LaTeX のすゝめ

介川 侑大

2020 年 12 月 21 日

目次

1 LaTeX とは 1
2 なぜ LaTeX なのか 3
3 LaTeX の書き方 4
3.1 基本的な書き方 4
3.2 効率的な書き方 5
4 パッケージの紹介 6
4.1 hyperref 6
4.2 titleps 7
4.3 siunitx 7
4.4 cleveref 7
4.5 tcolorbox 7
4.6 その他の主なパッケージ 7
5 おわりに 8
付録 A パッケージの作り方 9
A.1 パッケージの構成 9
A.2 パッケージの書き方 9
付録 B 文書のレイアウトの変更 10
B.1 余白設定 10
B.2 見出しのデザイン 10
B.3 ヘッダ、フッタの設定 11
B.4 目次の生成 11

1 LaTeX とは

まず、LaTeX とは何かを説明する前に、TeX について説明する必要がある。奥村氏の「美文書作成入門」という本では、TeX は次のように説明されている。

TEX は、組版ソフトです。

組版（typesetting）は印刷用語で、活字を組んで版（印刷用の板）を作ることを意味します。TEX は、コンピューターでテキストと図版をうまく配置して、版にあたるもの（PDF または PostScript ファイル）を出力する（タイプセットする）ためのソフトです。[*1]

TEX は処理方式としてバッチ処理を採用している。バッチ処理は、処理方式としてバッチ方式やマークアップ言語方式などと呼ばれている。この処理方式は、「作りたい文書を特定の言語によって記述し、それを文書作成システムに与えて出力を得る」[*2]というものである。TEX の場合は具体的に、テキストファイル（.tex）に作りたい文書のもととなるコマンドを記述し、それをコンパイルすることで出力を得る。今では、出力形式のほとんどが PDF である。TEX の処理の流れを図で表すと図 1 のようになる。

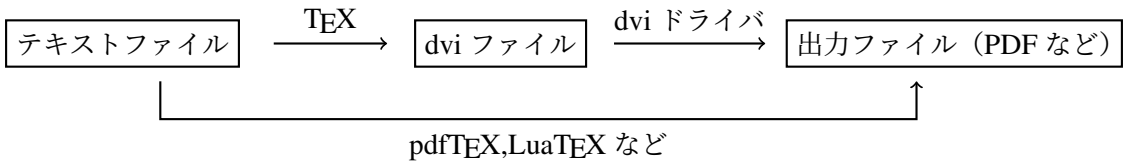

図 1：処理の流れ

日本で広く使われている pTEX や upTEX は図 1 のように dvi ファイルとして出力し、dvi ドライバ（dvipdfmx など）を用いて出力ファイル（PDF など）を得る。もともとの TEX もこのように処理が行われる。一方で、TEX の後に開発された pdfTEX と LuaTEX,XeTEX は中間ファイルとして dvi ファイルを出力せずに直接 PDF を出力することができる。pdfTEX は欧米語圏で主に使われているエンジン（マクロと区別するため TEX 本体をエンジンと呼ぶことがある）で、LuaTEX はその後継として注目されている。

LATEX[*3]は TEX のマクロ集として書かれている。マクロとは、複数のコマンドを一つにまとめたコマンドのことである。つまり、LATEX は TEX を用いて文書作成をより簡単にするものである。LATEX はエンジンによって微妙に異なる。例えば、pdfLATEX は pdfTEX 用に書かれた LATEX である。

以上の話は少し難しいが、LATEX で文書を作る際にはあまり必要にはならない。なぜなら、TEX Live が簡単に LATEX を使える環境を与えてくれるからである。TEX Live とは、TEX, LATEX 2_ε, BibTEX, ConTEXt, dvi ドライバなどの実行プログラムと、TEX を拡張する多数のマクロパッケージやフォントなどが収録された TEX ディストリビューションである。TEX Live はインターネットから簡単にインストールすることができる。TEX Live のインストール方法やアップデートの詳しい内容は、TEX Live ガイド 2020 [*4] で説明されている。パソコンの OS が Windows の人ならほとんどの人が TEX Live を使っている。私自身も TEX Live を利用して不自由なく LATEX を使えているので、TEX Live を利用することを勧める。また、最近では Cloud

