드는 다음과 같이 작성합니다.

```
1 \RequirePackage{plautopatch}
2 \documentclass[article,dvipdfmx]{jlreq}
3 \begin{document}
4   これは日本語\LaTeX 文書です。
5 \end{document}
```

これは日本語 \LaTeX 文書です。

이를 저장한 후, `up \LaTeX` 으로 컴파일하여 DVI 파일을 얻은 뒤, `dvipdfmx`를 실행해 이 DVI 파일을 PDF 파일로 변환하면 됩니다. 세로쓰기 문서를 작성하려면 클래스 옵션에 `tate`를 추가하면 됩니다.

한편, 오늘날에는 `Lua \LaTeX` 을 사용하기도 합니다. 많은 글꼴을 자유롭게 사용할 수 있고, 위의 두 엔진에서는 신경써야 하는 드라이버 및 패키지 충돌 문제에서도 자유롭기 때문입니다. 컴파일 하면 DVI 파일을 거치지 않고 PDF 파일을 바로 얻을 수도 있습니다. `Lua \LaTeX` 을 사용하여 일본어 문서를 작성할 때에도 마찬가지로 `jlreq` 클래스를 사용하며, 이때에는 `plautopatch` 패키지는 불러오지 않아도 됩니다. `Lua \LaTeX` 을 사용한 일본어 문서의 소스 코드는 다음과 같으며, 위와 같은 결과를 출력합니다.

```
1 \documentclass[article]{jlreq}
2 \begin{document}
3   これは日本語\LaTeX 文書です。
4 \end{document}
```

これは日本語 \LaTeX 文書です。

마찬가지로, 클래스 옵션으로 `tate`를 입력하면 세로쓰기 문서가 만들어집니다.

다른 언어의 문서 일부에서만 일본어를 출력하고 싶다면 `Lua \LaTeX` 엔진을 사용하고, `luatexja` 패키지를 불러오면 됩니다. 일본어 문서 작성에 관한 더 자세한 내용은 <https://ctan.math.illinois.edu/macros/latex/contrib/babel-contrib/japanese/japanese.pdf>를 참조하십시오.

제 27 장

글꼴 바꾸기

이번 장에서는 여러 패키지를 사용해 문서의 글꼴, 엄밀히는 타입페이스(typeface)를 바꾸는 방법을 알아보겠습니다.

\TeX 에서는 우리가 다른 프로그램에서 글꼴을 사용하는 것과는 다른 방식으로 글꼴을 처리합니다. 우리가 흔히 사용하는 TrueType 또는 OpenType 글꼴을 바로 사용할 수 없으며, \TeX 에서 사용할 수 있는 별도의 형식으로 제작된 글꼴만을 사용할 수 있습니다. CTAN에는 \TeX 에서 사용할 수 있는 여러 글꼴이 등록되어 있으며, 이들은 적절한 패키지를 불러와 사용할 수 있습니다.

하지만, \TeX 이 글꼴을 처리하는 방식의 문제점 때문에, CTAN의 많은 글꼴은 로마자만을 지원합니다. 특히, 동아시아의 문자들(한글, 한자, 가나 등)을 지원하는 글꼴은 얼마 되지 않습니다. 따라서, 문서를 작성할 때 CTAN에 등록되어 있지 않은 다양한 TrueType 및 OpenType 글꼴을 사용하려면 해당 기능을 지원하는 \XeTeX 이나 \LuaTeX 을 사용하는 것이 좋습니다.

27.1

글꼴을 지정하는 패키지로 글꼴 변경하기

\TeX 의 기본 로마자 글꼴은 도널드 커누스 박사가 개발한 ‘Computer Modern’입니다. 기본 글꼴이기에 많은 \LaTeX 문서는 이 글꼴을 사용하여 만들어집니다, 한편, 출판사의 편집을 거쳐 시중에 판매되는 많은 책에서는 Times나 Palatino 등의 글꼴을 사용합니다. 각 글꼴의 모습을 그림 27.1에서 확인해 보십시오.

Times는 `newtxtext` 패키지를, Palatino는 `newpxtext` 패키지를 불러와 사용할 수 있습니다. `newtx-math` 패키지나 `newpxmath` 패키지를 불러오면 수식 글꼴까지 본문 글꼴에 일치하도록 할 수 있습니다. 즉,

```
\usepackage{newtxtext,newtxmath}
```

를 전처리부에 입력하면 Times를 기본 글꼴로 하여 문서를 조판할 수 있습니다.

기본 산세리프 글꼴은 Computer Modern Sans Serif입니다. `helvet` 패키지를 사용하면 널리 사

Computer Modern: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris.

Times: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris.

Palatino: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris.

그림 27.1: \TeX 에서 자주 사용되는 로마자 세리프 글꼴

용되는 산세리프 글꼴인 Helvetica를 사용할 수 있습니다.¹ 글꼴의 자형을 그림 27.2에서 확인해보십시오.

Computer Modern Sans Serif: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna.

Helvetica: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna.

그림 27.2: \TeX 에서 자주 사용되는 로마자 산세리프 글꼴

예제 27.1 Times와 Palatino 중 원하는 글꼴을 기본 세리프 글꼴로, Helvetica를 기본 산세리프 글꼴로 지정하여 간단한 \LaTeX 문서를 조판해 보자. 패키지를 올바르게 불러왔다면 `\sffamily`나 `\textsf`를 사용해 Helvetica를 사용할 수 있을 것이다.

예제 27.2 세리프 글꼴로 New Century Schoolbook을 사용할 수 있도록 하는 `fouriernc` 패키지, 산세리프 글꼴로 Roboto를 사용할 수 있도록 하는 `roboto` 패키지를 불러와 간단한 문서를 작성해 보자.

이외에도, \TeX 에서는 수많은 로마자 글꼴을 사용할 수 있습니다. TUG의 Font Catalogue 페이지 (tug.org/FontCatalogue)에 접속하면 여러 글꼴을 볼 수 있습니다. 세리프, 산세리프 글꼴, 타자기(고정폭) 글꼴, 수식에서 사용할 수 있는 글꼴 등으로 구분되어 있으며, \TeX 에서 사용할 수 있는 다양한 글꼴과 그 글꼴을 사용하는 방법을 확인할 수 있습니다.

27.2

fontspec 패키지로 글꼴 변경하기

\XeTeX 과 \LuaTeX 엔진을 사용하면 앞서 설명하였듯 TrueType 및 OpenType 글꼴을 자유롭게 사용할 수 있습니다. 이때에는 `fontspec` 패키지가 필요합니다.

¹Helvetica는 같은 크기의 다른 글꼴에 비해 크게 보인다. 이때에는 패키지를 불러올 때 `scaled` 옵션을 지정해 주면 다른 글꼴과 시각적 크기가 비슷해지도록 할 수 있다.

패키지를 불러온 뒤에는 여러 제어 문자열을 사용해 문서에서 사용할 글꼴을 지정할 수 있습니다. 전처리부에서 `\setmainfont`, `\setsansfont`, `\setmonofont`를 사용해 문서의 세리프, 산세리프, 고정폭 글꼴을 지정합니다.

```
\setmainfont[<options>]{<font name>}
```

<options>: 글꼴에 지정할 옵션

**: 지정할 글꼴의 이름

`\setsansfont`, `\setmonofont`도 같은 방법으로 사용한다.

예를 들어, 문서에서 로마자 글꼴로 세리프체는 Times New Roman, 산세리프체는 Helvetica, 고정폭 서체는 Courier New를 사용하려 한다면, 전처리부에 아래와 같이 입력합니다.

```
1 \usepackage{fontspec}
2 \setmainfont{Times New Roman}
3 \setsansfont{Helvetica}
4 \setmonofont{Courier}
```

예제 27.3 로마자만을 사용하는 문서에서 세리프체 글꼴은 TeX Gyre Schola, 산세리프체 글꼴은 Roboto를 사용하고, 고정폭 글꼴은 T_EX 기본 글꼴을 사용하려 한다. fontspec 패키지를 불러온 후, 글꼴을 올바르게 사용하기 위한 코드를 작성해 보자.

선택 인자인 *<options>*에는 글꼴을 불러올 때의 다양한 옵션을 *<key>=<value>*의 형식으로 입력합니다. 글꼴에서 제공하는 다양한 OpenType 기능을 옵션을 지정함으로써 설정할 수 있습니다. 지정할 수 있는 옵션들은 타이포그래피와 관련된 전문적인 내용이므로 이 책에서는 그 내용을 다루지는 않겠습니다. 옵션을 지정하는 방법은 fontspec 패키지 안내서를 참조하십시오.

볼드체와 이탤릭체 지정하기

8.2절에서 설명했듯, L^AT_EX은 글꼴의 볼드체와 이탤릭체를 지정할 수 있습니다. 많은 글꼴은 특별한 설정 없이 볼드체와 이탤릭체가 자동으로 지정되며, `\bfseries`나 `\textbf`, `\itshape`나 `\textit`를 사용해 볼드체와 이탤릭체를 사용할 수 있습니다.

만약 글꼴의 볼드체와 이탤릭체가 제대로 표시되지 않는다면 이들 글꼴을 직접 지정해 주어야 합니다. `\set...font`의 선택 인자에 `UprightFont`(정체), `BoldFont`(볼드체 정체), `ItalicFont`(이탤릭체), `BoldItalicFont`(볼드 이탤릭체)를 키로, 각 글꼴 이름을 값으로 하여 내용을 입력해 주면 됩니다. 예를 들어, Times New Roman의 볼드체와 이탤릭체를 직접 지정하려면 다음과 같이 입력합니다. 이때 별표 *는 **에 입력된 내용으로 대체됩니다. `UprightFont`는 따로 지정하지 않으면 **에 입력된 값이 사용됩니다.

```
\setmainfont[ItalicFont = * Italic, BoldFont = * Bold, BoldItalicFont = *
Bold Italic]{Times New Roman}
```

27.3

한글·한자 글꼴 지정하기

X_YTeX이나 Lua_YTeX에서 kotex 패키지를 불러왔다면 fontspec 패키지와 같은 방법으로 글꼴을 지정합니다. kotex 패키지 내부에서 fontspec 패키지를 불러오므로, 따로 fontspec 패키지를 불러 오지 않아도 됩니다. 글꼴을 지정할 때에는 로마자 등 유럽 문자에 사용할 글꼴과 한글, 한자에 사용할 글꼴을 구분해 지정해야 합니다. 로마자의 글꼴은 fontspec 패키지를 불러왔을 때와 똑같이 지정하며, 한글과 한자 글꼴을 지정할 때에는 다음 제어 문자열을 `\set...font`와 같은 방법으로 사용하면 됩니다(문서에서 한자를 사용할 일이 없다면 한자 글꼴은 지정해 주지 않아도 됩니다).

```
\setmainhangulfont \setsanshangulfont \setmonohangulfont
\setmainhanjafont \setsanshangulfont \setmonohanjafont
```

앞서 로마자 글꼴을 지정한 예시에 추가하여, 한글 글꼴로 명조체는 Noto Serif CJK KR, 고딕체는 Noto Sans CJK KR, 고정폭은 Noto Sans Mono CJK KR를 사용하려 한다면 전처리부에 다음과 같이 입력합니다.

```
1 \usepackage[hangul]{kotex}
2 \setmainfont{Times New Roman}
3 \setsansfont{Helvetica}
4 \setmonofont{Courier}
5 \setmainhangulfont{Noto Serif CJK KR}
6 \setsanshangulfont{Noto Sans CJK KR}
7 \setmonohangulfont{Noto Sans Mono CJK KR}
8 \hanjabyhangulfont
```

마지막에 입력한 `\hanjabyhangulfont`는 한자를 식자할 때에도 한글 글꼴을 사용하도록 함을 뜻합니다.

예제 27.4 로마자와 한글 모두 세리프체·산세리프체·고정폭 글꼴로 각각 나눔명조, 나눔바른고딕, 나눔고딕코딩을 사용하도록 코드를 작성해 보자. (힌트: 로마자와 한글에 같은 글꼴을 사용할 때에는 kotex 패키지를 불러온 후 로마자·한글 글꼴 구분 없이 `\setmainfont`, `\setsansfont`, `\setmonofont`로 글꼴을 지정하면 된다.)

예제 27.5 로마자 세리프체 글꼴은 Source Serif Pro, 산세리프체 글꼴은 Source Sans Pro, 고정폭 글꼴은 Source Code Pro, 한글 및 한자 세리프체 글꼴은 Source Han Serif KR, 산세리프체 글꼴은 Source Han Sans KR, 고정폭 글꼴은 Source Han Mono KR를 사용하려 한다. kotex 패키지를 불러온 후, 글꼴을 올바르게 사용하기 위한 코드를 작성해 보자.

한글 글꼴의 볼드체와 이탤릭체 지정하기

로마자와 마찬가지로, 한글 글꼴을 지정했을 때 대응하는 이탤릭체와 볼드체 글꼴이 있다면 자동으로 인식되며, 그렇지 않은 경우에는 선택 인자를 지정하는 방식으로 직접 지정해 줄 수 있습니다. 예를 들어, 다음과 같이 사용합니다.


```

1 \usepackage[hangul]{kotex}
2 \setmainhangulfont[BoldFont = * Bold]{나눔명조}
3 \setsanshangulfont[BoldFont = * Bold]{나눔고딕}

```

한편, 전통적으로 한글 글꼴에는 이탤릭체에 대응하는 개념이 없으므로, 이에 주의해야 합니다. 일반적으로는 정체 글꼴을 강제로 기울여 표시하도록 하거나 다른 글꼴을 이탤릭체 대신 사용합니다. 경우에 따라서는 볼드체도 없지만, 볼드체는 강제로 글꼴을 볼드로 만들어 주는 기능을 사용하면 됩니다.

먼저, 전자의 방법, 즉 글꼴을 강제로 기울여 이탤릭체를 표시하려는 경우에는 글꼴 지정 제어 문자열의 `<options>` 부분에 `AutoFakeSlant`와 `AutoFakeBold`를 입력하면 됩니다. 예를 들어, 아래와 같이 사용합니다.

```

\setmainhangulfont
[AutoFakeSlant, AutoFakeBold]{Noto Serif CJK KR}

```

한편, 후자의 방법, 즉 별도의 글꼴을 지정하려는 경우에는 앞서 설명했던 대로, 글꼴 지정 제어 문자열의 선택 인자의 `ItalicFont`와 `BoldItalicFont`의 값에 원하는 글꼴을 지정하면 됩니다.

```

\setmainhangulfont
[BoldFont = KoPubWorld바탕체 Bold,
ItalicFont = 서울한강체 M, BoldItalicFont = 서울한강체 EB]%
{KoPubWorld바탕체 Light}

```

볼드체가 없는 글꼴에서는 `AutoFakeBold`를 옵션으로 지정하면 됩니다. 원래 글꼴의 모습을 바탕으로 볼드체를 비슷하게 만들어 줍니다.

예제 27.6 앞의 예제 26.5에서, 한글 글꼴의 이탤릭체는 강제로 기울인 꼴로 나타내도록 코드를 수정하자.

예제 27.7 앞의 예제 26.4에서, 나눔명조의 이탤릭체와 이탤릭 볼드체를 각각 나눔바른펜과 나눔바른펜 Bold로, 나눔바른고딕의 이탤릭체와 이탤릭 볼드체를 각각 나눔스퀘어와 나눔스퀘어 Bold로 조판하도록 코드를 수정하자.

`` 부분에는 글꼴의 이름뿐 아니라 글꼴 파일의 이름을 사용해서도 글꼴을 지정할 수 있습니다. `<options>` 부분에는 앞에서 알아본 옵션 외에도 다양한 옵션을 지정할 수 있습니다. 글꼴이 특수한 OpenType 기능을 지원하는 경우에는 그 기능을 사용하도록 할 수 있으며, 다양한 미세 조정을 할 수도 있습니다. 이에 대해서는 `fontspec` 패키지 문서와 `xetex-ko` 패키지 문서를 참조하십시오.

한편, 앞에서 알아본 `\set...font` 제어 문자열 외에도 글꼴을 지정할 수 있도록 하는 다양한 제어 문자열이 있습니다. 대표적으로 임시로 다른 글꼴을 사용하도록 하는 `\fontspec`, `\hangulfontspec`, `\hanjafontspec` 제어 문자열이 있습니다. 이 제어 문자열을 사용하는 방법이나, 이외에 사용할 수 있는 다른 제어 문자열에 대해서도 각 패키지 문서를 참조하시기 바랍니다.

제 28 장

색 사용하기

이번 장에서는 \LaTeX 에서 색을 정의하고, 색을 사용하는 방법을 알아봅니다. \TeX 은 다양한 색상을 사용하는 기능을 기본 기능으로 제공하지 않으므로, 색상을 사용하려면 `xcolor` 패키지를 불러와야 합니다. 이 패키지를 사용하여 문서에서 여러 가지 색상을 사용해 봅시다.

28.1

색상 지정하기

색상을 사용할 때에는 수를 사용해 해당 색의 정보를 정확하게 나타내야 합니다. 색상 정보를 나타내는 방법은 여러 가지가 있으며, 색상을 사용하는 목적(크게 인쇄용과 화면용으로 구분)과, 색을 만들어 내는 요소에 따라 구분합니다. 이렇게 구분한 각각의 방법을 ‘색상 모델’이라고 부릅니다. `xcolor` 패키지에서는 아래의 색상 모델을 지원합니다.

