第 2 回 TEX ゼミ

ゼミ担当者　：　昌山　智，田中　裕也，坂田　大輔
指導院生　　：　小椋　信弥，澤田　淳二
開催日　　：　2003 年 4 月 17 日

ゼミ内容: 本ゼミでは，TEX で文書を作成する際に，基本となる事項について説明する．次に，EPS 画像の貼り付け方法について説明し，何故，EPS 形式を使うのかについても説明する．その後，TEX で表を作成する方法について説明する．

1　TEX の基本

前回は，TEX の歴史，インストール方法，三木研究室でのスタイルファイルのダウンロード方法，および保存場所について説明した．今回は，TEX による文章作成に必要な知識と基本的な手順を説明する．

1.1　TEX における約束

1.1.1　TEX のファイル名

TEX では，原稿ファイルの拡張子を“ .tex ”にしてください．仮に拡張子を“ .tex ”にしなかった場合でも，TEXmac の場合は「拡張子が“ .tex ”ではありません．拡張子を“ .tex ”に変更しますか」と聞かれるので最初から拡張子を“ .tex ”にしてください．

1.1.2　TEX の最低限のルール

TEX では原稿ファイルを記述する上での注意事項がある．以下にそれを説明する．

1. TEX や LATEX の組版命令は，原則として“ ¥ ”で始まり半角スペースで終わる．組版命令に続く文字が，全角の空白や句読点，全角および半角の括弧や記号（@など），半角の数字，TEX や LATEX の命令であれば半角スペースの必要はない．

2. 半角の空白はいくつ空けても 1 つと見なされ，出力は変わらない．改行直後の半角の空白は無視される．

3. 全角文字の直後で改行すると無視され，半角文字の直後で改行すると改行は空白と見なされる．2つ以上続いた改行は，段落の切れ目として扱われる．

```
\documentclass[a4paper]{jarticle}
\begin{document}
This is a
pen.
これは，ペン
です.
\end{document}
```

このソースを TEX で処理すると，

This is **a pen.** これは，**ペンです．**

と出力される．ソースの 3 行目末〜4 行目の **a pen.** の部分と，5 行目末〜6 行目の**ペンです．**に相当する部分に注目すると，**a pen.** の部分には空白がある．これは，行末が半角文字の場合には，改行が単語間の空白として処理されているからである．これに対して，**ペンです．**には改行をしたにも関わらず空白がない．これは，行末が全角文字の場合には，特に何の処理もされていないからである．

4. 半角記号で，入力しても出力されない特殊な文字がある．次の半角文字はそのまま出力できない．

   `#`, `$`, `%`, `&`, `_`, `{`, `}`, `<`, `>`, `\`, `|`, `~`, `^`

   これらを出力するための手軽な方法は全角文字を使用することである．全角文字は TEX の内部で特別な意味を持っていないので，自由に使用できる．また，書いたままを出力するためには，`\verb` という命令や`\begin{verbatim}....\end{verbatim}`を使用する．`\verb` の場合は，引数の始まりと終わりを同じ区切り記号で挟み，数文字程度の場合に使用する．区切り記号の間に記述する文字列は基本的にどのようなものでもかまわないが，両端の区切り記号と同じ文字だけは使用できない．また，* 記号を両端の区切り文字として使用することもできない．これは，`\verb*` という別の命令があるからである．verbatim は行単位の場合に使用するのが一般的である．

5. 半角のカタカナは使用できない．半角のカタカナを入力してもエラーにはならないが，出力されない．

6. “ % ”以降は，改行コードも含めてコメントと見なされる．行中の“ % ”記号以下は改行コードも含めてコメントとして扱われるため，原稿ファイル

では改行しているのに，改行していないのと同様に処理したい場合などには“ % ”記号を利用する．

1.2 環境と命令について

LaTeX の最大の特徴は論理デザインが可能であるということである．これは，文章の論理構造に応じて原稿に埋め込まれたマクロ命令によって，自動的に体裁を整えて文章を作成する．その論理的な構成は，**「環境」**と呼ばれるマクロ命令によってなされる．環境とは`\begin`{何々} ....`\end` {何々} のように対になった命令のことをいう．Fig. 1 に itemize 環境の例を示す．

```
\begin{itemize}
      .
      .
      .
\end{itemize}
```

Fig. 1　環境の例

つまり，{何々} にあたる部分が環境という事になる．環境内は一種の別天地でいろいろな設定が環境の外側と異なる．Fig. 1 の itemize 環境は箇条書きを行うための環境である．このほかにも図版を張りこむための figure 環境や中央揃えにする center 環境などがある．

命令は，環境のように`\begin` {何々} ....`\end` {何々} という形をとらず，\ 命令 {引数} という書式で記述する．例えば，節を定義する`\section` 命令などがある．`\section` {はじめに} という命令の場合，section という命令に引数「はじめに」が渡されることで,「はじめに」という節があることを宣言している．

1.3 段落の区切り

TeX は，連続する改行記号を段落の区切りとしている．簡単に言えば，何も記述されていない改行記号だけの行があれば，それを段落の区切りとみなしている．段落の始まりでは，TeX が自動的に英文の場合には適当な分量だけ，和文の場合には全角一文字分の字下げをしてくれる．また，ワープロの場合と異なり，改行コードだけの行を続けても，縦方向の空白を空けることはできない．縦方向の空白を空ける場合にはそのための命令を使用する．(空白を空ける命令については，2.2.2 節を参照．)