[*1] [1]p.1
[*2] [2]p.2
[*3] 現在 LATEX が指すのは、古い LATEX の後にできた LATEX 2_ε のことである。
[*4] The TEX Live Guide の日本語版 https://www.tug.org/texlive/doc/texlive-ja/texlive-ja.pdf

LaTeX[*5]というオンラインで LaTeX が利用できるサイトができたため、Cloud LaTeX を利用する人も増えている。

2　なぜ LaTeX なのか

LaTeX を使って文書を作成することには、様々なメリットがある。ここでは主に 3 つの LaTeX を使うメリットを挙げる。

一つ目のメリットは、数式の組版である。LaTeX では、理数系の教科書でよく目にするような美しい数式を作ることができる。また、数式の入力形式も非常に優れていて、TeX 記法は数式記述の事実上の標準となっている。例えば、Wikipedia の数式記述、Web でよく使われる MathJax などは TeX 記法が使われている。Word でも TeX 形式で数式を入力することができる。ただし、TeX の数式フォントは利用可能ではない。LaTeX を用いれば、簡単に美しい数式を作ることができる。

二つ目のメリットは、処理方式がバッチ方式であることである。実は、処理方式は大きく二つに分けることができる。一つはバッチ方式で、もう一つは WYSIWYG 方式である。バッチ方式は 1 ページで述べたとおりである。WYSIWYG 方式とは、画面上に表れている文書がそのまま出力結果と常に一致しているように処理する方式である。文書作成によく使われるソフトである Word は WYSIWYG 方式を採用している。では、一体バッチ方式にはどのような特徴があるのか。その一つに、文書の論理構成とデザインを分離できることが挙げられる。多くの WYSIWYG なワープロソフトでも、見出しの作成などの内容の論理構成とデザインを分離する機能が備わっているが、使いこなすのは簡単ではない。その点、LaTeX では`\section{title}`とするだけで簡単にデザインと分離した形で見出しを作ることができる。ただし、デザイン性は WYSIWYG なワープロソフトの方が優れている。また、相互参照機能もバッチ方式の優れている点である。Word も相互参照機能が備わっているが、圧倒的に LaTeX の方が簡単に相互参照することができる。

三つ目のメリットは、参考文献を楽に引用できることである。LaTeX では BiBTeX を使って文献リストを自動で出力することができる。BiBTeX とは LaTeX における文献データベースである。BiBTeX は、文献情報が書かれた.bib ファイルを読み込み、文書内で引用することで自動で文献リストを出力できる。.bib ファイルは、例えば Google Scholor や東工大図書館でも入手することができ、また自分で BiB 形式に従って作ることも可能である。

その他にも、LaTeX は章や図、数式、脚注、参考文献などの番号を自動的に付してくれたり、目次や索引などの処理を自動で行ってくれるなど、多くの便利な機能がある。さらに、LaTeX には拡張機能を与えるパッケージが多く用意されていて、LaTeX でできることは無限大である。しかし、LaTeX はコマンドを入力する面倒や、書き初めに毎回クラスファイルやスタイルファイル（マクロパッケージ）を読み込む手間など、不便な点もある。これらのデメリットを克服する方法は次節に説明する。

[*5] https://cloudlatex.io/

3 LaTeX の書き方

このセクションでは、LaTeX の基本的な書き方と、私が実際にやっている効率的な書き方を簡単に紹介する。[*6]前提として、TeX Live 2020 をインストールして LaTeX が使える環境であるとする。文書の内容はテキストファイル（拡張子は.tex）に書く。基本的にエディタはどれでもよいが、TeX Studio というエディタを用いるとコマンド補完が充実しているので、より効率よく書き進めることができる。また以降の内容は、コンパイルに LuaLaTeX を用いていることを前提としている。LuaLaTeX は最先端の LaTeX で、日本語周りの開発も進んでおり、最近では簡単に日本語で LuaLaTeX を利用できるようになった。多くの場面で LuaLaTeX を用いるメリットがあるので、コンパイラには LuaLaTeX を用いることを勧める。コンパイルの仕方はエディタによって異なるので説明は省く。

