

# 한국과학영재학교 문맹퇴치운동

KSA를 위한 LATEX 초 입문자료

강병지, 김재호

2017년 2월 21일

version 1.0.0

## 차 례

차 례 . . . . .	1
1 LATEX 이해하기 . . . . .	2
1.1 LATEX이란 무엇인가요? . . . . .	2
1.2 왜 LATEX을 사용하나요? . . . . .	2
2 LATEX 시작하기 . . . . .	4
2.1 LATEX 설치하기 . . . . .	4
2.2 기본 사항 . . . . .	5
2.3 문제가 생기셨나요? . . . . .	8
3 문서 작성 시작하기 . . . . .	10
3.1 예시 코드 구경하기 . . . . .	10
3.2 글의 구조 . . . . .	12
3.3 Preamble의 기능 . . . . .	12
3.4 명령어 알아보기 . . . . .	13
3.5 문제가 생기셨나요?? . . . . .	13
4 좀 더 구체적으로 . . . . .	14
4.1 documentclass에 대하여 . . . . .	14
4.2 유용한 package 알아보기 . . . . .	15
4.3 유용한 환경 알아보기 . . . . .	16
4.4 새로운 명령 정의하기 . . . . .	17
4.5 수식 작성하기 . . . . .	18
4.6 표와 그림 . . . . .	23
4.7 문제가 생기셨나요? . . . . .	26
5 Appendix . . . . .	28
5.1 documentclass . . . . .	28
5.2 수식 모드 공백 기호 . . . . .	29
5.3 글 위, 아래 기호 . . . . .	29
5.4 표 열의 내용 정렬하기 . . . . .	29

## 들어가며

이 자료는 한국과학영재학교 학생을 위한 LATEX 멘토링 자료입니다. 입문자를 위해 설계되었으며, LATEX를 사용하는데 실제로 자주 사용되는 내용 위주로 구성하려고 노력했습니다. 내용의 흐름을 위해 자주 사용되지 않는 기능들에 대한 설명은 언급하지 않았습니다만, 그렇지 않더라도 LATEX의 수많은 기능들을 모두 언급할 수는 없는 점 이해해주셨으면 합니다. 이 글을 읽고 LATEX의 기본적인 사용 방법을 익히시고, 따로 필요한 기능들은 그때 그때 보다 전문적인 매뉴얼<sup>1</sup>을 참조하시거나, 인터넷에 검색하시며 사용하시면 됩니다. 이 글이 LATEX에 대한 진입장벽을 낮추는 데 도움이 되기를 기원합니다. 궁금한 점이 있으시면 ufolion@gmail.com이나 oojahooo@gmail.com로 연락해 주시면 답변 드리겠습니다.

## 1 LATEX 이해하기

### 1.1 LATEX이란 무엇인가요?

LATEX<sup>2</sup>은 간단히 말해서 문서 작성 도구 중 하나입니다. Microsoft Word나 한컴오피스 한글과 같이 우리가 컴퓨터상에서, 혹은 인쇄된 조판물로 보는 문서를 작성하기 위해 있는 시스템이라고 생각하시면 됩니다.

대신 앞서 언급한 워드, 한글과는 성격이 조금 다릅니다. 워드, 한글과 같은 프로그램은 WYSIWYG 편집기라고 부릅니다. WYSIWYG란 What You See Is What You Get, 작성 중인 그대로 화면에 띠워서, 어떤 식으로 출력될지 보면서 편집하는 프로그램을 말합니다. 반면에 LATEX은 문서 작성과 결과 확인이 분리되어 있습니다. 파일 프로그래밍을 하듯이 문서에 대한 코드를 짜고, 이를 LATEX 프로그램에 넣어서 실행시키면 우리가 볼 수 있는 PDF파일이 나온다고 생각하면 됩니다. 문서를 프로그래밍한다고 생각하셔도 되겠습니다.

문서를 어떻게 프로그래밍하느냐고요? 예를 들어 생각해봅시다. 우리가 워드프로세서에서 글을 크게 만들기 위해선 그 글을 드래그하고, 글씨를 키우는 버튼을 누르면 됩니다. 레이텍에서는 그 글 앞에 글씨를 크게하는 명령어를 붙이면 됩니다. \large Hello World! 처럼요. 예상하셨겠지만 여기서는 \large가 글씨를 크게 하는 명령어입니다.

### 1.2 왜 LATEX을 사용하나요?

그런데 1.1장만 읽고 나면 왜 LATEX을 사용해야 하는지 의문이 들 것입니다. 워드프로세서로 보면서 만들면 되는데 왜 문서를 코딩까지 해야 하는지 말입니다. LATEX이 워드프로세서가 개발되기 이전에 사용되던 구시대의 유물이라고 느껴질 수도 있겠죠. 하지만 LATEX은 지금도 이공계 다양한 분야에서 활발하게 사용되고 있는 도구입니다. 그렇다면 LATEX이 WYSIWYG 편집기에 비해 가지는 장점에는 무엇이 있길래 LATEX을 사용할까요?

#### 1.2.1 구조화& 규격화된 문서

우선 LATEX으로는 구조화되고, 규격화된 문서를 작성할 수 있습니다. 무슨 말이냐고요? 수학 계열 학회 논문을 보시면 쉽게 이해하실 수 있으실 겁니다. 수식, 그림마다 붙어있는 번호, 복잡한 상호 참조, 1.1, 1.2.1, ... 논리적인 번호 매기기, 각주, 참고문헌까지. 같은 학회의 논문은 이 모든 구조가 똑같은 방식으로 이루어져 있습니다. 이 복잡한걸 어떻게 했냐고요? 사실 직접 한게 아니에요. LATEX이 자동으로 해 준 거죠.

<sup>1</sup><https://tobi.oetiker.ch/lshort/lshort.pdf>  
<http://users.softlab.ntua.gr/~sivann/books/LaTeX%20-%20User's%20Guide%20and%20Reference%20Manual-lamport94.pdf>

<sup>2</sup>레이텍이라고 읽습니다

우선 줄간격, 여백 등의 기본적인 설정은 코드 몇 줄을 적어두는 걸로 충분합니다. 또한 `section`, `subsection` 등의 명령어를 통해 문서를 정리해두기만 하면 번호는 L<sup>A</sup>T<sub>E</sub>X이 자동으로 매겨줍니다. 그럼 역시 자동으로 번호가 매겨지고, 나중에 그 그림을 언급하고 싶을 때는 자동으로 그 그림에 맞는 번호를 찾아줍니다. 참고문헌은 bibtex와 같은 소프트웨어가 더욱 간단하게 규격에 맞게 reference를 작성할 수 있도록 도와주죠.

어쨌든 논문 하나를 적을 때마다 기본적인 설정을 코드로 설정해야 하니 워드나 한글에 비해 너무 귀찮지 않냐고요? 간단한 글을 작성할 때는 그런 걸 설정하지 않아도 충분하긴 하지만, 맞는 말이긴 합니다. 그렇지만 만약 그게 이미 다 작성되어 있다면 어떠시겠어요? 이미 모든 설정이 끝난 템플릿이 주어져 있다면, 거기에 쓰고 싶은 글만 채워 넣으면 되는 거 아니겠어요? 실제로 수학, 물리, 천산학 등의 많은 분야의 학회에서는 .tex파일로 논문 양식을 제공합니다. 그럼 오히려 형식에 신경쓰지 않고, 쓰고자 하는 글에 집중할 수 있죠. 꼭 논문이 아니더라도 우리 학교 일몰실 템플릿 하나만 만들어 놓으면, 혹은 친구에게 받으면, 그때부턴 L<sup>A</sup>T<sub>E</sub>X으로 작성하는게 그다지 어렵지 않을 겁니다. 규격화 되고, 구조화 되어 있으니 결과물도 예쁘고요.

### 1.2.2 간편한 수식 입력

L<sup>A</sup>T<sub>E</sub>X은 수식 입력이 간편합니다. 수식 입력 상으로는 사실상 표준으로 자리잡았다고 할 수 있죠. 워드나 한글에서도 수식 입력이 된다고요? 사실 그것도 L<sup>A</sup>T<sub>E</sub>X의 수식 입력 문법을 참조해서 만들어진 겁니다. 위키피디아에 있는 수식도 다 L<sup>A</sup>T<sub>E</sub>X문법에 맞춰서 코딩된 겁니다. 그리고 워드나 한글에서 새로운 창을 띄워서 수식을 입력하고 닫는 복잡한 과정은 L<sup>A</sup>T<sub>E</sub>X만큼 편하지도 않죠. L<sup>A</sup>T<sub>E</sub>X은 수식 입력이 간단하고, 만들어진 수식이 예쁩니다. 수식 입력에 관한 내용은 뒤에서 천천히 다루도록 하겠습니다.

### 1.2.3 안정성과 이식성

L<sup>A</sup>T<sub>E</sub>X파일은 .tex라는 확장자로 저장되는데, 사실 이는 본질적으로 텍스트 파일입니다. 메모장으로도 수정할 수 있죠. 이 파일 안에는 저자가 문서를 어떻게 만들지에 대한 모든 내용이 알차게 들어 있습니다. 반면 워드나 한글 파일 같은 경우에는 저자가 적고자 하는 내용 이외에도 수많은 불필요한 정보들을 포함하고 있습니다. 그렇기 때문에 느리고, 무겁고, 깨지기도 하죠. 아마 대부분 한글 문서의 버전이 달라 파일이 깨진 것을 본 경험이 있을 겁니다. 그렇지만 텍스트 파일을 전달한다면, 혹은 L<sup>A</sup>T<sub>E</sub>X으로 만들어진 pdf 파일을 전달한다면, 이러한 문제는 해결됩니다. 문서가 손상될 위험도 적고, 어느 컴퓨터에서나 열어볼 수 있죠.

마찬가지로 L<sup>A</sup>T<sub>E</sub>X으로 문서를 작성하는 것 역시 편리한 점이 많습니다. L<sup>A</sup>T<sub>E</sub>X파일을 수정하는 것은 본질적으로 텍스트 파일을 수정하는 것과 같기 때문에 그다지 높은 컴퓨터 성능을 필요로 하지 않습니다. 500페이지 책을 쓴다고 생각해 보세요. 한글, 워드 프로그램으로 그걸 여는데 얼마나 오래 걸리겠습니까. L<sup>A</sup>T<sub>E</sub>X으로 만들면 마지막에 한번 pdf로 바꾸는 과정만 거치면 됩니다.

### 1.2.4 수많은 확장 기능

여러분은 왜 Python이 많이 사용되는지 아시나요? Python은 쉽기도 하지만, 다양한 확장 기능을 통해 미분방정식, 데이터 처리 등 수많은 일을 할 수 있다는 장점이 있습니다. 한번쯤 경험해 보셨을 수 있겠지만, import를 통해 라이브러리를 불러와서 이런 일들을 하곤 합니다. L<sup>A</sup>T<sub>E</sub>X역시 비슷합니다. 뒤에서 설명드리겠지만 L<sup>A</sup>T<sub>E</sub>X에서는 \usepackage{}가 비슷한 역할을 합니다. L<sup>A</sup>T<sub>E</sub>X의 다양한 패키지를 활용하면 워드프로세서의 다양한 편의 기능을 구현하는 것은 물론, 프레젠테이션 자료, 악보, 그래프 그리기 등의 수많은 기능을 구현할 수 있습니다.

아 그리고 아까 L<sup>A</sup>T<sub>E</sub>X이 문서를 프로그래밍한다고 묘사했죠? L<sup>A</sup>T<sub>E</sub>X에서는 문서를 작성할 때 조건문을 사용하거나, 새로운 명령(\newcommand)을 만들어 이용할 수 있기 때문에, 배우시다 보면 상당히 편하게 이용하실 수 있으실 겁니다.

아 그리고 한 가지가 더 있네요. L<sup>A</sup>T<sub>E</sub>X은 무료입니다. 누구나 다운로드받아서 사용할 수 있죠. 이 외에도 미학적<sup>3</sup>, 철학적<sup>4</sup>인 관점에서 L<sup>A</sup>T<sub>E</sub>X의 다양한 장점을 찾으실 수 있습니다. 처음에만 어렵게 느껴질 뿐, 익숙해지면 여러분도 왜 L<sup>A</sup>T<sub>E</sub>X을 사용하는지 금방 알게 되실 겁니다.

