2016 年 4 月 22 日

須田 宇宙

TEX の使い方

目次

1 はじめに

この文章は，研究室のゼミでの報告書を，TEX を使って作成するためのひな形です．とりあえず使えるようになるため，要点を絞ってサンプルを掲載します．目次が付いていますが，ゼミでの報告書には目次は不要です．

まずは以下のコマンドを使って，自分のコンピュータにこのファイルと関連ファイルを取り込みましょう．

```
$ git clone http://git.mesh.cx/gitbucket/git/suda/semi.git
```

2 TEX とは？

TEX は，文章を作成する際に使用するソフトウェアとファイルフォーマットの総称です．HTML などと同じように，特定のルールにしたがってマークアップ（HTML でいうタグ）を含んだ文章を記述し，そのファイルをコンパイルすることで，見栄えのする文章が出力されます．出力ファイルの型式として，古くは DVI という型式や PostScript 型式がよく使われてきましたが，昨今では PDF を使用することが多いです．

ワープロソフトと比較して，無料である，数式に強い，殆どのコンピュータで使用できることなどが特徴です．しかも，記述するファイルはテキストファイルなので，普段親しんでいるテキストエディタを使用できますし，軽量です．また，テキストファイルなのでバージョン管理ツールとの相性も抜群です．

元々，クヌースさんという方が，自分の書いた本の仕上がりに満足できずに作成したソフトウェアなので，レイアウトや数式に関しては抜群の美しさを誇ります．反面，記述ルールを覚えるまでが面倒なのと，ルールに則っていないとエラーが表示され肝心の出力ファイルを得られません．本研究室では，ちょっとした文章から卒業研究の梗概，論文などの記述まで幅広く使っていきます．

3 基本的な使い方

3.1 TEX のサンプル

TEX のサンプルを見てみましょう．拡張子は「.tex」です．

hello.tex

```
\documentclass[a4j,10pt]{jsarticle}
\begin{document}
こんにちは世界
\end{document}
```

1 行目では，これから作成する文章の型式を指定しています．具体的には，用紙サイズが A4 で，文字サイズが 10 ポイント，最後の「jsarticle」で日本語の一般的な文章であることを示しています．jsarticle 以外には，論文などの長い文章を記述する jsbook があります．

TEX では，段落の区切りを表すために，文章と文章の間を一行開けてください．通常の改行は無視されます．通常は，文章は 1 行毎に改行しておいた方が，後で編集しやすいです．

3.2 TEX の環境

通常，TEX を使用する際にはテキストエディタとコンパイラ*1を使いますが，直感的でないので最初は難しいと感じるはずです．そこで，本研究室では TeXShop を標準の TEX 環境として利用します．また，複雑な数式を記述する際には LaTeXiT を使用します．

TeXShop を起動すると「名称未設定-1」と書かれたウィンドウが出現します．上の方に並んでいるボタンは，後で使用します．まずは，ボタンの下の白い部分に 3.1 で紹介した hello.tex を入力してみましょう．入力後に「タイプセット」ボタンを押すと，コンパイルが始まります．もし，ファイル名を付けていなければ保存のためのダイアログが表示されるので，今回は先ほどダウンロードした semi ディレクトリに好きな名前で保存してください．すると，コンパイルが実行され，入力ミスがなければ出力結果が画面に表示されるはずです．

エラーの場合は console ウィンドウが表示され，エラーの発生した行番号と内容が表示されるので，ソースコードを修正してください．稀に，文章構造に関わる致命的なエラーが起きた場合は，きちんと修正して再度タイプセットを実行しても，いつまでたってもエラーのままの場合があります．その場合，console の「作業ファイルを削除」をクリックすると治る場合があります．

3.3 機能を拡張する

TEX では，複雑な記述をまとめるためにマクロと呼ばれる機能があります．プログラム言語で言うメソッドのようなものと思ってください．そして，マクロを機能ごとにまとめたパッケージ（またはマクロファイル）が，あちこちで配布されています．この文章のソースコードを読み，どんなパッケージが使用されているか調べてみましょう．

パッケージを使うためには以下のように\usepackage を，\documentclass から\begindocument の間に記述します．ただし，標準的なパッケージは自動的に参照されますが，標準ではないパッケージを入手した場合は，tex ファイルと同じディレクトリに置くようにしてください．