3.1 基本的な書き方

LaTeX で文書を作るときは、いくつかルールがある。文書は次のような順番で書く。

1. `\documentclass[`**オプション**`]{ltjsarticle}`と書き、文書形式を指定する。ltjsarticle は LuaLaTeX 用で日本語用の記事を作るためのクラスファイルである。他にも ltjsreport や ltjsbook などがある。オプションはクラスファイルの説明書に説明されている。
2. `\usepackage[`**オプション**`]{`**パッケージ名**`}`コマンドを用いてマクロパッケージを読み込む。パッケージを読み込まないとできないことがあるので、必要なパッケージはここで全て読み込む。例えば、図を挿入するのに必要な graphicx パッケージ、高度な数式を入力するのに必要な amsmath パッケージなどである。
3. パッケージの設定、マクロの定義、再定義、文書情報の設定などを記述する。パッケージの設定はパッケージの説明書をもとに記述する。[*7]文書情報は`\title{`**タイトル**`}\author{`**筆者の名前**`}\date{`**日にち**`}`のようにして指定する。これはタイトルを作るときに必要である。マクロは必要ならここで作る。
4. document 環境内（`\begin{document}`と`\end{document}`の間）に文書の内容を書く。タイトルを作りたければ`\maketitle` を最初に書く。

文章の書き方には以下のようなルールがある。

- 段落と段落の間は一行空ける。
- 強制的に改行するときは、\\コマンドを使う。
- 半角スペースはいくつ重ねても一つとして処理される。半角スペースを明示的に出力するには~を半角スペースとして用いる。
- 相互参照するするには、参照される対象（図や表、節など）の直後（または同じ環境内）に

[*6] 詳しい書き方の基本は http://www.tufs.ac.jp/blog/is/g/sodan/texman.pdf を参照。また、LaTeX 2_ε で定義されているほぼ全てのコマンドは http://www.tug.org/texinfohtml/latex2e.html に解説されている。

[*7] ほとんどの場合必要ない。

例 3.1: 図の挿入マクロ

```
\newcommand{\myfig}[4][width=0.8\linewidth]{%
\begin{figure}[htbp]
\centering
\includegraphics[#1]{#2}
\caption{#3}\label{#4}
\end{figure}%
}
\myfig{titech.pdf}{caption}{label}
```

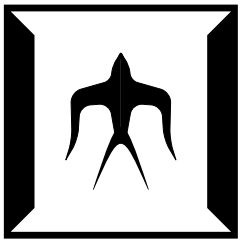

図 1：caption

\label{key}と書き、参照したい箇所に\ref{key}とする。こうして番号が参照できる。

以下に具体例を挙げる。

3.2 効率的な書き方

私が普段レポートを書く際にやっている、LaTeX での効率的な書き方を主に 2 つ紹介する。

一つは、マクロを作成することである。自分でよく行う作業を一つのコマンドにすることは生産性の向上につながる。マクロの定義は\newcommand{**コマンド名**}[**引数の数**]{**定義内容**}のように書く。例えば、よく数式の中で（）を用いるとき、括弧内の数式のサイズに合わせて\left(**数式**\right) のように囲むのは面倒である。これを\newcommand{\paren}[1]{\left(#1\right)}のようにマクロにすれば、一つのコマンドで囲むことができ、エディタ内の数式もごちゃごちゃしなくて済む。その他に、LaTeX での図の挿入は、図を挿入する機械が多いこともあり、マクロにすることに大きな意味がある。例えば、例 3.1 のようなマクロである。\linewidth は一行の長さであり、図の幅を\linewidth の 0.8 倍にすることで、文書の大きさやレイアウトに合わせて図を挿入することができる。このように、長さを表すコマンドは他にも色々あり、なるべくこれらを用いることは重要である。[*8]

さらに、if 文を用いればより高度なマクロを作ることができる。(例 3.2)

例 3.2: 数式マクロ

```
\newcommand{\bibun}[3]{%
\if 1#1
\frac{\mathrm{d}#2}{\mathrm{d}#3}
\else{%
\frac{\mathrm{d}^{#1}#2}{\mathrm{d}#3^{#1}}
}\fi%
}
\[\bibun{k}{f}{x},\bibun{k-1}{f}{x},\ldots,\bibun{2}{f}{x},\bibun{1}{f}{x}\]
```