## 2 L<sup>A</sup>T<sub>E</sub>X 시작하기

이제부터 L<sup>A</sup>T<sub>E</sub>X의 설치부터 문서 작성의 시작까지 설명을 드리려 합니다. 천천히 순서대로 따라오시면 여러분 누구나 할 수 있을겁니다. 따라오시는 도중에 문제가 생기시면 [문제가 생기셨나요?](#)를 참고해 보시고, 그래도 모르겠으면 컴퓨터를 잘 아는 친구에게 물어보도록 합시다.

### 2.1 L<sup>A</sup>T<sub>E</sub>X 설치하기

먼저, 당연한 이야기지만 L<sup>A</sup>T<sub>E</sub>X을 설치해야겠지요? 기본적으로 L<sup>A</sup>T<sub>E</sub>X에 관한 정보나 소식은 공식 홈페이지<sup>5</sup>에 잘 나와 있습니다.

L<sup>A</sup>T<sub>E</sub>X의 장점 중 하나는 다양한 운영체제나 기기에서 설치하여 사용할 수 있다는 점입니다. 그렇지만 구체적인 설치 방법은 조금씩 다릅니다. 이 교재에서는 일단 Microsoft Windows를 기본으로 설명하겠습니다. 다른 운영체제에 대한 정보는 [공식 홈페이지](#)를 참조해주세요. 그렇지만 Mac OS, Debian Linux의 경우 이러한 설치가 무척이나 간단하기 때문에 큰 설명이 필요하지 않으실 겁니다. 물론 윈도우도 그다지 복잡하지는 않습니다.

**T<sub>E</sub>XLive** 윈도우를 포함하여 대부분의 운영체제에서 L<sup>A</sup>T<sub>E</sub>X을 실행하고 관련 package를 사용하기 위해 T<sub>E</sub>XLive를 설치하여 사용합니다. T<sub>E</sub>XLive는 텍 사용자 모임 사이트<sup>6</sup>에서 설치할 수 있습니다. 영어가 어려우시다면 한글 텍 사용자 그룹<sup>7</sup>의 도움을 받는 게 좋습니다. 설명대로 천천히 진행하시다 보면 아마 어렵지 않게 설치하실 수 있으실 겁니다. 일반적인 인터넷 환경에서 1~ 2시간정도가 소요됩니다.

**ShareLaTeX** 앞에서 T<sub>E</sub>XLive를 설치하기도 했고, 운영체제에 따라 그 방법이 조금씩 다르다는 이야기를 했습니다만, L<sup>A</sup>T<sub>E</sub>X은 놀랍게도 인터넷에서 설치 없이도 실행할 수 있습니다. 바로 ShareLaTeX와 같은 웹사이트를 이용하는 방법입니다. 온라인에서 언제 어디서나, 게다가 공동편집을 통한 협업까지 쉽게 L<sup>A</sup>T<sub>E</sub>X을 이용할 수 있는 방법이죠.

**Tip: 에디터 설치하기** T<sub>E</sub>XLive를 설치할 때 특별히 TeXworks 에디터 설치를 취소하지 않는다면 아마 TeXworks 에디터가 기본적으로 제공될 겁니다. 하지만 기본 제공되는 에디터는 기능이 적기 때문에 굳이 추천드리지 않습니다. L<sup>A</sup>T<sub>E</sub>X의 특징이자 장점이 바로 입력 파일 자체는 그냥 텍스트로 이루어져 있다는 겁니다. 사실 그래서 텍스트를 편집할 수 있는 프로그램이라면 무엇을 쓰든 큰 상관은 없습니다. 다만 많은

<sup>3</sup><http://nitens.org/taraborelli/latex>

<sup>4</sup><http://ricardo.ecn.wfu.edu/~cottrell/wp.html>

<sup>5</sup><http://www.latex-project.org>

<sup>6</sup><http://www.tug.org/texlive/acquire-netinstall.html>

<sup>7</sup><http://www.ktug.org/xe/?mid=Install>

사람들이 추천하고 사용하는 에디터는 TeXstudio입니다. LATEX에 맞게 개발된 도구인 만큼 쉽고 편리하게 편집할 수 있는 환경을 제공해 주기 때문입니다.

설치 방법은 간단합니다. 다운로드 사이트<sup>8</sup>에 들어가 프로그램을 다운받고 설치하면 됩니다. 처음 사용하시는 분들의 이해를 돋도록 여기서부터는 TeXstudio를 기준으로 설명을 하도록 하겠습니다. 그렇지만 지금부터 설명하는 부분은 매우 기초적인 부분이기 때문에 다른 LATEX에디터를 사용하시더라도 기본적인 사용법만 아시면 2.2장을 이해하는 데는 크게 어려움이 없으실 겁니다.

## 2.2 기본 사항

자, LATEX과 에디터를 모두 설치했다면 문서를 작성해야겠죠? 이런 걸 배울 때 꼭 맨 처음 해보는 것이 있습니다. ‘Hello, world!’ 출력하기입니다. LATEX에서 ‘Hello, World!’, 나아가서 ‘안녕, 세계!’까지 출력하려면 어떻게 해야 할까요?

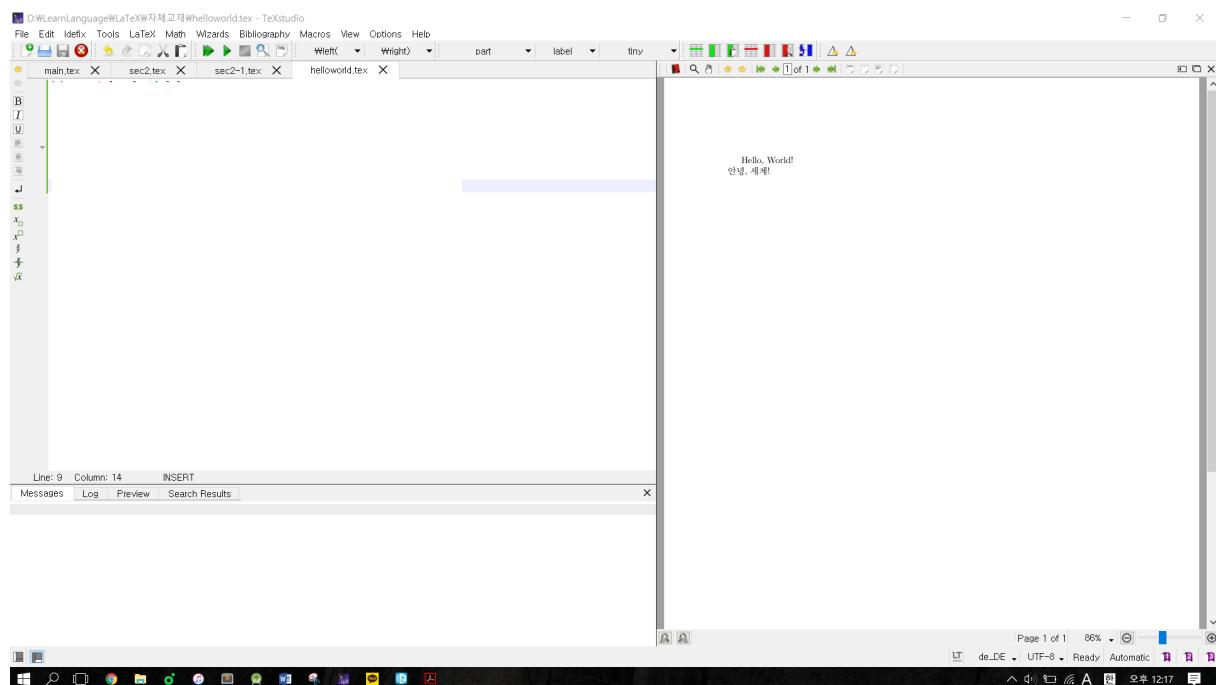


그림 1: TeXstudio에서 작업하는 모습

그림 1처럼 여러분은 에디터를 이용해서 문서를 작성할 수 있어야 합니다. 원하는 문서를 작성하려면 저런 화면에서 계속 타자를 쳐야 하는 겁니다. 그림 기준으로 왼쪽 화면에서 열심히 코딩하듯 입력하면서 오른쪽 화면에 그 결과물을 띄워보고, 또 편집하는 그런 식이죠. 일단 그림 1에서는 왼쪽 코드를 일부러 가렸습니다. 오른쪽 화면처럼 출력되려면 어떻게 입력해야 할까요?

직접 해봐야겠죠? TeXstudio를 실행하고, 파일을 만들어봅시다. 이제부터 차근차근 첫 번째 고비를 해결하는 겁니다.

### .tex 파일 생성부터 출력까지

1. TeXstudio를 실행하였다면, 왼쪽 위에 보이는 메뉴들을 봅시다. 새로 문서를 작성하기 위해 파일을

<sup>8</sup><http://texstudio.sourceforge.net/>

만들어야 하므로, File-New를 클릭하거나 그 바로 아래 아이콘을 클릭합니다. 단축키 Ctrl+N를 이용해도 됩니다.

2. 이제 뜬 창이 바로 입력 파일을 편집하는 곳입니다. 입력 파일은 미리 저장해둡시다. File-Save를 클릭하거나 아래 디스크 모양 아이콘을 클릭하거나 단축키 Ctrl+S를 이용합니다.
3. 입력 파일의 확장명은 tex입니다. (문서파일이름).tex 식으로 파일을 저장합니다.
4. 이제 입력을 시작하면 됩니다. 처음이니 깊게 생각하지 마시고 아래의 코드를 따라 쳐 보세요.

```
\documentclass{article}
\usepackage{kotex}
\usepackage{indentfirst}
\usepackage[left=2.5cm,right=2.5cm,top=3cm,bottom=3cm,a4paper]{geometry}

\begin{document}
Hello, World!\\
안녕, 세계!
\end{document}
```

5. 입력이 어느정도 되면 무려 ‘컴파일’이라는 것을 하게 됩니다. 그래야 문서를 출력할 수 있기 때문입니다.
6. 컴파일은 역시 위 메뉴에서 찾을 수 있습니다. 초록색 세모 아이콘을 클릭하거나 F6을 누르면 무언가가 돌아가는 모습이 보일 겁니다. ‘Process exited normally’라는 문구가 아래에 뜨면 성공한겁니다. 에러가 뜰 수도 있습니다.
7. 컴파일과 동시에 aux, log, out, pdf, synctex.gz, toc 등등 대부분은 알 수 없는 확장자의 파일처음 해보는 들이 생겨납니다. 당황하지 마세요. 잘 진행되고 있는 겁니다.
8. 컴파일이 성공적으로 끝나면 F7을 눌러봅시다. 오른쪽 화면에 무언가 뜰겁니다. 잠깐만 기다리면 여러분이 입력한 텍스트를 예쁘게 알맞게 출력시키면 어떻게 되는지 봅니다.

작업 도중 생성된 파일들은 각자 역할이 있고 쓰임이 있지만 일단 꼭 알아야 할 부분만 알아둡시다. 텍스트로 이루어진 tex 파일을 LATEX상에서 실행하여 pdf 파일로 출력물을 얻을 수 있다는 것입니다. 또한 그 상태에서 다시 편집하여 컴파일하면 알아서 변화된 내용대로 파일들이 수정됩니다. TeXstudio를 비롯해 많은 에디터들은 출력결과를 바로바로 확인하면서 편집할 수 있습니다. 이 덕에 LATEX를 WYSIWYG 편집기처럼 사용할 수 있죠.

**안녕, 세계!** 가장 기본적인 파일 생성 및 실행, 출력은 위 단계들만 잘 알아두면 됩니다. 계속 문서를 편집하다 보면 자연스럽게 익을 겁니다. 그럼 이제 정말로 어떻게 입력을 해야 문서라는 것이 만들어지는 것인지, 그 내용 부분으로 들어가 봅시다. 앞에 보여드렸던 그림 1을 다시 생각해 보는 겁니다. 모든 LATEX 문서 작성의 기본을, 이 예시를 통해 설명해드리겠습니다.