- RGB 빨강(Red), 초록(Green), 파랑(Blue) 3색의 강도로 색을 지정합니다. 화면에서의 기본 색 표현 방식이며, 각 빛의 강도가 낮을수록 어두운 색이, 높을수록 밝은 색이 만들어집니다.
- HSB 색상(Hue), 채도(Saturation), 명도(Brightness)의 값에 따라 색을 지정합니다. 이 색상 모델은 HSL이라고도 불리며, 화면에서의 또 다른 색 표현 방식입니다. 채도가 높을수록 선명한 색이, 낮을수록 무채색에 가까운 색이 만들어지며, 명도가 높을수록 밝고 선명한 색이, 낮을수록 어두운 색이 만들어집니다.
- CMYK 청록(Cyan), 자홍(Magenta), 노랑(Yellow), 검정(black) 4색의 농도를 통해 색을 지정합니다. 인쇄물에서의 기본 색 표현 방식이며, 각 색의 농도가 낮을수록 밝은 색이, 높을수록 어두운 색이 만들어집니다.
- gray 명도(Brightness)를 사용해 무채색을 지정합니다. 값이 높을수록 밝은 색이, 낮을수록 어두운 색이 만들어집니다.

같은 색상 모델을 사용하더라도 값을 지정하는 방법이 여러 가지 존재하며, 이를 표 28.1에 정리 하였습니다. `xcolor` 패키지의 안내서에서는 이외에 다른 모델도 확인할 수 있습니다.

표 28.1: xcolor 패키지에서 사용할 수 있는 색상 모델

| 색상 모델 | 색상 구성 방법 |
|-------|--|
| rgb | 빨강, 녹색, 파랑의 세기를 각각 0(어두움)부터 1(밝음)까지의 실수로 지정합니다. 예를 들어, 빨강은 <code>rgb(1, 0, 0)</code> 으로 나타냅니다. |
| RGB | 빨강, 녹색, 파랑의 세기를 각각 0(어두움)부터 255(밝음)까지의 정수로 지정합니다. 예를 들어, 녹색은 <code>RGB(0, 255, 0)</code> 으로 나타냅니다. |
| HTML | 빨강, 녹색, 파랑의 세기를 각각 0(어두움)부터 255(밝음)까지의 정수 중 지정해 두 자리 16진수로 이어 붙여 나타냅니다. 예를 들어, 파랑은 <code>#0000FF</code> 로 나타냅니다. |
| cmyk | 청록, 자홍, 노랑, 검정의 농도를 각각 0(밝음)부터 1(어두움)까지의 실수로 지정합니다. 예를 들어, 자홍색은 <code>cmyk(0, 1, 0, 0)</code> 으로 지정합니다. |
| hsb | 색상, 채도, 명도를 각각 0부터 1까지의 실수로 지정합니다. 예를 들어, 청록색은 <code>hsb(0.5, 1, 0.5)</code> 로 나타냅니다. |
| Hsb | 색상을 0부터 360까지의 실수로, 채도와 명도를 각각 0부터 1까지의 실수로 지정합니다. 예를 들어, 노랑은 <code>Hsb(60, 1, 0.5)</code> 로 나타냅니다. |
| gray | 무채색의 명도를 0(검정)부터 1(흰색)까지의 실수로 지정합니다. |

자신이 사용하고 싶은 색의 색상 값은 외부 도구를 사용해 확인할 수 있습니다. 예를 들어, Google에 ‘color picker’라고 검색하면 원하는 색을 직접 선택한 뒤 그 값을 확인할 수 있습니다. Photoshop 등 그래픽 관련 프로그램을 사용할 수도 있습니다.

미리 정의된 색상 사용하기

색상 모델과 수치를 사용해 사용하려는 색을 지정하는 일은 간단한 것은 아닙니다. 따라서, xcolor 패키지가 제공하는 24가지 기본 색상을 사용할 수도 있습니다. 표 28.2에 이러한 기본 색상을 나열하였습니다.

표 28.2: xcolor 패키지에서 제공하는 기본 색상

| | | | |
|------------|-------------|----------|----------|
| ■ black | ■ gray | ■ olive | ■ teal |
| ■ blue | ■ green | ■ orange | ■ violet |
| ■ brown | ■ lightgray | ■ pink | ■ white |
| ■ cyan | ■ lime | ■ purple | ■ yellow |
| ■ darkgray | ■ magenta | ■ red | |

또, 패키지를 불러올 때 `dvipsnames` 옵션을 지정하면 CMYK 색 공간에서 정의된 색상 68개를, `svgnames`나 `x11names` 옵션을 지정하면 RGB 색 공간에서 정의된 색상 151개 또는 317개를 추가로 사용할 수 있습니다. 패키지 안내서에서 각 옵션을 통해 사용할 수 있는 색상 표를 확인할 수 있습니다.

28.2

색상 정의하기

색을 ‘정의’하는 것은 색에 이름을 붙인다는 것을 뜻합니다. 매번 사용할 색의 정보를 색상 모델을 사용해 지정하는 것보다, 색에 기억하기 쉬운 이름을 붙이고 이 이름을 사용하는 것이 효율적입니다. 색을 정의할 때에는 색상 모델을 사용할 수도 있고, 이미 정의된 색을 혼합할 수도 있습니다.

색상 모델로 색상 정의하기

색상 모델과 값을 사용해 색을 정의할 때에는 `\definecolor` 제어 문자열을 사용합니다.

```
\definecolor{<name>}{<color model>}{<color spec>}
```

<name>: 정의할 색상의 이름

<color model>: 사용할 색상 모델

<color spec>: 해당 모델을 사용해 나타낸 색상 정보

<color model>에는 표 28.1의 첫 번째 열에 있는 모델 중 하나를 입력하고, <color spec>에는 그에 대응하는 색상 정보를 입력하면 됩니다.

예를 들어, `rgb(0.25, 0.5, 0.75)`를 `myblue`라는 이름으로, `cmk(0.2, 0.4, 0.6, 0.8)`를 `mydarkbrown`이라는 이름으로, `hsb(0.25, 0.5, 0.75)`를 `mylightgreen`이라는 이름으로 정의하려면

```
1 \definecolor{myblue}{rgb}{0.25, 0.5, 0.75}
2 \definecolor{mydarkbrown}{cmk}{0.2, 0.4, 0.6, 0.8}
3 \definecolor{mylightgreen}{hsb}{0.25, 0.5, 0.75}
```

를 입력하면 됩니다.

정의한 색들을 혼합하기

정의되어 있는 색들을 혼합해 새로운 색을 정의할 수도 있습니다. 이때에는 `\colorlet` 제어 문자열을 사용합니다.

```
\colorlet{<name>}{<color syntax>}
```

<name>: 정의할 색상의 이름

<color syntax>: 색상의 혼합식

이때, 색상 혼합식은 아래와 같이 입력합니다.

```
색상 혼합식 → <color 1>!<percentage>!<color 2>
```

<color 1>: 혼합할 첫 번째 색

<percentage>: 첫 번째 색을 혼합할 비율 (백분율로 입력)

<color 2>: 혼합할 두 번째 색

즉, `red!30!blue`는 red와 blue 색상을 3:7 비율로 섞은 색상을 지정합니다. 만약 `!(color 2)` 부분을 생략하면 `!white`를 입력한 것과 같은 결과를 얻게 됩니다. 이미 정의한 색을 연하게 사용하려 할 때 조금 더 간단하게 식을 입력할 수 있습니다. 예를 들어, `red!30`은 흰색에 빨간색을 30% 섞은 색상을 지정합니다. 이 두 색상을 각각 `mypurple`와 `mypink`로 지정하려면

```
1 \colorlet{mypurple}{red!30!blue}
2 \colorlet{mypink}{red!30}
```

과 같이 입력하면 됩니다.

위 문법을 연쇄적으로 사용해 여러 색을 한 번에 혼합할 수도 있습니다. 예를 들어,

```
\colorlet{mycolor}{cyan!30!magenta!60!yellow!80!black}
```

과 같이 입력했다면, cyan과 magenta를 3:7로 혼합한 색을 얻고, 이 색과 yellow를 6:4로 혼합한 색을 얻고, 이 색과 black을 8:2로 혼합한 색을 얻어 이 색을 mycolor라는 이름으로 정의하게 됩니다.

28.3

색상 사용하기

위의 과정을 거쳐 색을 지정하고 정의하는 방법을 알아보았다면, 문서에 실제로 색상을 사용해 봅시다. 패키지 안내서에서는 다양한 명령어를 설명하고 있지만, 이 책에서는 유용한 몇 가지만을 설명하겠습니다.

글자에 색상 적용하기

글자에 색상을 적용할 때에는 `\textcolor` 및 `\color` 제어 문자열을 사용합니다. 전자는 범위 지정 형, 후자는 서식 선언형 제어 문자열입니다.

```
\textcolor{<color name>}{<text>}
```

`<color name>`: 글자에 적용할 색상의 이름

`<text>`: 색상을 적용할 내용

```
\color{<color name>}
```

`<color name>`: 글자에 적용할 색상의 이름

색상의 이름을 사용하는 대신, 색상 모델과 색상의 값을 직접 입력해서 글자에 색을 입힐 수도 있습니다. 이때에는 `{<color name>}` 대신 `[<color model>]{<color spec>}`을 입력하면 됩니다. `<color model>`에는 색상 모델을, `<color spec>`에는 색상의 값을 입력하면 됩니다.

예를 들어, 다음 네 가지 코드는 모두 글자에 청록색을 지정하는 데 사용할 수 있습니다.

```
1 \textcolor{cyan}{청록색}
2 {\color{cyan}청록색}
```

```
3 \textcolor{cmyk}{1, 0, 0, 0}{청록색}
4 {\color{cmyk}{1, 0, 0, 0}{청록색}}
```

수식 입력 모드에서도 `\mathcolor` 및 `\color` 제어 문자열을 사용해 색을 지정할 수 있습니다. `\mathcolor` 제어 문자열은 `\textcolor` 제어 문자열과 사용법이 완전히 동일합니다.

```
\[ \int \mathcolor{cyan}f
\,,d\mathcolor{magenta}g =
\mathcolor{cyan}f\mathcolor{magenta}g -
\int \mathcolor{magenta}g
\,,d\mathcolor{cyan}f \]
```

$$\int f dg = fg - \int g df$$

글상자에 색 입히기

`xcolor` 패키지는 글상자에 색을 입히기 위한 제어 문자열도 제공합니다. 먼저, `\colorbox`는 지정한 색을 배경색으로 하는 글상자를 만듭니다.

```
\colorbox{<color>}{<text>}
```

<color>: 지정하려는 배경색의 이름

<text>: 상자 안에 입력할 내용

앞과 마찬가지로, `{<color>}` 대신 `[<color model>]{<color spec>}`을 사용해 색상 모델로도 배경색을 지정할 수 있습니다.

이 제어 문자열을 사용하면 `\colorbox{cyan!10}{형광펜 효과}`를 낼 수도 있습니다.

이 제어 문자열을 사용하면 **형광펜 효과**를 낼 수도 있습니다.

또, `\fcolorbox`는 지정한 두 개의 색을 각각 테두리 색과 배경색으로 하는 글상자를 만듭니다.

```
\fcolorbox{<f. color>}{<b. color>}{<text>}
```

<f. color>: 테두리 색상의 이름

<b. color>: 배경 색상의 이름

<text>: 상자 안에 입력할 내용

여기에서도 마찬가지로 색상 이름 대신 색상 모델과 값을 이용해 테두리 색과 배경색을 각각 지정할 수 있습니다.

형광펜을 칠하고 `\fcolorbox{cyan}{cyan!10}{테두리}`까지 돌렸습니다.

형광펜을 칠하고 **테두리**까지 돌렸습니다.

제 VIII 편

프레젠테이션 작성하기

제 29 장

소개와 기본 문서 구조

이번 장에서는 beamer 클래스를 소개하고, 이 클래스를 사용해 문서를 작성하기 위한 기본적인 코드 구조를 알아볼 것입니다.

29.1

L^AT_EX으로 프레젠테이션 문서 만들기

L^AT_EX으로는 일반적인 문서뿐만 아니라 슬라이드 쇼를 위한 프레젠테이션 문서를 만들 수도 있습니다. 프레젠테이션 문서를 작성하기 위한 클래스를 사용하고 해당 클래스의 문법을 사용하면 원하는 문서를 얻을 수 있습니다.

프레젠테이션을 작성하기 위한 클래스로는 L^AT_EX의 표준 클래스인 slides 클래스를 비롯하여, powerdot, seminar, prosper, beamer 클래스 등이 있습니다. 프레젠테이션 문서를 작성하기 위한 다양한 클래스를 CTAN의 Presentation 페이지(ctan.org/topics/presentation)에서 확인할 수 있습니다. 하지만 slides 클래스와 많은 다른 클래스는 더 이상 개발·사용되지 않고, beamer와 몇 가지 소수의 클래스만 오늘날까지 사용되고 있습니다. 이 책에서는 가장 널리 사용되는 beamer 클래스를 사용해 프레젠테이션 문서를 작성하는 방법을 설명하겠습니다.

폭넓은 분야에서 널리 사용되는 PowerPoint와는 달리, beamer는 학술적인 발표를 할 때 유용합니다. 특히, beamer는 L^AT_EX의 문법을 사용하여 문서를 작성하므로 수식 작성 기능을 포함한 L^AT_EX의 여러 기능을 그대로 사용할 수 있습니다. 슬라이드 쇼의 바탕이 되는 L^AT_EX 문서가 있다면 이 코드를 약간 가공해 프레젠테이션 문서를 손쉽게 만들 수도 있습니다. 반대로, 프레젠테이션 문서의 소스 코드를 바탕으로 몇 가지 처리만 해 준다면 출력용 문서 파일도 간단하게 만들 수 있습니다.

29.2

기본적인 문법

예시 문서 29.1은 beamer 클래스를 사용해 문서를 작성하기 위한 코드입니다. 이 코드를 컴파일 하면 2페이지짜리 PDF 파일이 생성됩니다.

예시 문서 29.1 First Beamer

```

1 \documentclass{beamer}
2 \usetheme{Madrid}
3
4 \title{My First Beamer Document}
5 \subtitle{A Brief Introduction}
6 \author{Woohyun Kang}
7 \institute[KAIST]{Korea Advanced
  Institute of Science and Technology}
8 \date{\today}
9
10 \begin{document}
11
12 \begin{frame}
13   \maketitle
14 \end{frame}
15
16 \begin{frame}{Introduction}
17   Beamer class is used to create
  presentation using \LaTeX. It can
18   \begin{itemize}
19     \item create a presentation with a
  professional-look,
20     \item typeset math formulae easily,
  and
21     \item make lecture notes based on
  the presentation easily.
22   \end{itemize}
23   It has many more amazing features.
24 \end{frame}
25
26 \end{document}

```

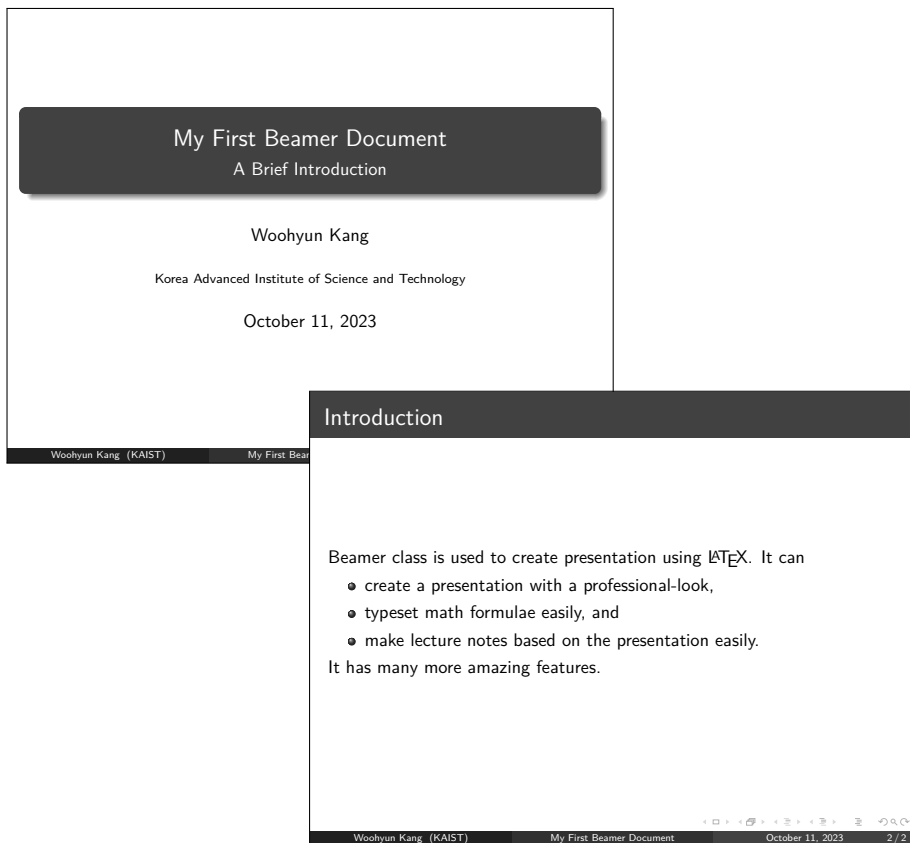


그림 29.1: 예시 문서 29.1의 컴파일 결과.