[*8] TeX で使える長さの単位 (pt,em,ex など) も重要である。

$$\frac{\mathrm{d}^k f}{\mathrm{d}x^k}, \frac{\mathrm{d}^{k-1} f}{\mathrm{d}x^{k-1}}, \ldots, \frac{\mathrm{d}^2 f}{\mathrm{d}x^2}, \frac{\mathrm{d}f}{\mathrm{d}x}$$

二つは、\input コマンドを活用することである。毎回プリアンブル（document 環境の前の部分）にパッケージを読み込みやマクロの定義を記述するのは非常に手間がかかる。\input{**ファイル名**}は別のファイルの内容をそのまま書かれた場所に挿入するコマンドである。つまり、別のファイルにパッケージの読み込みやマクロの定義を記述しておけば、\input コマンドで挿入するだけで済む。TEX Live 2020 からは\input コマンドに絶対パスを使用できるようになったため、パッケージの読み込みやマクロの定義を記述してあるファイルを決まった場所に保存しておけば、毎回 input するだけで文章を書き始めることができる。ただし、例えば Ti*k*Z などは重いパッケージなので使うときだけ読み込むようにし、ファイルにまとめるのはよく使うようなパッケージに限る方が良い。さらに、TEX Studio を用いて文書のテンプレートを作っておけばより楽に文書を書き始めることができる。

その他にも、パッケージの利用は論文執筆の効率化に効果的なことがある。例えば、cleveref や siunitx などである。今挙げた二つのパッケージは次節で紹介する。

4　パッケージの紹介

LATEX には多くのパッケージが存在する。TEX Live には、数千にも及ぶパッケージが収録されており、その他にも一般に公開されているパッケージも多数ある。ここでは TEX Live に収録されているパッケージの中で個人的に特に魅力を感じたパッケージを紹介する。TEX Live に収録されているパッケージの多くは説明書が付属している。コマンドラインに

```
$ texdoc <keyword>
```

と書けば、keyword に関連する説明書が適当なビューアにより表示されるので、詳しいパッケージの内容は説明書を参照してもらいたい。また、TEX Live には TeX Live Manager というソフトが付属していて、これを使うとパッケージやフォント、エンジンなどのの一覧を参照することができ、またそれらのアップデートも容易に行うことができる。

4.1　hyperref

このパッケージは必須といっていいほど重要なパッケージである。hyperref の主な役割は、PDF にハイパーリンクを付けることである。本文書もこのパッケージを使って作成した。そのため、参照した番号や URL はすべてリンク付きになっている。また、しおり（ブックマーク）も自動的に生成される。このパッケージのオプションの数は非常に多く、面倒な方は**??**にあるような設定にするとよい。また、リンク付きで URL を貼り付けることができる\url{URL}コマンドもある。

4.2 titleps

titleps は、ヘッダやフッタを作るためのパッケージである。よくヘッダとフッタを作るのに用いられるのは funcyhdr である。ネットでヘッダを作る方法を調べれば必ず funcyhdr パッケージが出てくる。しかし、個人的には titleps パッケージの方が楽にヘッダを作ることができるし、このパッケージはより高度なこともできる。例えば、既存のページスタイル（plain や empty など）を再定義することができる。姉妹関係にある titletoc, titlesec パッケージはそれぞれ、目次、見出しのデザインを編集できるマクロが用意されている。

4.3 siunitx

科学技術論文を作る際に重要視されるのは、物理量の単位である。単位は原則として国際単位系—SI—を使う必要がある。siunitx は正しく単位を記述するのに便利なパッケージである。例えば、`\SI{2.0e5}{\angstrom}`と入力すると 2.0×10^5 Å と出力される。このように、数値の入力も、特殊な単位の入力も簡単である。このコマンド以外に`\num{number}`,`\si{unit}`,`\ang{degrees}`などがある。

4.4 cleveref

このパッケージは文書作成を効率化する上で重要なものである。図や章を参照するとき、`\ref` だと番号しか出力されない。しかし、cleveref パッケージを使うと、図 1 や第 1 章のように自動で参照元の種類を判断して出力してくれる。コマンドは、`\cref{label}`,`\namecref{label}`,`\cpageref{label}`が与えられており、複数のラベルをカンマ区切りで引数に含んでもよい。日本語には対応してないが、日本語化できるようになっている。そのセットアップ作業の仕方は、ネットに解説されている。（例えばこのサイト）