余談ですが，% はコメントを表す記号です．

*1 コンパイラはさまざまな種類のものが存在します

hello.tex

```
\documentclass[a4j,10pt]{jsarticle}

\usepackage{okumacro} % 奥村先生による様々な便利機能
\usepackage[deluxe]{otf} % 日本語の太字をきちんと太字で表す
\usepackage{graphicx} % 図を貼り付ける際に使う．mediabb を使う場合は必須
\usepackage{mediabb} %PDF ファイルからサイズを自動取得する
\usepackage{amsmath} % 数式を，業界標準にする
\usepackage{enumerate} % 箇条書きを装飾する際に使う

\begin{document}
こんにちは世界
\end{document}
```

3.4 章・節・項について

章は\section コマンドで，節は\subsection コマンドで，項は\subsubsection コマンドで割り振ります．通常，章・節・図・式・表は，参照するための識別子を付加するため，\label コマンドと併用します．それぞれのコマンドの下にラベルを付けておいてください．以下に，具体的な例を示します．

章・節・項

```
\sectoin{説明}
\label{sec:description}
\subsection{章・節・項について}
\label{sec:section}
本文・・・
```

この内容が含まれた文章を作成すると，以下のように出力されます．

章・節・項の出力例

```
1 説明
1.1 章・節・項について
本文・・・
```

なお，章番号などは，TEX が割り振ってくれます．これにより，文章作成中に「この節は章として独立させよう」となった場合でも，ユーザは特に章番号を操作する必要はありません．また，文章を書いていると，「1 章に記したように」などと書くことがあります．ここで，章番号を手書きしていると，後々大きくずれて泣きながら訂正するはめに陥る場合があります．しかし，TEX では先に述べた label を使用することで，自動的に番号を合わせてくれます．

その場合，ref（reference：参照）を使用します．例では章番号を参照していますが，図でも表でも利用できます．

参照

```
\ref{sec:description}に書いたように・・・
```

なお，TEX は，コンパイル時にコンパイルしてから label 情報を作り直します．これは，とりあえずコンパイルしないと，図番号やページ番号が確定しないためです．コンパイルした時には 1 回前の label 情報が使用されているので，参照の状態をチェックしたい場合は続けて 2 回コンパイルしてみてください．特に，初回コンパイル時には章番号や目次内のページ番号が一切表示されなくて焦ることがあります．

3.5 空白について

スペースを空けようとしても，空けさせてくれない場合があります．そのために，スペースを空けるコマンドが用意されています[*2]．

縦のスペース \vspace{ <空ける量> }
横のスペース \hspace{ <空ける量> }

空ける量については，表 1 のように単位が定められています．よって，\hspace{ 1zw }などと入力すれば，全角 1 文字分空けてくれます．

表 1　空白の種類

単位	意味
cm	センチメートル
mm	ミリメートル
in	インチ
pt	ポイント
em	大文字の M の幅
ex	小文字の x の高さ
zw	全角文字の幅

3.6 数式の使い方

数式は，TEX の最も得意とする分野です．基本は equation コマンドを使って，以下のように記述します．

数式

```
\begin{equation}
y = x ^ 2
\end{equation}
```

すると，以下のように表示されます．数式番号は自動的につきます．

$$y = x^2 \tag{1}$$

また，文中にちょっとした数式を入れたければ，数式本体を$で囲ってください．例えば$ y=x^2 $とした場合，$y = x^2$ と表示されます．なお，数式コマンドについては省略します．慣れないうちは，LaTeXiT を併用してみてください．

3.7 図の貼付け方

図を貼り付ける場合は，PDF かそうでないか（JPEG や PNG）によって手間が異なります．本来の TEX だと，貼り付けようとしている図のピクセル数を与えなければなりません．しかし，PDF の場合は内部にピクセル数などのデータを保持しています．そのデータを，「mediabb」というマクロファイルが自動的に取得してくれます．

よって，PDF ファイルを貼り付けたい場合は，以下のように記述してください．includegraphics コマンドの width は紙面上の幅です．その次にファイル名を指定します．どちらも必ず指定してください．

[*2] この例は，見出し付き箇条書きの例でもあります

図

```
\begin{figure}[!htb]
\centering
\includegraphics[width=12cm]{branch.pdf}
\caption{プログラムの変更履歴とブランチ}
\label{fig:branch}
\end{figure}
```

なお，図のみの PDF の作り方ですが，PowerPoint などで図を書き→ PDF に変換したい部分を選択→右クリック→「図として保存...」→形式として PDF を選択するのが楽です．もしかしたら Mac OS X 限定かもしれません．