2 基本的な文書の編集

2.1 文字の修飾

文章を書くときに必要になってくる文字の修飾方法は LaTeX にも当然ある．ここでは，書体や文字サイズの変更方法について簡単に説明する．

2.1.1 文字サイズの変更

文字の大きさを変えるには，通常は Table 1 の 10 種類の命令を使用する．その出力方法を示す．

Table 1　文字サイズの変更

出力	ソース
sample	`{\tiny sample}`
sample	`{\scriptsize sample}`
sample	`{\footnotesize sample}`
sample	`{\small sample}`
sample	`{\normalsize sample}`
sample	`{\large sample}`
sample	`{\Large sample}`
sample	`{\LARGE sample}`
sample	`{\huge sample}`
sample	`{\Huge sample}`

2.1.2 書体の変更

書体の変更方法は Table 2 の通りである．

2.2 改ページと空白の開け方

2.2.1 改ページ

改ページは，通常 LaTeX が自動的に位置を決めて行う．しかし，ユーザが自ら指定することも可能である．次に改ページに関するコマンドをいくつか記す．

Table 2　フォントの変更

フォント名	出力	ソース
ローマン体	Roman.	\textrm{Roman.}
サンセリフ体	Sans Serif.	\textsf{Sans Serif.}
タイプライタ体	Typewriter.	\texttt{Typewriter.}
ボールド体	**Boldface.**	\textbf{Boldface.}
イタリック体	*Italic.*	\textit{Italic.}
スラント体	*Slanted.*	\textsl{Slanted.}
スモールキャピタル体	SMALL CAPS.	\textsc{Small Caps.}
明朝体	明朝体です.	\textmc{明朝体です．}
ゴシック体	**ゴシック体です．**	\textgt{ゴシック体です．}

\newpage　強制的に改ページしたい場合には，その場所で\newpage命令を使用する．二段組みをしている場合で，現在左の段であれば，\newpage 命令によって右側の段に移動する．

\clearpage　強制的に改ページしたい場合には，その場所で\clearpage命令を使用する．\newpage 命令との違いは，\clearpage命令の使用時に配置が決定されていない図表があれば，それらを全て出力してから改ページされることである．また，2 段組であるか無いかに関わらず，常に新しいページを起こす．

\cleardoublepage　次のページが右ページから始まるように，必要に応じて白紙のページを挿入して改ページしたい場合には，その場所で\cleardoublepage を使用する．

2.2.2　空白の開け方

先ほど説明したように，TEX では空白がいくつ続いても 1 つの空白とみなされる．1 つ以上の空白を空けるには，次の命令を使用する．

```
\vspace{height}
\hspace{weight}
```

\vspace{height}命令は”height”に記述された値だけ行間を空け，\hspace{weight}命令は”weight”に記述された値だけ文字と文字との間隔をあける．

```
\vspace{10mm}
```

この命令によって，

というように，行間を設定することが可能である．

\hspace{30mm}という命令では，　というように文字と文字の間隔を設定することができる．

また，段落間に適当な大きさの空白をあけたい場合には，次のような命令を使用する．

\smallskip は小さな空白を作る．

\medskip は中くらいの空白を作る．

\bigskip は大きな空白を作る．

このように，段落間の改行を操作することができる．

2.3　脚注

脚注を使用したい場合には，\footnote{ ···· } 命令を使用する．\footnote 命令を使用すると LATEX は組版時に自動的に番号を付け，引数に指定された文字列をページの下部に脚注として出力する．[1]

3　PostScript

パソコンで印刷物を作成する際には，画面上できれいに表示されるだけでなく，最終的に印刷物になったときにきれいな仕上がりになっていなければならない．そういった要求に応えたのが Adobe System 社が開発した PostScript である．

3.1　PostScript とは

PostScript は，強力なグラフィック機能を持ったプログラミング言語である．これはページ記述言語の一種で，PostScript 言語を用いると，印刷したいページの中身を記述することができる．1 つのページは必ず 1 つのグラフィックとして扱われる．PostScript は文字や画像を美

[1] 脚注はこのように出力される．

しく正確に印刷できるため，広告デザインや編集デザインなどにおいて広く使われている．

PostScript では，文字や図形や画像を扱うことができ，それらの属性やページ内での位置情報を指定できる．文字にはフォントや文字の大きさ，字飾りなどを指定することができ，図形は直線や円のほか，自由曲線を表現することが可能になっている．文字や図形は，ベジェ曲線を利用したベクトルデータ（図形中の主要な点の座標とそれらを結ぶ曲線の方程式のパラメータからなるデータ形式）として表現されるため，出力装置の最大解像度での精細な出力が可能となっている．

PS 言語の命令の列を直接プリンタに送る代わりにファイルの形で保存したものを，PS ファイルという．通常はテキストファイルなので，テキストエディタでの編集も行える．dvi も PS も元々は，プリンタや画面の解像度とは無関係に組版結果を記述するためにつくられたもので，dvi は文字情報に重点を置き，PS はあらゆる種類の画像を扱える点が異なる．