LATEX은 문서 스타일은 최대한 자동화시켜 알아서 꾸미고, 사용자는 그 내용에 집중할 수 있도록 도와줍니다. 그런 만큼 지금 만드려는 문서가 어떤 종류의 문서인지, 어떤 스타일로 만들어지길 원하는지, 또 어떤 기능을 사용해야 하는지 알려줄 필요가 있습니다. 이 부분을 preamble(서문)이라 합니다. 이제 그림 2의 코드를 보면서 이 preamble에 대해 이야기 해 봅시다.

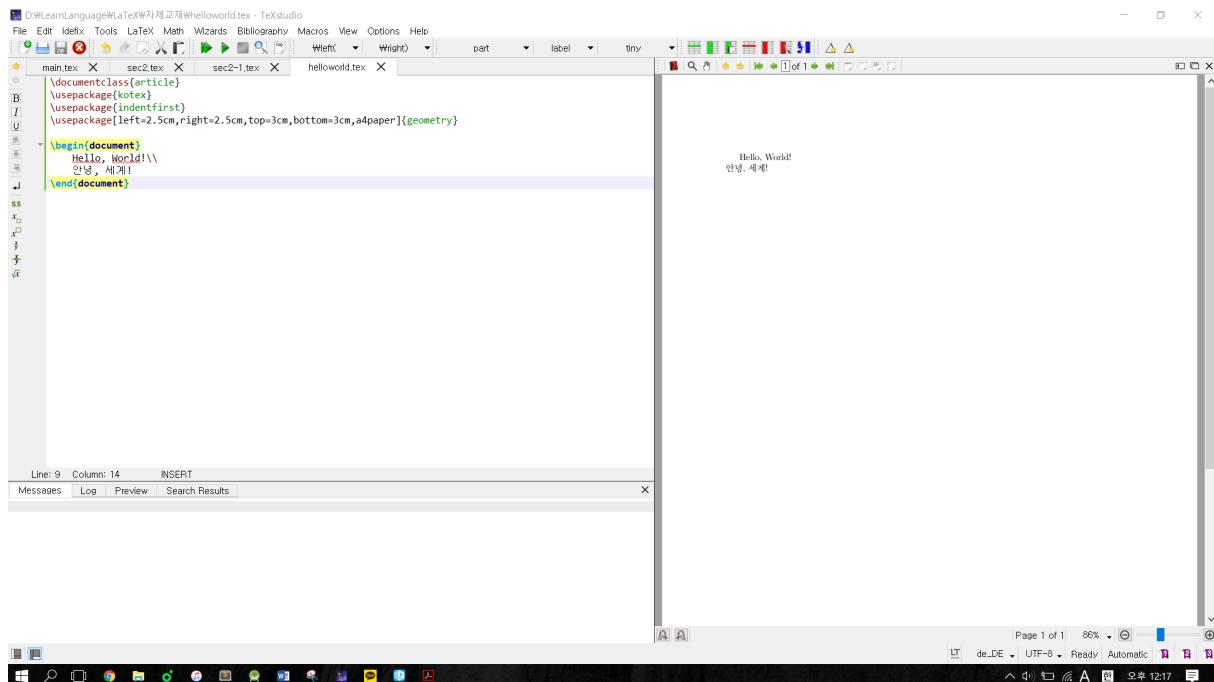


그림 2: 기본적인 코드의 모습

이것이 L<sup>A</sup>T<sub>E</sub>X문서 작업의 진짜 모습입니다. 코드를 본격적으로 살펴봅시다.

```
\documentclass{article}
\usepackage{kotex}
\usepackage[indentfirst]
\usepackage[left=2.5cm,right=2.5cm,top=3cm,bottom=3cm,a4paper]{geometry}

\begin{document}
Hello, World!\\
안녕, 세계!
\end{document}
```

아마 L<sup>A</sup>T<sub>E</sub>X을 처음 접해보는 분들께는 첫줄부터 무슨 의미인지 이해하기 힘들 겁니다. 천천히 저것들이 무슨 의미인지 봅시다.

**document class** 모든 것의 시작, \documentclass[option]{class}부터 이야기할까요? L<sup>A</sup>T<sub>E</sub>X으로 문서를 작성할 때에는 작성하려는 문서 유형을 처음 선언해주고 시작합니다. 문서 스타일을 자동으로 그 유형에 맞추어 주기 위해서입니다. 이것이 바로 \documentclass[option]{class}입니다. class, 즉 문서 유형은 무궁무진합니다. 기본적으로 article부터 letter, beamer까지 9가지 정도의 유형이 있습니다. 각 유형에 대한 설명은 표 1을 참고하시기 바랍니다. 물론 저 9가지 class들 외에도 매우 유용하고 좋은 class들이 많이 존재합니다. 이에 대해서는 필요할 때마다 이야기하겠습니다. class를 설정하고 나면 option을 추가로 설정할 수도, 설정하지 않을 수도 있습니다. 설정하지 않는다면 \documentclass{class} 식으로 써주시면 됩니다. option에 대한 설명 역시 표 2를 참고하면 됩니다.

‘안녕, 세계!’ 예시에서는 `article`이라는 class를 사용하였습니다. 사실 저정도 내용에서는 다른 class를 사용해도 큰 차이가 없겠지만 각 class마다 그 형태나 사용하는 기능이 다르기 때문에 실제 작성에서는 알맞은 class를 찾아 사용해야 합니다. 예시에서 중요한 점은 단 두 줄 쓰는 데에도 꼭 `\documentclass{}`를 써줘야 한다는 것입니다. 무슨 일이 있어도 첫 줄에 써 주세요!

**begin** 문서의 유형을 결정하였다면, 진짜 작성은 시작해야겠죠? 어떤 document class를 선택하든, 그 `document`의 시작을 선언할 필요가 있습니다. `\begin{document}`를 통해서 말입니다. 이 부분 역시 L<sup>A</sup>T<sub>E</sub>X 작성에서 항상 들어가는 부분입니다. tex 파일에서는 `\documentclass[option]{class}`부터 `\begin{document}`까지를 preamble(서문)이라고 합니다. 여기까지 마친 뒤 여러분이 하려고 했던 이야기들을 적어주시면 되는 겁니다. 또한, 하고 싶은 이야기가 전부 끝났다면, `\end{document}`로 끝마쳐야 합니다. 이것이 모든 문서의 기본입니다. 그림 2를 보시면 잘 나와 있듯이, 저렇게 문서의 모든 내용을 감싸주고 있다고 생각하시면 됩니다. 참고로, `\begin{}`, `\end{}` 구문은 굉장히 많은 용도로 쓰이기 때문에 잘 알아두셔야 합니다.

**package** 혹시 기본적인 것만으로는 표현할 수 없는 무언가를 표현해야 한다면, package를 찾게 될 겁니다. `\documentclass[option]{class}`와 `\begin{document}` 사이에는 해당 문서에서 특정 package를 사용할 수 있도록 선언해주는 `\usepackage[option]{package}`가 들어갈 수 있습니다. 마치 Python에서 사용할 모듈을 import하듯이 사용해야 하는, 사용하려는 package를 미리 불러오는 거죠. Python에서 모듈이 무궁무진하듯, package 역시 많은 사람들이 개발하고 있기 때문에 굉장히 다양하고, 얼마든지 필요한 package를 찾거나 직접 만들어서 추가할 수 있습니다. 그림 2의 ‘안녕, 세계!’라는 한글 역시 kotex라는 package를 추가하였기 때문에 사용할 수 있는 겁니다. 자주 쓰이는, 주요 package들에 대해서는 앞으로 차차 소개해 드릴 겁니다.

위 세 가지를 잘 기억해두고, 잘 써먹으면 L<sup>A</sup>T<sub>E</sub>X 문서의 기본이 완성된 것입니다. 이제 하고 싶은 이야기를 입력해 주시면 됩니다.

### 2.3 문제가 생기셨나요?

컴퓨터에 익숙하지 않은 분들은 시작부터 문제가 생기기도 합니다. 하지만 겁먹으실 필요는 없습니다. 지금부터 문제의 소지가 있을 만한 부분을 하나씩 짚어보도록 하죠.

- T<sub>E</sub>Xlive 설치 창에서 문제가 생기셨으면 여기를 봐 주세요. 한글 텍 사용자 그룹에서 권하는 대로 폴더 변경이 안 되나요? 사실 굳이 폴더를 변경하실 필요는 없습니다만, 오류가 생기는 이유는 T<sub>E</sub>XLive 설치 프로그램이 새로운 폴더를 자동으로 생성해주지 않기 때문입니다. 수동으로 원하는 폴더를 생성하고 다시 시도해 보세요.
- 커맨드창이라고 불리는 것들은 명령 프롬프트입니다. Windows 검색창에서 검색하셔서 실행하시면 됩니다. 참고로 커맨드창에서 tlmgr 명령을 하실 때는 커맨드창을 관리자 권한으로 실행시키셔야 합니다. 오른쪽 클릭하시면 하실 수 있습니다.
- TeXStudio가 한글 입력이 원활하지 않을 수 있습니다. 가끔 그런 경우가 있더라고요. 이건 그냥 그 프로그램상의 오류니까 굳이 깊게 생각하지 마시고 다른 에디터를 사용하세요. 에디터는 다양합니다.<sup>9</sup> 자신에게 맞는 에디터를 골라보세요.

---

<sup>9</sup><http://tex.stackexchange.com/questions/339/latex-editors-ides>

- L<sup>A</sup>T<sub>E</sub>X에만 해당되는 사항은 아니지만, 항상 오류가 나면 가장 먼저 의심해 보아야 할 것은 한글입니다. 안타까운 이야기지만 세계적으로 보았을 때 한글을 잘 처리할 수 있는 소프트웨어가 그다지 많지 않습니다. 어떤 프로그램이 실행되는 경로상에 한글과 같은 non-ASCII code가 포함되어 있으면 오류가 나는 경우가 상당히 많죠. 그런 의미에서 폴더 이름은 되도록 영어로 하는 것이 좋습니다. 꼭 L<sup>A</sup>T<sub>E</sub>X만을 위해서가 아니더라도요. 같은 맥락에서 Windows 사용자 이름이 한글로 되어 있으면 프로그램 설치에 문제가 생길 수 있습니다. L<sup>A</sup>T<sub>E</sub>X나 TeXStudio의 경우에는 어떤지 확실하게는 말씀드리기 어렵지만 문제가 생기신다면 의심하실 필요는 있습니다. 이건 윈도우를 초기화하지 않는 이상 바꾸기 상당히 까다로우니 주의해 주세요. 개인적으로는 초기화를 하더라도 바꾸는 것을 권장합니다. 다른 수많은 프로그램들을 위해서라도요.
- 아 그리고 참고로 알아두셨으면 하는 게 있네요. 아마 여러분이 pdf 파일을 보는 데 Adobe Acrobat Reader를 쓰시는 경우가 많으실텐데, 이 프로그램은 완성된 문서의 결과를 보는 데만 쓰셨으면 합니다. Acrobat Reader는 자신이 읽고 있는 pdf 파일을 잠금니다. 다시 말해 Acrobat Reader로 pdf 파일을 켜 놓으셨으면 그 파일은 컴파일이 되지 않을 겁니다. L<sup>A</sup>T<sub>E</sub>X전용 에디터들의 내장 뷰어나, SumatraPDF와 같은 프로그램은 특정 파일을 읽는 동안에도 그 파일의 수정을 허용하고, 뷰어를 끼다 켜지 않더라도 새로고침할 수 있지만, Acrobat은 그렇지 않습니다. 꼭 Acrobat만 그런 건 아니겠지만요. 어쨌든 pdf 뷰어를 켜놓고 컴파일이 되지 않는 경우 자신이 사용하는 pdf 뷰어의 특성일지 모른다는 것도 염두에 두셔야 합니다.