3.8 外部ファイル（ソースコードなど）の貼付け方

図の貼り付けと同じような感じですが，以下のようにします[*3]．itembox は，ファイル全体を角丸四角で囲み，ファイル名を表示します．実際のファイル名は，その次の verbatimtabinput で指定します．なお，verbatimtabinput の次の数値は，tab を置き換えるスペースの文字数です．枠線が必要なければ，itembox に関する記述は不要です．

図

```
\begin{itembox}[c]{Introduction1.java}
\verbatimtabinput[2]{Introduction1.java}
\ end{itembox}
```

3.9 箇条書き

箇条書きには「記号付き箇条書き」「番号付き箇条書き」「見出し付き箇条書き」の 3 つが用意されています．順番に例を示します．当然，項目はいくつあっても構いません．

3.9.1 記号付き箇条書きの書き方の例

```
\begin{itemize}
\item 1 番目の項目
\item 2 番目の項目
\end{itemize}
```

上記による表示例

- 1 番目の項目
- 2 番目の項目

3.9.2 番号付き箇条書きの書き方の例

```
\begin{enumerate}
\item 1 番目の項目
\item 2 番目の項目
\end{enumerate}
```

上記による表示例

1. 1 番目の項目

[*3] 3 行目の end の前にスペースが入っていますが，実際にはスペースは不要です

2. 2 番目の項目

3.9.3 見出し付き箇条書きの書き方の例

```
\begin{description}
\item[見出し 1] 1 番目の項目
\item[見出し 2] 2 番目の項目
\end{description}
```

上記による表示例

見出し 1 1 番目の項目
見出し 2 2 番目の項目

3.9.4 itemize で見出しを付ける書き方の例

以下の例では文章を見出しにつけていますが，本来は見出しに記号をつける場合に使います．文章見出しを付けたい場合は素直に description を使いましょう．

```
\begin{itemize}
\item[見出し 1] 1 番目の項目
\item[見出し 2] 2 番目の項目
\end{itemize}
```

上記による表示例

見出し 1 1 番目の項目
見出し 2 2 番目の項目

3.9.5 括弧を付けた番号付き箇条書きの書き方の例

括弧だけでなく好きな装飾を付けられます．また，1 と書いてあるのは数値を付けることを表します．他に A：大文字アルファベット，a：小文字アルファベット，I：大文字ローマン数字，i：小文字ローマン数字を使えます．

```
\begin{enumerate}[(1)]
\item 1 番目の項目
\item 2 番目の項目
\end{enumerate}
```

上記による表示例

(1) 1 番目の項目
(2) 2 番目の項目

3.9.6 行頭のスペースを変更する書き方の例

```
\begin{enumerate}[(1)]
\setlength{\rightskip}{3cm}
\setlength{\leftskip}{2cm}
\item 1 番目の項目
\item 2 番目の項目
```

```
\end{enumerate}
```

上記による表示例

(1) 1 番目の項目
(2) 2 番目の項目

3.9.7 箇条書き中断後に番号を継続する書き方の例

順番に解説すると，setcounter で変数に数値を代入することができます．まずは「mynumber」という変数を宣言し，最初の箇条書き終了時に変数「enumi」の値を「mynumber」に代入しておきます．この「enumi」という変数ですが，内部的に箇条書き番号を保持するための変数です．そして，箇条書き継続後に「mynumber」の値を「enumi」に代入することで，継続しているように見せています．

```
\newcounter{mynumber}
\begin{enumerate}[(1)]
\item 1 番目の項目
\item 2 番目の項目
\setcounter{mynumber}{\value{enumi}}
\end{enumerate}
ここで箇条書きを中止して，通常の文章を書きます．
その後，番号を継続したまま箇条書きを行います．
\begin{enumerate}[(1)]
\setcounter{enumi}{\value{mynumber}}
\item 3 番目の項目
\item 4 番目の項目
\end{enumerate}
```

上記による表示例

(1) 1 番目の項目
(2) 2 番目の項目

ここで箇条書きを中止して，通常の文章を書きます．その後，番号を継続したまま箇条書きを行います．

(3) 3 番目の項目
(4) 4 番目の項目

3.10 書体の変更について

TEX では，システムの定めた書体で出力されます．しかし，太字やゴシック体であれば使用できます．

ゴシック体を使いたい場合： \textsf{ゴシック体}→**ゴシック体**
太文字を使いたい場合： \textbf{太文字}→**太文字**
太いアルファベット： \textsf{alphabet}→ **alphabet**
等幅のアルファベット： \texttt{alphabet}→ `alphabet`