3.2 EPS（Encapsulated PostScript）とは

PS ファイルは，ページを記述したものであるため，中身が小さなグラフィックスの場合でもそのままでは別の PS ファイルに挿入できない．すなわち，ページの中にページを挿入できないということである．そこで，ページ内に挿入できるように，PS ファイルの内容にいくつかの制約事項を設ける方式が提案された．これが EPS 形式である．EPS ファイルは，一般の PS ファイルから「ページ」の概念をのぞいた限定的な PS ファイルであるといえる．EPS 形式のファイルには，PS 言語によるグラフィックス記述に加え，プレビュー用のビットマップ画像を収めることができる．EPS ファイルをエディタで開くとわかるが，ファイル内には

```
%BoundingBox: 0 0 413 139
```

のようなコメントが書かれている．これが外枠の座標を表している．このサイズにかかわらず，読み込む際は自由に，拡大・縮小できる．

3.3 EPS と BMP の違い

BMP の画像フォーマットは，基本的に画像を色の付いた点の集まりとして扱う．このような方式の画像は，通常何らかのアルゴリズムで画像データ自身が圧縮されており，ファイルの大きさを小さくするような工夫がされている．また，一般的に，表示にかかる計算量も少なくてすむ．しかし，コンピュータのモニタは，比較的解像度の低いピクセルを１つずつ表示するので，これらの画像ファイルを拡大または縮小しようとするのは困難となる．この様に画像フォーマットでは，画像を点の集まりとして考えているので解像度の変化に伴って品質が悪化することは否めない．

EPS は，このような問題点を解決するために，画像中に存在する各種要素を点に分解して保存するのではなく，出力の直前までは要素の属性を保持しておく方式である．すなわち，さきほども述べたとおり，ベクトル形式である．線分や円でも同様に，出力の直前までは線分であれば端点を，円であれば中心と半径をという具合に，図形の情報として保管しておく．もちろん，最終的にディスプレイやプリンタで出力する段階では，要素を点の集まりに分解しなければならないが，分解はデバイスの処理能力に応じて実行されるため，描画イメージが粗くなるという事態は避けられる（Fig. 2）．

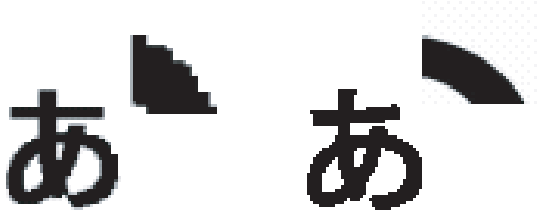

Fig. 2　BMP と EPS の違い

4　画像の貼り付け

ここでは，TEX へ画像を貼りこむ方法を説明する．今まで Microsoft Word などでは，画像は主に JPG を用いていたと思うが，ここ三木研では画像はすべて EPS 形式で貼り付ける．

4.1 画像の EPS への変換

まず，BMP や JPG の画像を EPS に変換する方法を説明する.BMP の画像を EPS ファイルとして保存する方法はいくつかのソフトで実行可能であるが，ここでは Adobe 社の Illustrator10J を例にとって解説する．まず Illustrator を起動する（Fig. 3）．

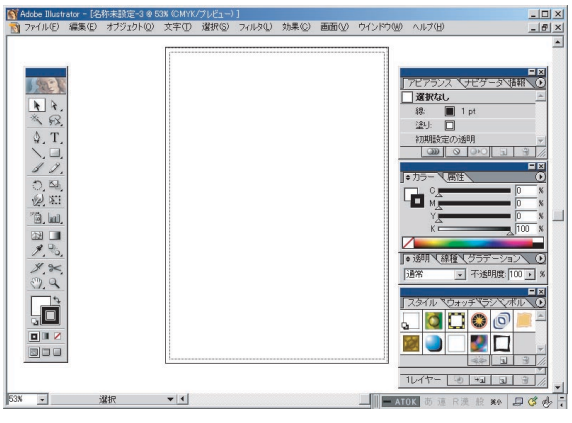

Fig. 3　Illustrator の起動画面

次に，適当な画像ファイルを開く．ただし画像が中央

に表示されている枠（バウンディング・ボックス）からはみ出すと，TEX に取り込んだときに画像が切れてしまうので，貼り付けた画像がはみ出さないように調整する（Fig. 4）．

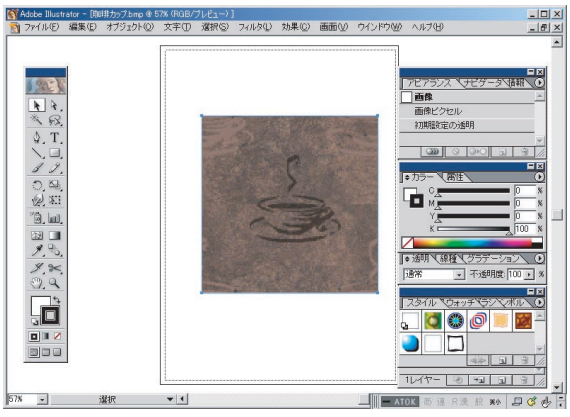

Fig. 4　バウンディング・ボックス内の画像

EPS へ変換するには，この調整した画像を EPS ファイルとして保存する．ちなみに，EPS はベクトル形式なので，画像のサイズをいくら縮小，拡大しようと，画像データが損なわれることはない．まず,「メニュー」の「ファイル」から「別名で保存」を選ぶ．デフォルトではファイルの種類が「Illustrator」でファイル名の部分の拡張子が「.ai」となっているはずなので，ファイルの種類から「Illustrator EPS」を選ぶ．ファイルの拡張子が「.eps」になるので，ここで「保存」ボタンを押す（Fig. 5）．