여기에 없는 문제들은 친구에게 물어보거나, 구글에 검색해 보세요. 아마 여러분과 같은 어려움을 겪고 있는 사람들이 상당히 많을 겁니다.

### 3 문서 작성 시작하기

#### 3.1 예시 코드 구경하기

그럼 지금부터는 본격적으로 문서 작성은 시작해 보도록 하겠습니다. 사실 진짜 기본적인 내용은 2장에서 모두 끝났기 때문에, 지금부터는 LATEX의 각 기능들을 어떻게 활용하는지에 초점을 맞추어 살펴보도록 하겠습니다. 아마 여기도 예시를 들어 설명하는 것이 보다 이해가 쉬울 겁니다. 간단한 예시 코드를 보도록 하죠.

```
\documentclass{article}
\usepackage{kotex}
\usepackage[left=2.5cm,right=2.5cm,top=3cm,bottom=3cm,a4paper]{geometry}
\author{KSAsstudent}
\date{\today}
\title{이게 레이텍입니다}
\begin{document}
    \maketitle
    \tableofcontents
    \vspace{3cm}
    \section{글 작성하기}
    \label{sec:intro}

    \subsection{기본적인 글}
    \label{sec:text}
    그냥 적으면 됩니다. ㅎㅎ

    \subsection{그림 넣기}
    \label{sec:image}
    \begin{center}
        \emph{뒤에서} 설명드릴게요.
    \end{center}

    \section{수식 작성하기}
    \ref{sec:image}와 크게 다르지 않습니다. \\
    뒤에서 설명드릴게요.

\end{document}
```

이 코드의 결과물은 다음 페이지에서 확인하실 수 있습니다.

# 이게 레이텍입니다

KSAsstudent

February 4, 2017

## Contents

1 글 작성하기	1
1.1 기본적인 글	1
1.2 그림 넣기	1
2 수식 작성하기	1

## 1 글 작성하기

### 1.1 기본적인 글

그냥 적으면 됩니다. ㅎ ㅎ

### 1.2 그림 넣기

뒤에서 설명드릴게요.

## 2 수식 작성하기

1.2와 크게 다르지 않습니다.

뒤에서 설명드릴게요.

이해하셨나요? 지금부터 천천히 하나씩 살펴봅시다.

### 3.2 글의 구조

일단 글의 전체적인 구조부터 파악해 봅시다. 앞서 2.2장에서 preamble에 대한 설명을 들으셨을 겁니다. `\documentclass` 부터 `\begin{document}` 까지가 preamble에 속합니다. 그 아래 `\begin{document}`부터 `\end{document}`까지는 글의 실질적인 내용에 속하죠.

글의 내용을 천천히 살펴봅시다. 여기서 사용된 `article`이라는 class는 section부터 시작해 subsection, subsubsection, paragraph 순으로 이어지는 문서 구조를 가지고 있습니다. 첫 번째 section의 첫 번째 subsection은 1.1, 두 번째 subsection은 1.2, ... 이런 식으로 구조화가 이루어지는 거죠. 앞서 말씀드렸듯이 `\begin{}`과 `\end{}`는 다양한 용도로 사용됩니다. `\begin{}`과 `\end{}`사이를 하나의 환경, environment라고 부르는데, 여기서는 center라는 environment를 정의하기 위해 사용되었죠. 특정 environment안의 글들은 그 environment의 특징을 띄게 됩니다. 여기서 center라는 environment는 글을 가운데 정렬하는 역할을 합니다.

거의 대부분의 L<sup>A</sup>T<sub>E</sub>X 코드들은 본질적으로 위 코드와 크게 다르지 않습니다. environment안에 environment가 있다거나 하는 복잡한 구조라도, 펼쳐놓고 보시면 이걸로도 충분히 이해하실 수 있습니다.

### 3.3 Preamble의 기능

그렇다면 지금부터는 아까 설명드린 preamble 각 줄이 어떤 역할을 하는지 보다 구체적으로 살펴보도록 하겠습니다. 사실 제가 예시로 든 것 말고도 수많은 package나 기능들이 있지만, 일단 그것들이 어떤 식으로 사용되는지 이해해 보시라는 의미에서 몇 가지 자주 쓰일만한 것들만 예시로 넣었습니다.

- 우선 맨 첫줄의 `\documentclass`에 대한 설명은 2.2장에서 충분히 했으니 다음 줄부터 설명하겠습니다. 다양한 document class들은 나중에 하나씩 살펴보도록 하고요.
- `\usepackage{kotex}`는 2.2장에서 잠깐 언급했지만, 한글을 사용하기 위해 사용되는 package입니다. 이 package 없이는 한글이 제대로 표시되지 않죠.
- 다음 줄의 `\usepackage[...]{geometry}` 부분을 살펴보도록 하겠습니다. `\usepackage{geometry}`는 geometry라는 이름의 package를 사용하겠다는 명령이고, 사이에 있는 대괄호는 그 geometry라는 package를 어떻게 사용할 것인지, 옵션을 지정하는 부분입니다. Geometry라는 이름에서, 그리고 옵션의 내용에서 짐작하실 수 있듯이 이 package는 문서의 여백을 설정할 수 있는 package입니다. 사실 굳이 지정을 안하셔도 L<sup>A</sup>T<sub>E</sub>X는 기본 설정대로 예쁜 문서를 만들어 줍니다. 그렇지만 문제가 있다면 여백이 조금 넓다는 거죠. 직접 한 번 해 보시면 아실 겁니다. 사실 이 여백은 어떻게 하면 가장 읽기 편한 문서를 만들 수 있을지, 다양한 방면에서 고려해 결정된 것이라곤 합니다. 그렇기에 저도 저 혼자 볼 문서는 기본 설정대로 만드는 경우가 많고요. 그렇지만 솔직히 여백이 너무 넓으면 부담스러운 경우가 많지 않습니까. 그럴 땐 이 패키지를 쓰시면 됩니다.
- 다음 세 줄은 연달아 author, date, title을 지정했네요. 별 거 없습니다. 말 그대로 지정한 거죠. 몇 줄 내려가 `\begin{document}`아래에 `\maketitle`이 보이시나요? 여기서는 여기서 지정한 값들을 바탕으로 제목을 만들어 줍니다. 아, `\today`는 뭐냐고요? date 자리에 오늘 날짜를 자동으로 입력해 주는 명령어입니다. 굳이 `\today`를 쓰지 않고 그냥 아무거나 넣으면 날짜 자리에 그게 뜰겁니다.

이 코드의 preamble은 이게 전부입니다. 간단하죠? 이 코드에서 사용된 것 말고도 preamble에서는 다양한 일을 할 수 있습니다. 앞서 1.2.4장에서 잠깐 언급했지만 새로운 명령을 정의해서 사용한다거나, 명령들이 어떻게 작동할지 구체적으로 설정하는 것도 가능합니다. 몇 가지는 뒤에서 천천히 소개하겠습니다.

### 3.4 명령어 알아보기

지금부터는 LATEX의 명령어들을 구체적으로 알아보겠습니다. 사실 명령어라고 거창하게 이름 붙일 것도 없습니다. 앞에서 눈치채셨겠지만 LATEX는 \로 시작하는 단어를 인식하고, 이는 모두 명령어라고 불리기 때문입니다. 잠시 3.3장에서 자세히 살펴보지 않은 유용한 LATEX 명령어들을 몇 가지 살펴보도록 하겠습니다.

우선 \maketitle은 앞에서 설명드렸으니 그 다음 \tableofcontents 부터 살펴보도록 하겠습니다. 예상하셨겠지만, \tableofcontents는 문서의 목차를 만들어 줍니다. 앞서 설명했듯 \section, \subsection 등을 이용하여 문서를 구조화시켜놓았다면 따로 별다른 설정을 하지 않아도 \tableofcontents 명령어 하나로 그럴듯한 목차를 만들어 줄 겁니다. 물론 목차 생성의 설정을 바꾸는 것도 가능합니다만, 여기서는 깊게 다루지 않도록 하겠습니다.

다음으로 보이는 명령어는 \vspace{3cm} 이네요. 이건 바로 알아보시지 못하셨을지도 모르겠네요. \vspace 란 vertical space의 줄임말로, 세로 여백을 조절할 수 있는 명령어입니다. 팔호 속의 3cm은 3cm의 여백을 두겠다는 의미죠. 센티미터 말고도 인치와 같은 다른 단위들을 사용하는 것도 가능합니다. 비슷한 방식으로 \hspace 명령어는 가로 방향 빈칸을 두기 위한 명령어로 사용됩니다.

다음은 \section{글 작성하기} 아래의 \label{sec:text}를 봅시다. label은 상호 참조를 위해 사용되는 명령어입니다. 글 작성하기라는 section을 뒤에서 참조하기 위해 이름을 붙여놓은 거죠. 이는 뒤에서 \ref{} 명령어와 함께 사용됩니다. \ref{sec:image}라는 명령어를 통해서 \label{sec:text}를 불러온 거죠. 여기서는 알아보지 못하시겠지만 이는 원하는 부분으로 하이퍼링크를 걸어줌과 함께 자동으로 번호를 매겨줍니다. 이 LATEX의 상호 참조 기능은 중간에 section, subsection을 추가하면 새롭게 번호가 매겨지기 때문에 무척 편리하고, 보다 구조화된 문서를 작성하는 데 도움이 됩니다. 물론 section, subsection 등에만 활용이 제한되어 있지 않고, 수식, 그림 등 다양한 부분에 사용 가능합니다.

마지막으로는 \emph{} 명령어를 구경해 봅시다. 우리가 문서의 특정한 부분에 특정한 효과를 적용하고 싶다면 위에서 언급한 \begin{center}, \end{center}와 같이 environment를 이용할 수 있습니다. 그렇지만 한 단어나, 한 글자와 같은 보다 짧은 구간에 적용되는 효과의 경우 일반적인 \begin{, \end}로 구성되는 방식 대신 \emph와 같은 방식이 사용되는 경우가 많습니다. 예상하실 수 있으시겠지만 emph 는 emphasis를 줄인 표현으로, 단어를 강조할 때 사용됩니다. 영문의 경우 이 명령어를 사용할 경우 *emphasis*와 같이 이탤릭체로 표현이 되지만, 한글과 한자의 경우 본래 이탤릭체가 없기 때문에<sup>10</sup> 강조 와 같은 형태로 나타나게 됩니다.

### 3.5 문제가 생기셨나요??

여기서는 위에서 언급한 것과 같은 기본적인 문서 작성 과정에서 문제가 발생했을 시 대처할 수 있는 몇 가지 방법 및 주의 사항에 대해 언급해 드리겠습니다.

- tableofcontents, reference 와 같은 기능들은 올바르게 작동하기 위해 두 번의 컴파일이 필요할 수 있습니다. citation 기능을 사용할 경우에는 tex engine을 바꿔 가며 순서에 맞게 컴파일할 필요가 있

<sup>10</sup>워드프로세서의 이탤릭 기능은 단순히 글자를 찌그러트린 것에 불과합니다. 실제로 한글이 이탤릭체를 지원하려면 이탤릭체가 포함된 한글 폰트가 필요합니다만, 영어와 달리 한글은 이탤릭체가 예쁘지도 않고, 전통적으로 사용되어 오지 않았기 때문에 이를 지원하는 한글 폰트는 거의 없다시피 합니다.

기도 합니다만 여기서는 자세히 언급하지 않겠습니다. 물론 TeXStudio와 같은 프로그램을 사용하면 간단하게 해결할 수 있는 문제이기는 합니다. 어쨌든 목차나 reference 기능이 올바로 작동하지 않더라도 너무 걱정하지 마시고 한번 더 컴파일을 해 보세요.

- 위에서 언급했듯이 L<sup>A</sup>T<sub>E</sub>X는 environment의 중첩 사용을 허용합니다. 즉 environment 안에 environment가 들어갈 수 있습니다. 그러나 이 때 environment는 그 중첩 순서가 맞아야 합니다. 다시 말해

```
\begin{environmentA} \begin{environmentB} \end{environmentB} \end{environmentA}
```

는 허용되지만

```
\begin{environmentA} \begin{environmentB} \end{environmentA} \end{environmentB}
```

는 허용되지 않습니다.