프레임

beamer에서는 슬라이드 쇼의 각 화면을 ‘프레임(frame)’이라고 부르며, 하나의 프레임은 하나의 frame 환경을 사용해 만듭니다. 일반적인 경우에는 하나의 프레임은 PDF 파일에서 하나의 페이지를 생성합니다. 즉, beamer의 프레임은 PowerPoint의 ‘슬라이드’와 비슷한 개념입니다. (참고로, beamer에서는 ‘슬라이드’라는 용어를 다른 의미로 사용하므로 슬라이드와 프레임을 혼동하지 않도록 주의하십시오.)

예시 문서 29.1의 코드를 잘 살펴 보면 frame 환경이 두 번 사용되었음을 알 수 있습니다. 첫 번째 프레임은 제목 페이지에, 두 번째 프레임은 설명 페이지에 대응됩니다. 예시 문서의 코드와 컴파일 결과물을 비교하면서 코드의 어떤 부분이 PDF 파일의 어느 부분에 대응하는지 살펴보고, 직접 beamer를 사용해 프레젠테이션 문서를 만들어 보십시오. 많은 \LaTeX 의 제어 문자열과 환경을 그대로 사용할 수 있으며 자주 사용되는 몇 가지 패키지(xcolor, amsmath, amssymb, amsthm, hyperref 패키지 등)는 beamer가 자동으로 불러오므로, 편리하게 문서를 작성할 수 있을 것입니다.

슬라이드

한 프레임에 있는 모든 내용을 한 번에 보여주는 대신 내용을 줄별로 차례로 보여주어, 현재 설명하는 내용에 청자가 집중하도록 할 수도 있습니다. 이때에는 각각의 프레임을 따로 만들 필요 없이, beamer의 ‘오버레이(overlay)’라는 기능을 사용하면 됩니다. 앞의 예시 문서 29.1에서 두 번째 frame 환경의 코드에 아래와 같이 `\pause` 제어 문자열을 추가해 보십시오.

```

1 \begin{frame}{Introduction}
2   Beamer class is used to create presentation using \LaTeX. \pause It can
3   \begin{itemize}
4     \item create a presentation with a professional-look, \pause
5     \item typeset math formulae easily, \pause and
6     \item make lecture notes based on the presentation easily.
7   \end{itemize} \pause
8   It has many more amazing features.
9 \end{frame}

```

이렇게 고친 뒤 컴파일 과정을 거치면 한 페이지로 조판되었던 두 번째 프레임이 네 개의 페이지로 늘어나고, 각 페이지에는 목록의 각 항목이 하나씩 추가되는 형태로 조판돼 있을 것입니다. 생성된 PDF 파일의 2~5페이지를 그림 29.2에 나타냈습니다.

이처럼 오버레이를 사용하면 프레임 상의 각 요소에 시간 차를 주어 표시할 수 있습니다. 오버레이를 사용했을 때 생성된 PDF 파일에서, 하나의 frame 환경이 생성한 각 페이지를 ‘슬라이드(slide)’라고 부릅니다. 기본적인 오버레이 기능은 `\pause` 제어 문자열로 충분하지만, 원하는 경우에는 더 다양한 방법으로 사용할 수도 있습니다. 이에 대해서는 제32장에서 자세히 다룹니다.

테마

beamer는 여러 가지 테마(theme)를 제공합니다. 테마는 머리글·바닥글·사이드 바의 디자인, 프레임의 제목의 색상, 목록의 각 항목 기호의 디자인 등 프레젠테이션 문서의 전체적인 디자인을

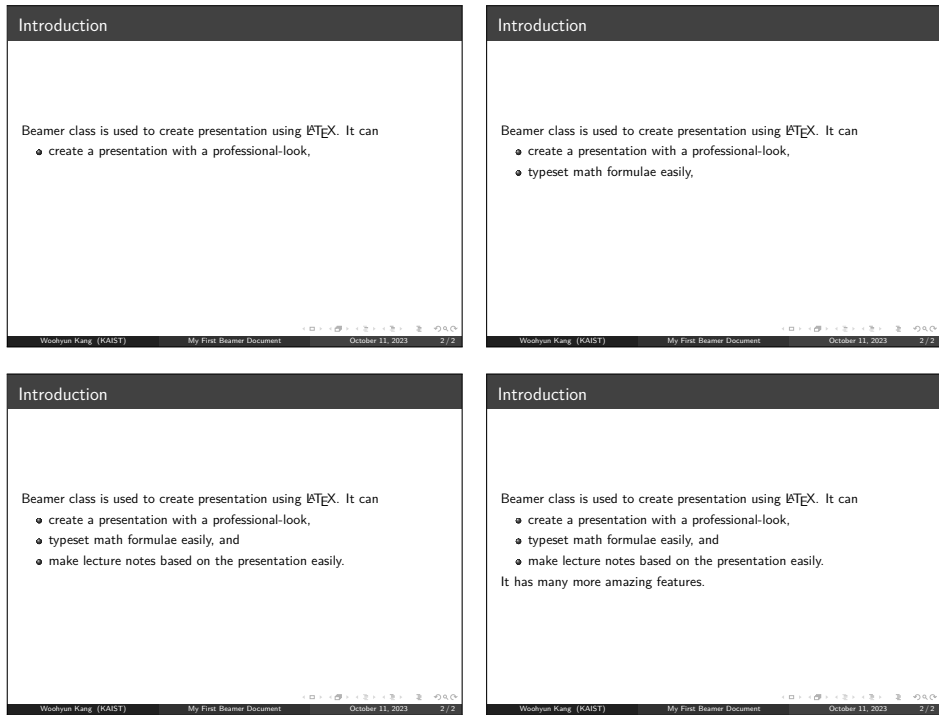


그림 29.2: 예시 문서 29.1에 오버레이를 사용한 모습.

결정합니다. 테마는 `\usetheme` 제어 문자열로 지정하며, 사용하려는 테마의 이름을 `\usetheme`의 인자로 입력하면 됩니다. 사용할 수 있는 테마의 목록은 beamer 안내서를 참조하십시오.

테마를 사용해 모든 요소를 한꺼번에 바꾸는 대신, 각각의 요소를 독립적으로 변경할 수도 있습니다. 이에 대해서는 제31장에서 자세히 다룹니다.

29.3

프레임 만들기

제 30 장

프레젠테이션의 형식 갖추기

어느 정도 형식을 갖춘 슬라이드 쇼의 첫 화면에는 제목과 저자, 날짜가 있고, 본문의 내용을 여러 부분으로 나누어 청자가 프레젠테이션에서 다룰 내용을 쉽게 파악할 수 있도록 합니다. 이번 장에서는 이때 사용할 수 있는 명령어를 설명합니다.

제 31 장

프레젠테이션의 레이아웃 바꾸기

31.1

클래스 옵션

먼저, 레이아웃 설정과 관련된 클래스 옵션을 소개하겠습니다. 페이지의 종횡비·크기와 글꼴 크기를 지정할 수 있습니다.

페이지의 종횡비는 `aspectratio` 옵션으로 지정합니다. 등호 뒤에 숫자를 두 개~네 개 입력하면 됩니다.

31.2

테마

beamer는 여러 가지 테마를 제공합니다.

테마는 외부 테마(outer theme), 내부 테마(inner theme), 색상 테마, 글꼴 테마로 구분되며, 이들을 독립적으로 지정할 수도 있습니다.

한편, 이들 요소를 한꺼번에 바꾸는 대신, 각 요소를 독립적으로 변경할 수도 있습니다. 각 요소가 어떤 모습으로 조판되는지는 템플릿(template)이 결정합니다.

31.3

프레임의 구성 요소

하나의 프레임은 다음 8가지 요소로 구성됩니다.

- 1 | 머리글과 바닥글
- 2 | 왼쪽과 오른쪽 사이드바
- 3 | 내비게이션 바
- 4 | 내비게이션 기호
- 5 | 로고

6 | 프레임의 제목

7 | 배경

8 | 프레임의 내용

제 32 장

오버레이 사용하기

제29장에서 설명했듯, 오버레이는 하나의 프레임에 있는 내용을 여러 슬라이드에 걸쳐 표시하는 기능입니다. 보통은 `\pause`만으로도 충분하지만, 내용을 순차적으로 표시하는 것뿐 아니라 더 다양한 효과를 내고 싶다면 이번 장의 내용이 도움이 될 것입니다. 예를 들어, 특정 슬라이드 ‘이후’ 대신 슬라이드 ‘에서만’ 내용이 표시되게끔 하거나, 목록에서 각 항목을 순차적으로 표시하되 항목이 처음 나타나는 슬라이드에서는 강조 표시를 하는 등의 방법을 알아봅니다.

32.1

오버레이 조건

beamer의 많은 제어 문자열과 환경에는 화살괄호 `<...>`를 사용해 지정하는 *<overlay spec.>*이라는 인자를 사용할 수 있습니다. 여기에는 해당 명령어가 유효하도록 할 슬라이드의 번호를 입력합니다.

예를 들어, `\alert`에 오버레이 설정을 하면 지정된 슬라이드에서만 강조 효과가 적용되고, `\item`에 오버레이 설정을 하면 지정된 슬라이드에서만 해당 항목이 표시됩니다.

슬라이드의 번호는 ‘1-3’처럼 범위를 사용할 수도, ‘2, 4, 7’처럼 쉼표를 사용할 수도, 이들을 혼합할 수도 있습니다. 또, 3슬라이드 이후를 모두 지정하려면 ‘3-’을, 처음부터 4슬라이드까지만을 지정하려면 ‘-4’을 입력하면 됩니다. ‘-3, 5, 7-’이라고 입력하면 4슬라이드와 6슬라이드를 제외한 모든 슬라이드를 지정하게 됩니다.

예를 들어, 다음 코드를 봅시다.

```

1 \begin{frame}
2   \frametitle{Definition of a Group}
3   \begin{definition}
4     A \alert{group} is a set  $(G)$  equipped with an operation  $(*)$  such that
5     \begin{enumerate}
6       \item associativity holds; that is,
7         
$$(a*b)*c = a*(b*c)$$

8       for any element  $(a,b,c)$  in  $(G)$ .

```

```

9      \item the identity exists; that is, for any element  $(a)$  in  $(G)$ ,
      there exists an element  $(e)$  in  $(G)$  such that
10      $[ a * e = e * a = a. ]$ 
11     \item the inverse exists; that is, for each element  $(a)$  in  $(G)$ 
      there exists an element  $(a^{-1})$  in  $(G)$  such that
12      $[ a * a^{-1} = a^{-1} * a = e. ]$ 
13     \end{enumerate}
14     \end{definition}
15     \end{frame}

```

여기에서 6행, 9행, 11행에 있는 `\item`을 각각

```

\item<2-> associativity ...
\item<3-> the identity ...
\item<4-> the inverse ...

```

로 고친 후 컴파일 해 봅시다. 첫 번째 슬라이드에서는 ‘A group is ...such that’까지만 표시되고, 두 번째 슬라이드에서는 첫 번째 항목도 함께, 세 번째 슬라이드에서는 두 번째 항목도 함께 표시되며, 네 번째 슬라이드에서는 프레임의 모든 내용이 표시됩니다.

32.2

명령어에 오버레이 조건 지정하기

오버레이 조건을 지정할 수 있는 명령을 알아보겠습니다.

32.3

가려진 항목을 흐리게 표시하기

가려진 항목을 ‘정말로’ 보이지 않게 하는 대신, 흐린 색으로 표시할 수도 있습니다. 이때에는 `\setbeamercovered` 제어 문자열을 사용합니다.

제 33 장

유인물과 노트 만들기

이번 장에서는 beamer의 소스 코드를 바탕으로 유인물과 노트를 만드는 방법을 설명합니다. 프레젠테이션 문서의 소스 코드에서 몇 가지 코드만 바꾸면 문서 전반의 서식을 신경 쓸 필요 없이 간편하게 유인물 또는 노트 버전 문서를 얻을 수 있습니다.

부록

부록에서는 \LaTeX 의 문법 외적인 내용을 설명합니다. \TeX 배포판을 컴퓨터에 설치하고 관리하는 방법과, 컴파일 과정에서 발생한 문제를 해결하는 방법을 다룹니다.

부록 A

컴퓨터에 T_EX 설치해서 사용하기

이 책의 첫머리에서는 Overleaf를 사용해 L^AT_EX 문서를 작성하는 방법을 설명하였지만, 복잡한 문서를 만든다면 Overleaf 등의 온라인 플랫폼을 사용하는 것이 불편할 것입니다. 이때에는 컴퓨터에 T_EX를 설치해 문서를 작성하는 것이 더 편리합니다.

이번 장에서는 Overleaf 등의 온라인 플랫폼을 통해 T_EX를 사용하지 않고, 컴퓨터에 T_EX를 설치해 사용하는 방법을 알아보겠습니다. T_EX를 설치하고 사용할 때에는 (1) T_EX에 관련된 모든 프로그램과 파일을 관리하는 프로그램인 T_EX 배포판과 (2) 문서를 편집하고 확인할 때 사용하는 편집기 겸 뷰어가 필요합니다. 이들을 설치하고 관리하는 방법을 알아보시다.

A.1

T_EX 배포판 설치

T_EX 배포판에는 여러 종류가 있으며, 그중 T_EX Live와 MikT_EX이 자주 사용됩니다. 두 배포판 모두 Windows, macOS, 리눅스, 유닉스에서 사용할 수 있습니다. 두 T_EX 배포판에는 몇 가지 차이점이 있으며, 그 특징을 그림 A.1에 정리하였습니다. 두 배포판 중 어느 것을 설치하더라도 조판 결과는 동일하므로, 자신이 사용하기에 편리한 것을 사용하시면 됩니다.

T_EX 배포판에 관한 자세한 정보는 TUG의 The T_EX Live Guide—2023¹을 참조하십시오.

T_EX Live 설치 TUG의 T_EX Live 페이지(tug.org/texlive)에서 안내에 따라 설치 프로그램을 내려받은 후, 해당 프로그램을 실행하면 됩니다. T_EX Live를 설치할 때에는 인터넷 연결이 필요하며, 1시간 이상의 긴 시간이 소요될 수 있습니다. 설치가 완료되면 프로그램 목록에 기본 편집기인 T_EXworks와 관리 프로그램인 T_EXLive Manager가 나타날 것입니다.

macOS에서는 T_EX Live의 macOS용 버전인 MacT_EX을 TUG의 MacT_EX 페이지(tug.org/mactex)의 안내에 따라 설치하면 됩니다. 설치 과정이 완료되면 Launchpad에 기본 편집기인 T_EXShop과 관리 프로그램인 T_EX Live utility가 나타날 것입니다.

¹URL: tug.org/texlive/doc/texlive-en/texlive-en.html

MikTeX 설치 MikTeX 홈페이지의 Download 페이지(miktex.org/download)에서 운영 체제에 맞는 설치 파일을 내려받습니다. 내려받은 프로그램을 실행하면 컴퓨터에 MikTeX이 설치됩니다. 설치 과정이 완료되면 프로그램 목록에 기본 편집기인 TeXworks와 관리 도구인 MikTeX Console이 나타납니다.

TeX Live의 특징

- 처음 설치할 때 필요한 프로그램과 패키지를 모두 설치합니다. 인터넷을 통해 설치가 진행되며, 5 GB의 패키지를 내려받으며 설치에 2시간 이상의 긴 시간이 소요됩니다.
- 컴파일 과정에서 내부적으로 사용하는 프로그래밍 언어인 Perl을 함께 설치합니다.
- MikTeX에 비해 컴파일 속도가 빠릅니다.
- 매년 새로운 버전이 출시됩니다. 여러 버전을 한 컴퓨터에 동시에 설치할 수도 있고, 단순히 업데이트 과정을 통해 최신 버전만을 사용할 수도 있습니다.
- Windows에서 사용자의 계정 이름이나, 작업한 파일의 이름 또는 경로에 한글이 포함되어 있으면 설치 및 문서 컴파일 과정에서 오류가 발생할 수 있습니다.

MikTeX의 특징

- 처음 설치할 때, 필수적인 패키지만을 내려받고 그 외 필요한 패키지는 문서를 컴파일하다 필요하면 그때 그때 내려받습니다. 필요한 패키지를 내려받을 때는 인터넷 연결이 필요합니다.
- 컴파일 과정에서 내부적으로 사용하는 프로그래밍 언어인 Perl이 기본적으로 설치되지 않습니다. 이전에 컴퓨터에 설치한 적이 없다면 별도 설치가 필요합니다.
- TeX Live에 비해 컴파일 속도가 느립니다.
- 새로운 버전이 출시되면 업데이트 과정을 통해 새 버전을 사용할 수 있습니다.
- 사용자의 계정 이름이나, 작업한 파일의 이름 또는 경로에 한글이 포함되어 있더라도 문제가 발생하지 않습니다.

그림 A.1: TeX Live와 MikTeX의 차이점

A.2

편집기 및 뷰어

TeX 문서는 플레인 텍스트 형식이므로, 어떤 텍스트 편집기를 사용하더라도 문제 없이 소스 코드를 작성할 수 있습니다. 운영 체제의 기본 프로그램인 메모장(Windows)이나 텍스트 편집기(macOS)를 사용할 수도 있고, 프로그래밍을 할 때 사용하는 VSCode, Xcode, Vim 등을 사용할 수도 있습니다. 또, 컴파일 과정을 거쳐 얻은 PDF 파일은 아무 PDF 뷰어를 통해 확인할 수 있습니다.