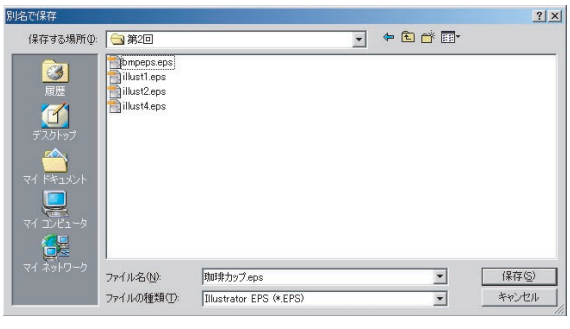

Fig. 5　画像の保存

次に，EPS の形式の設定を行う．先ほどの保存ボタンを押すと（Fig. 6）のような画面が表示されるので，そのまま「OK」ボタンを押す．プレビュー画像は，EPS ファイルを直接表示できないアプリケーションで表示される．必要なければ，プレビューの「形式」で「なし」を選択する．

最後に警告画面が表示されるが，気にせず「続行」を押す（Fig. 7）．

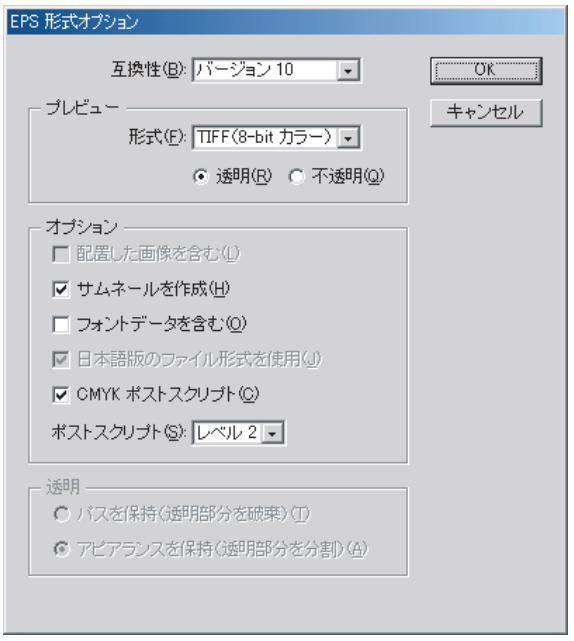

Fig. 6　EPS 形式オプション

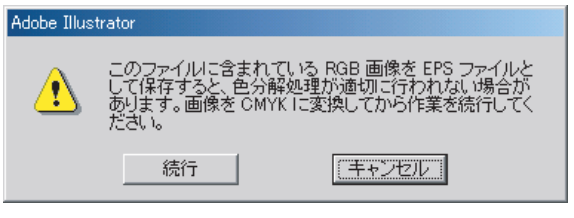

Fig. 7　警告画面

以上が，EPS ファイルとして画像を保存する方法である．

4.2 TEX への図の挿入

TEX に図を挿入するには graphics またはそれを拡張した graphicx というパッケージを使う．graphicx の方が graphics より高性能であり，DVIOUT とも相性がよいので以下では graphicx を使う．このパッケージを用いるには，LATEX のプリアンブルに

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

と書いておく．そして，図 EPS.eps を入れたいところに

```
\includegraphics[width=7.5cm, height=5cm]{EPS.eps}
```

のように書く．これは図 EPS.eps を幅が 7.5cm，高さが 5cm になるように拡大縮小してその場所に出力することを意味する．また

```
\includegraphics[width=7.5cm, clip]{EPS.eps}
```

のようにすると，縦横比（aspect ratio）を保ったまま，幅 7.5cm に収まるように拡大・縮小する．一般的にはこちらの方を用いるとよい．また，通常論文などを書く場合には，

```
\begin{figure}[htbp]
\begin{center}
\includegraphics[width=7.5cm, clip]{tefu/EPS.eps}
\caption{図の解説をここに書く}
\label{図の参照名をここに書く}
\end{center}
\end{figure}
```

のように，figure 環境に入れて用いる．figure 環境は，図版を貼り込む領域を確保するための環境である．`\caption` の引数には，図の説明を入れ，`\label` の引数には，図の参照名を入力する．本文中で図の参照を行うときにこの参照名を用いると，図の番号を自動的に付けてくれる．たとえば，参照名が ”EPS7 ”である図を本文中で参照するには，たとえば

図`\ref{EPS7}` を参照のこと

と入力する．この出力結果は，

図 3.8 を参照のこと

のようになる．

以上が，TEX への基本的な図の貼り付け方である．

5 表の作成

図版と同様，LATEX には表を最適な場所に配置するための環境も用意されている．本章では，表を配置したり作成したりする環境について解説する．

5.1 表組みの基本

最も基本的な表組みの例を Table 3 に示す．

上の表組みを出力するソースは Fig. 8 のとおりである．

Table 3 最も基本的な表

書名	数量	金額
進化する人工物	1	1400 円（税別）
未来革命	1	1700 円（税別）

```
\begin{center}
  \begin{tabular}{lcr}
    書名 &  数量 & 金額  \\
    進化する人工物 & 1 & 1400 円（税別）\\
    未来革命 & 1 & 1700 円（税別）\\
  \end{tabular}
\end{center}
```