- L<sup>A</sup>T<sub>E</sub>X의 모든 명령어는 대소문자를 구분합니다. 꼭 유의하세요. 그리고 명령어의 기준은 \이 시작될 때부터 알파벳이 아닌 문자가 올 때까지입니다. 다시 말해 명령어를 모두 입력하고 나면 빈 칸을 추가해 주어야 L<sup>A</sup>T<sub>E</sub>X이 명령어가 끝났다는 점을 인식할 수 있습니다. 그렇지 않으면 없는 명령어를 사용했다며 에러를 배출하겠죠. 또한 새로운 명령어를 만들 때, 숫자나 괄호가 들어가는 명령어를 사용할 수도 없습니다.

## 4 좀 더 구체적으로

3장에서는 기본적인 문서의 구조와 다양한 명령어들을 사용하는 기본적인 방법에 대해서 익혔습니다. 그렇지만 사실 L<sup>A</sup>T<sub>E</sub>X에는 기능이 무수히 많고, L<sup>A</sup>T<sub>E</sub>X을 워드프로세서와 같이, 그리고 그 이상의 수준으로 사용하기 위해서는 이를 익힐 필요가 있습니다. 물론 모든 기능을 익힐 필요는 없고, 익힐 수도 없겠죠. 그렇기 때문에 기본적인 기능을 익힌 뒤에는 자신이 자주 작성하는 문서에 사용되는 기능들을 위주로 필요할 때마다 알아 나가는 것이 바람직합니다. 지금부터 설명할 내용은 참고할 만한 기능들 중 비교적 유용하다 생각하는 것들을 모아 놓은 것들입니다. 한 번 훑어 보시고 필요할 때마다 찾아 쓰시는 게 바람직할 겁니다. 복잡하고 자주 사용되지 않는 기능들은 여기에 언급하지 않았으니 필요하시다면 보다 전문적인 매뉴얼<sup>11</sup>이나, 인터넷 검색을 활용하시면 됩니다.

### 4.1 documentclass에 대하여

여기서는 문서 작성에 사용할만한 documentclass에 대해 보다 자세히 살펴보도록 하겠습니다. 위에서는 article class만 사용했지만 사실 다른 class도 자주 사용되거든요. 일반적인 문서를 작성할 때에는 article class보다는 memoir class가 더 자주 사용됩니다. memoir class는 여러 외부 패키지들을 모아 하나의 class로 만든 것으로 article, report 등의 class에 비해 다양한 기능을 가지고 있고 다양한 설정을 할 수 있기 때문에 무척 편리합니다. 사실 class라고 부르는 개념도 여러 package나 몇 가지 설정들을 미리 불러오는 정도에 불과하기 때문에 L<sup>A</sup>T<sub>E</sub>X를 자유자재로 다룰 수 있다면 무슨 class를 사용하나 큰 차이가 없겠지만, 익숙하지 않다면 기본적인 설정들이 적절하게 되어 있는 memoir와 같은 class를 사용하는게 편하겠죠.

---

<sup>11</sup><https://tobi.oetiker.ch/lshort/lshort.pdf>  
<http://users.softlab.ntua.gr/~sivann/books/LaTeX%20-%20User's%20Guide%20and%20Reference%20Manual-lamport94.pdf>

**oblivoir class** 제가 여러분들에게 일반적인 문서 작성 용도로 추천드리는 class는 `oblivoir` 입니다. 이 class는 `memoir` class를 한글 사용에 적합하도록 변형시킨 것으로, 큰 고민 없이 사용하기에 가장 편하실 겁니다. 나중에 `LATEX`에 익숙해지게 되면 커스터마이징해서 사용하기에도 무척이나 유리한 class입니다.

**다른 유용한 class들** 일반적이지 않은 문서 작성은 하실 때에는 `LATEX`의 다른 class들을 눈여겨 보시는 것이 좋습니다. 특정 작업에 특화된 다양한 class들이 있기 때문에 만약 관련 작업들을 하신다면 그런 class를 사용하는게 바람직하겠죠. 표 5.1에서는 `LATEX`에 있어서 상당히 기본적인 class만을 모아 설명하고 있습니다만, 그것 말고도 유용한 class가 몇 가지 있거든요. 여기서는 표 5.1에서 언급된 것을 포함한 몇 가지 class들의 특징들을 설명하도록 하겠습니다.

**book** class는 말 그대로 책을 만들 때 사용됩니다. 좌우로 펼쳐 보는 책이기 때문에 홀/짝 페이지별로 좌우 여백이 다르다는 것이 다른 class와의 가장 큰 차이라고 생각하시면 됩니다.

**beamer** class는 프레젠테이션 자료를 만들 때 사용됩니다. 일반적으로 프레젠테이션 자료를 만들 때는 `powerpoint`나 `keynote` 등의 프로그램을 사용합니다만, `LATEX`으로도 상당히 다양한 시각적 효과를 가진 프레젠테이션 자료를 만드는 것이 가능합니다. `LATEX`으로 프레젠테이션 자료를 만들 때 장점은 구조화된 문서, 수식 입력의 편의성 등을 들 수 있겠습니다. 윤상현 선생님의 프레젠테이션 자료를 생각하시면 이해가 쉬울 겁니다. 그렇지만 특별히 구조화된 프레젠테이션을 하고, `LATEX`의 기능을 이용하고자 하는 게 아니라면 `IguanaTeX`와 같은 `powerpoint` 용 `LATEX`수식 작성 플러그인 등을 사용하는 것도 하나의 방법입니다.

**standalone** class는 용지의 크기가 정해져 있지 않고, 내용에 맞게 크기가 변합니다. 수식 몇 줄을 기록해 두는 용도로 깔끔하고 유용하죠.

**exam** class는 시험지를 만드는 데 특화된 class입니다. 파일 하나로 답안이 표시된 파일과 그렇지 않은 파일을 만들고, 배점을 정하는 등의 기능이 있죠. 멘토링 자료를 만들 때 유용합니다.

이를 제외하고도 `LATEX`에는 독특한 class들이 있습니다만, 자주 사용되지는 않을 겁니다. 여기서 언급한 class들도 그다지 자주 사용할 일은 없겠지만요.

## 4.2 유용한 package 알아보기

여기서는 `LATEX`의 몇 가지 유용한 package를 알아보도록 하겠습니다. `LATEX`은 수많은 package를 어떻게 활용하느냐에 따라 훨씬 보기 좋은 문서를 쉽게 만들 수 있습니다. 물론 package 역시 수가 많고 계속 발전하고 있기 때문에 모두 익히기는 어렵습니다. 여기서는 몇 가지 자주 사용되는 package의 사용법을 정리해 보도록 하겠습니다. 나중에 `LATEX`으로 특정 기능을 구현하고 싶다면 인터넷에 검색해 어떤 package를 사용하는 게 좋을 지 알아보세요. `kotex`, `geometry`, `amsmath`, `amssymb`, `graphicx` 등의 일부 package는 각각 3.3장, 4.5.1장, 4.6.2장 등에서 설명하기 때문에 따로 언급하지 않겠습니다.

**setspace** package는 줄 간격을 조절할 때 사용됩니다. option을 어떻게 주느냐에 따라 더블스페이싱 등을 설정할 수 있죠.

```
\usepackage[doublespacing]{setspace} \usepackage[onehalfspacing]{setspace}
```

등을 이용하면 됩니다. `\doublespacing` 등의 명령을 사용하는 것도 가능합니다. 그렇지만 `memoir`, `oblivoir` class를 사용할 경우 class 내에서 이미 `setspace` package가 설정되어 있기 때문에 이를 바로 사용할 수 없습니다.

```
\begin{Spacing}{2}
Hello, world!
\end{Spacing}
```

과 같이 memoir계열 class의 기능인 Spacing envoriment를 사용하거나

```
\DisemulatePackage{setspace}
\usepackage{setspace}
```

와 같은 방식으로 이미 정의된 setspace 설정을 바꿀 수 있습니다. 참고로 oblivious class의 경우에는 줄 간격이 한글에 적합하게 설정되어 있습니다. 영문 문서를 작성할 때 쓰이는 class들과 기본 설정에 차이가 있으니 유의해 주세요.

**indentfirst** 는 그다지 많은 기능을 가진 package가 아닙니다. 그냥 첫 번째 문단을 들여쓰기 해 주는 역할을 하죠. 첫 번째 문단을 들여쓰기 할 것인지에 대한 문제는 상당히 의미 없으면서도 논란이 되기도 하는 문제입니다. 그렇지만 결국 특정한 문서 형식을 만족시켜야 하는 경우가 아니라면 이는 저작자의 취향에 달린 문제이기도 하죠. LATEX는 기본적으로 첫 번째 문단을 들여쓰기하지 않습니다. 그렇지만 만약 하고 싶으시다면 preamble에 \usepackage{indentfirst}를 추가해 주세요.

**float** 이 package는 4.6장을 읽고 나신 후에 그 유용함을 알게 되실 겁니다. 이 package를 사용하면 figure 를 보다 쉽게 설정할 수 있습니다. Preamble에 \usepackage{float}을 추가하시면 figure 위치 지정에 H라는 위치 인자를 이용하실 수 있는데 이는 !를 이용한 위치 지정과 유사하게 figure를 강제로 원하는 위치에 고정시킵니다. 또한

```
\floatstyle{boxed}
\restylefloat{figure}
```

등을 통해 figure에 테두리를 추가하는 등 다양한 작업이 가능합니다.

**morefloats** 4.6장에서는 float, 떠다니는 개체에 대해 다룹니다. 그런데 LATEX는 기본적으로 떠다니는 개체를 18개까지만 다룰 수 있습니다. 만약 더 많은 수를 다루고 싶다면 morefloats package를 활용해야 하죠.

### 4.3 유용한 환경 알아보기

이 장에서는 LATEX의 유용한 환경(environment)에 대해 다루도록 하겠습니다. LATEX은 기본 기능으로, 혹은 package를 통해서 다양한 environment를 제공합니다. 여기서는 그 중 자주 사용되는 것들 몇 가지를 살펴보도록 하겠습니다.

**center** 이 environment는 3.4장에서 언급했듯이 가운데 정렬을 해 주는 environment입니다. 이와 비슷한 방식으로 flushleft, flushright environment는 각각 왼쪽, 오른쪽 정렬을 해 줍니다.

**verbatim** 이 environment는 코드를 표현하는 데 사용됩니다. 일반적으로 LATEX코드는 컴파일을 하면 그 명령어대로 모양이 바뀌게 되죠. 그렇지만 verbatim envoriment를 사용하게 되면 코드를 모양 그대로 나타낼 수 있습니다. 정보과학 관련 자료에 많이 사용됩니다. 이 문서처럼요. 그렇지만 보다 다양한 기능을 가진 verbatim environment를 사용하고 싶으시다면 fancyvrb package를 이용하여 Verbatim environment를 사용해 보는 것도 좋습니다.<sup>12</sup>

---

<sup>12</sup>LATEX은 대소문자를 구분합니다

`itemize` 이 environment는 내용을 item별로 정렬해서 나타내는 데 사용됩니다. 문제가 생기셨나요? 의 구조를 보시면 이해가 쉬우실 겁니다. 이와 비슷한 environment로는 번호를 매겨 정렬해주는 `enumerate`, `labeling`과 함께 정렬해주는 `description` environment 등을 들 수 있습니다.

**abstract** 논문에는 초록, abstract가 있습니다. 어떻게 하면 이 `abstract`를 예쁘게 넣을 수 있을까요? LATEX에서는 별 고민 없이 `abstract` environment를 이용하면 됩니다. 별 설명이 필요 없죠.

이 외에도 인용문을 위한 `quote`, 시를 위한 `verse` 등 LATEX에는 특별한 상황을 위한 수많은 environment가 존재합니다. 이를 알고 있다면 특정 작업을 하기 수월하겠습니다. 물론 많은 environment는 각자의 package를 필요로 하기 때문에 어떤 Package가 필요한지 알고 있어야겠죠. 그렇지만 아마 한국과학영재학교 학생의 일반적인 문서 작성에는 이 정도면 큰 무리가 없지 않을까 싶습니다.