한편, TeXworks, TeXShop, TeXstudio, TeXmaker 등 TeX 문서 작성에 특화된 다양한 편집기 겸 뷰어도 있습니다. 이 프로그램들은 TeX을 사용해 문서를 작성할 때에는 ‘코드 작성 → 컴파일 → 생성된 문서 확인’의 과정을 하나의 프로그램 안에서 할 수 있게 해 줍니다. 편집기에서 코드를 작성한 후, 터미널을 열어 복잡한 명령어를 입력할 필요 없이 프로그램 안에 있는 ‘조판’ 버튼을 누르면 바로 컴파일이 시작됩니다. 프로그램의 뷰어를 통해 PDF 파일도 같이 볼 수 있으며, 컴파일이 완료되면 실시간으로 업데이트 되기 때문에 편리하게 문서를 작성할 수 있습니다. 이러한 프로그램에서는 편집기와 뷰어가 연동되어, PDF 파일 상의 특정 지점을 클릭하면 편집기에서 해당 부분의 코드를 보여 주고, 그 반대의 경우도 가능합니다.

A.3

컴퓨터에서 문서 컴파일하기

\TeX 문서 작성용 편집기를 사용하지 않는다면, 터미널을 사용해 컴파일을 할 수 있습니다. 운영 체제의 터미널(Windows라면 PowerShell, macOS라면 터미널 등)을 실행한 후, 컴파일할 엔진과 컴파일할 파일의 이름을 입력하면 됩니다. 예를 들어, \LaTeX 엔진으로 `~/Documents/mydocument.tex` 파일을 컴파일하려 한다면

```
$ latex ~/Documents/mydocument.tex
```

를 입력한 후 리턴 키를 눌러 명령을 실행하면 됩니다. 파일 이름에 마침표가 없는 경우에는 `‘.tex’` 부분을 생략해도 됩니다. 사용하려는 엔진에 따라 `latex` 대신 `pdflatex`, `xelatex`, `lualatex` 등을 입력하면 됩니다. 26.2절에서 설명한 일본어 문서를 작성하는 경우, `uplatex`을 사용할 수도 있습니다.

`latex` 또는 `uplatex` 명령어를 사용해 `.tex` 파일을 컴파일한 경우에는 PDF 파일 대신 DVI 파일을 얻습니다. 이때에는 `dvipdfmx` 명령어를 사용해 DVI 파일을 PDF 파일로 변환할 수 있습니다.

```
$ dvipdfmx ~/Documents/mydocument.dvi
```

위 명령어를 입력하면 `mydocument.dvi` 파일을 `mydocument.pdf` 파일로 변환할 수 있습니다. 마찬가지로 파일 이름에 마침표가 없는 경우에는 `‘.dvi’` 부분을 생략해도 됩니다.

참고 문헌 목록, 인용 표지와 색인 생성하기

참고 문헌을 인용하고 목록을 조판하기 위한 \BibTeX 은 `bibtex` 명령어를 사용해 실행합니다. 인자로 사용하는 `.aux` 파일을 지정하면 됩니다. 예를 들어, `~/Documents/mydocument.tex` 파일에서 참고 문헌 목록을 조판하려는 경우

```
$ bibtex ~/Documents/mydocument.aux
```

라고 입력하면 됩니다. 파일 이름에 마침표가 없는 경우에는 `‘.aux’` 부분을 생략해도 됩니다.

마지막으로, 색인을 생성하려는 경우에는 `MakeIndex` 또는 `xindy`를 사용하게 됩니다. 이때에는 제 20장에서 설명한 것처럼 `makeindex` 명령어나 `texindy` 명령어를 사용하면 됩니다.

```
1 $ makeindex ~/Documents/mydocument.idx
2 $ texindy -L english -I latex -C utf8 ~/Documents/mydocument.idx
```

사용하려는 프로그램에 따라 위 두 줄 중 하나를 골라 사용하십시오.

부록 B

패키지 및 프로그램 관리하기

이번 장에서는 \TeX 배포판과 패키지 파일을 관리하는 방법을 다룹니다. 컴퓨터에 설치한 \TeX 배포판에 따라 방법이 달라지므로, 이를 기준으로 구분하여 방법을 설명하겠습니다.

B.1

\TeX Live

컴퓨터에 설치한 \TeX 배포판이 \TeX Live인 경우에 대해 설명하겠습니다(\TeX Live를 포함하는 Mac\TeX 을 관리하는 방법은 다음 절에서 따로 다루겠습니다). \TeX Live Shell이라는 앱을 사용합니다.

Windows에서는 시작 메뉴의 앱 목록에서 ‘ \TeX Live 2023 → TLShell \TeX Live Manager’를 클릭해서, 그 외의 운영 체제에서는 터미널을 열고 `tlshell`을 입력해 관리 프로그램인 \TeX Live Shell을 열 수 있습니다.

이 프로그램에서는 현재 컴퓨터에 설치된 패키지와 설치되지 않은 패키지, 또 업데이트할 수 있는 패키지의 목록을 확인할 수 있습니다. 화면 가운데 ‘Package List’ 부분에서 목록에 표시할 조건을 선택한 후, 패키지 이름 왼쪽의 동그라미를 클릭한 뒤 업데이트하거나 삭제, 설치할 수 있습니다. 창 오른쪽의 ‘Update all’을 클릭해 간단히 모든 패키지를 업데이트할 수도 있습니다. 가능한 경우 관리 프로그램 자체의 업데이트도 할 수 있습니다.

새로운 버전의 \TeX Live 사용하기

\TeX Live는 매년 새로운 버전이 공개되며, 이전 버전은 일정 시점이 되면 더 이상 패키지의 업데이트를 제공하지 않습니다. 따라서, 새로운 버전의 \TeX Live를 사용하려면 \TeX Live Shell을 통해 업데이트를 설치하는 것이 아니라, TUG에서 새로운 버전의 \TeX Live를 새로 설치해야 합니다(설치 과정은 부록 A를 참조하세요). 만약 이전 버전의 \TeX Live가 필요하지 않다면 다음 절의 안내에 따라 해당 버전을 삭제할 수 있습니다.

TeX Live 삭제하기

더 이상 TeX를 사용하지 않는다면, TeX 배포판을 삭제해 공간 낭비를 줄이는 것이 좋습니다. 또, TeX Live는 새로운 버전이 설치되더라도 이전 버전이 사라지지 않기 때문에, 불필요하다면 오래된 버전의 배포판을 삭제해 주는 것이 좋습니다. 먼저, Windows에서는 아래의 과정을 통해 TeX Live를 삭제할 수 있습니다.

- 1 | 시작 메뉴를 열고, 설치된 앱 목록에서 'TeX Live 2023 → Uninstall TeX Live'를 클릭합니다. 컴퓨터에 설치된 버전에 따라 연도가 다르게 표시될 수 있습니다.
- 2 | 화면의 안내에 따라 삭제 과정을 완료합니다.

다른 운영 체제에서 TeX Live를 삭제하려면 터미널을 열고,

```
> tlmgr uninstall --all
```

를 실행합니다. 이후에는 자동으로 TeX Live가 삭제됩니다.

B.2

MacTeX

macOS에서는 앞 절에서 다룬 TeX Live Shell을 사용할 수도 있고, 대신 TeX Live Utility를 사용할 수도 있습니다. Launchpad에서 아이콘을 클릭하여 이 앱을 실행합니다.

'Updates' 탭에서는 현재 업데이트할 수 있는 패키지를 확인할 수 있으며, 목록에서 Control-클릭 후 나타나는 메뉴에서 'Update All Packages'를 클릭해 사용 가능한 모든 업데이트를 설치할 수도 있습니다. 'Packages' 탭에서는 CTAN에 현재 등록된 모든 패키지의 목록을 확인할 수 있습니다. 패키지를 더블 클릭하면 정보를 확인할 수 있으며, 패키지 문서가 존재한다면 여기에서 바로 읽을 수도 있습니다. TeX Live Utility 자체의 업데이트도 할 수 있으며, 메뉴 막대에서 'TeX Live Utility → Check for Updates...'를 클릭해 진행할 수 있습니다.

MacTeX도 TeX의 핵심 부분은 TeX Live를 통해 관리되므로, 새로운 연도의 버전이 출시되면 이전 버전의 패키지 업데이트 제공이 중단됩니다. 새로운 업데이트를 받으려면 부록 A의 내용을 참고해 TUG에서 새로운 버전의 MacTeX을 설치하십시오.

macOS에서 MacTeX을 삭제하려면 다음의 과정을 따릅니다.

- 1 | 먼저, TeX Live를 삭제합니다. Finder에서 '이동 → 폴더로 이동...'을 클릭합니다.
- 2 | `/usr/local/texlive`를 입력하고 리턴 키를 누릅니다.
- 3 | 삭제하려는 버전의 폴더를 선택하고 삭제합니다. 이중 `texmf-local` 폴더에는 여러 버전의 TeX Live에서 공유하는 파일이 저장되어 있으며, 컴퓨터에서 TeX를 완전히 제거하려면 `texmf-local` 폴더까지 삭제합니다. 삭제 과정에서 시스템이 관리자의 암호를 요구할 수 있습니다.
- 4 | 이제 TeX와 관련된 앱을 모두 삭제합니다. TeX Live를 업데이트하려는 것이라면 이 과정은 진행하지 마십시오. Finder의 왼쪽 사이드바에서 '응용 프로그램'을 클릭합니다.
- 5 | `TeX` 폴더를 선택하고 삭제합니다.

B.3

MiKTeX

컴퓨터에 설치한 \TeX 배포판이 \TeX Live가 아닌 MiKTeX이라면, MiKTeX Console을 사용해 관리할 수 있습니다. 설치된 앱 목록에서 ‘MiKTeX Console’을 클릭해 실행합니다.

- 1 | 맨 처음에는 사용자 모드로 진행할지, 관리자 모드로 진행할지 선택해야 합니다. 자신이 어떤 모드를 선택해야 하는지 모르겠다면 우선 사용자 모드를 선택해 진행하고, 막히는 것이 있다면 ‘File → Switch to MiKTeX administrator mode’를 선택해 관리자 모드로 전환하면 됩니다.
- 2 | 왼쪽에서 ‘Updates’를 선택하면 MiKTeX을 업데이트할 수 있습니다. ‘Check for updates’를 클릭해 업데이트를 확인하고, 가능한 업데이트가 있다면 ‘Update now’를 클릭해 업데이트를 진행합니다.
- 3 | ‘Documentation’을 선택하면 다양한 문서를 읽을 수 있습니다. 원하는 문서를 검색하고 더블 클릭하면 됩니다.
- 4 | ‘Packages’를 선택하면 패키지의 목록을 확인할 수 있습니다. 현재 컴퓨터에 설치된 패키지를 확인하고, 필요한 경우 설치·삭제·업데이트할 수 있습니다.
- 5 | 컴퓨터에서 MiKTeX을 삭제하거나 설정을 초기화하려면 ‘Cleanup’을 선택합니다. 이곳의 안내에 따라 진행합니다.

MiKTeX은 연도에 따른 버전 구분 없이 모든 패키지의 업데이트가 제공됩니다. \TeX Live와 달리 재설치 등을 통한 업그레이드 과정을 거치지 않으며, 필요한 경우 MiKTeX Console에서 \TeX 배포판의 업데이트를 진행하면 됩니다.

부록 C

문제 해결하기

가장 먼저, 이번 장에서는 문제를 해결하는 방법을 알아봅니다. 제II편에서 설명하였듯 문제는 심각한 정도에 따라 ‘오류’와 ‘경고’로 구분됩니다.

C.1

오류

‘오류(error)’는 심각한 문제로, 주로 소스 파일의 문법에 문제가 있는 경우 발생합니다. 많은 오류는 오타에 의해 발생하며, 주의해서 사용해야 하는 특수 문자를 잘못 입력한 경우에도 발생할 수 있습니다. 아래 목록에는 제I편부터 제IV편까지에서 다루었던 내용에서 발생할 수 있는 오류 메시지를 정리하였습니다.

기본적인 오류 메시지

- Can be used only in preamble.
\documentclass나 \usepackage가 document 환경 안에서 쓰였을 때 발생합니다. 이들은 전처리부에 입력해야 합니다.
- Paragraph ended before (cs) was complete.
제어 문자열의 인자를 지정할 때, 중괄호를 닫지 않거나 인자에 빈 줄이 포함되면 발생합니다. 중괄호의 개수가 맞는지, 또 인자에 빈 줄이 포함돼 있지는 않은지 확인하세요.
- Too many }'s.
여는 중괄호보다 닫는 중괄호가 더 많을 때 발생합니다. 제어 문자열의 인자를 지정할 때 개수를 헷갈리거나 여는 중괄호를 입력하는 것을 잊었을 수 있습니다.
- Missing { inserted. 또는 Missing } inserted.
중괄호가 잘못 사용되었을 때 발생합니다.
- Undefined control sequence.

자주 보게 될 오류 메시지로, 입력한 제어 문자열이 정의되어 있지 않음을 뜻합니다. 제어 문자열에 오타는 없는지, 대소문자가 틀리진 않았는지, 제어 문자열 뒤의 띄어쓰기는 잘 되었는지 확인하세요. 제어 문자열을 사용하기 위해 불러와야 하는 패키지를 불러오지 않았을 때에도 이 오류가 발생합니다.

- Environment $\langle unknown \rangle$ undefined.

입력한 이름의 환경이 정의되지 않았을 때 발생합니다. 환경의 이름을 입력할 때 오타가 있었거나, 환경을 사용하기 위해 불러와야 할 패키지를 불러오지 않았을 때 주로 발생합니다.

- $\backslash begin\{\langle env_1 \rangle\}$ on input line $\langle n \rangle$ ended by $\backslash end\{\langle env_2 \rangle\}$.

시작한 환경의 이름과 끝낸 환경의 이름이 다를 때 발생합니다. 여러 개의 환경을 중첩해 사용할 때 환경을 열고 닫는 순서를 헷갈리거나, $\backslash begin$ 또는 $\backslash end$ 의 짝이 맞지 않으면 발생합니다.

클래스 및 패키지 관련 오류 메시지

- File ' $\langle cls \rangle$.cls' not found.

문서 클래스를 지정할 때, $\langle cls \rangle$ 라는 이름의 클래스가 존재하지 않으면 발생합니다. 문서 클래스의 이름이 옳게 입력되었는지 확인하세요.

- $\backslash usepackage$ before $\backslash documentclass$.

전처리부에서 $\backslash documentclass$ 가 입력되기 전에 $\backslash usepackage$ 가 입력된 경우 발생합니다. $\backslash documentclass$ 가 $\backslash usepackage$ 앞에 오도록 해 주세요.

- File ' $\langle pkg \rangle$.sty' not found.

패키지를 불러올 때, $\langle pkg \rangle$ 라는 이름의 패키지가 존재하지 않으면 발생합니다. 패키지의 이름이 옳게 입력되었는지 확인하세요.

- Unknown option ' $\langle option \rangle$ ' for package ' $\langle pkg \rangle$ '.

패키지를 불러올 때 옵션을 잘못 지정한 경우 발생합니다.

문자 입력 관련 오류 메시지

- Unicode character $\langle char \rangle$ not set up for use with LaTeX.

소스 파일에 입력한 문자 $\langle char \rangle$ 을 조판하기 위한 적절한 설정이 갖추어지지 않았을 때 발생합니다. 예를 들어, 소스 파일에 한글을 입력했는데 kotex 패키지를 불러오지 않은 경우 등이 해당합니다.

- Misplaced alignment tab character &.

&는 array 환경, align 환경 등에서 열 구분자로 사용됩니다. 적절하지 않은 곳에서 &를 입력하면 이 오류가 발생합니다. 문자 '&'를 출력하려면 $\backslash \&$ 를 입력하세요.

- You can't use 'macro parameter character #' in $\langle mode \rangle$ mode.

제어 문자열을 정의할 때 매개 변수 지정자로 사용하는 #을 적절하지 않은 곳에 입력하면 발생합니다. 문자 '#'를 출력하려면 $\backslash \#$ 를 입력하세요.

- There's no line here to end.

문단과 문단 사이에 추가 간격을 주기 위해 문단 끝에 `\\`을 입력하면 발생합니다. 이러한 경우에는 `\\` 대신 25.1절에서 다룬 `\bigskip` 등의 제어 문자열을 대신 사용하는 것이 적절합니다.

모드 관련 오류 메시지

- Missing \$ inserted.

수식 입력 모드에서만 사용할 수 있는 `^`나 `_`, 또는 수식 입력 모드 전용 제어 문자열을 텍스트 모드에서 입력했음을 뜻합니다. 수식 입력 모드를 시작한 후 수식 입력 모드를 끝내지 않았을 때에도 이 오류가 발생합니다.

- $\langle cs \rangle$ allowed only in math mode.

수식 입력 모드에서만 허용되는 제어 문자열 $\langle cs \rangle$ 를 텍스트 모드에서 입력했을 때 발생합니다.

- Bad math environment delimiter.

`\(, \), \[, \]`를 잘못 사용하면 발생합니다. 짝은 맞는지, 수식 입력 모드 안에서 또 수식 입력 모드를 시작하진 않았는지 확인하세요.

표를 작성할 때의 오류 메시지

- LaTeX Error: Illegal character in array arg.

`tabular` 또는 `array` 환경의 인자에 사용할 수 없는 내용을 입력했을 때 발생합니다.

- Misplaced `\noalign`.

`\hline` 제어 문자열 앞에 `\\`을 입력하지 않은 경우에 발생합니다. 가로선을 긋기 위해 `\hline`을 사용하기 전에는 줄을 나누어 주어야 합니다.

- Misplaced `\omit`.

`\multicolumn`을 입력한 칸에 다른 내용이 더 입력되어 있을 때 발생합니다. 해당 칸에 입력하려는 내용은 `\multicolumn`의 인자로만 지정해야 합니다.