Fig. 8 最も基本的な表のソース

center 環境で囲まれた部分は，中央揃えが行われる．実際の表に関する記述は 2 行目の`\begin{tabuler}{lcr}` から始まる．

`\begin{tabular}...\end{tabular}`までが表そのものを出力する tabular 環境である．tabular 環境の書式と引数の指定をそれぞれ，Fig. 9 および Table 4 にまとめる．

```
\begin{tabular}{引数 (列指定)}
                ⋮
\end{tabular}
```

Fig. 9 tabular 環境の書式

引数（列指定）は列の数だけ並べる．Table 3 では列指定は lcr なので，1 列目は左寄せ，2 列目は中央，3 列目は右寄せになっている．また，列の幅を固定したい場合は p{} を使う．例えば p{3zw} とした場合，その列は全角 3 文字分の幅に固定することができる．

列の区切りは`&` で行う．また，行の記述が終わったあとは，改行を意味する`\\`が必要になる．

5.2 罫線

5.2.1 罫線の引き方

前節で基本的な表の作成を解説した．この節では表にはなくてはならない罫線の引き方を説明する．実際に前節で作成した表組みに罫線を加えたものを Table 5 に，そのソースを Fig. 10 に示す．

縦罫線の引き方は { 列指定 } の中の該当個所に半角縦線 | を入れる．また 2 重の縦罫線にしたいときは ‖ と書く．

Table 4　tabular 環境の引数

指定	解説
l	左寄せ
c	中央
r	右寄せ
p{ }	左寄せ，長さ指定
|	縦の罫線
||	縦の二重罫線

Table 5　罫線を引いた表

書名	数量	金額
進化する人工物	1	1400 円（税別）
未来革命	1	1700 円（税別）

横罫線は\hline で引かれる．\hline\hline と書くと，2 重の横罫線になる．

5.2.2　部分的に罫線を引く方法

\hline 命令を使用すると，表の幅の分だけ横の罫線を出力する．しかし，場合によっては，1，2，4 列目だけに引きたい場合がある．例えば，Table 6 のような表を作りたいとする．

Fig. 11 に示したように，この場合には\hline 命令の代わりに\cline 命令を使う．
\cline 命令は\cline{数字 - 数字}のように罫線を引きはじめたい列から引き終わりたい列までの数字を書く．

5.3　1 行あたりの要素数の変更

表を組んでいると，たとえば要素数は 4 つだが，見出しの行だけ 4 つの要素の欄をまとめて使用したいとか，表の要素は左揃えで出力したいが，見出しの要素だけは中央揃えにしたいということがある．たとえば，Table 7 のように出力したいという場合がある．

Table 7 は「請求書」が 3 列分まとめて中央揃えで出力されているのがわかる．Table 7 のソースは Fig. 12 のようになる．

列をまとめるには，\multicolumn という命令を用いる．この\multicolumn 命令は 3 つの引数を持っており，前から順にまとめる欄の数，まとめた欄における要素の

```
\begin{tabular}{|l|c|r|} \hline
 書名 & 数量 & 金額 \\ \hline \hline
  進化する人工物 & 1 & 1400 円（税別）\\
  未来革命 & 1 & 1700 円（税別）\\ \hline
\end{tabular}
```

Fig. 10　罫線を引いた表のソース

Table 6　部分的に罫線を引いた表

出版社	書名	著者	定価
A 社	TEX とは何か	鈴木一郎	1400 円
B 社	TEX の活用術		1800 円

```
\begin{tabular}{|c|c|c|c|} \hline
 出版社 & 書名 & 著者 & 定価 \\
 \hline\hline
 A 社 & \TeX とは何か & 鈴木一郎 & 1400 円 \\
 \cline{1-2}\cline{4-4}
 B 社 & \TeX の活用術 &  & 1800 円 \\\hline
\end{tabular}
```

Fig. 11　部分的に罫線を引いた表のソース

Table 7　1 行目をまとめた表

請求書		
書名	数量	金額
進化する人工物	1	1400 円（税別）
未来革命	1	1700 円（税別）

```
\begin{tabular}{|l|c|r|}
  \multicolumn{3}{|c|}{\textbf{請求書}}\\
 \hline\hline
  \multicolumn{1}{|c|}{書名} & 数量 & 金額\\
 \hline\hline
  進化する人工物 & 1 & 1400 円（税別）\\
  未来革命 & 1 & 1700 円（税別）\\\hline
\end{tabular}
```

Fig. 12　1 行目をまとめた表のソース

配置位置，要素となっている．Fig. 12 では，

```
\multicolumn{3}{|c|}{\textbf{請求書
}} \\\hline\hline
```

となっているが，この行は次のような意図によって指定されている．

1. **tabular 環境で作成されるすべての欄をまとめて見出しにしたい**

   Fig. 12 の tabular 環境で作成される欄の総数は 3 つである．そこでまず，\multicolumn 命令で最初の欄以降の 3 つの欄をまとめる (\multicolumn 命令の第 1 引数を"{3}"にする)．

2. **見出しは中央揃えにしたい**

   \multicolumn 命令でまとめた 3 つの欄を，新たに要素数が 1 つで，要素を中央揃えで配置する欄として定義しなおす．なお，要素の左右には縦罫を引く (\multicolumn 命令の第 2 引数を"{|c|}"にする)．

3. **見出しの欄には，太字で請求書と出力したい**

   \multicolumn 命令で新たに設定された欄に，"請求書"という文字列を入れる (\multicolumn 命令の第 3 引数を"{\textbf{請求書}}"にする)．