4.5 tcolorbox

tcolorbox の主な役割は文章などを囲む枠を作ることであり、私が一番魅力的だと感じたパッケージである。Ti*k*Z パッケージを基に枠を生成するため多少重いパッケージである。Ti*k*Z は LaTeX で描画できるようにするパッケージで、かなり高度な描画を可能とする。マニュアルは 1000 ページにもおよび、使いこなすのは難しい。しかし、tcolorbox は枠を作ることに特化した形で、簡単に一からオリジナルの枠を作ることができる。tcolorbox のマニュアルは 500 ページにおよぶ大作であるが、中身は具体例に富み非常に分かりやすい。tcolorbox に関する日本語の情報は非常に少ないので、マニュアルを参考にするとよい。

4.6 その他の主なパッケージ

表 1：パッケージ一覧

パッケージ	概要
geometry	ページの余白設定。
luatexja	LuaTEX を日本語で使うのに必要。ltjsclasses に含まれる。
graphicx	画像を処理する。図の挿入に必要。
amsmath	数式を記述する。
mathtools	amsmath を用いた数式記述。
amssymb	特殊文字
caption	図に説明を付ける。
subcaption	一つの caption に複数図を挿入。
xcolor	文字に色を付ける。
wrapfig	図に文章をオーバーラップさせる。
float	図の配置をコントロール。
pdfpages	PDF を結合。
easylist	箇条書きを簡単に重ねる。
enumitem	箇条書きを細かくコントロール。
fontawsome5	アイコンのような文字を出力。
xltabular	tabularx と longtable を合わせたパッケージ。高度な表を作成。
attachfile2	PDF にファイルを添付。
mhchem	化学式を記述。
listings	プログラム言語を挿入。
minted	プログラム言語を挿入。
pdfcomment	PDF の注釈を挿入。
bm	ベクトル表記で使われる太字斜体の文字を記述。
bookmark	ブックマーク（しおり）を作成。
ulem	アンダーラインを引く。
comment	複数行にわたってコメントアウト。
nicematrix	行列（線形代数）の要素を綺麗に記述。
xpause	高度なマクロを作成。
ifthen	場合分け、繰り返しを行う。
subfiles	文章を複数のファイルに分けて編集。

5 おわりに

ここまで、LATEX の基本的な情報から LATEX を使うメリット、文書作成に役立つ情報などを紹介してきた。特に、文書を効率的に作る方法は重要な知識である。しかし、多くの人にはあまり知られていない。今後は、そのような情報を発信していくことを目的に活動していこうと

考えている。さらに、本書はまだまだ詳しい説明が十分でないので、これからも内容を書き足していこうと考えている。[*9]

付録 A　パッケージの作り方

本文では\input{filename}を使うことを勧めたが、パッケージの作成はそれほど難しくなかったので紹介する。まずスタイルファイル（パッケージ）を置くディレクトリを作成し、そのディレクトリを任意のディレクトリから\usepackage で読み込めるよう設定する。やり方は、コマンドプロンプトを開き、

```
$ mkdir C:\texlive\2020\texmf-dist\tex\lualatex\hoge
$ cd C:\texlive\2020\texmf-dist\tex\lualatex\hoge
$ mktexlsr
```

と書けばよい。作ったパッケージはここに保存する。スタイルファイルの拡張子は.sty であり、普段作成している tex ファイルの拡張子.tex を.sty に変更するだけで基本的にパッケージとして扱うことができる。

A.1　パッケージの構成

パッケージの構成は以下のようなものである。

1. **Identification**
   パッケージが対応する LaTeX のバージョンを示し、パッケージ情報を記述する。
2. **Preliminary declarations**
   オプションに必要な他のパッケージの読み込み、コマンドの定義を行う。
3. **option**
   オプションの宣言、処理を行う。
4. **More declarations**
   パッケージの内容を記述する。

A.2　パッケージの書き方

スタイルファイルの最初に、パッケージの情報を記述する。

例 付録 A.1: Identification

```
\NeedsTeXFormat{LaTeX2e}[2020/12/04]
\ProvidesPackage{本パッケージ名}[2020/12/04]
```