- Extra alignment tab has been changed to `\cr`.

생성한 열의 수보다 더 많은 열에 내용을 입력했을 때 발생합니다. `&`를 지나치게 많이 사용하지 않았는지, `\\`를 잊지 않았는지 확인하십시오.

특정 명령어의 오류 메시지

- No `\title` given.

`\maketitle`을 입력하였지만 `\title`을 지정하지 않았을 때 발생합니다.

- Something's wrong--perhaps a missing `\item`.

목록 작성 환경(`enumerate` 등)에서 `\item`이 사용되지 않았을 때 발생합니다.

- Lonley `\item`--perhaps a missing list environment.

`\item` 제어 문자열이 목록 작성 환경 바깥에서 사용되었을 때 발생합니다.

- Too deeply nested.
itemize 또는 enumerate 환경을 다섯 번 이상, description 환경을 일곱 번 이상 중첩하여 사용했을 때 발생합니다.
- `\verb` illegal in argument.
`\verb` 제어 문자열을 인자로 지정했을 때 발생합니다. 이 제어 문자열은 정의가 복잡하여, 다른 제어 문자열의 인자로 지정할 수 없습니다.
- `\verb` ended by end of line.
`\verb` 제어 문자열을 시작하고 끝내지 않았을 때 발생합니다. 그대로 조판하려는 문자열의 범위를 문법에 맞게 지정했는지 확인하세요.
- Double subscript. 또는 Double superscript.
위 첨자나 아래 첨자를 적절한 중괄호 없이 여러 번 사용했을 때 발생합니다.
- Missing `\right`. inserted.
수식 입력 모드에서 `\left`를 입력하고 `\right`를 입력하지 않았을 때 발생합니다.
- Extra `\middle`. 또는 Extra `\right`.
수식 입력 모드에서 `\left`가 입력되지 않았는데 `\middle` 또는 `\right`가 입력되었을 때 발생합니다.
- Too many columns in eqnarray environment.
eqnarray 환경에서 &가 한 줄에서 3개 이상 사용되면 발생합니다.

C.2

경고

‘경고(warning)’는 오류에 비해서는 덜 심각한 문제로, 주로 \LaTeX 소스 파일의 문법에는 문제가 없지만 조판 결과가 만족스럽지 않을 수 있는 경우 발생합니다.

오버풀과 언더풀

가장 쉽게 발생하는 경고 메시지입니다. 여기에서는 이 메시지가 왜 표시되는지 간략하게만 설명하겠습니다. 오버풀과 언더풀에 대한 자세한 내용은 24.4절에서 다룹니다.

- Overfull `\hbox` ($\langle width \rangle$ too wide) ...
 \TeX 이 문서의 내용을 조판할 때, 적절한 줄바꿈 지점을 찾지 못해 글줄의 길이가 길어졌을 때 발생합니다. 오류가 발생한 곳의 조판 결과를 보면 해당 부분의 내용이 오른쪽 여백 부분을 침범했음을 확인할 수 있습니다.
- Underfull `\hbox` (badness $\langle num \rangle$) ...
위와 마찬가지로, \TeX 이 적절한 줄바꿈 지점을 찾지 못했을 때 발생합니다. 오버풀과 다르게, 문서의 내용이 페이지의 여백을 침범하지는 않지만 단어 사이의 간격이 지나치게 넓어집니다.

- Overfull `\vbox` ($\langle height \rangle$ too high) ...

\TeX 이 문서의 내용을 조판할 때, 적절한 페이지 나눔 지점을 찾지 못했을 때 발생합니다. 오류가 발생한 곳의 조판 결과를 보면 해당 부분의 내용이 아래쪽 여백을 침범했음을 확인할 수 있습니다.

- Underfull `\vbox` (badness $\langle num \rangle$) ...

위와 마찬가지로, \TeX 이 적절한 페이지 나눔 지점을 찾지 못했을 때 발생합니다. 오버풀과 다르게, 문서의 내용이 페이지의 여백을 침범하지는 않지만 문단과 문단 사이의 간격이 지나치게 넓어집니다.

문서 클래스의 옵션 중 `draft`를 지정하면, 오버풀이 발생한 곳을 \TeX 이 표시해 주므로 검토 과정에서 더 쉽게 확인할 수 있습니다. 기본값은 이러한 표시를 하지 않는 `final`입니다.

기타 경고 메시지

- Some font shapes were not available, defaults substituted.

지정한 글꼴 스타일을 사용할 수 없을 때 발생합니다. 예를 들어, `\scshape`와 `\bfseries`를 함께 사용하려고 하면 발생할 수 있습니다. 다른 글꼴 스타일을 사용하세요.

- No `\author` given.

`\maketitle`을 입력하였지만 `\author`를 지정하지 않았을 때 발생합니다. `\author`를 사용해 저자를 지정하세요.

- Command $\langle cs \rangle$ invalid in math mode.

텍스트 모드에서만 사용할 수 있는 제어 문자열 $\langle cs \rangle$ 를 수식 입력 모드에서 입력했을 때 발생합니다. 해당 제어 문자열과 동일한 동작을 하는 수식 입력 모드용 명령어를 사용하거나, 해당 제어 문자열을 `\text`의 인자로 사용하세요.

- Unused global option(s): [$\langle options \rangle$].

`\documentclass`에 입력한 옵션이 불러온 문서 클래스에 존재하지 않는 옵션이고, 불러온 패키지 중 어느 것도 이 옵션을 사용하지 않을 때 발생합니다. 옵션이 옳게 지정되었는지 확인하세요.

C.3

인터넷에 검색하기

\LaTeX 으로 문서를 작성하다가 원인을 도저히 알 수 없는 문제가 발생했을 때, 특정 기능은 어떻게 구현하는지 궁금할 때, \TeX 과 \LaTeX 의 깊은 내용이 궁금할 때 등 많은 경우에는 인터넷 검색을 통해 궁금증을 해결할 수 있습니다. 이번 절에서는 인터넷 검색을 통해 궁금증을 해결할 때 참고할 사항을 몇 가지 알려드리겠습니다.

- 기초적인 내용이 아니라면 영어로 검색한다. 대부분의 경우, 영어로 검색했을 때 양질의 답변을 많이 얻을 수 있습니다. 국내에서도 많은 사람들이 사용하는 Word, 아래아한글, 포토샵 등과

달리, \TeX 은 비교적 한정된 분야의 사람들이 사용하므로, 좋은 정보를 많이 찾기 어렵습니다. 영어가 어렵더라도 영어로 정보를 찾는 습관을 들이는 것이 좋습니다. 다만, 아주 기초적인 내용은 한국어로 작성된 글도 몇몇 있으므로 기본적인 내용은 한국어로 검색해도 좋은 결과를 얻을 수 있습니다.

- 한글 및 한국어 관련 기능은 한국어로 검색한다. 당연한 이야기이지만, 한글과 한국어를 사용해 문서를 작성하는 사람은 아주 높은 확률로 한국인입니다. 이런 경우에는 오히려 영어로 검색했을 때 좋은 답변을 찾기 힘들 것입니다.
- 특정 주제에 대한 자세한 내용이 궁금하다면 매뉴얼 형식의 웹사이트를 사용한다. 특정 주제의 문법에 관한 많은 내용을 정확히 알고 싶다면, \LaTeX 2\epsilon unofficial reference manual이나 Overleaf 등 매뉴얼 형식의 웹사이트에서 정보를 찾는 것이 적절합니다.
- 특정 기능을 구현하는 방법, 특정 주제에 관한 궁금증은 질문 및 답변 형식의 웹사이트를 사용한다. 구체적인 기능을 어떻게 구현하는지 알고 싶거나, 특정 주제에 관한 질문은 Stack Exchange 나 KTUG의 질문 게시판 등 질문·답변 형식의 웹사이트를 사용하는 것이 좋습니다. 질문 및 답변 웹사이트에서는 일반적으로 답변 작성자가 예시 코드를 제시하므로, 이를 참고해 더 쉽게 해당 기능을 사용하는 방법을 알 수 있습니다.
- 패키지의 사용법은 패키지 매뉴얼을 통해 확인한다. 이 패키지가 어떤 기능을 제공하는지, 어떻게 사용하는지 궁금하다면 해당 패키지 매뉴얼을 읽는 것이 좋습니다. 패키지 매뉴얼은 CTAN 이라는 웹사이트에서 확인할 수 있습니다. 컴퓨터 터미널의 명령 창에 `texdoc <pkg name>`을 입력하거나, \TeX 배포판의 관리 도구를 사용해 확인할 수도 있습니다. 예를 들어, kotex 패키지의 문서는 터미널에 `texdoc kotex`을 입력해 확인할 수 있습니다.
- 질문을 올릴 때에는 최소 동작 예제를 첨부한다. 문서에 문제가 있어 질문을 올리든, 특정 기능을 구현하는 방법이 궁금해서 질문을 올리든, 답변 작성자가 참고할 수 있는 최소 동작 예제(Minimal Working Example, MWE)를 꼭 함께 첨부하세요. 문제가 있었다면 그 문제가 발생하는 문서를 첨부하고, 기능을 구현하는 방법이 궁금하다면 최소한의 내용과 함께 자신이 원하는 출력 결과 이미지를 첨부하세요. 자신이 작성하던 문서 자체를 그대로 올릴 필요는 없습니다. 최소 동작 예제를 첨부하는 것은 \TeX 질문·답변 사이트의 기본적인 예절입니다. 답변 작성자는 자신의 소중한 시간을 할애해 여러분에게 답변을 제공하는 것이므로, 답변 작성자를 배려해 질문을 작성하는 것이 예의입니다.

아래는 인터넷에 검색할 때 참고하면 좋은 웹사이트입니다. Google 등 검색 엔진에 관련 검색어를 입력하면, 아래의 웹사이트를 금방 발견할 수 있을 것입니다.

- Stack Exchange 질문을 올리면 다른 사용자들이 답변을 다는 형식으로 운영되는 웹사이트입니다. 이미 수많은 질문과 답변이 축적되었기 때문에 검색만 하더라도 원하는 내용을 찾을 수 있을 것입니다. 그래도 궁금한 내용을 찾을 수 없다면, 직접 질문을 올려 여러 사람들의 답변을 기대할 수도 있습니다.

- KTUG 제1장에서 설명하였듯 한글 및 한국어 문서 작성에 관련된 내용을 많이 찾을 수 있습니다. 꼭 한국어 또는 한글에 관한 내용이 아니더라도, 많은 사용자들이 한국어로 질문을 올리고 답변을 달기도 합니다.
- Overleaf \LaTeX 의 다양한 문법을 주제에 따라 분류해 설명하는 웹사이트입니다. 자주 사용되는 패키지의 사용법도 간략히 소개하고 있습니다.
- \LaTeX 2 ϵ unofficial reference manual Overleaf처럼 \LaTeX 의 문법을 주제에 따라 분류해 설명하는 웹사이트입니다. Overleaf에 비해 더 깊은 내용을 다루고 있습니다. 패키지를 통해 구현되는 \LaTeX 자체의 기능이 아닌 내용은 거의 다루지 않습니다.
- CTAN 이름은 ‘Comprehensive \TeX Archive Network’를 줄인 것으로, 이름대로 \TeX 에 관한 수많은 파일이 저장되어 있습니다. 많은 패키지 문서와 가이드를 이 웹사이트에서 찾을 수 있습니다.

부록 D

템플릿

이번 장에서는 몇 가지 템플릿을 제공합니다. 문서를 작성할 때 요긴하게 사용할 수 있을 것입니다.

먼저, 다음은 article 클래스의 문서 템플릿입니다. 한국어 문서를 작성하는 것이 아니라면 제13행과 제14행은 삭제하십시오.

```

1 \documentclass[a4paper]{article}
2 \usepackage{mathtools,amssymb}
3 \newcommand\N{\mathbb N}
4 \newcommand\Z{\mathbb Z}
5 \newcommand\Q{\mathbb Q}
6 \newcommand\C{\mathbb C}
7 \newcommand\R{\mathbb R}
8 \usepackage{amsthm}
9 \newtheorem{proposition}{Proposition}
10 \newtheorem{theorem}{Theorem}
11 \newtheorem{corollary}{Corollary}
12 \newtheorem{remark}{Remark}
13 \theoremstyle{definition}
14 \newtheorem{definition}{Definition}
15 \theoremstyle{remark}
16 \newtheorem{remark}{Remark}
17 ⟨other packages, definitions, etc.⟩
18 \usepackage{hyperref}
19 \usepackage[hangul]{kotex}
20 \usepackage{ob-mathleading}
21 \title{⟨title here⟩}
22 \author{⟨author here⟩}
23 \date{\today}
24 \begin{document}
25 \maketitle
26 ⟨content here⟩
27 \end{document}

```

다음은 report 클래스와 book 클래스의 문서 템플릿입니다. report 클래스에서는 `\frontmatter`, `\mainmatter`, `\backmatter`를 빼고 사용하십시오.

```

1 \documentclass[a4paper]{book}
2 \usepackage{mathtools,amssymb}
3 \newcommand\N{\mathbb N}
4 \newcommand\Z{\mathbb Z}
5 \newcommand\Q{\mathbb Q}
6 \newcommand\C{\mathbb C}
7 \newcommand\R{\mathbb R}
8 \usepackage{amsthm}
9 \newtheorem{proposition}{Proposition}
10 \newtheorem{theorem}{Theorem}
11 \newtheorem{corollary}{Corollary}
12 \newtheorem{remark}{Remark}
13 \theoremstyle{definition}
14 \newtheorem{definition}{Definition}
15 \theoremstyle{remark}
16 \newtheorem{remark}{Remark}
17 \usepackage{makeidx}
18 \makeindex
19 (other packages, definitions, etc.)
20 \usepackage{hyperref}
21 \usepackage[hangul]{kotex}
22 \usepackage{ob-mathleading}
23 \title{{title here}}
24 \author{{author here}}
25 \date{\today}
26 \begin{document}
27 \frontmatter
28 \maketitle
29 \tableofcontents
30 \mainmatter
31 {content here}
32 \backmatter
33 \bibliography{{bibliography file here}}
34 \printindex
35 \end{document}

```

부록 E

유용한 패키지

이 책에서는 여러 패키지를 소개하였습니다. 이번 장에서는 이 책에서 설명한 패키지를 포함하여, 여러 가지 유용한 패키지를 소개할 것입니다.

`amsmath` 패키지는 수식을 작성하는 데 유용한 많은 제어 문자열과 환경을 정의합니다. 행렬을 작성하는 환경, 정렬된 수식을 작성하는 환경, 연산자를 입력하는 제어 문자열, 수식 도중 일반 텍스트를 입력하는 제어 문자열 등이 있습니다. 수식을 작성하는 많은 문서에서 필수적으로 불러옵니다.

패키지 매뉴얼은 `texdoc amsdoc`을 입력해 확인할 수 있습니다.

`amssymb` 패키지는 수식에서 사용되는 많은 기호를 정의합니다. 또, `amsfonts` 패키지를 불러와 프락투르체와 칠판 볼드체를 사용할 수 있게 됩니다. 수식 글꼴을 변경하는 패키지를 불러오지 않았다면, 이 패키지를 불러와 다양한 기호를 입력하게 됩니다.

`amsthm`

`array` 패키지는 \LaTeX 의 기본 `tabular` 및 `array` 환경을 확장하고 몇 가지 버그를 수정합니다. 표의 열 생성 인자로 `m`, `b`를 사용할 수 있게 되며, `!`와 `>`, `<` 인자를 사용해 표의 형식을 변경할 수 있습니다.

패키지 매뉴얼은 `texdoc array`를 입력해 확인할 수 있습니다.

`babel` 패키지는 문서의 언어 설정을 담당합니다.

`booktabs` 패키지는 더 깔끔한 디자인의 표를 작성할 수 있도록 해 줍니다. 가로줄과 세로줄을 사용해 칸을 구분하는 대신, 세로줄은 사용하지 않고 가로줄을 두께를 다르게 하여 내용을 구분합니다.

패키지 매뉴얼은 `texdoc booktabs`를 입력해 확인할 수 있습니다.

`calc`

`caption`

`cleveref`

`csquotes`

`ctex`

`enumitem`

fancyhdr

float

fontenc

fontspec 패키지는 Xe_{La}TeX이나 Lua_{La}TeX 엔진에서 문서의 글꼴을 변경할 수 있도록 해 줍니다. OTF 및 TTF 글꼴을 사용하려면 이 패키지를 불러오면 됩니다.

패키지 매뉴얼은 `texdoc fontspec`을 입력해 확인할 수 있습니다.

geometry

graphicx

helvet

hyperref

inputenc

kotex 패키지는 L_AT_EX에서 한국어 문서 작성 및 한글 입력을 위한 설정을 도와 줍니다. 이 패키지는 사실 컴파일 엔진에 따라 다른 패키지를 불러 오는 역할을 합니다. PDF_{La}TeX에서는 `kotex-utf` 패키지를, Xe_{La}TeX에서는 `xetex-ko` 패키지를, Lua_{La}TeX에서는 `luatex-ko` 패키지를 불러옵니다.