   また，Fig. 12 の 4 行目にみられる 2 つの\multicolumn は，本来なら左揃えや右揃えである 1，3 列目を，見出しだけ中央揃えにするという意図がある．

5.4 横幅の決まった表

全体の横幅の決まった表は tabular 環境の代わりに tabular*環境を使う．Fig. 13 のように使う．

```
\begin{tabular*}{(幅)}{@{\extra
colsep{\fill}}(列指定)}
      ⋮
\end{tabular}
```

Fig. 13　横幅を決める表の書式

幅を合わせるために列間に均等に空きが入る．Table 8，Fig. 14 は幅を 80mm にしたときの表と，そのソースである．

5.5 table 環境

これまで，tabular 環境の解説をしてきたが，tabular 環境はあくまで表を作るだけなので，表の上部に説明を加えたり，本文中でその表を参照するなどということはできない．そこで用いるのが，表全体の領域を確保するための table 環境である．table 環境を用いれば，

Table 8　横幅を 80mm にした表

請求書		
書名	数量	金額
進化する人工物	1	1400 円 (税別)
未来革命	1	1700 円 (税別)

```
\begin{tabular*}{80mm}{@{\extracolsep{\fi
ll}}|p{10zw}|r|r|}\hline
 \multicolumn{3}{|c|}{\textbf{請求書}} \\
 \hline\hline
 書名 & 数量 & 金額 \\ \hline
 進化する人工物 & 1 & 1400 円 (税別)\\
 未来革命 & 1 & 1700 円 (税別)\\ \hline
\end{tabular*}
```

Fig. 14　横幅を 80mm にしたときの表のソース

```
\caption{説明}
```

で表の説明をつけたり，

```
\label{参照名}
```

で，参照名をつけたりできる．一般的に表の説明は表の上に来るように書く．

後で説明するように，mikilab スタイルでは本文中での表の参照は\tbref を用いる．たとえば，

```
··· については，\fbref{table1}を参照のこと．
```

とソースに記述しておくと，出力結果は

··· については，Table 2 を参照のこと．

のようになる．

また，\begin{table*}とすると，2 段組のクラスファイルを使用しているときに，段組を抜いて出力することができる．

6 スタイルファイル

6.1 スタイルファイルとは？

LaTeX 2_εでは，文書ファイルには文書構造だけを指定し，文書構造と実際のレイアウトとの対応は，クラスファイル (cls ファイル) という別ファイルで定義するという方法をとっている．また，それらクラスファイルを補う形で，いろいろなスタイルファイル (sty ファイル) を必要に応じて文書ファイルのプリアンブルで読み込ませる．

Fig. 15　ヘッダ，題目および著者名

6.1.1　mikilab.sty について

月例発表会スタイルファイル（mikilab.sty）は，知的システムデザイン研究室の月例発表会のレジュメ作成用のスタイルオプションファイルで，そのためのマクロがいくつか登録されている．今回の TEX ゼミでは，それらマクロの用い方を解説する．

mikilab スタイルを用いるには，原稿のプリアンブルに次のように記述する．

```
\usepackage{mikilab}
```

6.1.2　mikilab.sty のマクロ

mikilab.sty には，月例発表会のレジュメ用のマクロがいくつか登録されているが，具体的には以下のとおりである．なお，各マクロ命令の詳しい説明は，第 7 節のコマンド集にあるのでそちらを参照すること．

- **ヘッダ**

  1 ページ目のヘッダに月例発表会の通算回数と開催年・月を記述するための，`\beginheader` というマクロ命令が用意されている．

- **題目**

  日本語題目を入力するために`\title` というマクロ命令が用意されている．題目を出力するのに `\maketitle` は必要としない．

- **著者名**

  著者名を入力するために`\author` というマクロ命令が用意されている．連名の場合には「,」で区切る．著者名を出力するのに`\maketitle` は必要としない

以上のマクロ命令を用いると，Fig. 15 のような出力が得られる．

- **概要**

  レジュメの概要は**英文**で 4 行程度にまとめておくこと．概要の記述には abstract 環境を利用する．出力は，たとえば次のようになる．なお，太字の **Abstract:**は自動的に加えられる．

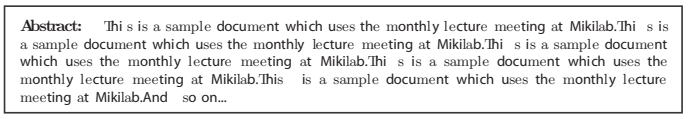

Fig. 16　概要の出力

- **図，表および式の参照**

  図，表，式の参照は通常`\ref` 命令が使われるが，mikilab スタイルでは新たなマクロが登録されている．図を参照するには`\fgref`，表を参照するには`\tbref`，数式を参照するには`\eqref` を用いる．

7　mikilab スタイルで使用できるコマンド一覧表

7.1　先頭に関するコマンド

\documentclass	mikilab スタイルでは，デフォルトで\documentclass[a4paper,10pt]{jarticle}と表示される．[] の中で，用紙の大きさ，基本となる文字の大きさ，用紙の方向，段組み，印刷の種類，標題・概要ページの扱い，試し刷り，章の開始ページの設定などを決めることができる． 次に {} の中では，文書クラスごとのデフォルトのクラスオプションを設定できる．今回の jarticle の場合では，A４用紙・標準文字サイズ 10pt・片面印刷・1 段組み・試し刷りの設定 OFF というデフォルトになっています．また，jreport に変更すると，章（section）が使用可能になる．