#### 4.4 새로운 명령 정의하기

4.2장과 4.3장에서는 다양한 LATEX명령어를 불러와서 사용하는 방법에 대해 익혔습니다. 그렇지만 간혹 필요한 명령들을 간단하게 사용하기 힘든 경우도 있습니다. 이럴 때는 명령어를 만들 수 있습니다. LATEX에 어느정도 익숙해졌다면 자주 작성하는 문서마다 사용되는 명령어와 스타일을 지정한 package를 만들어 사용하는 것도 가능합니다만, 이는 상대적으로 복잡하기 때문에 여기서는 자세히 설명하지 않고, 간단한 명령을 정의하여 사용하는 법을 소개하고 마무리하도록 하겠습니다.

새로운 명령을 정의하는 방법은 그다지 어렵지 않습니다. 기본적으로 preamble에 `\newcommand` 명령을 사용하는 것으로 충분하죠. 예시를 들어 봅시다. 여러분은 LATEX, 이 신기한 모양의 글자가 어떻게 만들 어졌다고 생각하세요? 이는 사실 레이텍의 기본 명령어 `\LaTeX`를 사용한 것입니다. 그렇지만 이 문서에는 LATEX라는 표현이 무척 많이 사용되었고, 매번 `\LaTeX`를 입력하기는 상당히 귀찮습니다. 그래서 사실 이 문서를 작성할 때는 `\newcommand{\lt}{\LaTeX}`를 preamble에 추가하여 LATEX 대신 `\lt`를 사용하는 것으로 LATEX의 입력을 대신했습니다. 대충 이해가 되시나요? `\newcommand`뒤의 중괄호 `{ }`에는 새로운 명령의 이름을, 그 뒤에는 원한다면 대괄호 `[]`를 사용하여 num 인자를<sup>13</sup>,<sup>14</sup>,<sup>15</sup>, 그 뒤의 중괄호 `{ }`에는 원하는 명령을 적으면 됩니다.

두 명령을 하나로 묶고 싶다면 어떻게 할까요? 예를 들어 math mode에서 `\lceil` `\rceil` 명령, `\lceil` `\rceil` 명령을 사용하면 각각 `\lceil` `\rceil`이 나타납니다.

그렇지만 우리가 원하는 ceiling function을 만들기 위해 `\lceil` `\rceil`을 통해 `[3.5]`을 사용하기는 너무 번거롭죠. 이 때 `mathtools` package에서 지원하는 `DeclarePairedDelimiter` 명령을 사용하면 간단하게 두 개를 하나로 묶을 수 있습니다. `\DeclarePairedDelimiter{\ceil}{\lceil}{\rceil}`를 preamble에 추가하면 `\ceil{3.5}`로 `[3.5]`와 같은 결과를 얻을 수 있죠.

새로운 환경을 정의하는 것도 가능합니다. `\newenvironment` 명령을 사용하면 됩니다. 이렇듯 LATEX은 수많은 명령을 간단하게 정의해서 사용할 수 있는 방법이 있고, LATEX을 익힐수록 이를 더 능숙하게 사용할 수 있을 것입니다. 그렇지만 사실 처음 사용하는 입장에서 이러한 명령어를 정의하는 방법을 열심히 공부할 필요는 없다고 봅니다. 그렇기에 여기서는 이 정도 소개로 마무리하도록 하겠습니다.

---

<sup>13</sup>생략한 경우 0입니다

<sup>14</sup>num 인자는 괄호 안의 내용을 명령 속 원하는 내용으로 치환합니다. 자세한 사용법은 전문적인 매뉴얼을 참고하세요

<sup>15</sup><https://tobi.oetiker.ch/lshort/lshort.pdf>

## 4.5 수식 작성하기

드디어 LATEX의 꽃, 수식에 대해 설명을 드릴 차례입니다. 다른 어떤 편집기보다도 강력하고 편리한 수식 편집 기능을 가지고 있는 LATEX인 만큼, 실질적으로 수식 편집의 표준으로 자리 잡았기 때문에 다양한 곳에서 이제 배우는 것들을 이용할 수 있을 겁니다. 역시나 예시부터 볼까요?

```
\documentclass{article}
\usepackage{kotex}
\usepackage{setspace}
\usepackage{amsmath}
\usepackage{amssymb}
\usepackage[left=2.5cm,right=2.5cm,top=3cm,bottom=3cm,a4paper]{geometry}
\begin{document}

    \doublespacing
    \noindent 수식을 입력해 봅시다.\\
    막 내용을 입력하다 중간에 짧은 수식을 넣을 수 있습니다.\\
    예를 들어 $c \in A$ 이렇게 말입니다.\\
    조금 길거나 큰 수식을 따로 입력할 수도 있습니다.\\
    예를 들어
    \[\left[ \begin{array}{l}
        a \in f(a) \\
        a \notin f(a)
    \end{array} \right]
    이렇게 말입니다.\\
    심지어
    \begin{aligned}
        |f(x)| &= |sf_1(x) + tf_2(x)| && \text{\textbackslash le } |sf_1(x)| + |tf_2(x)| \text{\textbackslash nonumber} \\
        \&= |s||f_1(x)| + |t||f_2(x)| \text{\textbackslash nonumber} \\
        \&= |s|c_1|g(x)| + |t|c_2|g(x)| \text{\textbackslash nonumber} \\
        \&= (|s|c_1 + |t|c_2)|g(x)| \text{\textbackslash nonumber}
    \end{aligned}
    이런 것도 됩니다.\\
    \LaTeX 은 정말 많은 수식 요소들을 지원해줍니다.\\
    $a^{bcd}_{efg}$와 같은 첨자는 물론이고, $\sum$, $\int$같은 것도 지원합니다.
    \[\frac{\pi}{2}\]
    이렇게 분수도 잘 됩니다.
\end{document}
```

역시 바로 다음 페이지에서 결과물을 확인하실 수 있습니다.

수식을 입력해 봅시다.

막 내용을 입력하다 중간에 짧은 수식을 넣을 수 있습니다.

예를 들어  $c \in A$  이렇게 말입니다.

조금 길거나 큰 수식을 따로 입력할 수도 있습니다.

예를 들어

$$\begin{cases} a \in f(a) \\ a \notin f(a) \end{cases}$$

이렇게 말입니다.

심지어

$$\begin{aligned} |f(x)| &= |sf_1(x) + tf_2(x)| \leq |sf_1(x)| + |tf_2(x)| \\ &= |s||f_1(x)| + |t||f_2(x)| \\ &\leq |s|c_1|g(x)| + |t|c_2|g(x)| \\ &= (|s|c_1 + |t|c_2)|g(x)| \end{aligned}$$

이런 것도 됩니다.

L<sup>A</sup>T<sub>E</sub>X은 정말 많은 수식 요소들을 지원해줍니다.

$a_{efg}^{bcd}$ 와 같은 첨자는 물론이고,  $\sum$ ,  $\int$ 같은 것도 지원합니다.

$$\frac{\pi}{2}$$

이렇게 분수도 잘 됩니다.

예시를 잘 보면서, 천천히 시작부터 보도록 합시다.

#### 4.5.1 패키지 사용하기

보통 수식을 제대로 입력하려면 두 가지 package를 사용해야 합니다. amsmath와 amssymb이 바로 그것들입니다. 수식을 입력해야 하는 문서라면 `\usepackage{amsmath}`과 `\usepackage{amssymb}`을 preamble에 입력해주세요. 참고로 한 번에 `\usepackage{amsmath, amssymb}`라 쓸 수 있습니다. 수식은 웬만한 문서에 들어가기 때문에 이 한 줄을 기본적으로 입력해 두는 것을 추천합니다. 패키지를 불러왔다면, 이제 문서 어디든 수식을 입력할 준비가 된 것입니다.

#### 4.5.2 입력 시작하기

L<sup>A</sup>T<sub>E</sub>X에서 수식을 입력하기 위해서는 따로 수식 편집기를 칸다거나, 그림을 추가하듯 추가하는 것이 아니라 기호 또는 command를 통해 수식을 입력함을 표시하면 됩니다. 그 방식에는 크게 세 가지 정도가 있습니다.

**짧은 수식 삽입하기** 길고 크고 염청난 수식을 입력하는 것이 아니라면, 보통의 경우는 수식을 문장의 중간에 넣고 싶을 겁니다. 이 때에는 내용을 입력하다가 수식을 입력하고 싶을 때  $$...$$ 를 이용하면 됩니다. 보통 수식에서 많이 쓰이는 기호들은 저 달려 표시 안에 있어야 L<sup>A</sup>T<sub>E</sub>X에서 인식할 수 있습니다. 또 저 안에 있으면 글꼴 자체가 수식에 맞도록 바뀝니다. 이 외에도 사람이 보기 좋은 수식으로 알아서 맞추어 주도록 하는 마법의 기호입니다. 다만 조금 길거나 여러 줄로 넘어가는 수식은 입력하기 어렵습니다. 꼭 일반적인 문장 중간에 넣고 싶을 때 사용해주시기 바랍니다.

**긴 수식 삽입하기, 수식만 따로 삽입하기** 길고 크고 염청난 수식을 입력하고 싶거나, 그래야 한다면 조금 다른 방법을 이용해야 합니다. 사실 정확히 말하자면 꼭 길지 않더라도 수식을 문장과 분리하여 따로 삽입 하려면 이 방법을 이용해야 합니다. 바로 `\[...\]`를 사용하는 겁니다. 이 안에 있는 모든 텍스트는 수식 입력으로 인식해서 수식으로 바꿔줍니다. 여러 행의 수식은 보통 이 방식으로 입력해줘야 하며 위의 방식보다 안정적으로 수식을 작성할 수 있습니다. 다만 이 방식 역시 모든 것이 가능한 것은 아닙니다. 그래도 웬만한 수식은 여기까지의 방법만 알아도 충분히 가능합니다.

**특별한 수식들** 위 두 가지 방법이 아닌 다른 방법으로 수식을 입력하는 경우도 있습니다. 예시에서는 `\begin{align}`이 이에 해당한다고 볼 수 있습니다. `\begin{align}`에 대한 자세한 설명은 뒤 4.5.3장에서 하도록 하겠습니다. 이 외에도 `\begin{equation}`, `\begin{displaymath}` 등의 수식 환경을 이용할 수 있습니다.

세 가지 방법 중 상황에 맞게 한 가지를 골라 수식을 입력하기 시작하면 됩니다.

#### 4.5.3 다양한 기능

L<sup>A</sup>T<sub>E</sub>X가 수식 입력에서 그렇게 두각을 나타내고 실질적인 업계 표준이 되어 다른 편집기가 베낄 정도로 뛰어난 이유는 위에서 말씀드렸던 입력의 용이성 외에도 수식에 필요한 기능을 거의 전부 가지고 있으며 이를 미려한 디자인으로 출력해준다는 것에 있습니다. 이제 그 많은 기능들 중 자주 필요로 하는 기능들 위주로 몇가지 소개해드리려 합니다.

**빈칸 띄우기** 수식 모드에서 단순한 빈 칸과 줄바꿈은 적용되지 않습니다. 수식에서 모든 임의로 설정하지 않은 공백은 그 수식에 맞게 알아서 설정됩니다. 이 공백을 임의로 제어하기 위해서는 \,, \quad 또는 \qquad와 같은 특별한 명령으로 지정해야 합니다. 자세한 정보는 표 3을 참조하세요.

**글꼴 바꾸기** 수식 모드에서 모든 글자는 수식에 맞는 글꼴로 바뀌어 표현됩니다. 이 때문에 알파벳을 입력하면 이를 변수로 취급하여 변수 글꼴로 바꿔줍니다. 일반 텍스트를 수식 안에서 사용하고 싶다면 텍스트를 \text{...} 안에 넣으면 됩니다.