패키지 매뉴얼은 `texdoc kotex`를 입력해 확인할 수 있습니다. 엔진에 따라 `kotex kotex-utf`, `texdoc xetexko`, `texdoc luatexko`를 입력하면 엔진에 맞는 자세한 내용을 읽을 수 있습니다.

listings

luatexja

makeindex

mathtools

multicol

multirow

newpx, newpxmath, newpxtext

newtx, newtxmath, newtxtext

pgfplots

plautopatch

polyglossia

ragged2e

subcaption

tabularray

tabularx

tcolorbox

textcomp

tikz

tikz-cd

titlesec

titletoc

unicode-math 패키지는 Xe_{La}TeX이나 Lua_{TeX} 엔진에서 수식 글꼴을 변경할 수 있도록 합니다. 수식 작성을 지원하는 OTF 글꼴을 사용해 T_EX에서 수식을 조판하도록 합니다.

wrapfig

xcolor

부록 F

예제 정답

이번 장에서는 예제의 정답을 설명합니다.

참고 문헌

- [1] Michel Goossens Frank Mittelbach. *The L^AT_EX Companion*. Addison-Wesley, second edition, 2004.
- [2] Donald E. Knuth. *The T_EXBook*. Addison-Wesley Professional, 1986.
- [3] Leslie Lamport. *L^AT_EX: A Document Preparation System*. Addison-Wesley, second edition, 1994.
- [4] Frank Mittelbach and David Carlisle. *A new implementation of L^AT_EX's tabular and array environment*. L^AT_EX3 Project, 2.5g edition, June 2023.
- [5] Tobias Oetiker, Hubert Partl, Irene Hyna, and Elisabeth Schlegl. *The Not So Short Introduction to L^AT_EX 2_ε*, version 6.1 edition, March 2021. Accessed: October 2, 2022.
- [6] Scott Pakin. *The Comprehensive L^AT_EX Symbol List*, May 2021. Accessed: October 2, 2022.
- [7] Oren Patashnik. *BibT_EXing*, February 1988. Accessed: February 5, 2023.

제어 문자열과 환경 찾아보기

| | | |
|--------------------------------------|--|---------------------------------------|
| <code>\!</code> , 68 | <code>\approx(\approx)</code> , 78 | <code>\bibliography</code> , 157 |
| <code>\#(\#)</code> , 18 | <code>\Approxcolon(\approx::)</code> , 79 | <code>\bibliographystyle</code> , 159 |
| <code>\\$(\\$)</code> , 18 | <code>\approxcolon(\approx)</code> , 79 | <code>\Big</code> , 95 |
| <code>\%(\%)</code> , 18 | <code>\approxeq(\cong)</code> , 78 | <code>\big</code> , 95 |
| <code>\&(\&)</code> , 18, 96 | <code>\ar</code> , 110 | <code>\bigcap</code> , 75 |
| <code>*</code> , 18 | <code>\arabic</code> , 180 | <code>\bigcirc</code> , 74 |
| <code>\,</code> , 68 | <code>\arccos(\arccos)</code> , 75 | <code>\bigcup</code> , 75 |
| <code>\:</code> , 68 | <code>\arcsin(\arcsin)</code> , 75 | <code>\Bigg</code> , 95 |
| <code>\;</code> , 68 | <code>\arctan(\arctan)</code> , 75 | <code>\bigg</code> , 95 |
| <code>\></code> , 68 | <code>\arg(\arg)</code> , 75 | <code>\Biggl</code> , 94 |
| <code>_(_)</code> , 18, 72 | <code>array</code> 환경, 99, 119, 121 | <code>\biggl</code> , 94 |
| <code>\^</code> , 18, 72 | <code>\arrow</code> , 110 | <code>\Biggm</code> , 95 |
| <code>\~</code> , 18 | <code>\ast(\ast)</code> , 74 | <code>\biggm</code> , 95 |
| <code>\ </code> , 18, 96 | <code>\asymp(\asymp)</code> , 78 | <code>\Biggr</code> , 94 |
| <code>\{(\{)</code> , 18 | <code>\author</code> , 39 | <code>\biggr</code> , 94 |
| <code>\}(\})</code> , 18 | | <code>\Bigl</code> , 94 |
| <code>?`</code> , 34 | B | <code>\bigl</code> , 94 |
| <code>!`</code> , 34 | <code>\backepsilon(\epsilon)</code> , 78 | <code>\Bigm</code> , 95 |
| A | <code>\backmatter</code> , 42 | <code>\bigm</code> , 95 |
| <code>abstract</code> 환경, 48 | <code>\backprime(\prime)</code> , 86 | <code>\bigodot</code> , 75 |
| <code>\abstractname</code> , 48 | <code>\backsim(\sim)</code> , 78 | <code>\bigoplus</code> , 75 |
| <code>\acute</code> , 86 | <code>\backsimeq(\backsimeq)</code> , 78 | <code>\bigotimes</code> , 75 |
| <code>\addtocounter</code> , 180 | <code>\backslash(\backslash)</code> , 83, 86 | <code>\Bigr</code> , 94 |
| <code>\addtolength</code> , 186 | <code>\bar</code> , 86 | <code>\bigr</code> , 94 |
| <code>\aleph(\aleph)</code> , 82 | <code>\barwedge(\barwedge)</code> , 74 | <code>\bigskip</code> , 202 |
| <code>align</code> 환경, 104 | <code>\baselineskip</code> , 199 | <code>\bigsqcup</code> , 75 |
| <code>\Alph</code> , 180 | <code>\baselinestretch</code> , 199 | <code>\bigstar(\star)</code> , 86 |
| <code>\alph</code> , 180 | <code>\Bbbk(\Bbbk)</code> , 83 | <code>\bigtriangledown</code> , 74 |
| <code>\alpha(\alpha)</code> , 82 | <code>\because(\because)</code> , 78 | <code>\bigtriangleup</code> , 74 |
| <code>\amalg(\amalg)</code> , 74 | <code>\beta(\beta)</code> , 82 | <code>\biguplus</code> , 75 |
| <code>\and</code> , 39 | <code>\beth(\beth)</code> , 82 | <code>\bigvee</code> , 75 |
| <code>\angle(\angle)</code> , 86 | <code>\between(\between)</code> , 78 | <code>\bigwedge</code> , 75 |
| <code>\appendix</code> , 41 | <code>\bf</code> , 55, 90 | <code>\binom</code> , 71 |
| | <code>\bfseries</code> , 55 | <code>\blacklozenge</code> , 86 |

| | | |
|--|--|--|
| <code>\blacksquare</code> (■), 86 | <code>\circlearrowleft</code> (↺), 81 | <code>\curlyvee</code> (\curlyvee), 74 |
| <code>\blacktriangle</code> (▲), 86 | <code>\circlearrowright</code> (↻), 81 | <code>\curlywedge</code> (\curlywedge), 74 |
| <code>\blacktriangledown</code> (▼), 86 | <code>\circledast</code> (⊛), 74 | <code>\curvearrowleft</code> (↶), 81 |
| <code>\blacktriangleleft</code> (◄), 80 | <code>\circledcirc</code> (◎), 74 | <code>\curvearrowright</code> (↷), 81 |
| <code>\blacktriangleright</code> (►), 80 | <code>\circledR</code> (®), 83 | |
| <code>Bmatrix</code> 환경, 96 | <code>\circledS</code> (Ⓢ), 83 | D |
| <code>bmatrix</code> 환경, 96 | <code>\cite</code> , 158 | <code>\dagger</code> (†), 74 |
| <code>\bmod</code> , 77 | <code>\cleardoublepage</code> , 137 | <code>\daleth</code> (ℵ), 82 |
| <code>\boldsymbol</code> , 90 | <code>\clearpage</code> , 137 | <code>\dar</code> , 111 |
| <code>\bot</code> (⊥), 83 | <code>\cline</code> , 100, 130 | <code>\Dashcolon</code> (--:), 79 |
| <code>\bowtie</code> (⋈), 78 | <code>\clubsuit</code> (♣), 86 | <code>\dashcolon</code> (-:), 79 |
| <code>\Box</code> (□), 86 | <code>\colon</code> (:), 85 | <code>\dashleftarrow</code> (←--), 81 |
| <code>\boxdot</code> (⊠), 74 | <code>\Colonapprox</code> (::≈), 79 | <code>\dashrightarrow</code> (→--), 81 |
| <code>\boxminus</code> (⊖), 74 | <code>\colonapprox</code> (:≈), 79 | <code>\dashv</code> (⊥), 78 |
| <code>\boxplus</code> (⊕), 74 | <code>\Colondash</code> (::—), 79 | <code>\date</code> , 39 |
| <code>\boxtimes</code> (⊗), 74 | <code>\colondash</code> (:—), 79 | <code>\dbinom</code> , 71 |
| <code>\breve</code> , 86 | <code>\Coloneq</code> (::=), 79 | <code>\dblcolon</code> (::), 79 |
| <code>\bullet</code> (•), 74 | <code>\coloneq</code> (:=), 79 | <code>\ddagger</code> (‡), 74 |
| <code>\Bumpeq</code> (≈), 78 | <code>\Colonsim</code> (::~), 79 | <code>\ddddot</code> , 86 |
| <code>\bumpeq</code> (≐), 78 | <code>\colonsim</code> (:~), 79 | <code>\ddot</code> , 86 |
| | <code>\color</code> , 218 | <code>\ddots</code> (⋮), 85 |
| C | <code>\colorbox</code> , 219 | <code>\DeclareMathOperator</code> , 91 |
| <code>\cal</code> , 90 | <code>\colorlet</code> , 217 | <code>\DeclareMathOperator*</code> , 91 |
| <code>\Cap</code> (⋈), 74 | <code>\columnsep</code> , 197 | <code>\def</code> , 175 |
| <code>\cap</code> (∩), 74 | <code>\columnseprule</code> , 197 | <code>\definecolor</code> , 217 |
| <code>\caption</code> , 137 | <code>\columnwidth</code> , 197 | <code>\deg</code> (deg), 75 |
| <code>\caption*</code> , 137 | <code>\complement</code> (℄), 83 | <code>\Delta</code> (Δ), 82 |
| <code>cases</code> 환경, 97 | <code>\cong</code> (≅), 78 | <code>\delta</code> (δ), 82 |
| <code>\cdot</code> (·), 74 | <code>\contentsname</code> , 146 | <code>description</code> 환경, 46 |
| <code>\cdot</code> (·), 85 | <code>\coprod</code> (∏), 75 | <code>\det</code> (det), 75 |
| <code>\cdots</code> (⋯), 85 | <code>\cos</code> (cos), 75 | <code>\dfrac</code> , 70 |
| <code>Center</code> 환경, 51 | <code>\cosh</code> (cosh), 75 | <code>\diagdown</code> (↘), 86 |
| <code>center</code> 환경, 50 | <code>\cot</code> (cot), 75 | <code>\diagup</code> (↗), 86 |
| <code>\centerdot</code> (⋅), 74 | <code>\coth</code> (coth), 75 | <code>\Diamond</code> (◇), 86 |
| <code>\Centering</code> , 51 | <code>\counterwithin</code> , 182 | <code>\diamond</code> (◊), 74 |
| <code>\centering</code> , 50 | <code>\counterwithin*</code> , 183 | <code>\diamondsuit</code> (◇), 86 |
| <code>\cfrac</code> , 71 | <code>\counterwithout</code> , 182 | <code>\digamma</code> (ƒ), 82 |
| <code>\chapter</code> , 40 | <code>\counterwithout*</code> , 183 | <code>\dim</code> (dim), 75 |
| <code>\check</code> , 86 | <code>\csc</code> (csc), 75 | <code>displaymath</code> 환경, 67 |
| <code>\chi</code> (χ), 82 | <code>\Cup</code> (⊃), 74 | <code>\displaystyle</code> , 69 |
| <code>\choose</code> , 73 | <code>\cup</code> (∪), 74 | <code>\div</code> (÷), 74 |
| <code>\circ</code> (○), 74 | <code>\curlyeqprec</code> (≍), 78 | <code>\divideontimes</code> (⋈), 74 |
| <code>\circeq</code> (≐), 78 | <code>\curlyeqsucc</code> (≎), 78 | <code>\dler</code> , 111 |

document 환경, 17
`\documentclass`, 23
`\dot`, 86
`\dotemph`, 54
`\doteq`(\doteq), 78
`\doteqdot`(\doteqdot), 78
`\dotplus`($\dot{+}$), 74
`\dots`(\dots), 85
`\doublebarwedge`($\overline{\wedge}$), 74
`\Downarrow`(\Downarrow), 81, 83
`\downarrow`(\downarrow), 81, 83
`\downdownarrows`(\Downarrow), 81
`\downharpoonleft`(\lrcorner), 81
`\downharpoonright`(\rccorner), 81
`\drar`, 111

E

`\ell`(ℓ), 83
`\em`, 53
`\emph`, 53
`\emptyset`(\emptyset), 86
`\enspace`, 201
`\ensuremath`, 174
 enumerate 환경, 45
`\epsilon`(ϵ), 82
`\eqcirc`(\circ), 78
`\Eqcolon`(\equiv), 79
`\eqcolon`(\equiv), 79
 eqnarray 환경, 106
`\eqref`, 141
`\eqslantgtr`(\gtrless), 80
`\eqslantless`(\lessgtr), 80
 equation 환경, 67, 101
`\equiv`(\equiv), 78
`\eta`(η), 82
`\eth`(\eth), 86
`\evensidemargin`, 195
`\exists`(\exists), 83
`\exp`(\exp), 75

F

`\fallingdotseq`(\fallingdotseq), 78
`\fcolorbox`, 219
 figure 환경, 135
 figure* 환경, 198

`\figurename`, 137
`\Finv`(\Finv), 83
 FlushLeft 환경, 51
 flushleft 환경, 50
 FlushRight 환경, 51
 flushright 환경, 50
`\fnsymbol`, 180
`\footnote`, 42
`\footnotesize`, 56
`\forall`(\forall), 83
`\frac`, 70
 frame 환경, 225
`\frontmatter`, 42
`\frown`(\frown), 78

G

`\Game`(\Game), 83
`\Gamma`(Γ), 82
`\gamma`(γ), 82
`\gana`, 181
 gather 환경, 104
`\gcd`(\gcd), 75
`\ge`(\geq), 79
`\geq`(\geq), 79
`\geqq`(\geqq), 80
`\geqslant`(\geqslant), 80
`\gets`(\leftarrow), 81
`\gg`(\gg), 79
`\ggg`(\ggg), 80
`\gimel`(\gimel), 82
`\gnapprox`(\gtrapprox), 80
`\gneq`(\gtrneq), 80
`\gneqq`(\gtrneqq), 80
`\gnsim`(\gtrsim), 80
`\grave`, 86
`\gtrapprox`(\gtrapprox), 80
`\gtrdot`(\gtrdot), 80
`\gtreqless`(\gtrlessgtr), 80
`\gtreqqlless`(\gtrlessgtr), 80
`\gtrless`(\gtrless), 80
`\gtrsim`(\gtrsim), 80
`\guillemetleft`(\llcorner), 33
`\guillemetright`(\lrcorner), 33
`\guilsinglleft`(\langle), 33

`\guilsinglright`(\rangle), 33
`\gvertneqq`(\gtrneqq), 80

H

`\hanjabyhangulfont`, 212
`\hanjanum`, 181
`\hat`, 86
`\hbar`(\hbar), 83
`\heartsuit`(\heartsuit), 86
`\hfil`, 202
`\hfill`, 202
`\hline`, 100, 130
`\Hnum`, 181
`\hNum`, 181
`\hnum`, 181
`\hom`(\hom), 75
`\hookleftarrow`(\hookleftarrow), 81
`\hookrightarrow`(\hookrightarrow), 81
`\hRoman`, 181
`\hroman`, 181
`\hslash`(\hslash), 83
`\hspace`, 201
`\hspace*`, 201
`\Huge`, 56
`\huge`, 56
`\hyperlink`, 143
`\hypertarget`, 143