[*9] https://yudaisukegawa.github.io/latex/

ブラケット内の日付は任意である。例では、パッケージ「本パッケージ名」が LaTeX 2_ε を必要としていることを意味している。次に外部からパッケージの読み込みを行う。コマンドは \RequirePackage を用いるが、機能も使い方も\usepackage とほぼ同じである。次に、オプションの設定を行う。

例 付録 A.2: Options

```
\DeclareOption{オプション名}{オプションが選択されたときに実行するコード}
\ExecuteOptions{default option list}
\ProcessOptions\relax% 選択されたオプションの実行
```

最後に記述したいパッケージの内容を書く。

付録 B　文書のレイアウトの変更

LaTeX では、とくにユーザーが設定せずとも綺麗に文書が仕上がる。しかし、ときには仕上がりが気に入らず自分で変えたいと思うこともあるが、デザインの変更となると途端に難易度があがってしまう。LaTeX にはデザインの変更をなるべく簡単にできるようにしたパッケージがあるので、それらを使ったレイアウトを変更する方法を紹介する。

B.1　余白設定

余白設定には geometry パッケージが便利である。

B.2　見出しのデザイン

見出しのデザインの変更には titlesec パッケージを使う。基本的な使い方は、このウェブサイトを参照。オプションの explicit を追加しておき、見出し本体を#1 で表せるようにしておくとよい。この場合、コマンド\titleformat の使い方に注意する必要がある。

例 付録 B.1: titleformat

```
%\usepackage[explicit]{titlesec}
\titleformat{\section}[hang]{\Large}{\thesection}{1em}{#1}[]
\titleformat{\section}[hang]{\Large}{\thesection\quad#1}{0em}{}[]
\titleformat{\section}[hang]{\Large}{}{0em}{\thesection\quad#1}[]
\titleformat{\section}[display]{\Large}{\thesection\quad#1}{0em}{}[]
\titleformat{\section}[display]{\Large}{}{0em}{\thesection\quad#1}[]
```

上の例では、ほぼ同じ出力が得られるが、下の 3 つの出力は見出し周りのスペースに違いがある。同じ出力が得られる上二つに対し、三つ目は左に変な空白が、四つ目は下に余白が、五つ目は上に余白ができてしまう。かなり大胆な見出しをデザインしたければ、二つ目の使い方

を選択するとよい。ただし、Overfull hbox...（一行に収まらず、分け方もわからないのではみ出してしまうよ）といった警告が出ることがあるので、気になる人は 0em の部分に大きな負の値うを入れるとよい。

例 付録 B.2: 見出しのデザイン

```
%\usepackage[explicit]{titlesec}
\titleformat{\section}[hang]{\Large\bfseries}{\begin{tikzpicture}
\pgfmathparse{100*rnd}
\edef\tmp{\pgfmathresult}
\pgfmathparse{100*sin(rnd*pi/2 r)}
\edef\tmq{\pgfmathresult}
\filldraw[line width=3pt,draw=cyan!\tmp!magenta!\tmq!yellow,
fill=white,rounded corners] (0,0) rectangle (\linewidth-3pt,1);
\node[] at (\linewidth/2,0.5) {\thesection~~#1};
\node at (\linewidth-2em,0.5) {\small \if@link
\hyperlink{mokuji}{\color{magenta}目次へ戻る}\else \fi};
\end{tikzpicture}
}{-10pt}{}[]
\titlespacing{\section}{0pt}{4mm}{2mm}
```

B.3 ヘッダ、フッタの設定

ヘッダ、フッタの設定には、titleps パッケージが便利である。[*10]titlesec パッケージと併用する場合、titlesec パッケージのオプションに pagestyles を追加するだけで titleps パッケージの読み込みは必要ない。

B.4 目次の生成

目次の出力形式を変更するには、titletoc パッケージを使う。このパッケージの使い方の前に、LaTeX でどのように目次がつくられるのか簡単に説明する。コンパイルする際に出力されるサブファイルの中には、拡張子が.toc であるファイルがあり、目次はこのファイルに基づいて生成される。このファイルは、\addcontentsline{toc}{section}{**見出し内容**}によって生成されている。例えば、\chapter の中では、次のように書かれている。