**그리스 문자 넣기** 수식에는 그리스 문자가 굉장히 많이 들어갑니다. 매우 보편적인 파이부터, 웬만한 그리스 문자는 한 번씩 쓰게 되죠. LATEX은 아주 예쁜 그리스 문자들을 출력해줍니다. 물론 대문자, 소문자 모두 말입니다. 아래 그림을 참고해주세요.

$\alpha$	\alpha	$\theta$	\theta	$\circ$	\circ	$\tau$	\tau
$\beta$	\beta	$\vartheta$	\vartheta	$\pi$	\pi	$\upsilon$	\upsilon
$\gamma$	\gamma	$\Gamma$	\Gamma	$\varpi$	\varpi	$\phi$	\phi
$\delta$	\delta	$\kappa$	\kappa	$\rho$	\rho	$\varphi$	\varphi
$\epsilon$	\epsilon	$\lambda$	\lambda	$\varrho$	\varrho	$\chi$	\chi
$\varepsilon$	\varepsilon	$\mu$	\mu	$\sigma$	\sigma	$\psi$	\psi
$\zeta$	\zeta	$\nu$	\nu	$\varsigma$	\varsigma	$\omega$	\omega
$\eta$	\eta	$\xi$	\xi				
$\Gamma$	\Gamma	$\Lambda$	\Lambda	$\Sigma$	\Sigma	$\Psi$	\Psi
$\Delta$	\Delta	$\Xi$	\Xi	$\Upsilon$	\Upsilon	$\Omega$	\Omega
$\Theta$	\Theta	$\Pi$	\Pi	$\Phi$	\Phi		

Table 1: Greek Letters

이 그림은 <http://web.ift.uib.no/Teori/KURS/WRK/TeX/symALL.html>에서 가져온 것으로 더 많은 정보는 직접 사이트에 들어가서 확인해주세요.

**벡터 표시, 밑줄 등 줄긋기** 벡터를 표시하거나 선분을 표시하는 등 글자 위에 줄을 긋거나 밑줄을 쳐야 할 때가 있습니다. 벡터는 \overrightarrow{...}와 같은 식으로 표현할 수 있습니다. \overline{...}처럼 그냥 선 역시 그릴 수 있습니다. 자세한 것은 표 4를 확인해주세요.

**열 맞춰 쓰기** 수식을 여러 행 쓰다보면 등호와 같이 열을 맞춰 쓰고 싶을 때가 있습니다. \begin{align} 을 사용하면 열을 맞추어 쓰고 싶은 여러 행 수식을 마치 표 작성하듯 쉽게 입력할 수 있습니다. 표 작성과 방식이 굉장히 비슷하기 때문에, 4.6장을 미리 보고 익히셔도 좋습니다. 방식을 설명해드리자면, 먼저 예시를 봅시다. 예시에서 볼 수 있듯 등호, 부등호처럼 같은 열에 맞춰 쓰고 싶은 기호들 앞에 공통된 기호를 하나 입력했습니다. &입니다. 이 기호는 표에서도 비슷한 역할을 하므로 기억해 둡시다. 즉 같은 열에 맞추어 쓰고 싶은 부분 바로 앞에 &를 붙이고, 줄바꿈은 아시는 대로 \\를 이용하시면 됩니다.

**여러 줄 묶기** 혹은 대괄호로 여러 행의 식을 묶어 표현해야 할 때도 있습니다. 여러 개의 식으로 정의되는 함수를 표현한다거나 할 때, 큰 중괄호로 식들을 묶는 방식을 많이 보셨죠? 바로 그것을 쉽게 표현할 수 있다는 겁니다. 예시에도 나와 있듯이 `\[` 안에 `\begin{array}{...}`를 이용하는 방법입니다. 이 환경 역시도 표 작성 방식과 거의 일치합니다. 구체적이고 자세한 것은 역시나 4.6장을 먼저 읽어서 이해하시는 것을 추천드립니다. 간략히 설명을 드리자면, ...에는 묶고 싶은 수식의 열과 각 열의 정렬 방식에 따라 알맞은 문자를 넣습니다. array 환경 앞에는 `\left.`, 그리고 중괄호와 같이 묶는 데 쓸 기호를 입력합니다. array 환경 안에서 묶을 식들을 예시와 같이 입력하고, array 환경이 끝나고 `\right.`을 입력하면 됩니다.

이 외에도 정말 수많은 기능과 command들이 존재합니다. 수식 기호 관련한 command 및 문법 정보는 위키피디아의 TeX 문법<sup>16</sup> 문서를 확인하면 대부분 찾을 수 있습니다.

---

<sup>16</sup>[https://en.wikipedia.org/wiki/Help:Displaying\\_a\\_formula](https://en.wikipedia.org/wiki/Help:Displaying_a_formula)

## 4.6 표와 그림

이제 거의 다 왔습니다. 초 입문서의 끝이 보입니다.

이번에는 표와 그림을 삽입하는 방법에 대해서 알아봅시다. 표와 그림은 LATEX환경에서 ‘떠다니는 개체’라는 개념으로 이해됩니다. 이는 표, 그림등으로 된 하나의 개체가 떠다니듯 존재해 그 위치를 일정한 규칙에 맞춰 유동적으로 결정한다는 의미라고 보시면 됩니다.

언제나처럼 예시부터 볼까요?

```
\documentclass{article}
\usepackage{kotex}
\usepackage{setspace}
\usepackage[pdftex]{color, graphicx}
\usepackage[left=2.5cm, right=2.5cm, top=3cm, bottom=3cm, a4paper]{geometry}
\begin{document}
\double spacing
\noindent 이번에는 표와 그림에 대해서 봅시다.\\
먼저 표는 이렇게 그리면 됩니다.

\begin{table}[hp]
\centering
\begin{tabular}{|c|c|} \hline
 그냥 & 이런\\ \hline
 식으로 & 하면\\ \hline
 표가 & 됩니다\\ \hline
\end{tabular}
\caption{캡션은 이렇게 답니다}
\label{tab:eg}
\end{table}
참 쉽죠?\\
다음은 그림입니다.

\begin{figure}[hp]
\includegraphics[width=0.9\textwidth]{OhMyGirl.jpg}
\caption{그림도 캡션이 됩니다}
\label{fig:eg}
\end{figure}
\end{document}
```

결과는 다음 페이지에서 확인하시면 됩니다.

이번에는 표와 그림에 대해서 봅시다.

먼저 표는 이렇게 그리면 됩니다. 참 쉽죠?

그냥	이런
식으로	하면
표가	됩니다

Table 1: 캡션은 이렇게 답니다

다음은 그림입니다.



Figure 1: 그림도 캡션이 됩니다

그럼 각각 어떤 식으로 삽입하는지 이제부터 알아보도록 합시다.

#### 4.6.1 표

먼저 표부터 살펴봅시다. 표를 삽입하는 것은 두가지 절차로 나누어 생각하면 쉽습니다. 표를 그리기 위한 과정과 표를 일정한 위치에 삽입하기 위한 과정이 그 절차들입니다. 절차를 나눠 천천히 익혀봅시다.

**표 그리기** 표를 그리는 데에는 `\begin{tabular}{pos}`를 이용하면 됩니다. 여기서 pos에 들어가는 것은 위의 수식 입력하기에서 array 환경에서도 봤던 바로 그것입니다. 표의 형태를 미리 정의하는 과정인데, 표가 몇 열로 이루어지는지, 각 열은 어떤 정렬로 할지, 각 열 사이 세로선은 넣을지 말지 결정할 수 있습니다. 예시에서는 `|c|c|`라고 되어 있군요. 여기서 `|`는 세로선을 의미하고 각 알파벳은 하나의 열을 뜻합니다. 특히 `c`는 center, 즉 가운데 정렬을 의미합니다. 이런 식으로 표의 형태를 어느 정도 결정해주는 겁니다. pos에 들어가는 글자들에 대한 전체 정보는 표 5를 참고해주세요. pos까지 결정했다면 표를 그리기 시작하면 됩니다. 각 열은 &로, 각 행은 \\\로 구분하여 작성합니다. 가로줄을 원한다면 각 행 사이에 `\hline`을 입력하면 됩니다. 열 병합을 원한다면 `\multicolumn{number}{pos}{text}`을 이용하세요. number엔 병합할 열 개수, pos는 새로 병합한 열의 형태, text엔 내용을 입력하면 됩니다.

**표 삽입하기** 표를 다 그렸으면 문서에 삽입을 해야겠죠? 예시를 봅시다. `tabular` 환경을 감싸고 있는 것 이 있습니다. `\begin{table}[pos]`입니다. 그려진 표를 위치시키기 위한 환경입니다. 이 환경에서도 pos가 있습니다. 여기서 pos는 표의 위치를 결정하는 부분입니다. h, p, t, b, !로 이루어집니다. 여기서 h는 당장 입력하고 있는 바로 그곳을 뜻합니다. p는 새로운 페이지를 뜻합니다. 다시 말해 떠다니는 개체가 모이는 특정 페이지를 뜻합니다. t는 페이지 맨 위를 뜻합니다. b는 페이지 맨 아래를 뜻합니다. !는 개체를 위치시키는 일정한 규칙을 무시하고 선언한 대로 위치하도록 강제한다는 뜻입니다. 대신 개체, 즉 표가 출력 과정에서 아예 위치되지 못할 수도 있습니다. 이 문자들을 이용하여 '`!hbp`'와 같은 식으로 표를 위치시킬 방법을 선언합니다. `!hbp`라면 기존 규칙을 무시하고 당장 입력하고 있는 곳에 위치시키거나, 그게 불가능 하면 페이지 맨 아래에 위치시키거나, 그것도 불가능하면 새 페이지에 위치시키겠다는 말입니다. `table` 환경에서는 `\centering`으로 표를 가운데에 위치시킬 수도 있습니다. 또 `tabular`로 표 작성이 끝난 뒤 `table` 환경이 끝나기 전에 `\caption{text}`으로 캡션을 달 수도 있습니다. 표를 참조하고 싶다면 캡션 command 바로 뒤에서 `\label{}`을 이용하면 됩니다.

**LaTeX Table Generator** 이정도만 익히면 웬만한 표들은 작성이 가능할겁니다. 그렇지만 아마 이런 식으로만 이야기를 하면 조금 당황스러우실지도 모르겠습니다. 어느 정도 방법은 이해가 되더라도, 표를 이런 식으로 만든다는 게 굉장히 귀찮고 막막하게 느껴지실 수도 있겠죠. 하지만 사실 많은 `LATEX` 사용자들은 표를 만들 때 이런 방법을 사용해 한 칸 한 칸 코딩해 나가지 않습니다. 귀찮기도 하거니와, 굳이 그렇게 해야 할 필요가 없거든요. 표는 그래도 GUI, Graphical User Interface를 이용해서 그리는 게 낫겠죠. GUI를 통해 구현된 표를 `LATEX` code로 바꾸는 방법은 무궁무진 합니다. 엑셀 표를 `LATEX` 수식으로 변환시켜주는 플러그인이나, `LATEX` 표를 만들어 주는 웹사이트 등을 활용해 봅시다. 혹은 `LATEX` 에디터 중 GUI로 표 생성을 도와주는 에디터가 있기도 합니다. 여기서는 간단하게 `Table Generator`라는 웹 페이지를 소개하고 마치도록 하겠습니다. 사용법은 설명할 필요가 전혀 없을 정도로 간단합니다. 이젠 정말 어렵게 느끼실 이유가 없을 겁니다.

### 4.6.2 그림

이제 그림을 넣어봅시다. 그림은 초 입문자 분들께서는 package의 사용이나 그 과정이 이해되기 힘들겁니다. 확장자가 eps나 다른 것인가에 따라 컴파일 과정이 달라지고, package도 조금씩 바꿔서 써야 합니다. 일단 여기서는 보편적으로 쓰이는 일반 확장자들에 대한 방법 위주로 설명을 드리겠습니다.