I

`\idotsint`($\int \dots \int$), 75
`\iff`(\iff), 80
`\iiint`(\iiint), 75
`\iint`(\iint), 75
`\iint`(\iint), 75
`\Im`(\Im), 83
`\imath`(\imath), 83
`\impliedby`(\impliedby), 80
`\implies`(\implies), 80
`\in`(\in), 83
`\include`, 190
`\includegraphics`, 133
`\includeonly`, 190
`\indent`, 199
`\indexname`, 167
`\inf`(\inf), 75

| | | |
|--|--|--|
| <code>\infty</code> (∞), 86 | <code>\leftleftarrows</code> (\Leftrightarrow), 81 | 81 |
| <code>\injlim</code> (inj lim), 75 | <code>\Leftrightarrow</code> (\Leftrightarrow), 81 | <code>\longmapsto</code> (\longmapsto), 81 |
| <code>\input</code> , 189 | <code>\leftrightharpoonup</code> (\leftrightharpoonup), 81 | <code>\Longrightarrow</code> (\Longrightarrow), 81 |
| <code>\int</code> (\int), 75 | <code>\leftrightsquigarrow</code> (\leftrightsquigarrow), 81 | <code>\longrightarrow</code> (\longrightarrow), 81 |
| <code>\intercal</code> (\intercal), 74 | <code>\leftthreetimes</code> (\leftthreetimes), 74 | <code>\looparrowleft</code> (\looparrowleft), 81 |
| <code>\intertext</code> , 105 | <code>\leq</code> (\leq), 79 | <code>\looparrowright</code> (\looparrowright), 81 |
| <code>\iota</code> (ι), 82 | <code>\leqq</code> (\leqq), 80 | <code>\lor</code> (\vee), 74 |
| <code>\it</code> , 55, 90 | <code>\leqslant</code> (\leqslant), 80 | <code>\lozenge</code> (\diamond), 86 |
| <code>\item</code> , 45 | <code>\lessapprox</code> (\lessapprox), 80 | <code>\lparen</code> ($()$), 83 |
| <code>itemize</code> 환경, 45 | <code>\lessdot</code> (\lessdot), 80 | <code>\Lsh</code> (\lsh), 81 |
| <code>\itshape</code> , 55 | <code>\lesseqgtr</code> (\lesseqgtr), 80 | <code>\ltimes</code> (\ltimes), 74 |
| J | <code>\lesseqgtr</code> (\lesseqgtr), 80 | <code>\lVert</code> (\lVert), 84 |
| <code>\jaso</code> , 181 | <code>\lessgtr</code> (\lessgtr), 80 | <code>\lvert</code> ($ $), 84 |
| <code>\jmath</code> (j), 83 | <code>\lessssim</code> (\lessssim), 80 | <code>\lvertneqq</code> (\lvertneqq), 80 |
| <code>\Join</code> (\Join), 78 | <code>\let</code> , 175 | M |
| <code>\justifying</code> , 51 | <code>\lfloor</code> (\lfloor), 83 | <code>\mainmatter</code> , 42 |
| K | <code>\lg</code> (\lg), 75 | <code>\makeindex</code> , 162 |
| <code>\kappa</code> (κ), 82 | <code>\lhd</code> (\lhd), 74 | <code>\maketitle</code> , 39 |
| <code>\ker</code> (\ker), 75 | <code>\lim</code> (\lim), 75 | <code>\mapsto</code> (\mapsto), 81 |
| L | <code>\liminf</code> (\liminf), 75 | <code>math</code> 환경, 67 |
| <code>\label</code> , 141 | <code>\limits</code> , 92 | <code>\mathbb</code> , 89 |
| <code>\Lambda</code> (Λ), 82 | <code>\limsup</code> (\limsup), 75 | <code>\mathbf</code> , 89 |
| <code>\lambda</code> (λ), 82 | <code>\listfigurename</code> , 146 | <code>\mathcal</code> , 89 |
| <code>\land</code> (\wedge), 74 | <code>\listoffigures</code> , 145 | <code>\mathcolor</code> , 219 |
| <code>\angle</code> (\angle), 83 | <code>\listoftables</code> , 145 | <code>\mathellipsis</code> (\dots), 85 |
| <code>\lar</code> , 111 | <code>\listtablename</code> , 146 | <code>\mathfrak</code> , 89 |
| <code>\LARGE</code> , 56 | <code>\ll</code> (\ll), 79 | <code>\mathit</code> , 89 |
| <code>\Large</code> , 56 | <code>\Lleftarrow</code> (\Lleftarrow), 81 | <code>\mathnormal</code> , 89 |
| <code>\large</code> , 56 | <code>\lll</code> (\lll), 80 | <code>\mathrm</code> , 89 |
| <code>\lbrace</code> ($\{$), 83 | <code>\ln</code> (\ln), 75 | <code>\mathsf</code> , 89 |
| <code>\lbrack</code> ($[$), 83 | <code>\lnapprox</code> (\lnapprox), 80 | <code>\mathtt</code> , 89 |
| <code>\lceil</code> (\lceil), 83 | <code>\lneq</code> (\lneq), 80 | <code>matrix</code> 환경, 96 |
| <code>\ldotp</code> (\ldotp), 85 | <code>\lneqq</code> (\lneqq), 80 | <code>\max</code> (\max), 75 |
| <code>\ldots</code> (\dots), 85 | <code>\lnot</code> (\neg), 86 | <code>\maxof</code> , 184 |
| <code>\le</code> (\leq), 79 | <code>\lnsim</code> (\lnsim), 80 | <code>\mdseries</code> , 55 |
| <code>\leadsto</code> (\leadsto), 81 | <code>\log</code> (\log), 75 | <code>\measuredangle</code> (\measuredangle), 86 |
| <code>\left</code> , 93 | <code>\Longleftarrow</code> (\Longleftarrow), 81 | <code>\medskip</code> , 202 |
| <code>\Leftarrow</code> (\Leftarrow), 81 | <code>\longleftarrow</code> (\longleftarrow), 81 | <code>\medspace</code> , 68 |
| <code>\leftarrow</code> (\leftarrow), 81 | <code>\Longleftrightarrow</code> (\Longleftrightarrow), 81 | <code>\mho</code> (\mho), 86 |
| <code>\leftarrowtail</code> (\leftarrowtail), 81 | <code>\longleftarrowtail</code> (\longleftarrowtail), 81 | <code>\mid</code> ($ $), 78 |
| <code>\leftharpoondown</code> (\leftharpoondown), 81 | <code>\longleftrightarrow</code> (\longleftrightarrow), 81 | <code>\middle</code> , 94 |
| <code>\leftharpoonup</code> (\leftharpoonup), 81 | | <code>\min</code> (\min), 75 |

| | | |
|--|--|--|
| <code>\minof</code> , 184 | <code>\nleltrightarrow</code> (\nleftrightarrow), 81 | <code>\ojaso</code> , 181 |
| <code>\mit</code> , 90 | <code>\nleq</code> (\nleq), 80 | <code>\oldstylenums</code> , 36 |
| <code>\mod</code> , 77 | <code>\nleqq</code> (\nleqq), 80 | <code>\Omega</code> (Ω), 82 |
| <code>\models</code> (\models), 78 | <code>\nleqslant</code> (\nleqslant), 80 | <code>\omega</code> (ω), 82 |
| <code>\mp</code> (\mp), 74 | <code>\nless</code> (\nless), 80 | <code>\ominus</code> (\ominus), 74 |
| <code>\mspace</code> , 202 | <code>\nmid</code> (\nmid), 78 | <code>\onecolumn</code> , 196 |
| <code>\mu</code> (μ), 82 | <code>\nocite</code> , 158 | <code>\onum</code> , 181 |
| <code>\multicols</code> 환경, 198 | <code>\noindent</code> , 199 | <code>\operatorname</code> , 91 |
| <code>\multicolumn</code> , 127 | <code>\nolimits</code> , 92 | <code>\operatorname*</code> , 91 |
| <code>\multimap</code> (\multimap), 78 | <code>\nonumber</code> , 103 | <code>\oplus</code> (\oplus), 74 |
| <code>\multirow</code> , 128 | <code>\normalfont</code> , 55 | <code>\oslash</code> (\oslash), 74 |
| N | <code>\normalsize</code> , 56 | <code>\otimes</code> (\otimes), 74 |
| <code>\nabla</code> (∇), 86 | <code>\not</code> , 78 | <code>\over</code> , 73 |
| <code>\ncong</code> (\ncong), 78 | <code>\nparallel</code> (\nparallel), 78 | <code>\overbrace</code> , 87 |
| <code>\ne</code> (\neq), 79 | <code>\nprec</code> (\nprec), 78 | <code>\overbracket</code> , 87 |
| <code>\nearrow</code> (\nearrow), 81 | <code>\npreceq</code> (\npreceq), 78 | <code>\overleftarrow</code> , 87 |
| <code>\neg</code> (\neg), 86 | <code>\nrightarrow</code> (\nrightarrow), 81 | <code>\overleftrightharrow</code> , 87 |
| <code>\negmedspace</code> , 68 | <code>\nshortmid</code> (\nshortmid), 78 | <code>\overline</code> , 87 |
| <code>\negthickspace</code> , 68 | <code>\nshortparallel</code> (\nshortparallel), 78 | <code>\overrightarrow</code> , 87 |
| <code>\negthinspace</code> , 68 | <code>\nsim</code> (\sim), 78 | <code>\owns</code> (\owns), 83 |
| <code>\neq</code> (\neq), 79 | <code>\nsubseteq</code> (\nsubseteq), 79 | P |
| <code>\newcolumn</code> , 198 | <code>\nsubseteqq</code> (\nsubseteqq), 79 | <code>\pageref</code> , 141 |
| <code>\newcommand</code> , 172 | <code>\nsucc</code> (\nsucc), 78 | <code>\paperheight</code> , 195 |
| <code>\newcommand*</code> , 172 | <code>\nsucceq</code> (\nsucceq), 78 | <code>\paperwidth</code> , 195 |
| <code>\NewCommandCopy</code> , 175 | <code>\nsupseteq</code> (\nsupseteq), 79 | <code>\par</code> , 18 |
| <code>\newcounter</code> , 179 | <code>\nsupseteqq</code> (\nsupseteqq), 79 | <code>\paragraph</code> , 40 |
| <code>\NewDocumentCommand</code> , 175 | <code>\ntriangleleft</code> (\ntriangleleft), 80 | <code>\parallel</code> (\parallel), 78 |
| <code>\newenvironment</code> , 178 | <code>\ntrianglelefteq</code> (\ntrianglelefteq), 80 | <code>\parindent</code> , 199 |
| <code>\newenvironment*</code> , 178 | <code>\ntriangleright</code> (\ntriangleright), 80 | <code>\parskip</code> , 199 |
| <code>\newlength</code> , 185 | <code>\ntrianglerighteq</code> (\ntrianglerighteq), 80 | <code>\part</code> , 40 |
| <code>\newpage</code> , 197 | <code>\nu</code> (ν), 82 | <code>\partial</code> (∂), 83 |
| <code>\newtheorem</code> , 59 | <code>\nVDash</code> (\nVDash), 78 | <code>\peng</code> , 181 |
| <code>\newtheorem*</code> , 59 | <code>\nvDash</code> (\nvDash), 78 | <code>\perp</code> (\perp), 78 |
| <code>\newtheoremstyle</code> , 62 | <code>\nvdash</code> (\nvdash), 78 | <code>\pgana</code> , 181 |
| <code>\nexists</code> (\nexists), 83 | <code>\nwarrow</code> (\nwarrow), 81 | <code>\Phi</code> (Φ), 82 |
| <code>\ngeq</code> (\ngeq), 80 | O | <code>\phi</code> (ϕ), 82 |
| <code>\ngeqq</code> (\ngeqq), 80 | <code>\oddsidemargin</code> , 195 | <code>\Pi</code> (Π), 82 |
| <code>\ngeqslant</code> (\ngeqslant), 80 | <code>\odot</code> (\odot), 74 | <code>\pi</code> (π), 82 |
| <code>\ngtr</code> (\ngtr), 80 | <code>\oeng</code> , 181 | <code>\pitchfork</code> (\pitchfork), 78 |
| <code>\ni</code> (\ni), 83 | <code>\of</code> , 73 | <code>\pjaso</code> , 181 |
| <code>\nLeftarrow</code> (\nLeftarrow), 81 | <code>\ogana</code> , 181 | <code>\pm</code> (\pm), 74 |
| <code>\nleftarrow</code> (\nleftarrow), 81 | <code>\oint</code> (\oint), 75 | <code>\pmatrix</code> 환경, 96 |
| <code>\nLeftrightarrow</code> (\nLeftrightarrow), 81 | | <code>\pmd</code> , 90 |

| | | |
|---|--|--|
| <code>\pmod</code> , 77 | <code>\refstepcounter</code> , 180 | <code>\setcounter</code> , 180 |
| <code>\pnum</code> , 181 | <code>\renewcommand</code> , 175 | <code>\setlength</code> , 186 |
| <code>\pod</code> , 77 | <code>\renewcommand*</code> , 175 | <code>\setlist</code> , 47 |
| <code>\Pr(Pr)</code> , 75 | <code>\RenewCommandCopy</code> , 175 | <code>\setmainfont</code> , 211 |
| <code>\prec(<)</code> , 78 | <code>\RenewDocumentCommand</code> ,
175 | <code>\setminus()</code> , 74 |
| <code>\precapprox(\simeq)</code> , 78 | <code>\renewenvironment</code> , 178 | <code>\setmonofont</code> , 211 |
| <code>\preccurlyeq(\lessapprox)</code> , 78 | <code>\renewenvironment*</code> , 178 | <code>\setsansfont</code> , 211 |
| <code>\preceq(\leq)</code> , 78 | <code>\rfloor()</code> , 83 | <code>\sf</code> , 55, 90 |
| <code>\precnapprox(\lessapprox)</code> , 78 | <code>\rhd(>)</code> , 74 | <code>\sffamily</code> , 55 |
| <code>\precnsim(\lessapprox)</code> , 78 | <code>\rho(\rho)</code> , 82 | <code>\shortintertext</code> , 105 |
| <code>\precsim(\lesssim)</code> , 78 | <code>\right</code> , 93 | <code>\shortmid()</code> , 78 |
| <code>\prime</code> , 72, 86 | <code>\Rightarrow(\Rightarrow)</code> , 81 | <code>\shortparallel()</code> , 78 |
| <code>\printindex</code> , 162 | <code>\rightarrow(\rightarrow)</code> , 81 | <code>\Sigma(\Sigma)</code> , 82 |
| <code>\prod(\prod)</code> , 75 | <code>\rightarrowtail(\rightarrowtail)</code> , 81 | <code>\sigma(\sigma)</code> , 82 |
| <code>\projlim(projlim)</code> , 75 | <code>\rightharpoonowdown(\rightharpoonowdown)</code> , 81 | <code>\sim(\sim)</code> , 78 |
| proof 환경, 63 | <code>\rightharpoonup(\rightharpoonup)</code> , 81 | <code>\Simcolon(\sim::)</code> , 79 |
| <code>\proofname</code> , 63 | <code>\rightleftarrows(\rightleftarrows)</code> , 81 | <code>\simcolon(\sim::)</code> , 79 |
| <code>\propto(\propto)</code> , 78 | <code>\rightleftharpoons(\rightleftharpoons)</code> , 81 | <code>\simeq(\simeq)</code> , 78 |
| <code>\Psi(\Psi)</code> , 82 | <code>\rightrightarrows(\rightrightarrows)</code> , 81 | <code>\sin(\sin)</code> , 75 |
| <code>\psi(\psi)</code> , 82 | <code>\rightsquigarrow(\rightsquigarrow)</code> , 81 | <code>\sinh(\sinh)</code> , 75 |
| Q | <code>\rightthreetimes(\rightthreetimes)</code> , 74 | <code>\sl</code> , 55 |
| <code>\qed</code> , 63 | <code>\risingdotseq(\risingdotseq)</code> , 78 | <code>\slshape</code> , 55 |
| <code>\qedhere</code> , 63 | <code>\rm</code> , 55, 90 | <code>\small</code> , 56 |
| <code>\qedsymbol</code> , 64 | <code>\rmfamily</code> , 55 | <code>\smallfrown(\smallfrown)</code> , 78 |
| <code>\quad</code> , 68, 201 | <code>\Roman</code> , 180 | smallmatrix 환경, 97 |
| <code>\quad</code> , 68, 201 | <code>\roman</code> , 180 | <code>\smallsetminus()</code> , 74 |
| quotation 환경, 49 | <code>\root</code> , 73 | <code>\smallskip</code> , 202 |
| quote 환경, 49 | <code>\rparen()</code> , 83 | <code>\smallsmile(\smallsmile)</code> , 78 |
| <code>\quotedblbase()</code> , 33 | <code>\Rsh()</code> , 81 | <code>\smile(\smile)</code> , 78 |
| <code>\quotesinglbase()</code> , 33 | <code>\rtimes(\rtimes)</code> , 74 | <code>\sp</code> , 72 |
| R | <code>\rule</code> , 203 | <code>\space</code> , 18 |
| <code>\RaggedLeft</code> , 51 | <code>\rVert()</code> , 84 | <code>\spadesuit(\spadesuit)</code> , 86 |
| <code>\raggedleft</code> , 50 | <code>\rvert()</code> , 84 | <code>\sphericalangle(\sphericalangle)</code> , 86 |
| <code>\RaggedRight</code> , 51 | S | <code>\sqcap()</code> , 74 |
| <code>\raggedright</code> , 50 | <code>\sb</code> , 72 | <code>\sqcup()</code> , 74 |
| <code>\rangle()</code> , 83 | <code>\sc</code> , 55 | <code>\sqrt</code> , 73 |
| <code>\rar</code> , 111 | <code>\scriptscriptstyle</code> , 69 | <code>\sqsubset(\sqsubset)</code> , 79 |
| <code>\rbrace()</code> , 83 | <code>\scriptsize</code> , 56 | <code>\sqsubseteq(\sqsubseteq)</code> , 79 |
| <code>\rbrack()</code> , 83 | <code>\scriptstyle</code> , 69 | <code>\square(\square)</code> , 86 |
| <code>\rceil()</code> , 83 | <code>\scshape</code> , 55 | <code>\stackbin</code> , 88 |
| <code>\Re(\Re)</code> , 83 | <code>\searrow(\searrow)</code> , 81 | <code>\stackrel</code> , 88 |
| <code>\ref</code> , 141 | <code>\sec(sec)</code> , 75 | <code>\star(*)</code> , 74 |
| <code>\refname</code> , 160 | <code>\section</code> , 40 | <code>\stepcounter</code> , 180 |