\usepackage	LaTeX は，新たなマクロを定義した「パッケージ」と呼ばれるファイルを読み込むことで，命令を変更したり追加したりすることができる．いくつかのパッケージは標準の LaTeX の命令が持っている制限事項を取り払い，またあるパッケージは LaTeX が使用できるフォントの数を増やし，別のパッケージはたくさんの命令や環境を追加する． つまり，パッケージを読み込むことで，LaTeX に表現可能な文書の世界を大きく広げてあげるのです．パッケージは通常，拡張子が“ .sty ”のファイルによって提供されます． 三木研究室では，mikilab.sty を使うことにより，すべての環境が整います．

\onecolumn	新規ページを起こして，以降を一段組みとします．
\twocolumn	新規ページを起こして，以降を二段組みとします．

具体例：abstract 環境を二段抜きで出力する場合

```
\documentclass[a4j,twocolumn]{jarticle}
\usepackage{mikilab}
\pagestyle{empty}
\begin{document}
\twocolumn[%
\begin{center}
{\LARGE\textbf{二段抜き}}\\ \vspace{10mm}
{\large\textbf{abstract 部}}\\ \vspace{3mm}
この部分は二段抜きされている部分です
\vspace{3mm}
\end{center}]
この部分から２段組が始まります．
\newpage
２段組の右側です
\end{document}
```

Fig. 17　ソースファイル

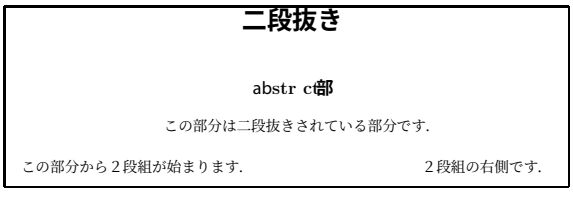

Fig. 18　Fig. 17 の実行結果

\pagestyle	ページ番号を出力するのか，そしてどこに出力するのかをプリアンブルで設定することができます．引数が plain なら，各ページの下部にページ番号を出力し，empty なら，ページ番号を一切出力しません．
\setcounter{page}{0}	ページ番号の初期値を変更します．これでは 0 ページからになります．
\thispagestyle	特定のページだけページ番号の出力方式を変えたい場合に，そのページの適当な部分に記述します．引数は\pagestyle と同じです．

7.2 月例用ヘッダに関するコマンド

\beginheader	１ページ目のヘッダに月例発表会の通算回数と開催年･月を記述するための，\beginheader というマクロ命令を用意しました．Fig. 19 を参照し，以下のような書式で題目を入力してください．
\title	題目を入力するために￥title というマクロ命令を用意しました．Fig. 19 を参照し，Fig. 20 のような書式で題目を入力してください．
\author	著者名を入力するために￥author というマクロ命令を用意しました．Fig. 19 のような書式で題目を入力してください．

ヘッダの作成例

```
\begin{document}
\beginheader{99}{2001}{4}
\title%
{月例発表会レジュメ用スタイルファイル（mikilab.sty）の使い方}%
{How to use mikilab.sty}
\author{昌山　智，坂田　大輔，田中　裕也}{Satoru MASAYAMA, Daisuke SAKATA, Yuya TANAKA}
\endheader
\end{document}
```

Fig. 19　ヘッダのソース

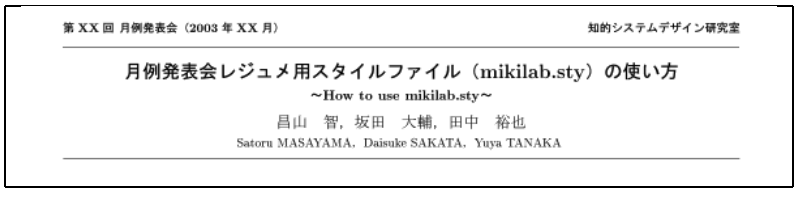

Fig. 20　ヘッダの出力例

7.3 アブストラクトに関するコマンド

\abstract	abstract 環境は，レジュメの概要を記述するための環境です．概要は，英語で 4 行程度にまとめて記述してください．

abstract の使用例

```
\begin{abstract}
This is a sample document which uses the
monthly lecture meeting at Mikilab. This is
a sample document which uses the monthly
lecture meeting at Mikilab. \end{abstract}
```

Fig. 21　ソースファイル

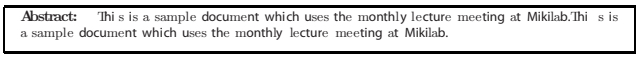

Fig. 22　abstract の出力例

7.4 参照に関するコマンド

Table 9 に参照に関するコマンドを示す．

7.5 フォントおよびサイズに関するコマンド

フォントに関するコマンドを Table 10 に，サイズに関するコマンドを Table 11 に示す．

Table 9　参照に関するコマンド

参照の種類	ソース	出力例
通常の参照	\ref{(参照名)}	1.1
図の参照	\fgref{(参照名)}	Fig.1.1
表の参照	\tbref{(参照名)}	Table.1.1
数式の参照	\eqref{(参照名)}	式 (1.1)

Table 10　フォントの変更

フォント名[2]	ソース	出力例
ボールド体	\textbf{(文字列)}	**ボールド体 BoldFace**
タイプライタ体	\texttt{(文字列)}	タイプライタ体 TypeWriter
イタリック体	\textit{(文字列)}	イタリック体 *Italic*
ゴシック体	\textgt{(文字列)}	**ゴシック体**

7.6　脚注に関するコマンド

脚注を表示するには，\footnote 命令を用います.