例 付録 B.3: addcontentsline

```
\addcontentsline{toc}{chapter}%
{\protect\numberline{\@chapapp\thechapter\@chappos}#1}%
```

[*10] fancyhdr パッケージでもよいが扱いづらい。

一つ目の引数は.toc の拡張子のファイルに見出し情報を記述することを意味している。二つ目の引数は、chapter(他 section,　subsection など) の形式[*11]に従って本文中の目次に出力することを意味している。これによって、toc ファイルには例えば次のように記述される。

例 付録 B.4: toc ファイル

```
\contentsline {chapter}{\numberline {第 1 章}ガンマ関数}{1}{chapter.1}%
\contentsline {chapter}{\numberline {第 2 章}ベータ関数}{3}{chapter.2}%
\contentsline {section}{\numberline {2.1}定義}{3}{section.2.1}%
                                        ⋮
```

目次はこのファイルをもとに、\@starttoc{toc}という\tableofcontents のコアになるコマンドによってリストアップされる。他の例では、\listoffigures と \listoftables は、それぞれ.lof、.lot ファイルに書かれている図目次、表目次の内容を \@starttoc{lof},\@starttoc{lot}によって出力している。目次のデザインを変えるには、\@starttoc を用いて\tableofcontents を再定義するのと、リストアップ形式 (chapter,section など) を再定義するの二つ行う。前者は簡単に行うことができる。後者は、titletoc パッケージの\titlecontents{section}[leftmargin]{above-code}{numbered-entry-format}{numberless-entry-format}{filler-page-format}[bellow-code] コマンドを使って変更する。例えば次のようにする。

例 付録 B.5: 目次のデザインの変更

```
%\usepackage{titletoc}
\titlecontents{section}[1.5em]{}{\makebox[2em][l]{\large\thecontentslabel}
\large}{\hspace*{2em}\large}{\titlerule*{ }\large\thecontentspage}
\titlecontents{subsection}[3.5em]{}{\makebox[2.5em][l]{
\thecontentslabel}}{}{\titlerule*{. }\thecontentspage}
\makeatletter
\renewcommand{\tableofcontents}{%
\noindent\textbf{\LARGE 目次}\bigskip\@starttoc{toc}\bigskip}
\makeatother
```

また、titletoc パッケージでは小目次を作る機能が備わっている。注意すべきは、hyperref パッケージの前に titletoc パッケージを読み込む必要があることである。小目次を作るには次のようにする。

[*11] LaTeX には\l@chapter,\l@section のようなコマンドが定義されていて、二つ目の引数によってこれらのコマンドが選択され、その定義に従って目次が生成されている。

例 付録 B.6: 小目次の作成

```
\startcontents[name]
\printcontents[name]{prefix}{start-level}[toc-depth]{toc-code}
```

name には、章内に小目次を作るなら chapter、部内なら part とするが、書かなくても問題ない。prefix には、例えば l を入れると、出力形式に lchapter, lsection が選択される。[*12]何も入れなければ定義済みの出力形式（chapter, section など）が選択される仕組みになっている。start-level には 0(chapter) や 1(section) などと数字を入れる。toc-code にはとくに書く必要はない。章見出しの後に小目次を作りたければ以下のようにする。

例 付録 B.7: 小目次付きの章見出し

```
%\usepackage[explicit]{titlesec}\usepackage{titletoc}
\titleformat{\chapter}[display]{\normalfont\huge\bfseries}{\thechapter}{1em}{
#1}[\vspace{1em}{\normalsize\normalfont\makebox[\linewidth][c]{目次}
\startcontents[chapters]
\printcontents[chapters]{}{1}[2]{}\bigskip]
```

参考文献

[1] 奥村晴彦 黒木裕介, LaTeX 2ε 美文書作成入門, 改訂第 7 版, ser. LaTeX 2ε 美文書作成入門 (ラテック・ツー・イーびぶんしょさくせいにゅうもん). 東京, Japan: 技術評論社, 2017.

[2] 大野義夫, TEX 入門. 共立出版株式会社, 1995.

[*12] この場合、\titlecontents{lsection}... のようにして lsection という出力形式を定義しておく必要がある。