| | | |
|-------------------------------------|---|---|
| subequations 환경, 103 | <code>\text</code> , 69 | <code>\texteightoldstyle(8)</code> , 36 |
| <code>\subparagraph</code> , 40 | <code>\textacutedbl(")</code> , 37 | <code>\textellipsis(...)</code> , 35 |
| <code>\subsection</code> , 40 | <code>\textasciicute(')</code> , 37 | <code>\textemdash(—)</code> , 35 |
| <code>\Subset(⊆)</code> , 79 | <code>\textasciibreve(˘)</code> , 37 | <code>\textendash(–)</code> , 35 |
| <code>\subset(⊂)</code> , 79 | <code>\textasciicaron(ˇ)</code> , 37 | <code>\textestimated(ℰ)</code> , 37 |
| <code>\subseteq(⊆)</code> , 79 | <code>\textasciicircum(ˆ)</code> , 18, 37 | <code>\texteuro(€)</code> , 35 |
| <code>\subseteqq(⊆)</code> , 79 | | <code>\textexclamdown</code> , 34, 35 |
| <code>\subsetneq(⊂)</code> , 79 | <code>\textasciidieresis(¨)</code> , 37 | <code>\textfiveoldstyle(5)</code> , 36 |
| <code>\subsetneqq(⊂)</code> , 79 | <code>\textasciigrave(`)</code> , 37 | <code>\textflorin(f)</code> , 35 |
| <code>\substack</code> , 93 | <code>\textasciimacron(ˉ)</code> , 37 | <code>\textfouroldstyle(4)</code> , 36 |
| <code>\subsubsection</code> , 40 | <code>\textasciitilde(~)</code> , 18, 37 | <code>\textfractionsolidus(/)</code> , 36 |
| <code>\succ(⋗)</code> , 78 | <code>\textbackslash(\)</code> , 18, 35 | <code>\textgravedbl(˘)</code> , 37 |
| <code>\succapprox(⋗)</code> , 78 | <code>\textbaht(฿)</code> , 35 | <code>\textgreater(>)</code> , 36 |
| <code>\succcurlyeq(⋗)</code> , 78 | <code>\textbar()</code> , 37 | <code>\textguarani(₲)</code> , 35 |
| <code>\succeq(⋗)</code> , 78 | <code>\textbardbl(‖)</code> , 37 | <code>\textheight</code> , 195 |
| <code>\succnapprox(⋗)</code> , 78 | <code>\textbf</code> , 55 | <code>\textinterrobang(‽)</code> , 35 |
| <code>\succnsim(⋗)</code> , 78 | <code>\textbigcircle(⊙)</code> , 37 | <code>\textinterrobangdown(‽)</code> , 35 |
| <code>\succsim(⋗)</code> , 78 | <code>\textblank(b)</code> , 37 | <code>\textit</code> , 55 |
| <code>\sum(∑)</code> , 75 | <code>\textborn(✶)</code> , 36 | <code>\textlangle(⟨)</code> , 36 |
| <code>\sup(sup)</code> , 75 | <code>\textbraceleft(⟨)</code> , 36 | <code>\textlbrackdbl(⌋)</code> , 36 |
| <code>\Supset(⊇)</code> , 79 | <code>\textbraceright(⟩)</code> , 36 | <code>\textleaf(♻)</code> , 36 |
| <code>\supset(⊃)</code> , 79 | <code>\textbrokenbar(⌋)</code> , 37 | <code>\textleftarrow(←)</code> , 35 |
| <code>\supseteq(⊃)</code> , 79 | <code>\textbullet(•)</code> , 37 | <code>\textless(<)</code> , 36 |
| <code>\supseteqq(⊃)</code> , 79 | <code>\textcent(¢)</code> , 35 | <code>\textlira(₺)</code> , 35 |
| <code>\supsetneq(⊃)</code> , 79 | <code>\textcentoldstyle(c)</code> , 35 | <code>\textlnot(¬)</code> , 36 |
| <code>\supsetneqq(⊃)</code> , 79 | <code>\textcircledP(Ⓟ)</code> , 35 | <code>\textlquill(⌋)</code> , 36 |
| <code>\surd(√)</code> , 86 | <code>\textcolonmonetary(₯)</code> , 35 | <code>\textmarried(∞)</code> , 36 |
| <code>\swapnumbers</code> , 61 | <code>\textcolor</code> , 218 | <code>\textmd</code> , 55 |
| <code>\swarrow(↘)</code> , 81 | <code>\textcopleft(⊙)</code> , 35 | <code>\textmho(℧)</code> , 37 |
| T | <code>\textcopyright(©)</code> , 35 | <code>\textminus(−)</code> , 33, 36 |
| table 환경, 135 | <code>\textcurrency(₹)</code> , 35 | <code>\textmu(μ)</code> , 37 |
| table* 환경, 198 | <code>\textdagger(†)</code> , 37 | <code>\textmusicalnote(♩)</code> , 37 |
| <code>\tablename</code> , 137 | <code>\textdaggerdbl(‡)</code> , 37 | <code>\textnaira(₦)</code> , 35 |
| <code>\tableofcontents</code> , 145 | <code>\textdblhyphen(=)</code> , 37 | <code>\textnineoldstyle(9)</code> , 36 |
| tabular 환경, 119, 121 | <code>\textdegree(°)</code> , 37 | <code>\textnormal</code> , 55 |
| tabular* 환경, 130 | <code>\textdied(†)</code> , 36 | <code>\textnumero(Nº)</code> , 37 |
| tabularx 환경, 131 | <code>\textdiscount(ℒ)</code> , 37 | <code>\textohm(Ω)</code> , 37 |
| <code>\tag</code> , 102 | <code>\textdiv(÷)</code> , 36 | <code>\textonehalf(½)</code> , 36 |
| <code>\tag*</code> , 102 | <code>\textdivorced(⚭)</code> , 36 | <code>\textoneoldstyle(1)</code> , 36 |
| <code>\tan(tan)</code> , 75 | <code>\textdollar(\$)</code> , 35 | <code>\textonequarter(¼)</code> , 36 |
| <code>\tanh(tanh)</code> , 75 | <code>\textdollaroldstyle(\$)</code> , 35 | <code>\textonesuperior(¹)</code> , 36 |
| <code>\tau(τ)</code> , 82 | | <code>\textopenbullet(◦)</code> , 37 |
| <code>\tbinom</code> , 71 | <code>\textdong(₫)</code> , 35 | |
| | <code>\textdownarrow(↓)</code> , 35 | |

| | | |
|---|---|--|
| <code>\textordfeminine</code> (^a), 35 | <code>\textthreequarters</code> (³ / ₄), 36 | <code>\triangleq</code> (\triangleq), 80 |
| <code>\textordmasculine</code> (^o), 35 | <code>\textthreequartersemdash</code> (—), 35 | <code>\triangleright</code> (\triangleright), 74 |
| <code>\textparagraph</code> (¶), 37 | <code>\textthreesuperior</code> (³), 36 | <code>\trianglerighteq</code> (\trianglerighteq), 80 |
| <code>\textperiodcentered</code> (·), 35 | <code>\texttildelow</code> (\sim), 37 | <code>\tt</code> , 55, 90 |
| <code>\textpertenthousand</code> (‰), 37 | <code>\texttimes</code> (\times), 36 | <code>\ttfamily</code> , 55 |
| <code>\textperthousand</code> (‰), 37 | <code>\texttrademark</code> (™), 35 | <code>\twocolumn</code> , 196 |
| <code>\textpeso</code> (₱), 35 | <code>\texttt</code> , 55 | <code>\twoheadleftarrow</code> (\twoheadleftarrow), 81 |
| <code>\textpilcrow</code> (¶), 37 | <code>\texttwelveudash</code> (—), 35 | <code>\twoheadrightarrow</code> (\twoheadrightarrow), 81 |
| <code>\textpm</code> (±), 36 | <code>\texttwooldstyle</code> (2), 36 | |
| <code>\textquestiondown</code> , 34, 35 | <code>\texttwosuperior</code> (²), 36 | U |
| <code>\textquotedbl</code> ("), 35 | <code>\textunderscore</code> ($_$), 37 | <code>\uar</code> , 111 |
| <code>\textquotedblleft</code> (“), 35 | <code>\textup</code> , 55 | <code>\ular</code> , 111 |
| <code>\textquotedblright</code> (”), 35 | <code>\textuparrow</code> (↑), 35 | <code>\underbrace</code> , 87 |
| <code>\textquoteleft</code> (‘), 35 | <code>\textvisiblespace</code> ($_$), 37 | <code>\underbracket</code> , 87 |
| <code>\textquoteright</code> (’), 35 | <code>\textwidth</code> , 195 | <code>\underleftarrow</code> , 87 |
| <code>\textquotesingle</code> (‘), 35 | <code>\textwon</code> (₩), 35 | <code>\underleftrightharrow</code> , 87 |
| <code>\textquotestraightbase</code> (,), 35 | <code>\textyen</code> (¥), 35 | <code>\underline</code> , 53, 87 |
| <code>\textquotestraightdblbase</code> (,,), 35 | <code>\textzerooldstyle</code> (o), 36 | <code>\underrightarrow</code> , 87 |
| <code>\texttriangle()</code> , 36 | <code>\tfrac</code> , 70 | <code>\unlhd</code> (\triangleleft), 74 |
| <code>\textrbrackdbl</code> (⌋), 36 | <code>\thanks</code> , 39 | <code>\unrhd</code> (\triangleright), 74 |
| <code>\textrecipe</code> (R), 37 | <code>\the</code> , 186 | <code>\Uparrow</code> (↑), 81, 83 |
| <code>\textreferencemark</code> (※), 37 | <code>\the<counter></code> , 182 | <code>\uparrow</code> (↑), 81, 83 |
| <code>\textregistered</code> (®), 35 | <code>\theoremstyle</code> , 62 | <code>\Updownarrow</code> (↑↓), 81, 83 |
| <code>\textrightarrow</code> (→), 35 | <code>\therefore</code> (\therefore), 78 | <code>\updownarrow</code> (↑↓), 81, 83 |
| <code>\textrm</code> , 55 | <code>\Theta</code> (Θ), 82 | <code>\upharpoonleft</code> (⌈), 81 |
| <code>\textrquill</code> (}), 36 | <code>\theta</code> (θ), 82 | <code>\upharpoonright</code> (⌋), 81 |
| <code>\textsc</code> , 55 | <code>\thickapprox</code> (\approx), 78 | <code>\uplus</code> (\uplus), 74 |
| <code>\textsection</code> (§), 37 | <code>\thicksim</code> (\sim), 78 | <code>\upshape</code> , 55 |
| <code>\textservicemark</code> (SM), 35 | <code>\thickspace</code> , 68 | <code>\Upsilon</code> (Υ), 82 |
| <code>\textsevenoldstyle</code> (7), 36 | <code>\thinspace</code> , 68 | <code>\upsilon</code> (υ), 82 |
| <code>\textsf</code> , 55 | <code>\tilde</code> , 86 | <code>\upuparrows</code> (↑↑), 81 |
| <code>\textsixoldstyle</code> (6), 36 | <code>\times</code> (\times), 74 | <code>\urar</code> , 111 |
| <code>\textsl</code> , 55 | <code>\tiny</code> , 56 | <code>\usepackage</code> , 17 |
| <code>\textsterling</code> (£), 35 | <code>\title</code> , 39 | |
| <code>\textstyle</code> , 69 | <code>titlepage</code> 환경, 40 | V |
| <code>\textsubscript</code> , 57 | <code>\to</code> (→), 81 | <code>\value</code> , 182 |
| <code>\textsuperscript</code> , 36, 57 | <code>\top</code> (T), 83 | <code>\varDelta</code> (Δ), 82 |
| <code>\textsurd</code> (√), 36 | <code>\topmargin</code> , 195 | <code>\varepsilon</code> (ε), 82 |
| <code>\textthreeoldstyle</code> (3), 36 | <code>\triangle</code> (Δ), 86 | <code>\varGamma</code> (Γ), 82 |
| | <code>\triangledown</code> (▽), 86 | <code>\varinjlim</code> (\varinjlim), 75 |
| | <code>\triangleleft</code> (\triangleleft), 74 | <code>\varkappa</code> (κ), 82 |
| | <code>\trianglelefteq</code> (\trianglelefteq), 80 | <code>\varLambda</code> (Λ), 82 |
| | | <code>\varliminf</code> (\varliminf), 75 |
| | | <code>\varlimsup</code> (\varlimsup), 75 |
| | | <code>\varnothing</code> (∅), 86 |

$\backslash\mathrm{varOmega}(\Omega)$, 82
 $\backslash\mathrm{varPhi}(\Phi)$, 82
 $\backslash\mathrm{varphi}(\varphi)$, 82
 $\backslash\mathrm{varPi}(\Pi)$, 82
 $\backslash\mathrm{varpi}(\varpi)$, 82
 $\backslash\mathrm{varprojlim}(\varprojlim)$, 75
 $\backslash\mathrm{varpropto}(\propto)$, 78
 $\backslash\mathrm{varPsi}(\Psi)$, 82
 $\backslash\mathrm{varrho}(\varrho)$, 82
 $\backslash\mathrm{varSigma}(\Sigma)$, 82
 $\backslash\mathrm{varsubsetneq}(\subsetneq)$, 79
 $\backslash\mathrm{varsubsetneqq}(\subsetneqq)$, 79
 $\backslash\mathrm{varsupsetneq}(\supsetneq)$, 79
 $\backslash\mathrm{varsupsetneqq}(\supsetneqq)$, 79
 $\backslash\mathrm{varTheta}(\Theta)$, 82
 $\backslash\mathrm{vartheta}(\vartheta)$, 82
 $\backslash\mathrm{vartriangle}(\triangle)$, 86
 $\backslash\mathrm{vartriangleleft}(\triangleleft)$, 80
 $\backslash\mathrm{vartriangleright}(\triangleright)$, 80
 $\backslash\mathrm{varUpsilon}(\Upsilon)$, 82
 $\backslash\mathrm{varXi}(\Xi)$, 82
 $\backslash\mathrm{vcentcolon}(:)$, 79
 $\backslash\mathrm{Vdash}(\Vdash)$, 78
 $\backslash\mathrm{VDash}(\=)$, 78
 $\backslash\mathrm{vdash}(\vdash)$, 78
 $\backslash\mathrm{vdots}(\vdots)$, 85
 $\backslash\mathrm{vec}$, 86
 $\backslash\mathrm{vee}(\vee)$, 74
 $\backslash\mathrm{veebar}(\veebar)$, 74
 $\backslash\mathrm{verb}$, 58

$\backslash\mathrm{verb*}$, 58
 $\mathrm{verbatim}$ 환경, 57
 $\mathrm{verbatim*}$ 환경, 58
 verse 환경, 48
 $\backslash\mathrm{Vert}(\|)$, 83
 $\backslash\mathrm{vert}()$, 83
 $\mathrm{Vmatrix}$ 환경, 96
 $\mathrm{vmatrix}$ 환경, 96
 $\backslash\mathrm{vspace}$, 202
 $\backslash\mathrm{vspace*}$, 202
 $\backslash\mathrm{Vvdash}(\Vvdash)$, 78

W

$\backslash\mathrm{wedge}(\wedge)$, 74
 $\backslash\mathrm{widehat}$, 87
 $\backslash\mathrm{widetilde}$, 87
 $\backslash\mathrm{widthof}$, 187
 $\backslash\mathrm{wp}(\wp)$, 83
 $\backslash\mathrm{wr}()$, 74

X

$\backslash\mathrm{xhookleftarrow}$, 88
 $\backslash\mathrm{xhookrightarrow}$, 88
 $\backslash\mathrm{Xi}(\Xi)$, 82
 $\backslash\mathrm{xi}(\xi)$, 82
 $\backslash\mathrm{xLeftarrow}$, 88
 $\backslash\mathrm{xleftarrow}$, 88
 $\backslash\mathrm{xleftharpoonup}$, 88
 $\backslash\mathrm{xleftharpoonup}$, 88
 $\backslash\mathrm{xLeftrightarrow}$, 88

$\backslash\mathrm{xletrightarrow}$, 88
 $\backslash\mathrm{xleftrightharpoons}$, 88
 $\backslash\mathrm{xmapsto}$, 88
 $\backslash\mathrm{xrightarrow}$, 88
 $\backslash\mathrm{xrightarrow}$, 88
 $\backslash\mathrm{xrightharpoonup}$, 88
 $\backslash\mathrm{xrightharpoonup}$, 88
 $\backslash\mathrm{xrightleftharpoons}$, 88

Z

$\backslash\mathrm{zeta}(\zeta)$, 82

가

$\backslash\mathrm{가}$, 143
 $\backslash\mathrm{과}$, 143

ㄴ

$\backslash\mathrm{ㄴ}$, 143

ㄹ

$\backslash\mathrm{라}$, 143
 $\backslash\mathrm{로}$, 143
 $\backslash\mathrm{를}$, 143

ㅇ

$\backslash\mathrm{와}$, 143
 $\backslash\mathrm{으로}$, 143
 $\backslash\mathrm{은}$, 143
 $\backslash\mathrm{을}$, 143
 $\backslash\mathrm{이}$, 143
 $\backslash\mathrm{이라}$, 143

패키지 찾아보기

A

amsmath 패키지, 70
amssymb 패키지, 70
amsthm 패키지, 59
array 패키지, 121

B

babel 패키지, 205

C

calc 패키지, 184, 186
caption 패키지, 137
ctex 패키지, 207

E

enumitem 패키지, 46

F

fontenc 패키지, 31
fontspec 패키지, 210

G

geometry 패키지, 196

graphicx 패키지, 133

H

helvet 패키지, 209
hyperref 패키지, 143

I

inputenc 패키지, 32

K

kotex 패키지, 206

L

luatexja 패키지, 208

M

makeidx 패키지, 162
mathtools 패키지, 70
multicol 패키지, 198
multirow 패키지, 128

N

newpxmath 패키지, 209

newpxtext 패키지, 209

newtxmath 패키지, 209

newtxtext 패키지, 209

P

plautopatch 패키지, 208
polyglossia 패키지, 205, 206

R

ragged2e 패키지, 51

S

subcaption 패키지, 137

T

tabularx 패키지, 131
textcomp 패키지, 37
tikz-cd 패키지, 109
titlesec 패키지, 41
titletoc 패키지, 146

X

xcolor 패키지, 215