ソース

```
このページの下\footnote{これが脚注です}にあるのが脚注です.
```

出力

このページの下[a]にあるのが脚注です.

[a] これが脚注です

7.7　見出しに関するコマンド

見出しに関するコマンドは以下のようなものが存在するが，プリアンブルで指定したクラス（\documentclass[]{クラス}）によって，結果は異なる．今回の例では jreport クラスによる例を紹介する.

\chapter	章の設定であり，部の次にに大きいまとまりであります．しかし，jarticle クラスと tarticle クラスには用意されていません.
\section	節の設定であり，３番目に大きいまとまりであります.
\subsection	項（小節）の設定であり，４番目に大きいまとまりであります.
\subsubsection	目（小々節）の設定であり，５番目に大きいまとまりであります.

具体例：見出しの使用

Table 11　サイズの変更

ソース	出力例
{\small Sample}	Sample
{\large Sample}	Sample
{\Large Sample}	Sample
{\LARGE Sample}	Sample
{\huge Sample}	Sample
{\Huge Sample}	Sample

```
\documentclass[a4paper,10pt]{jreport}
\usepackage{mikilab}
\pagestyle{empty}
\begin{document}
\chapter{これが章です.}
\section{これが節です.}
\subsection{これが小節です. }
\subsubsection{これが小々節です.}
\end{document}
```

Fig. 23　ソースファイル

第1章　これが章です.

1.1　これが節です.

1.1.1　これが小節です.

1.1.1.1　これが小々節です.

Fig. 24　Fig. 23 の実行結果

7.8　空白調節に関するコマンド

\hspace{}	命令の書かれた行と次行とのあいだを，引数の値だけ空ける.
\vspace{}	命令の書かれた箇所を，引数の値だけ空ける.

具体例：空白調節の使用

```
\section{空白調節なし}
かなり空いてます.
```

Fig. 25　調節前のソースファイル

第1章　空白調節なし

かなり空いてます.

Fig. 26　Fig. 25 の実行結果

```
\section{空白調節あり}
\vspace{-15mm}
狭くなりました.
```

Fig. 27　調節後のソースファイル

第1章　空白調節あり

狭くなりました.

Fig. 28　Fig. 27 の実行結果

7.9　環境に関するコマンド

documentclass 命令	「あらかじめ定義された設定」を選択し，文書の基本的な体裁を指定する命令であります. 具体的には，文書の種類（文書クラス）とそのオプション（クラスオプション）を定義する働きをもっています.
document 環境	LaTeX に「そこが組版し，出力すべき文章である」ことを伝えます. また，documentclass 命令と document 環境の間の領域のことを「プリアンブル」といいます.

LaTeX の基本原稿構造

figure 環境	「図版を張り込む領域を確保する環境」です．figure 環境の中には，実際の図表を出力させるための\includegaraphics 命令や，必要に応じて\caption 命令や\label 命令を指定します．後者の 2 つは必ず指定しなければならないわけではありませんが，文書全体をわかりやすくするために，極力指定しましょう。
\caption{}	図版の説明文を指定します．{} の中に説明文を記述してください.
\includegraphics[]{}	図版を貼り込むための命令です．[] にはオプション引数として，貼り込み後の画像サイズを，{} には引数として，貼り込みたい画像ファイル名（eps）が必要となります..
\label{}	図版の参照用の目印を付け加えます.

```
\document[クラスオプション]{文書オプション}
プリアンブル
\begin{document}
原稿の本体
\end{document}
```

Fig. 29　基本構造

figure 環境 の基本原稿構造

```
\begin{figure}
            実際の図版
            \caption{図版の説明文}
            \label{ラベル}
\end{figure}
```

Fig. 30　figure 環境

table 環境	「表を張り込む領域を確保する環境」です．詳細は figure 環境と同じなので省略します．なお，二段組みの文書内で二段抜きの表を作成したい場合は，table*環境を用いてください．

tabular 環境	同レジュメ第 5 節に詳細があります．

verbatim 環境	\verb 命令によって，特殊文字を出力させることができました．しかし，プログラムリストのように複数行に渡る原稿をそのまま出力させるときは面倒です．このようなときに，verbatim 環境を使います．この環境内に書かれた文章は，そのまま出力されます．

itemize 環境	itemize 環境は，複数の項目を見出し付き箇条書きにしたいときに用います．itemize 環境の中では，各項目は\item 命令を使用して並べます．また，箇条書きは第 4 レベルまで入れ子状態にすることも可能です．

```
\begin{itemize}
 \item 第 1 レベルの項目 1
 \item 第 1 レベルの項目 2
  \begin{itemize }
   \item 第 2 レベルの項目 1
   \item 第 2 レベルの項目 2
  \end{itemize}
 \item 第 1 レベルの項目 3
\end{itemize}
```

Fig. 31　ソース

- 第 1 レベルの項目 1
- 第 1 レベルの項目 2
  - 第 2 レベルの項目 1
  - 第 2 レベルの項目 2
- 第 1 レベルの項目 3

Fig. 32　出力

enumerate 環境	番号などを付けた箇条書きのための環境です．itemize 環境同様，enumerate 環境も入れ子（第 4 レベルまで）にすることが可能です．