**package 사용하기** png, jpg, pdf, mps 확장자를 가지는 그림 파일들은 packge의 사용을 필요로 합니다. `\usepackage[pdftex]{color,graphicx}`를 이용합시다. 예시에도 잘 나와 있는 이 package를 이용해야 앞에서 말씀드린 확장자의 그림 파일들을 삽입할 수 있습니다. 뒤에서 그림을 삽입할 때 사용하는 command를 설명해드릴건데, 당연하지만 이 command에는 삽입하려는 그림 파일 이름을 입력해야 합니다.

**그림 삽입하기** 이제 정말 그림을 넣어봅시다. 그림도 어떻게 보면 표처럼 두 가지 절차로 나누어집니다. 그림을 불러오는 과정과 위치시키는 과정입니다.

먼저 불러오는 과정은 `\includegraphics[key=value...]{imagefile}`를 이용하면 됩니다. key에는 height, width등이 들어가며, 각 key의 value를 고정시켜줄 수 있습니다. 예를 들어, `width=0.5\textwidth`라고 해주면 그림의 가로 길이를 문단 가로 길이의 0.5배로 고정시킨다는 뜻이 되는 겁니다. imagefile에는 파일명을 입력하시면 됩니다.

**그림 위치시키기** 다음으로 위치시키는 과정은 기본적으로 위의 table 환경 설명에서 table 대신 figure를 입력하면 됩니다. 조금 더 자세히 다시 설명해드리자면, 그림을 위치시키기 위해 `\begin{figure}[pos]`를 이용한다는 것입니다. 여기서 pos는 table에서와 마찬가지로 h, p, b, t, !로 이루어집니다.

또한 `\includegraphics[key=value]{imagefile}` 뒤에 캡션과 라벨 역시 사용할 수 있도록 해줍니다.

**심화 기능들** 여기서 조금 더 심화적인 기능을 써볼까요? 지금까지의 방법으로는 그림은 새로운 줄 혹은 페이지에 위치해 자리 차지를 엄청 크게 합니다. 마이크로소프트 워드에서 ‘위/아래’라고 되어 있는 레이아웃에 해당한다는 말입니다. 그림을 텍스트가 둘러싸게 위치시키고 싶을 때가 많을 겁니다. 마이크로소프트 워드에서 ‘정사각형’이라고 되어 있는 레이아웃에 해당하도록 할 수 있다는 겁니다.

일단 `\usepackage{wrapfig}`를 preamble에 추가해줍니다.

그리고 그림을 위치시키고 싶을 때 `\begin{wrapfigure}{pos}{width}`를 이용하면 됩니다. 여기서 말하는 pos는 문단의 오른쪽에 위치시킬지 왼쪽에 위치시킬지 결정하는 부분입니다. 왼쪽은 l 또는 L, 오른쪽은 r 또는 R을 입력하면 됩니다. width에는 말 그대로 그림을 넣을 자리의 가로 길이를 설정하는 부분입니다. 주의하세요. 그림의 가로 길이가 아닙니다. 그림을 넣을 자리의 가로 길이입니다.

이번에는 한 줄에 여러 개의 그림을 넣어봅시다. 역시나 package를 추가해야 합니다.

`\usepackage{subcaption}`을 추가해줍니다. 그 다음은 쉽습니다. 기본적으로 그림을 위치시킬 때 쓰는 figure 환경 안에서, 넣고 싶은 그림마다 `\begin{subfigure}{width}`를 써주면 됩니다. width에는 역시나 각 그림이 들어갈 자리의 가로 길이를 써주면 됩니다. 물론 각 그림 하나하나마다 캡션을 달고 참조시킬 수 있습니다. figure 환경에서 캡션 달듯이 subfigure에 달면 됩니다.

역시나 이정도만 알고 계신다면 일반적인 그림 파일들은 삽입할 수 있을 겁니다.

## 4.7 문제가 생기셨나요?

물론 수식을 작성하실 때, 표나 그림을 삽입하실 때에도 문제는 발생할 수 있습니다. 어떤 문제가 생기든 겁먹지 마세요. 문제를 해결하기 위해 다시 돌아왔습니다. 지금부터 하나하나 짚어봅시다.

- align환경을 이용하실 때 원하는 모양대로 안 나오거나 에러가 떴나요? 혹시 줄바꿈을 제대로 해주었는지 확인해보세요. 열을 맞추기 위해 &를 입력하고 나면 그 다음 &이 나오기 전에 꼭 줄바꿈을 해줘야 합니다.
- 수식 모드에서는 텍스트 모드와 달라 주의할 점들이 많습니다. 특히 공백 문자는 전부 무시해버린다는 사실은 꼭 기억해두고 본문에서 설명드린 강제 공백 명령어들을 이용해주세요. 다만 LATEX에서 강제로 디자인에 영향을 주는 것은 언제 어떤 문제가 생길지 모르는 행동입니다. 너무 남용하지는 말아주세요.
- 수식 모드의 공백 기호에서 반각은 \ 입니다. 그냥 역슬래시 하나라 굉장히 당황스럽겠지만, 모든 명령어가 그렇듯 역슬래시 하나만 입력하고 스페이스 혹은 중괄호와 같은 비문자를 입력해주면 됩니다.
- 반대로 텍스트 모드이기 때문에 수식 모드와 구분해야 하는 경우도 있습니다. 생각보다 많은 부호, 기호들이 텍스트 모드에서는 동작하지 않습니다. | 같은 기호들은 전부 수식 모드 안에서만 동작하죠. 이 점 유의해주세요. 만약 에디터로 TeXStudio를 사용하고 계신다면 더더욱 주의해주세요. 쓸 수 있는 명령어들의 자동완성 기능은 이 에디터의 특장점이지만 텍스트 모드에서 쓸 수 없는 수식 모드 전용 명령어들까지 텍스트 모드에서 자동완성 시켜줍니다. 자동완성 된다고 안심하지 마세요.
- 표나 그림에 캡션을 달고 그 캡션에 label을 달게 되는 경우가 많습니다. 이 때 주의할 점이 있습니다. 꼭 \caption{text} 뒤에 \label{text}을 입력해주세요. 순서가 바뀌면 번호가 잘못 매겨지는 경우가 간혹 있습니다.
- 조금 전에 LATEX환경에서 강제로 디자인에 영향을 주는 것이 좋지 못한 행동이라고 말씀을 드렸습니다. 이것이 정말 크게 드러나는 부분이 표, 그림에서 위치선정 부분입니다. ‘떠다니는 개체’라고 불리는 표, 그림은 그 이름 만큼이나 자유롭고 유동적으로 위치가 결정됩니다. 만약 별일이 없다면 LATEX에서 일정한 규칙에 맞게 배치를 해줍니다. 그런데 table이나 figure 환경에서 !를 사용한다거나, 어떻 게든 사용자가 원하는 자리에 두기 위해서 다양한 실험을 하는 분들이 계실 겁니다. 이런 행동, 정말 주의하셔야 합니다. 본문에서 설명 드렸듯이 !를 사용하다가는 LATEX이 개체를 배치할 자리를 못 찾고 아예 삽입이 안 될 수도 있습니다. 또 원하는 자리에 두기 위한 지나친 노력은 대부분 에러, 위닝, 배드박스를 부르는 원인이 됩니다. LATEX의 자리 배정 규칙을 믿고, 어느 정도는 유연하게 받아들이는 것이 상책입니다. 그게 어쩌면 제일 예쁠지도 몰라요.
- 그림을 넣으실 때 파일 확장자까지 입력해주지 않는다면 아주 약간의 문제가 생길 수 있습니다. 당장은 문제가 없어보일 수도 있지만 확장자까지 꼭 적어주시기 바랍니다. 참고로, eps 확장자의 그림 파일을 삽입하고 싶다면 package의 pdftex 부분을 driver로 바꿔주거나, 확장자를 입력하지 않으면 됩니다. 역시나 전자의 방법을 추천드립니다.

여기에 없는 문제들은 친구에게 물어보거나, 구글에 검색해 보세요. 아마 여러분과 같은 어려움을 겪고 있는 사람들이 상당히 많을 겁니다.

## 5 Appendix

### 5.1 documentclass

article	과학 저널, 프레젠테이션, 짧은 보고서, 프로그램 문서, 초 대장 등의 글을 쓸 때 사용합니다.
IEEEtran	IEEE <sup>17</sup> 양식에 맞도록 글을 쓸 때 사용합니다.
proc	article 중 회의록을 작성하기 위해 사용합니다.
report	Chapter를 포함하는 긴 보고서, 작은 책, 학위 논문 등에 쓰입니다.
book	책자를 만들 때 사용합니다.
slides	슬라이드를 만들 때 사용합니다. big sans serif 폰트를 사용합니다.
memoir	book class보다 개선된 책자 레이아웃입니다.
letter	letter를 쓸 때 사용합니다.
beamer	발표자료를 만들 때 사용합니다.

표 1: Document Classes

10pt, 11pt, 12pt	문서의 글자 크기를 결정합니다. 아무 옵션도 적용하지 않으면 10pt로 간주합니다.
a4paper, letterpaper, ...	용지 크기 및 모양을 설정합니다. 기본은 letterpaper입니다. 다만 상황에 따라 기본 크기가 다르게 설정될 수 있으므로, 될 수 있으면 직접 명시하는 것이 좋습니다.
fleqn	기본 가운데 정렬로 되어 있는 수식을 왼쪽정렬합니다. 후에 자세히 설명해드리겠지만, 수식만 따로 쓰는 경우 기본 가운데 정렬입니다. 이 때 이 option이 있으면 왼쪽으로 정렬됩니다.
leqno	역시 후에 자세히 다루겠지만, 수식을 쓸 때 줄마다 numbering이 됩니다. 이 때 기본 오른쪽에 numbering이 뜨는 것을 왼쪽으로 바꿔줍니다.
titlepage, notitlepage	표지를 따로 만들지 말지를 결정합니다. article은 없는 것이, report나 book은 있는 것이 기본입니다.
twocolumn	2단 편집을 할 수 있도록 합니다. 왼쪽 오른쪽 단을 나눠서 쓰는 것을 생각하면 됩니다.
twoside, oneside	양면인쇄할지 단면인쇄할지 결정합니다. article이나 report에서는 단면인쇄, book에서는 양면인쇄를 기본으로 합니다. 이 설정은 프린트 설정에 따라 적용이 안 되거나 문제가 생길 수 있으니 주의해주세요.
landscape	용지 방향을 가로로 바꿔줍니다.
openright, openany	새로운장을 항상 오른쪽 페이지에서 시작할지, 바로 다음 새 페이지에서 시작할지 설정합니다. article에서는 chapter가 존재하지 않아 쓸 수 없습니다. report에서는 새 페이지, book에서는 오른쪽 페이지가 기본입니다.

표 2: Document Class Options

## 5.2 수식 모드 공백 기호

\,	전각의 $\frac{3}{18}$
\:	전각의 $\frac{4}{18}$
\;	전각의 $\frac{5}{18}$
\	반각
\quad	전각
\quad\quad	전각의 2배

표 3: 강제 공백 기호들

## 5.3 글 위, 아래 기호

\overrightarrow{text}, \overleftarrow{text}	text 위에 오른쪽, 왼쪽 화살표를 그립니다.
\overleftrightarrow{text}	text 위에 양쪽 화살표를 그립니다.
\overline{text}, \underline{text}	text 위, 아래에 수평선을 그립니다.
\overbrace{text}, \underbrace{text}	text 위, 아래에 중괄호를 그립니다.

표 4: 글자 위, 아래에 그려지는 기호들

## 5.4 표 열의 내용 정렬하기

l	왼쪽 정렬된 열
c	가운데 정렬된 열
r	오른쪽 정렬된 열
p{'width'}	'width'에 맞춰 양쪽정렬 및 자동줄바꿈을 하며 위쪽정렬합니다.
m{'width'}	'width'에 맞춰 양쪽정렬 및 자동줄바꿈을 하며 가운데정렬합니다.(array package 필요)
b{'width'}	'width'에 맞춰 양쪽정렬 및 자동줄바꿈을 하며 아래쪽정렬합니다.(array package 필요)
	세로줄
	세로 두 줄

\*'width'는 열의 너비로 cm, pt 혹은 textwidth command를 사용하여 나타냅니다.

표 5: 정렬하는 명령어들